Dimensionality Hyper-Reduction and Machine Learning for Dynamical Systems with Varying Parameters

Syuzanna Sargsyan

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2016

Reading Committee:

J. Nathan Kutz, Chair

Steven L. Brunton

Eric Shea-Brown

Maryam Fazel

Program Authorized to Offer Degree:
Applied Mathematics

University of Washington

Abstract

Dimensionality Hyper-Reduction and Machine Learning for Dynamical Systems with Varying Parameters

Syuzanna Sargsyan

Chair of the Supervisory Committee:
Professor J. Nathan Kutz
Applied Mathematics

This work demonstrates methods for hyper reduction and efficient computation of solutions of dynamical systems using optimization and machine learning techniques. We consider nonlinear partial differential equations that under discretization yield a system of nonlinear differential equations. Discretization can be performed using finite element techniques, spectral methods, finite-difference methods, or in the case of conservation laws, finite-volume methods. However, such complex systems generate very large-scale problems leading to very long simulation times. Therefore, these models are not practical to use for real-time scenarios. Reduced order models simplify this type of computation and often assure significant speedups for these types of problems. One such method is described in this work. We demonstrate the synthesis of sparse sampling and machine learning to characterize and model complex, nonlinear dynamical systems over a range of bifurcation parameters. It is shown that nearly optimal sensor placement locations can be achieved by applying a genetic algorithm on a generalization of the discrete empirical interpolation method with varying parameters. On the other hand, there is no guarantee that these methods preserve the problem structure which can lead to instability and inaccuracy. In this work this issue was addressed for systems that are solved with finite volume methods. This was done by formulating a constrained optimization problem and solving it nearly as efficiently as the unconstrained

problem. The introduced constraints insure that the most important properties of the model are preserved despite the fact that the dimensionality of the system was greatly reduced. Exemplary results are computed for Euler equation for an inviscid compressible flow and the complex Ginzburg-Landau equation.

TABLE OF CONTENTS

LIST OF FIGURES

Figure Number Page

LIST OF TABLES

ACKNOWLEDGMENTS

Firstly, I would like to express my deepest and sincere appreciation to my advisor Prof. J. Nathan Kutz for the continuous support, patience and encouragement during my Ph.D study and guidance in related research. His advice helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor for my Ph.D study.

Similarly, I would like to thank my collaborators Steven L. Brunton and Kevin T. Carlberg for their guidance and invaluable advice that helped me to become a researcher. I would also like to thank my previous advisors Armen Kamalyan and Mher Martirosyan helping me to build my mathematical background and strong analytical skills.

I am grateful to the department of Applied Mathematics for all the support I had during my studies. This work would not be possible without my committee members Eric Shea-Brown and Maryam Fazel. I would like to thank them for their time and efforts.

Thank you also my friend Bethany Lusch. Without you I would not be a person I am right now. I would also like to sincerely thank my family mom, dad, sister, brother and grandmother for their constant love, endless support and encouragement. Lastly, I would like to express my dearest appreciation to my partner Albert Tamazyan for his patience, support and for being there for me.

DEDICATION

To my family, Georgie, Armine, Svetlana, Anzhelika and Hrach

Chapter 1

INTRODUCTION

1.1 Reduced Order Models

The theoretical study of nonlinear dynamical systems pervades the physical, biological and engineering sciences. Today, these studies are increasingly driven by computational simulations that are of growing complexity and dimension due to increasing computational power and resolution in numerical discretization schemes. Yet most dynamics of interest are known ultimately to be low-dimensional in nature [31], thus contrasting, and in antithesis to, the high-dimensional nature of scientific computing. Reduced order models (ROMs) are of growing importance in scientific applications and computing as they help reduce the computational complexity and time needed to solve large-scale, engineering systems [74, 11]. Specifically, ROMs provide a principled approach to approximating high-dimensional spatio-temporal systems. They first perform an offline expensive computations to build an approximation basis for the state of the model. This is followed by an inexpensive online stage, when the model is projected onto the subspace spanned by the basis computed on the offline stage. However, the complexity of the projection stage remains challenging due to higher-order nonlinear terms [10, 29]. The empirical interpolation method (EIM), and the simplified discrete empirical interpolation method (DEIM) for the proper orthogonal decomposition (POD) [60, 45], overcomes this difficulty by providing a computationally efficient method for discretely (sparsely) sampling and evaluating the nonlinearity. These methods ensure that the computational complexity of ROMs scale favorably with the rank of the approximation, even for complex nonlinearities.

An alternative computational strategy for handling the nonlinearity is based upon machine learning techniques of dimensionality reduction whereby libraries of *learned* POD modes

can be constructed and inner products pre-computed for a number of distinct dynamical regimes of the nonlinear dynamical system [16, 12, 72, 55]. This strategy also evokes the power of compressive sensing for efficiently identifying the active POD subspace necessary for a low-dimensional Galerkin-POD truncation [60, 45]. One of the main points of this work is to combine the power of the DEIM with the library building strategy. Specifically, we show that building libraries that encode the nonlinearities allows one to (i) take advantage of the DEIM to evaluate the nonlinearities, (ii) more robustly classify the dynamical regime the system is in, and (iii) identify the discrete sensor locations to construct a nonlinear model reduction. Our method allows for orders-of-magnitude improvement in computational speed and memory requirements, thus making it highly attractive for computational physics simulations. We demonstrate the full integration of the methods on a canonical model of mathematical physics and nonlinear dynamical systems, the cubic-quintic Ginzburg-Landau (CQGLE) equation. However, the methodology can be more broadly applied to nonlinear dynamical systems. In biophysical applications, for instance, advances in neuroscience over the past decade have revealed two critical, and seemingly ubiquitous, phenomena: (i) that meaningful input/output of signals in high-dimensional networks of neurons are encoded in low-dimensional patterns of dynamic activity [48, 76, 75, 56, 44, 81], and (ii) that sparsity plays a key role in encoding and decoding strategies, especially in neural computation [41, 65]. This typifies the state-of-the-art in neuroscience, which manifests the most sophisticated functionality in information processing and computation known in any system.

1.2 ROM methodology

Although a variety of dimensionality-reduction techniques exist, the ROM methodology considered here is based upon the proper orthogonal decomposition [60, 45]. The POD method is ubiquitous in the dimensionality reduction of physical systems. It is alternatively referred to as principal components analysis (PCA) [67], the Karhunen–Loève (KL) decomposition, empirical orthogonal functions (EOF) [59], or the Hotelling transform [46, 47]. Snapshots (measurements) of many nonlinear dynamical systems often exhibit low-dimensional phe-

nomena [31], so that the majority of variance/energy is contained in a few modes computed from a singular value decomposition (SVD). For such a case, the POD basis is typically truncated at a pre-determined cut-off value, such as when the modal basis contain 99% of the variance, so that only the first r-modes (r-rank truncation) are kept. There are numerous additional criteria for the truncation cut-off, and recent results derive a hard-threshold value for truncation that is optimal for systems with well-characterized noise [42]. The SVD acts as a filter, and often the truncated modes correspond to random fluctuations and disturbances. If the data considered are generated by a dynamical system (nonlinear system of ordinary differential equations of order n), it is then possible to substitute the truncated POD expansion into the governing equation and obtain Galerkin projected dynamics on the rank-r basis modes [45, 55]. Recall that we are assuming that the nonlinear dynamical systems under consideration exhibit low-dimensional attractors, thus the Galerkin truncation with only a few modes should provide an accurate prediction of the evolution of the system [16, 12]. Note that it has also been shown recently that it is possible to obtain a *sketched*-SVD by randomly projecting the data initially and then computing the SVD [39, 43, 73].

1.3 Hyper Reduction: Sparse Sampling

For physical systems that require high spacial resolution, however, the above described methods could lead to infeasibly high computations. Efficiently managing the computation of the nonlinearity (inner products) in dimensionality reduction schemes is of paramount importance. This was recognized early on in the reduced order modeling community, and a variety of techniques where proposed to accomplish the task. Among the first methods used was the technique of Everson and Sirovich developed for gappy data [37]. In their sparse sampling scheme, random measurements were used to perform reconstruction tasks of inner products. Willcox [85] and Karniadakis [87] built on these ideas by advocating principled approaches for selecting sampling locations for Gappy POD. Carlberg et al. [24] developed the Gauss-Newton with approximated tensors (GNAT) scheme that is successfully applied to finite volume models arising in problems in turbulent, compressible fluid dynamics.

The EIM was also developed for the purpose of efficiently managing the computation of the nonlinearity. And as with Gappy POD, principled techniques for sparse measurements where also advocated early on in its history [64]. A variant of this techniques, the DEIM method, was specifically tailored to POD with Galerkin projection. Indeed, the DEIM approximates the nonlinearity by using a small, discrete sampling of spatial points that are determined in an algorithmic way. This ensures that the computational cost of evaluating the nonlinearity remains proportional to the rank of the reduced POD basis. As an example, consider the case of an r-mode POD-Galerkin truncation. A simple cubic nonlinearity requires that the POD-Galerkin approximation be cubed, resulting in r^3 operations to evaluate the nonlinear term. The DEIM approximates the cubic nonlinearity by using $O(r)$ discrete sample points of the nonlinearity, thus preserving a low-dimensional ($O(r)$) computation, as desired. The DEIM approach combines projection with interpolation. Specifically, the DEIM uses selected interpolation indices to specify an interpolation-based projection for a nearly optimal ℓ_2 subspace approximating the nonlinearity. The EIM/DEIM are not the only methods developed to reduce the complexity of evaluating nonlinear terms, see for instance the missing point estimation (MPE) [5], "best points" method [64], or gappy POD [37, 85, 87, 24] methods already mentioned. However, they have been successful in a large number of diverse applications and models [29]. In any case, the MPE, gappy POD, and EIM/DEIM use a small selected set of spatial grid points to avoid evaluation of the expensive inner products required to evaluate nonlinear terms (See background section on POD modeling).

The discrete sampling points given by the DEIM to evaluate the nonlinearity get a new interpretation in the current work. Specifically, in the chapter 2 of this work we show them to be the effective locations for placing sensors in the nonlinear dynamical system in order to (i) determine the dynamic regime of the system, (ii) reconstruct the current state of the system, and (iii) produce a POD-Galerkin prediction (nonlinear model reduction) of the future state of the system. Such tasks are accomplished by using ideas of sparse representation [86] and compressive sensing [33, 34, 22, 19, 23, 21, 8, 9]. In particular, the theory of compressive sensing shows that a small number of measurements are sufficient to perform a reconstruction

provided there exists a sparse representation (or basis) of the data. Sparsity techniques have also been shown to be highly effective for numerical solution schemes [79, 62]. In our case, the sparse basis is generated from a library learning procedure of a nonlinear dynamical system that exhibits low-rank dynamics. More than that, however, we also build libraries of the *nonlinearities*, thus pre-computing the low-dimensional structures observed in the different dynamical states of the nonlinear dynamical system. This allows for more robust dynamical classification as well as allowing easy evaluation of the nonlinear terms through the DEIM. The combination of library building, compressive sensing and the DEIM is demonstrated to be a highly effective and intuitively appealing methodology for scientific computing applications. It further highlights the need in modern scientific computing of nonlinear dynamical systems to integrate a variety of data-driven modeling strategies, many of which are being developed under the aegis of dimensionality reduction, in order to most efficiently simulate large-scale systems.

The various sparse sampling techniques proposed above are not optimal, but have been shown to be sufficiently robust to provide accurate reconstructions of the high-dimensional system. In the chapter 3 of this work we build upon the DEIM library learning framework, showing that nearly-optimal sparse sampling can be achieved with a short online, genetic algorithm search from the learned DEIM libraries. This improves both the classification and reconstruction accuracy of the sparse sampling. The proposed algorithm consists of two stages: (i) an offline stage that produces initial sparse sensor locations, and (ii) an online stage that uses a short, genetic search algorithm for producing nearly optimal sensor locations. The technique optimizes for both reconstruction error and classification efficacy, leading to an attractive online modification of commonly used gappy POD methods.

1.4 Physical Interpretation

The ideas presented here are more than just numerical efficiencies. Indeed, the methodology identifies the underlying modal structures that drive the dynamics of the nonlinear dynamical system, thus helping to understand the fundamental interactions and physics of

the system. Throughout the development of 20th-century physics and engineering sciences, the understanding of many canonical problems has been driven by recasting the problem into its *natural* basis (mode) set. The majority of classical problems from mathematical physics are linear Sturm-Liouville problems whose ideal modal representations are generated from eigenfunction decompositions, i.e. special functions. In quantum mechanics, for instance, Gauss-Hermite (denoted by $H_n(x)$) polynomials are the natural basis elements for understanding the harmonic oscillator. Likewise, spherical harmonics (denoted by $Y_l^m(\theta, \varphi)$) are critical in the computation of atomic orbital electron configurations as well as in representation of gravitational fields, the magnetic fields of planetary bodies and stars, and the characterization of the cosmic microwave background radiation.

For modern dynamical systems, nonlinearity plays a dominant role and shapes the underlying modes, thus necessitating a new approach, such as that presented here, for extracting these critical spatio-temporal structures. Remarkably, although nonlinearity creates new modal structures, it does not destroy the underlying low-dimensional nature of the dynamics. Distinct physical regimes may be obtained by varying bifurcation parameters, and these regimes will typically have different local bases and physical interactions. Instead of developing a global interpolated model, which may obscure these distinct physical mechanisms, we advocate a hierarchy of models along with sparse sampling and library learning to classify and characterize the system parameters from a few online measurements. Methods that take advantage of such underlying structure are critical for developing theoretical understanding and garnering insight into the fundamental interactions of the physical, engineering and biological systems under consideration.

1.5 Finite Volume Methods and Conservation Laws

The methods discussed thus far apply for the finite difference discretization of PDE models. However, many systems arising in physics are solved using finite volume discretization schemes. It is one of the most popular approaches for solving a partial differential equations governed by conservation laws. The finite volume method is extensively used in compress-

ible fluid dynamics, fluid mechanics, semiconductor device simulations and many other fields where the flux term is of great importance. It is the discretization of the integral form of the conservation laws [57]. However, if high spatial resolution is required, the method can lead to intractable computations. This issue arises in time-critical applications such as the design and optimization of flow control.

As stated above, to overcome this computational bottleneck, reduced order models (ROMs) were developed. These methods approximate the system of equations and unknowns describing the model. They first perform an offline expensive computation and build a basis to approximate the solution. The next stage is an inexpensive online stage, where the model is projected onto the subspace spanned by the previously computed basis vectors from the offline stage. The projection stage includes computations of the nonlinear terms at each step which has a high computational complexity. Along with the methods discussed in this work, several other methods have been developed to simplify these calculations. One such method, called the Gauss-Newton with approximated tensors (GNAT) [24], is a hyper-reduction scheme that proved to be successful in many problems including those arising in turbulent, compressible fluid flows. Although this method is shown to work in many scenarios, it does not necessarily preserve the structure of the model. This happens since the method does not enforce the conservation laws to be true on the full computational domain or on some large part of it. Thus even small computational errors lead to violation of this property, resulting in dissipation of key quantities. To overcome this difficulty, chapter 4 proposes a new hyper-reduction technique that will explicitly enforce the conservation laws on the full computational domain or its subdomains. This means even if the dimension of the system is highly reduced the conserved quantities will still obey the conservation laws. Exemplary results are computed for the quasi-one-dimensional Euler equation for an inviscid compressible flow of a nozzle with continuously varying cross sectional area, and shown to outperform the Galerkin, least-squares Petrov-Galerkin and GNAT results.

Chapter 2

NONLINEAR MODEL REDUCTION FOR DYNAMICAL SYSTEMS USING SPARSE SENSOR LOCATIONS FROM LEARNED LIBRARIES

2.1 Background for Model Reduction

Our innovations are built upon two key methods which are used for model reduction and approximating nonlinear dynamical systems. The first approach is the well-known POD-Galerkin method, which is used to reduce the dimension of systems in a principled way. However, computing the form of the nonlinearity in the reduced-order system is an expensive offline computation, as inner products of the full high-dimensional system must still be computed. Online evaluation of the nonlinear terms in the reduced-order model may remain expensive, as these typically involve dense matrix or tensor operations of the same order as the degree of nonlinearity. The second approach highlighted is the DEIM algorithm [29] which reduces the complexity of evaluating the nonlinear terms. In particular, it gives a principled way to sparsely sample the nonlinearity in order to approximate the nonlinear terms in a low-dimensional way.

2.1.1 POD

Consider a high-dimensional system of nonlinear differential equations that can arise, for example, from the finite-difference discretization of a partial differential equation:

$$\frac{d\mathbf{u}(t)}{dt} = L\mathbf{u}(t) + N(\mathbf{u}(t)), \tag{2.1}$$

where $\mathbf{u}(t) = [u_1(t)\ u_2(t)\ \cdots\ u_n(t)]^T \in \mathbb{R}^n$ and $n \gg 1$. Typically under discretization of a single spatial variable, $u_j(t) = u(x_j, t)$ is the value of the field of interest at the spatial

Table 2.1: DEIM algorithm for finding approximation basis for the nonlinearity and its interpolation indices.

DEIM algorithm	
Basis	
• collect data, construct snapshot matrix	$\mathbf{X} = [\mathbf{u}(t_1)\ \mathbf{u}(t_2)\ \cdots\ \mathbf{u}(t_p)]$
• construct nonlinear snapshot matrix	$\mathbf{N} = [N(\mathbf{u}(t_1))\ N(\mathbf{u}(t_2))\ \cdots\ N(\mathbf{u}(t_p))]$
• singular value decomposition of $\mathbf{N}$	$\mathbf{N} = \mathbf{\Xi}\mathbf{\Sigma}_N\mathbf{W}_N^*$
• construct approximating basis (first m columns)	$\mathbf{\Xi}_m = [\boldsymbol{\xi}_1\ \boldsymbol{\xi}_2\ \cdots\ \boldsymbol{\xi}_m]$
Interpolation Indices (Iteration Loop)	
• choose the first index (initialization)	$[\rho, \gamma_1] = \max\|\boldsymbol{\xi}_1\|$
• approximate $\boldsymbol{\xi}_j$ by $\boldsymbol{\xi}_1, ..., \boldsymbol{\xi}_{j-1}$ at indices $\gamma_1, ..., \gamma_{j-1}$	Solve for $\mathbf{c}$: $\mathbf{P}^T\boldsymbol{\xi}_j = \mathbf{P}^T\mathbf{\Xi}_{j-1}\mathbf{c}$ with $\mathbf{P} = [\mathbf{e}_{\gamma_1}\ \cdots\ \mathbf{e}_{\gamma_{j-1}}]$
• select γ_j and loop ($j = 2, 3, ..., m$)	$[\rho, \gamma_j] = \max\|\boldsymbol{\xi}_j - \mathbf{\Xi}_{j-1}\mathbf{c}\|$

location x_j. The linear part of the dynamics is given by $L \in \mathbb{R}^{n\times n}$ and the nonlinear terms are in the vector $N(\mathbf{u}(t)) = [N_1(\mathbf{u}(t))\quad N_2(\mathbf{u}(t))\quad \cdots\quad N_n(\mathbf{u}(t))]^T \in \mathbb{R}^n$. The nonlinear function is evaluated component-wise at the n spatial grid points used for discretization.

For achieving high-accuracy solutions, n is typically required to be a very large number, thus making the computation of the solution expensive and/or intractable. The POD-Galerkin method is a principled dimensionality-reduction scheme that approximates the function $\mathbf{u}(t)$ with rank-r-optimal basis functions where $r \ll n$. These optimal basis functions are computed from a singular value decomposition of a series of temporal snapshots of the nonlinear dynamical system. Specifically, suppose snapshots of the state, $\mathbf{u}(t_j)$ with $j = 1, 2, \cdots, p$, are collected. The snapshot matrix $\mathbf{X} = [\mathbf{u}(t_1)\ \mathbf{u}(t_2)\ \cdots\ \mathbf{u}(t_p)] \in \mathbb{R}^{n\times p}$ is constructed and

the SVD of $\mathbf{X}$ is computed: $\mathbf{X} = \mathbf{\Phi\Sigma W}^*$. The r-dimensional basis for optimally approximating $\mathbf{u}(t)$ is given by the first r columns of matrix $\mathbf{\Phi}$, denoted by $\mathbf{\Phi}_r$. Thus the POD-Galerkin approximation is given by

$$\mathbf{u}(t) \approx \mathbf{\Phi}_r \mathbf{a}(t) \tag{2.2}$$

where $\mathbf{a}(t) \in \mathbb{R}^r$ is the time-dependent coefficient vector and $r \ll n$. Plugging this modal expansion into the governing equation (2.1) and applying orthogonality (multiplying by $\mathbf{\Phi}_r^T$) gives the dimensionally reduced evolution

$$\frac{d\mathbf{a}(t)}{dt} = \mathbf{\Phi}_r^T L \mathbf{\Phi}_r \mathbf{a}(t) + \mathbf{\Phi}_r^T N(\mathbf{\Phi}_r \mathbf{a}(t)). \tag{2.3}$$

By solving this system of much smaller dimension, the solution of a high-dimensional nonlinear dynamical system can be approximated.

This standard POD procedure [45] has been a ubiquitous algorithm in the reduced-order modeling community. However, it also helps illustrate the need for innovations such as the DEIM, Gappy POD and/or MPE. Consider the nonlinear component of the low-dimensional evolution (2.3): $\mathbf{\Phi}_r^T N(\mathbf{\Phi}_r \mathbf{a}(t))$. For a simple nonlinearity such as $N(u(x,t)) = u(x,t)^3$, consider its impact on a spatially-discretized, two-mode POD expansion: $u(x,t) = a_1(t)\phi_1(x) + a_2(t)\phi_2(x)$. The algorithm for computing the nonlinearity would require the evaluation of

$$u(x,t)^3 = a_1^3\phi_1^3 + 3a_1^2 a_2 \phi_1^2 \phi_2 + 3a_1 a_2^2 \phi_1 \phi_2^2 + a_2^3 \phi_2^3 \,. \tag{2.4}$$

The dynamics of $a_1(t)$ and $a_2(t)$ would then be computed by projecting onto the low-dimensional basis set by taking the inner product of this nonlinear term with respect to both ϕ_1 and ϕ_2. Thus the number of computations not only doubles, but the inner products must be computed with the n-dimensional vectors. Methods such as the DEIM overcome this high-dimensional computation and instead produce an $O(r)$ dimensional evaluation of the nonlinear terms.

2.1.2 DEIM

As outlined in the previous section, the shortcomings of the POD-Galerkin method are generally due to the evaluation of the nonlinear term $N(\mathbf{\Phi}_r\mathbf{a}(t))$. To avoid this difficulty, the DEIM approximates $\mathbf{N} = N(\mathbf{\Phi}_r\mathbf{a}(t))$ through projection and interpolation instead of evaluating it directly. A considerable reduction in complexity is achieved by the DEIM because evaluating the approximate nonlinear term does not require a prolongation of the reduced state variables back to the original high dimensional state approximation required to evaluate the nonlinearity in the POD approximation. The DEIM therefore improves the efficiency of the POD approximation and achieves a complexity reduction of the nonlinear term with a complexity proportional to the number of reduced variables. The DEIM constructs these specially selected interpolation indices that specify an interpolation-based projection to provide a nearly ℓ_2 optimal subspace approximation to the nonlinear term without the expense of orthogonal projection [29].

In the DEIM, a low-rank representation of the nonlinearity, taken as snapshots in time, is computed from the singular value decomposition

$$\mathbf{N} = [N(\mathbf{u}_1)\ N(\mathbf{u}_2)\ \cdots\ N(\mathbf{u}_p)] = \mathbf{\Xi}\mathbf{\Sigma}_N\mathbf{W}_N^* \tag{2.5}$$

where the matrix $\mathbf{\Xi}$ contains the optimal (in an ℓ_2 sense) basis set for spanning the nonlinearity. Specifically, we consider the rank-m basis set $\mathbf{\Xi}_m = [\boldsymbol{\xi}_1\ \boldsymbol{\xi}_2\ \cdots\ \boldsymbol{\xi}_m]$ that approximates the nonlinear function ($m \ll n$ and $m \sim r$). The approximation to the nonlinearity $\mathbf{N}$ is given by:

$$\mathbf{N} \approx \mathbf{\Xi}_m\mathbf{c}(t) \tag{2.6}$$

where $\mathbf{c}(t)$ is similar to $\mathbf{a}(t)$ in (2.2). Since this is a highly overdetermined system, a suitable vector $\mathbf{c}(t)$ can be found by selecting only m rows of the system. The DEIM algorithm was specifically developed to identify which m rows to evaluate.

The DEIM algorithm begins by considering the vectors $\mathbf{e}_{\gamma_j} \in \mathbf{R}^n$ which are the γ_j-th column of the n dimensional identity matrix. We can then construct the projection matrix

$\mathbf{P} = [\mathbf{e}_{\gamma_1}\ \mathbf{e}_{\gamma_2}\ \cdots\ \mathbf{e}_{\gamma_m}]$ which is chosen so that $\mathbf{P}^T\boldsymbol{\Xi}_m$ is nonsingular. Then $\mathbf{c}(t)$ is uniquely defined from

$$\mathbf{P}^T\mathbf{N} = \mathbf{P}^T\boldsymbol{\Xi}_m\mathbf{c}(t), \tag{2.7}$$

and thus,

$$\mathbf{N} \approx \boldsymbol{\Xi}_m(\mathbf{P}^T\boldsymbol{\Xi}_m)^{-1}\mathbf{P}^T\mathbf{N}. \tag{2.8}$$

The tremendous advantage of this result for nonlinear model reduction is that the term $\mathbf{P}^T\mathbf{N}$ requires evaluation of nonlinearity only at m indices, where $m \ll n$. The DEIM further proposes a principled method for choosing the basis vectors $\boldsymbol{\xi}_j$ and indices γ_j. The DEIM algorithm, which is based upon a greedy-like search, is detailed in [29] and further demonstrated in Table 2.1.

2.1.3 Application to ROMs

The POD and DEIM provide a number of advantages for nonlinear model reduction of nonlinear dynamical systems. POD provides a principled way to construct an r-dimensional subspace $\boldsymbol{\Phi}_r$ characterizing the dynamics. The DEIM augments the POD method by providing a method to evaluate the problematic nonlinear terms using an m-dimensional subspace $\boldsymbol{\Xi}_m$ that represents the nonlinearity. Thus a small number of points, specifically m, can be sampled to approximate the nonlinear terms in the ROM.

The method proposed here capitalizes on these methods by building low-dimensional libraries associated with the full nonlinear system dynamics as well as the specific nonlinearities. Moreover, the sparse measurement locations computed by the DEIM are found to be highly effective for sensor placement. Such sensors, as will be shown in what follows, can be used with sparse representation and compressive sensing to (i) identify dynamical regimes, (ii) reconstruct the full state of the system, and (iii) provide an efficient nonlinear model reduction and POD-Galerkin prediction for the future state. Moreover, we show that point measurements of the nonlinearity of the dynamical system can be much more robust to noise for accomplishing the above tasks.

The concept of library building of low-rank "features" from data is well established in the computer science community. In the reduced-order modeling community, it has recently become an issue of intense investigation. Indeed, a variety of recent works, for instance from Amsallem, Charbel and co-workers [4, 30] and Peherstorfer and Willcox [70, 69, 68], have produced libraries of ROM models that can be selected and/or interpolated through measurement and classification (typically clustered with k-means type algorithms). Alternatively, cluster-based reduced order models use a k-means clustering to build a Markov transition model between dynamical states [50]. These recent innovations are similar to the ideas advocated here. However, the focus of this work is on determining nearly optimal sparse sensor locations that work across all the libraries. Further, we build two sets of libraries: one for the full dynamics and a second for the nonlinearity so as to make it computationally efficient with the DEIM strategy. Before these more formal techniques based upon machine learning were developed, it was already realized that parameter domains could be decomposed into subdomains and a local ROM/POD computed in each subdomain. Patera *et al.* [36] used a partitioning based on a binary tree whereas Amsallem *et al.* [1] used a Voronoi Tessellation of the domain. Such methods were closely related to the work of Du and Gunzburger [35] where the data snapshots were partitioned into subsets and multiple reduced bases computed. The multiple bases were then recombined into a single basis, so it doesn't lead to a library per se. For a review of these domain partitioning strategies, please see Ref. [2].

2.2 Model Problem

One of the canonical nonlinear PDEs in mathematical physics and pattern forming systems is the Ginzburg-Landau (GL) equation and its many-variants [31]. It has been used to model a variety of physical systems from condensed matter to biological waves. Here we consider a variant of the GL equation arising in mode-locked laser theory that has cubic and quintic

nonlinear terms and a fourth-order derivative [54]:

$$iU_t + \left(\frac{1}{2} - i\tau\right) U_{xx} - i\kappa U_{xxxx} + (1 - i\mu)|U|^2 U \\ +(\nu - i\varepsilon)|U|^4 U - i\gamma U = 0, \tag{2.9}$$

where $U(x,t)$ is a complex valued function of space and time. Under discretization of the spatial variable, $U(x,t)$ becomes a vector $\mathbf{u}$ with n components, i.e. $\mathbf{u}_j(t) = U(x_j,t)$ with $j = 1, 2, \cdots n$.

An efficient and exponentially accurate numerical solution to (2.9) can be found using standard spectral methods [55]. Specifically, the equation is solved by Fourier transforming in the spatial dimension and then time-stepping with an adaptive 4th-order Runge-Kutta method. The extent of the spatial domain is $x \in [-20, 20]$ with $n = 1024$ discretized points. Note that in what follows, the indices for evaluation of the nonlinear term correspond to the collocation points away from the center spatial point of the computational domain so that x_0 is the 513th point in the domain. Here, we allow the parameters $\beta = (\tau, \kappa, \mu, \nu, \epsilon, \gamma)$ to vary in order to discover various dynamical regimes that exhibit low-rank structure and stable attractors. Table 2.2 shows six different parameter regimes that have unique low-dimensional attractors (see [72]). The evolution of the system for parameter regimes β_1, β_3 and β_5 is illustrated in Fig. **??**. Such stereotypical low-dimensional behaviors, which are commonly observed in pattern forming systems [31], will serve as the basis for our library building methodology, especially in regards to using a small number of measurements to identify the β_j regime, reconstruct the solution, and project a future state. Although our results are demonstrated on this specific PDE, the methodology is quite general.

2.3 Libraries

As can be seen from Fig. **??** and Table 2.2, generic initial conditions evolve towards a variety of low-dimensional attractors. This suggests that each dynamic regime, with a given β_j, can be approximated by a small number of modes via a POD reduction. These modes will constitute our *library modes* in what follows. For each of the six regimes β_j in Table 2.2, we

	τ	κ	μ	ν	ϵ	γ	description
β_1	-0.3	-0.05	1.45	0	-0.1	-0.5	3-hump, localized
β_2	-0.3	-0.05	1.4	0	-0.1	-0.5	localized, side lobes
β_3	0.08	0	0.66	-0.1	-0.1	-0.1	breather
β_4	0.125	0	1	-0.6	-0.1	-0.1	exploding soliton
β_5	0.08	-0.05	0.6	-0.1	-0.1	-0.1	fat soliton
β_6	0.08	-0.05	0.5	-0.1	-0.1	-0.1	dissipative soliton

Table 2.2: Values of the parameters from equation (2.9) that lead to six distinct dynamical regimes. To exemplify our algorithm, the first, third and fifth regimes will be discussed in this chapter.

build a library of POD modes. The number of POD modes r is selected to capture 99% of the total variance (energy). For the β_1, β_2, β_5 and β_6 regimes, only a single mode is required so that $r = 1$. For the β_3 regime $r = 6$, whereas for the β_4 regime, $r = 14$ in order to capture the fluctuations observed. Figure 2.3(a) illustrates the library POD modes in differing colors for all of the β_j regimes except β_4. The exclusion of the β_4 modes in this visualization is simply due to the large number ($r = 14$) necessary in comparison to the other dynamical regimes. As illustrated in Fig. 2.2, library building is the first step in a training module aimed at *learning* the low-rank dynamical behavior of a nonlinear dynamical system.

In practice, a dynamical system such as (2.9) may change over time due to evolution or modulation of the parameters β_j. Thus the dynamics may evolve from one attractor to another with some prescribed transition time (typically on the order of $O(1)$ time for (2.9)). One of the primary goals of this and previous [16, 13] work is to find *optimal and sparse sensor locations* whereby limited measurements of the system are taken in order to classify the dynamical regime. Interestingly, the previous efforts [16] used expert-in-the-loop knowledge to help select the optimal measurement positions. For the simple model considered

here, such expert knowledge can be acquired from familiarity with the POD library modes and considering locations of maximal variance. However, for a more general system, this is a difficult task that could greatly benefit from a more principled mathematical approach. The DEIM algorithm will provide this approach. Moreover, as required by the DEIM, we also build low-rank libraries for the cubic and quintic terms associated with the dynamical regimes β_j. In doing so, we not only find effective sensor locations, but we also circumvent the computational difficulties of the POD in evaluating the nonlinear terms.

To library build, consider the following linear and nonlinear functions associated with the governing equations (2.9) for a given parameter regime β_j:

$$N_L(U) = U \tag{2.10a}$$

$$N_3(U) = |U|^2 U \tag{2.10b}$$

$$N_5(U) = |U|^4 U \tag{2.10c}$$

$$N_{NL}(U) = (i+\mu)|U|^2 U + (i\nu+\epsilon)|U|^4 U\,, \tag{2.10d}$$

where the second and third terms are the standard cubic and quintic nonlinearities of (2.9) and the last term enforces their prescribed relative weighting.

Associated with each nonlinearity (2.10) are a set of measurements and snapshot matrices. For a snapshot matrix sampled at p temporal locations $[\mathbf{u}_1\ \mathbf{u}_2\ \cdots\ \mathbf{u}_p] \in \mathbb{R}^{n\times p}$, we can construct the nonlinear $\mathbb{R}^{n\times p}$snapshot matrices

$$\mathbf{N}_L = [\mathbf{u}_1\ \mathbf{u}_2\ \cdots\ \mathbf{u}_p] \tag{2.11a}$$

$$\mathbf{N}_3 = [N_3(\mathbf{u}_1)\ N_3(\mathbf{u}_2)\ \cdots\ N_3(\mathbf{u}_p)] \tag{2.11b}$$

$$\mathbf{N}_5 = [N_5(\mathbf{u}_1)\ N_5(\mathbf{u}_2)\ \cdots\ N_5(\mathbf{u}_p)] \tag{2.11c}$$

$$\mathbf{N}_{NL} = [N_{NL}(\mathbf{u}_1)\ N_{NL}(\mathbf{u}_2)\ \cdots\ N_{NL}(\mathbf{u}_p)]. \tag{2.11d}$$

The singular value decomposition of these matrices will give a basis for approximation of each of the nonlinearities for a given β_j as well as the standard snapshot matrix of POD. Specifically, the SVD gives the library of modes: $\mathbf{\Phi}_{L,\beta_j}$, $\mathbf{\Phi}_{3,\beta_j}$, $\mathbf{\Phi}_{5,\beta_j}$ and $\mathbf{\Phi}_{NL,\beta_j}$ (See Fig. 2.2).

The POD modes can be arranged in a collection of library elements, $\mathbf{\Psi}_L$, $\mathbf{\Psi}_3$, $\mathbf{\Psi}_5$ or $\mathbf{\Psi}_{NL}$, by concatenating the POD modes from each of the different β_j regimes. Thus the construction of multiple libraries would take the form

$$\mathbf{\Psi}_L = [\mathbf{\Phi}_{L,\beta_1}\ \mathbf{\Phi}_{L,\beta_2}\ \cdots\ \mathbf{\Phi}_{L,\beta_6}] \tag{2.12a}$$

$$\mathbf{\Psi}_3 = [\mathbf{\Phi}_{3,\beta_1}\ \mathbf{\Phi}_{3,\beta_2}\ \cdots\ \mathbf{\Phi}_{3,\beta_6}] \tag{2.12b}$$

$$\mathbf{\Psi}_5 = [\mathbf{\Phi}_{5,\beta_1}\ \mathbf{\Phi}_{5,\beta_2}\ \cdots\ \mathbf{\Phi}_{5,\beta_6}] \tag{2.12c}$$

$$\mathbf{\Psi}_{NL} = [\mathbf{\Phi}_{NL,\beta_1}\ \mathbf{\Phi}_{NL,\beta_2}\ \cdots\ \mathbf{\Phi}_{NL,\beta_6}]\,. \tag{2.12d}$$

The number of basis elements (rank) for the cubic and quintic terms in a given POD library coincides with the rank r required for each β_j, i.e. $r = m$. Note that the library $\mathbf{\Psi}_L$ is the library containing the POD modes used for POD-Galerkin projections of the future state. It is also the only library constructed in previous work [16, 12]. Figure 2.3(b,c) shows the cubic and quintic library modes for (2.9). They can be compared to the standard POD modes illustrated in Fig. 2.3(a). Although the modes look quite similar, we will show that the classification can be improved using the libraries of nonlinearities. Further, evaluation of the nonlinearities through the DEIM now remains a low-order computation.

2.4 DEIM for sensor locations

The idea of using a limited (sparse) number of sensors to characterize the dynamics has previously been considered in [16, 12, 72]. However, no algorithm was specified to determine the best locations for the sensors, although optimal sensor placement has been investigated in the context of categorical decisions [13]. Indeed, the previous work relied on expert-in-the-loop selection of the sensors in order to classify the dynamics. Interestingly, the DEIM algorithm gives a principled way to discretely and sparsely sample the nonlinearity in order to evaluate the various inner products for a POD reduction. These same DEIM spatial sampling locations make good sensor locations for classification and reconstruction. Since the interpolation indices from the DEIM algorithm [29] correspond to the entries with largest magnitude of the residual error between the chosen basis and its approximation at each step (see last line

of the table 2.1), it becomes interesting to see what the classification/reconstruction will be if we pick these locations for sensors. As demonstrated in Fig. 2.2, determining the sensor locations is part of a training module.

We apply the DEIM algorithm outlined in Table 2.1 on the nonlinear POD (SVD) library modes ($\mathbf{\Psi}_3$, $\mathbf{\Psi}_5$ or $\mathbf{\Psi}_{NL}$) computed from (2.11) and (2.12). The application of the algorithm yields the DEIM interpolation locations which we will call our *sensor locations*. Note that the indices indicate the distance away from the center of the computational grid. Thus $x_0 = 0$, $x_{\pm 1} = dx$, $x_{\pm 2} = 2dx$, etc. Or more generally, the index n corresponds to $x_n = n\,dx$. Thus the indices depend on the specific discretization of the domain. Sensor locations are computed for each of the nonlinearities: $\mathbf{\Phi}_{3,\beta_j}$, $\mathbf{\Phi}_{5,\beta_j}$ and $\mathbf{\Phi}_{NL,\beta_j}$ for $j = 1, 2, 3$. Each dynamical regime β_j and nonlinear library gives a unique set of sensor locations. Our goal is to evaluate the placement of 3 sensors. Table 2.3 and its accompanying figure gives a vector of the indices for the locations $\mathbf{x}_{\beta_j}$ of the 3 sensors found for three regimes β_1, β_3 and β_5 using the libraries $\mathbf{\Phi}_{3,\beta_j}$, $\mathbf{\Phi}_{5,\beta_j}$ and $\mathbf{\Phi}_{NL,\beta_j}$. Also represented are the 3 sensor locations when all three β_j regimes are combined into a single library, i.e. the best sensor locations for the combined dynamic library is identified. This regime is represented in Table 2.3 by $\mathbf{x}_{\beta_{\text{all}}}$.

Application of the DEIM algorithm results in the measurement matrix $\mathbf{P}$ of (2.8). For 3 sensors, generically it takes the form

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & \cdots & & & & & \cdots & 0 \\ 0 & \cdots & 0 & 1 & 0 & \cdots & & \cdots & 0 \\ 0 & \cdots & & & \cdots & 0 & 1\ 0 & \cdots & 0 \end{bmatrix} \tag{2.13}$$

where the specific columns containing the nonzero entries are given by the indices found from the DEIM and shown in Table 2.3. More precisely, this matrix is *exactly* the output of the DEIM algorithm. In our scenario, the construction of the P matrix is made for each nonlinearity as well as for each dynamical regime β_j. This gives the sensor locations for the sparse sensing scheme presented in the next section. Figure 2.4 illustrates the locations of the sensors and the value of library modes at the prescribed locations for both the cubic and quintic nonlinearities. Figure 2.5 shows a histogram of the various sensor location

	Cubic $\lvert U\rvert^2U$				Quintic $\lvert U\rvert^4U$				Nonlinear $N(U)$			
Sensor	$\mathbf{x}_{\beta_1}$	$\mathbf{x}_{\beta_3}$	$\mathbf{x}_{\beta_5}$	$\mathbf{x}_{\beta_{all}}$	$\mathbf{x}_{\beta_1}$	$\mathbf{x}_{\beta_3}$	$\mathbf{x}_{\beta_5}$	$\mathbf{x}_{\beta_{all}}$	$\mathbf{x}_{\beta_1}$	$\mathbf{x}_{\beta_3}$	$\mathbf{x}_{\beta_5}$	$\mathbf{x}_{\beta_{all}}$
one	0	0	0	0	0	0	0	0	0	9	0	0
two	5	15	12	6	4	13	10	6	6	21	6	6
three	13	26	17	22	13	23	15	20	13	32	15	13

Table 2.3: Summary of sensor location vectors (indices for evaluation) from the DEIM algorithm. The table summarizes the findings from Fig. 2.4, giving precise grid cells to be used in evaluating the nonlinear inner products in the Galerkin-POD approximation.

evaluations. In particular, the three dominant locations are at the indices $n = 0, 6$ and 13. Thus they will be used in what follows.

2.5 Classification

Our goal is to make use of recent innovations in sparse sampling and compressive sensing [33, 34, 22, 19, 23, 21, 8, 9] for characterizing the nonlinear dynamical system [16, 12, 72]. Specifically, we wish to use a limited number of sensors for classifying the dynamical regime of the system. With this classification, a reconstruction of the full state space can be accomplished and a POD-Galerkin prediction can be computed for its future. In general, if we have a sparse measurement $\tilde{\mathbf{u}} \in \mathbf{R}^q$, where q is the number of measurements, then

$$\tilde{\mathbf{u}} = \mathbf{P}\mathbf{u}\,, \tag{2.14}$$

where $\mathbf{u}$ is the full state vector and $\mathbf{P}$ is the sampling matrix determined by the DEIM given by (2.13). In the previous section, we constructed the matrix $\mathbf{P}$ for $q = 3$.

The full state vector $\mathbf{u}$ can be approximated with the POD library modes ($\mathbf{u} = \mathbf{\Psi}_L\mathbf{c}$),

therefore

$$\tilde{\mathbf{u}} = \mathbf{P}\boldsymbol{\Psi}_L \mathbf{c}, \tag{2.15}$$

where $\boldsymbol{\Psi}_L$ is the low-rank matrix whose columns are POD basis vectors concatenated across all β regimes and $\mathbf{c}$ is the coefficient vector giving the projection of $\mathbf{u}$ onto these POD modes. If $\mathbf{P}\boldsymbol{\Psi}_L$ obeys the restricted isometry property [20] and $\mathbf{u}$ is sufficiently sparse in $\boldsymbol{\Psi}_L$, then it is possible to solve the highly-underdetermined system (2.15) with the sparsest vector $\mathbf{c}$. Mathematically, this is equivalent to the optimization problem

$$\mathbf{c} = \arg\min_{\mathbf{c}'} ||\mathbf{c}'||_0, \quad \text{subject to} \quad \tilde{\mathbf{u}} = \mathbf{P}\boldsymbol{\Psi}_L \mathbf{c}.$$

Minimizing the l_0 norm is computationally an *np*-hard problem. However, It has been proven that under certain conditions, a sparse solution of equation (2.15) can be found by minimizing the l_1 norm instead [34, 19] so that

$$\mathbf{c} = \arg\min_{\mathbf{c}'} ||\mathbf{c}'||_1, \quad \text{subject to} \quad \tilde{\mathbf{u}} = \mathbf{P}\boldsymbol{\Psi}_L \mathbf{c}. \tag{2.16}$$

The last equation can be solved through standard convex optimization methods such as the CVX package for Matlab.

To classify the dynamical regime from limited measurements $\tilde{\mathbf{u}}$ (specifically 3 spatial measurements), we use the sensor locations matrix $\mathbf{P}$ found from the DEIM on the libraries of nonlinearities. Here, the sensor locations used for $\mathbf{P}$ are from all the library elements combined and the nonlinearity $N(U)$ (See the last column in Table 2.3 remarked with red boxes), i.e. $n = 0, 6$ and 13. Suppose we have a linear measurement $\tilde{\mathbf{u}}$, then we can construct the vectors $\tilde{\mathbf{u}}_3 = |\tilde{\mathbf{u}}|^2\tilde{\mathbf{u}}$ and $\tilde{\mathbf{u}}_5 = |\tilde{\mathbf{u}}|^4\tilde{\mathbf{u}}$ and classify them using the libraries of nonlinearities. Specifically, the nonlinear classification is accomplished with:

$$\mathbf{c}_3 = \arg\min_{\mathbf{c}_3'} ||\mathbf{c}_3'||_1, \quad \text{subject to} \quad \tilde{\mathbf{u}}_3 = \mathbf{P}\boldsymbol{\Psi}_3 \mathbf{c}_3 \tag{2.17a}$$

$$\mathbf{c}_5 = \arg\min_{\mathbf{c}_5'} ||\mathbf{c}_5'||_1, \quad \text{subject to} \quad \tilde{\mathbf{u}}_5 = \mathbf{P}\boldsymbol{\Psi}_5 \mathbf{c}_5\,. \tag{2.17b}$$

Figures 2.6 and 2.7 show the coefficient vectors $\mathbf{c}_3$ and $\mathbf{c}_5$ respectively for measurements performed in the β_1, β_3 and β_5 regimes. The vectors $\mathbf{c}_3$ and $\mathbf{c}_5$ clearly act as accurate indicator

functions for the dynamical regime, better than even simple linear measurements [16, 12]. Indeed, the DEIM algorithm for sensor location does as well as expert-in-the-loop selections [16, 12, 72], but requires no extensive and pre-existing knowledge about the dynamical libraries. We can also make a categorical decision, with similar results, about the dynamical regime the dynamics belongs to by computing error of projection onto a given library and considering which has the smallest error. This is the same as sparse representation used for image classification [86].

The above analysis assumes that there is no noise in the measurements or the system itself. However, most sensors are subject to noise fluctuations which can impact the ability of a scheme such as this to correctly identify β_j. As a consequence, we also perform the classification task with noisy data. First, assume that we collect linear measurements which have additive noise. Denote this data by

$$\bar{\mathbf{u}} = \tilde{\mathbf{u}} + \mathcal{N}(0, \sigma^2) \tag{2.18}$$

where $\mathcal{N}(0, \sigma^2)$ is a Gaussian distributed noise term with variance σ^2 .

In order to evaluate the classification, we need to once again compute the nonlinear terms and run the optimization algorithm for computing the library coefficients and the associated dynamical regime. The statistical result for 400 trials when $\sigma = 0.2$ is shown in Table 2.4. One can see that the noise introduces misclassification errors to the original 100%-accurate classification scheme. However, multiple measurements still give an accurate classification overall with the exception of using the quintic library in the β_3 regime.

Interestingly, if point measurements of the nonlinearity are considered, then the results can improve drastically. For instance, in optics, measurements (full field or point measurements) are made of the intensity of the field rather than the field itself. This represents a simple form of a nonlinear measurement. Thus consider the nonlinear point (spatial) measurements subject to noise:

$$\bar{\mathbf{u}}_3 = |\tilde{\mathbf{u}}|^2\tilde{\mathbf{u}} + \mathcal{N}(0, \sigma^2) \tag{2.19a}$$

$$\bar{\mathbf{u}}_5 = |\tilde{\mathbf{u}}|^4\tilde{\mathbf{u}} + \mathcal{N}(0, \sigma^2)\,. \tag{2.19b}$$

The classification results for this case are also shown in Table 2.4. Note the clear improvement (100% accuracy) in using point measurements of the nonlinearity for classification tasks. Thus if the noise is driven by the sensor itself, then point measurements of the nonlinearity may be quite advantageous.

2.6 Reconstruction and the Galerkin-POD Approximation

The classification step of the last section identifies the dynamical regime of the nonlinear dynamical system by using sparsity promoting ℓ_1 optimization on the learned libraries. Once the correct β_j regime is determined, reconstruction of the solution and a future state prediction can be achieved through the POD-Galerkin approximation. Specifically, once the dynamical regime β_j has been identified, then a subset of modes $\mathbf{\Psi}_L \rightarrow \mathbf{\Phi}_{L,\beta_j}$ form the correct modal basis for a POD-Galerkin approximation.

To be more precise, recall that only a limited number of measurements are made as in (2.14). But now $\mathbf{u} = \mathbf{\Phi}_{L,\beta_j}\mathbf{c}$ where the vector $\mathbf{c}$ is now the projection onto the smaller set of library modes associated with a single β_j. Thus instead of (2.15), we now we have

$$\tilde{\mathbf{u}} = \mathbf{P}\mathbf{\Phi}_{L,\beta_j}\mathbf{c}\,. \tag{2.20}$$

Unlike the classification step, we can now determine $\mathbf{c}$ by simply solving the above equation using a standard Moore-Penrose pseudo-inverse operator $\dagger$ [82] so that $\mathbf{c} = (\mathbf{P}\mathbf{\Phi}_{L,\beta_j})^\dagger\tilde{\mathbf{u}}$, i.e. it solves for $\mathbf{c}$ by minimizing the ℓ_2 norm. With $\mathbf{c}$ determined, the reconstruction of the solution thus follows:

$$\mathbf{u} = \mathbf{\Phi}_{L,\beta_j}(\mathbf{P}\mathbf{\Phi}_{L,\beta_j})^\dagger\tilde{\mathbf{u}} \tag{2.21}$$

This is the reconstruction of the system given the sparse measurement vector $\tilde{\mathbf{u}}$ and a classification β_j. The POD-Galerkin approximation for the future state can then be accomplished by using (2.3) and with the DEIM algorithm for evaluating the nonlinearities (2.8). The initial condition for the POD-Galerkin is given from (2.21). Thus as advocated in previous work [16, 12], accurate classification is accomplished with ℓ_1 optimization (decoding) while

the more standard ℓ_2 norm is used for reconstruction and POD-Galerkin projection (encoding). Figure 2.2 illustrates the execution state outlined here for classification, reconstruction and projection.

The computational efficiency of the proposed method can be evaluated. Of course, there is no computational savings in the training stage of the algorithm. Indeed, there is some computational overhead associated with building the libraries (an SVD evaluation of $O(N^3)$) and running the DEIM algorithm (of $O(N)$). However, once the training stage is done, then the compressive sensing, which is a small ℓ_1 optimization procedure can be used to identify the correct POD modes and project into the future with a Galerkin-POD approximation. In the case of the CQGLE considered here, this provides a computational savings of three orders of magnitude, both in computational time and memory requirements. This highlights the potentially transformative use of reduced-order models in computational physics and for simulations of multi-scale nonlinear dynamical systems.

2.7 Conclusions and Outlook

In conclusion, we advocate a general theoretical framework for nonlinear dynamical systems whereby low-rank libraries representing the optimal modal basis are constructed, or learned, from snapshot sampling of the dynamics. In order to make model reduction methods such as POD computationally efficient, especially in evaluating the nonlinear terms of the governing equations, libraries of nonlinearities are also constructed during the learning stage. This allows for the application of the discrete empirical interpolation method which identifies a limited number of spatial sampling locations that can allow for reconstruction of the nonlinear terms in a low-dimensional manner. Such sparse sampling of the nonlinearity is directly related to compressive sensing strategies whereby a small number of sensors can be used to characterize the dynamics of the nonlinear system. Indeed, the POD method, when combined with the DEIM and compressive sensing, can (i) correctly identifying the dynamical parameter regime, (ii) reconstruct the full state dynamics and (iii) produce a low-rank prediction of the future state of the nonlinear dynamical system. All of these tasks

are accomplished in a low-dimensional way, unlike standard POD-Galerkin models whose nonlinearities can prove to be computationally inefficient.

To be more precise about our learning algorithm for the nonlinear dynamical system, we construct the library modes representing the dynamics by the ℓ_2-optimal proper orthogonal decomposition. Several libraries are constructed: one for linear snapshot measurements, one for each nonlinear term, and one which combines all the nonlinear terms together with their prescribed weightings. The DEIM algorithm then allows us to identify sparse measurement locations capable of both classifying the dynamics regime of the nonlinear dynamical system and efficiently evaluating the inner products of the nonlinear terms for a POD-Galerkin projection of the system. Indeed, the dynamical state is identified from limited noisy measurements using the sparsity promoting ℓ_1 norm and the compressive sensing architecture. The strategy for building modal libraries by concatenating truncated POD libraries across a range of relevant bifurcation parameters may be viewed as a simple dimensionality-reduction implementation of machine learning. The resulting modal libraries are a natural sparse basis for the application of compressive sensing. After the expensive one-time library-building procedure, accurate identification, projection, and reconstruction may be performed entirely in a low-dimensional framework.

The method is effective for nonlinear dynamical systems where POD approximations are relevant. Thus it can be applied only to systems where low-dimensional attractors are the key dynamical features of interest. Specifically, it fails if such attractors are not present, if wave propagation is the dominant dynamical feature (although modifications exist to account for this), and/or intrinsically high-dimensional systems such as turbulent flows. More broadly, the methodology can also be extended to complex systems that exhibit low-dimensional attractors in their repertoire of dynamical behaviors. As an example, one need only consider encoding schemes in neuro-sensory systems whereby the collective behavior of networked neurons produce emergent low-dimensional patterns of activity for given stimulus [81, 53].

With three DEIM determined sensor locations, it is possible to accurately classify bifurcation regimes, reconstruct the low-dimensional content, and simulate the Galerkin projected

dynamics of the complex Ginzburg Landau equation. In addition, we investigate the performance of sparse representation with the addition of sensor noise. For moderate noise levels, the method accurately classifies the correct dynamic regime. Point measurements of the nonlinearity dramatically improve the classification procedure. Interestingly, the DIEMs algorithm not only provides efficient sensor positioning, it also helps perform POD-Galerking truncations in a fully low-rank manner, thus avoiding the computational expense of evaluating nonlinear terms using the POD methodology. Overall, the combination of ℓ_2 low-rank representations and ℓ_1 sparse sampling enables efficient characterization and manipulation of low-rank dynamical systems.

For modern nonlinear dynamical systems, it is known that nonlinearity plays a dominant role and shapes the underlying spatio-temporal dynamics and modal structures, thus necessitating a new approach, such as that presented here, for extracting these critical structures. As has been demonstrated, although nonlinearity drives new modal structures, it does not destroy the underlying low-dimensional nature of the dynamics. Methods that take advantage of such underlying structure are critical for developing theoretical understanding and garnering insight into the fundamental interactions of a vast array of physical, engineering and biological systems.

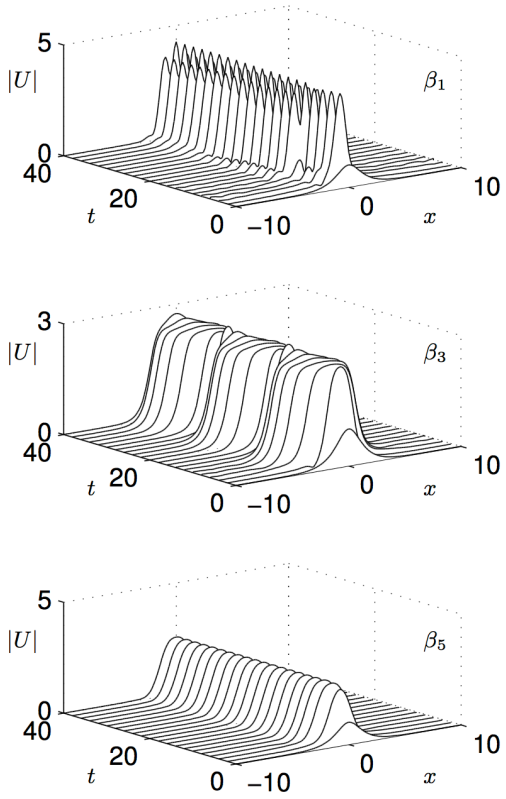


Figure 2.1: Evolution dynamics of (2.9) for the parameter regimes β_1, β_3 and β_5 over the time interval $t \in [0, 40]$. The initial transients are quickly attenuated away, leaving the stable attractor for the given β_j regime. Sampling of the dynamics for library building occurs once the transients have decayed.

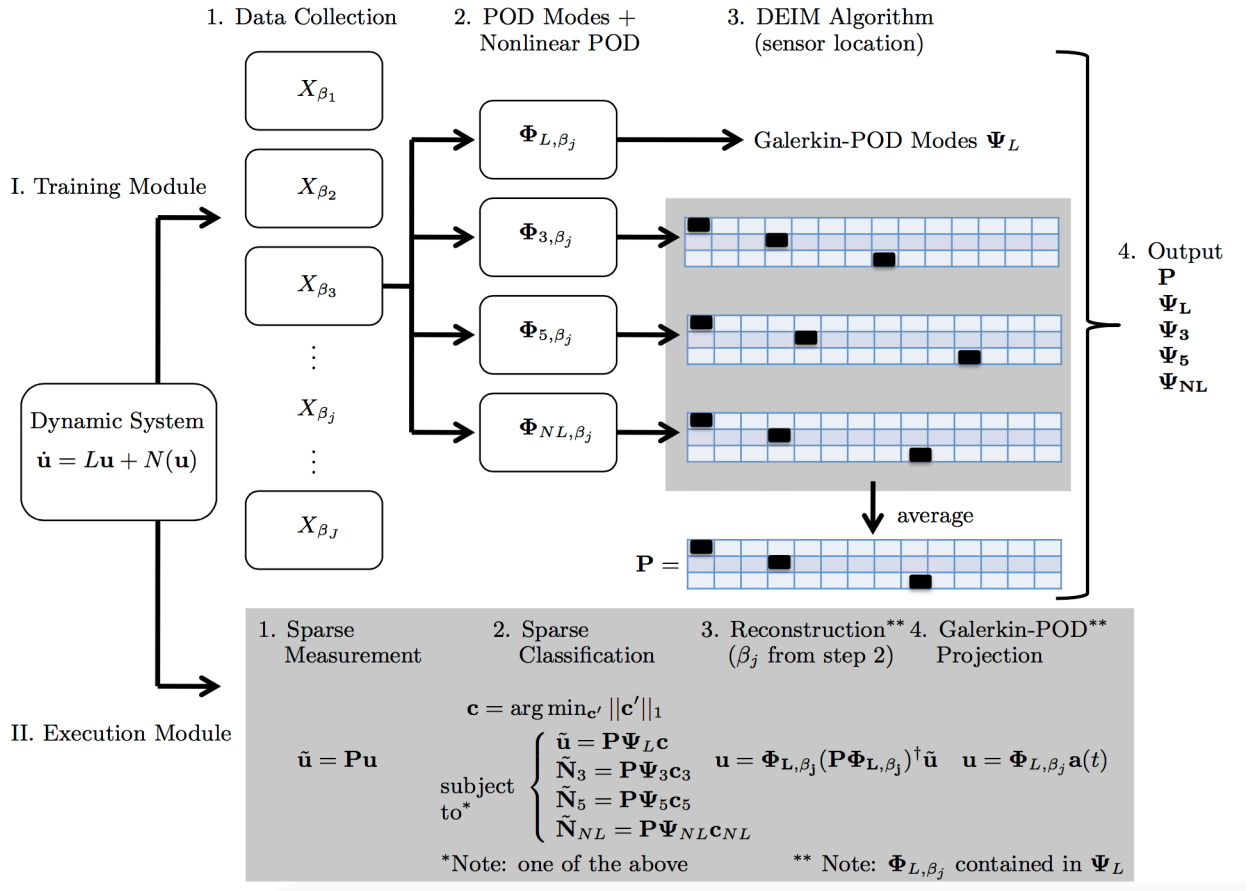


Figure 2.2: Training and execution modules for the library learning and sensor location optimization with the DEIM. The training module samples the various dynamical regimes $(\beta_1, \beta_2, \cdots, \beta_J)$ through snapshots. For each dynamical regime, low-rank libraries are constructed for the nonlinearities of the nonlinear dynamical system $(\mathbf{\Phi}_{L,\beta_j}, \mathbf{\Phi}_{3,\beta_j}, \mathbf{\Phi}_{5,\beta_j}, \mathbf{\Phi}_{NL,\beta_j})$. The DEIM algorithm is then used to select sparse sampling locations and construct the projection matrix $\mathbf{P}$. The execution module uses the sampling locations to classify the dynamical regime β_j of the nonlinear dynamical system, reconstruct its full state $(\mathbf{u} = \mathbf{\Phi}_{\mathbf{L},\beta_\mathbf{j}}(\mathbf{P}\mathbf{\Phi}_{\mathbf{L},\beta_\mathbf{j}})^\dagger\tilde{\mathbf{u}})$, and provide a low-rank Galerkin-POD approximation for its future $(\mathbf{u} = \mathbf{\Phi}_{L,\beta_j}\mathbf{a}(t))$. Note that $(\mathbf{P}\mathbf{\Phi}_{\mathbf{L},\beta_\mathbf{j}})^\dagger$ denotes the Moore-Penrose pseudo-inverse of $(\mathbf{P}\mathbf{\Phi}_{\mathbf{L},\beta_\mathbf{j}})$.

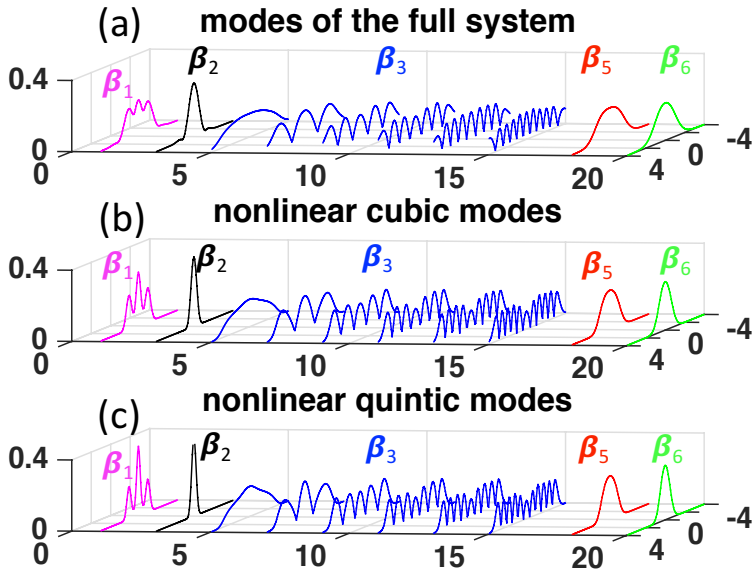


Figure 2.3: Library modes for (a) the full system, (b) the cubic nonlinearity, and (c) the quintic nonlinearity. The modes are color coded by their dynamical regime from β_1 to β_6 as given in Table 2.2. The rank-r for each library is chosen by selecting the modes that comprise 99% of the total variance for a given dynamical regime.

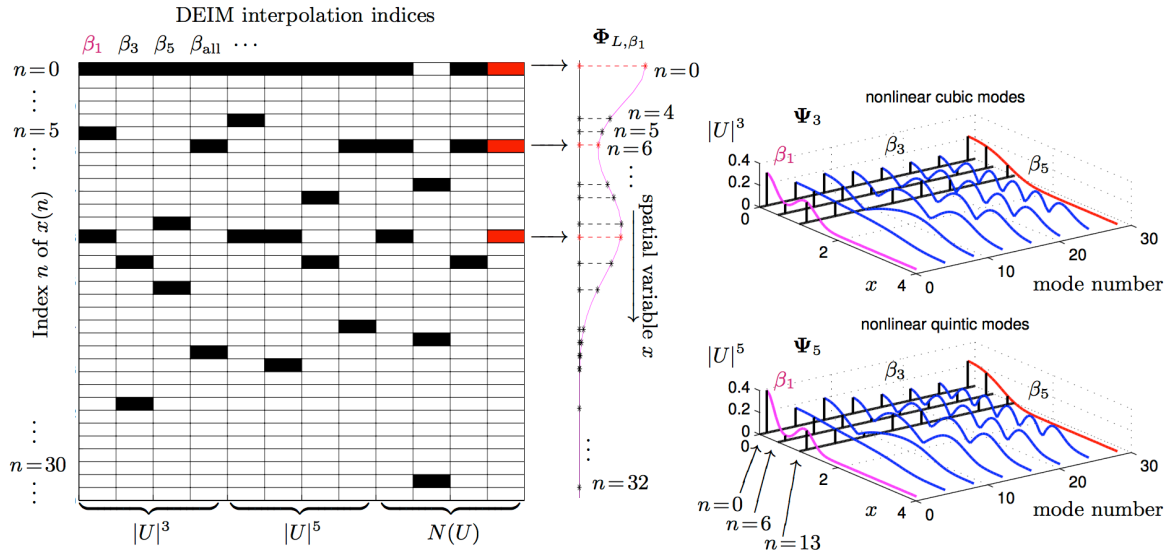


Figure 2.4: Location of indices determined by the DEIM for the libraries of nonlinearities $|U|^3$, $|U|^5$ and $N(U)$. The spatial domain $x \in [-20, 20]$ is discretized on a periodic domain with $n = 1024$ points. The center point of the domain corresponds to $x(0) = 0$. The index values are the number of grid points ndx away from the center grid point, e.g. $x(5) = 5dx$. The left grid shows the location of the DEIM indices (black boxes) determined by the algorithm in Table 2.1 for the regimes β_1, β_3 and β_5 as well as the combination of all three regimes together β_{all}. The middle panel shows the library mode Φ_{L,β_1} (laid out vertically) as a function of the spatial variable $x(n)$. Indicated on this transverse mode are the measurement locations for the different DEIM nonlinearities and β_j regimes. The right two panels show the β_1, β_3 and β_5 modes with the black lines indicating the measurement locations for $n = 0, 6$ and 13. This allows one to visualize where the measurement occur on the mode structures.

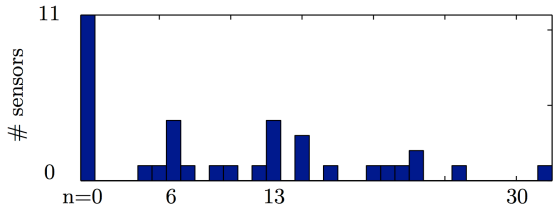


Figure 2.5: Histogram of sensor placement locations based upon Fig. 2.4. The top three sensor location indices are located at $n = 0, 6$ and 13. Thus we use these locations in our simulations for classification and reconstruction.

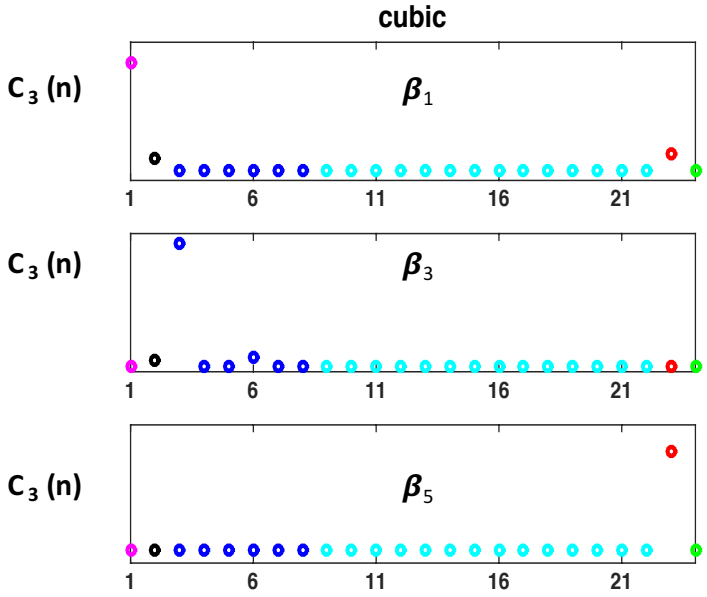


Figure 2.6: The values of the 24×1 projection vector $\mathbf{c}$ from solving using a cubic measurement $\tilde{\mathbf{u}}_3 = |\tilde{\mathbf{u}}|^2\tilde{\mathbf{u}}$ and the cubic library $\mathbf{\Psi}_3$ in (2.17a). The three panels show the dominant vector component to be in the β_1, β_3 and β_5 regime respectively, thus showing that it correctly identifies each dynamical regime from 3 measurement locations. The values of the colored circles correspond to the expression strength of the different library elements of Fig. 2.3.

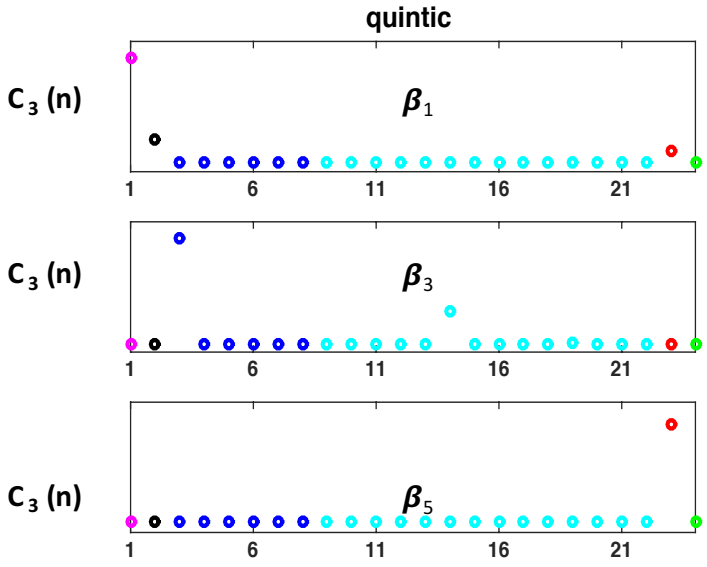


Figure 2.7: The values of the 24×1 projection vector $\mathbf{c}$ from solving using a quintic measurement $\tilde{\mathbf{u}}_5 = |\tilde{\mathbf{u}}|^4\tilde{\mathbf{u}}$ and the quintic library $\mathbf{\Psi}_5$ in (2.17b). The three panels show the dominant vector component to be in the β_1, β_3 and β_5 regime respectively, thus showing again that point measurements of the nonlinearity correctly identify each dynamical regime from 3 measurement locations. The values of the colored circles correspond to the expression strength of the different library elements of Fig. 2.3.

β_1 regime	β_1	β_2	β_3	β_4	β_5	β_6
$\lvert\bar{u}\rvert^2\bar{u}$	**98.75**	0	1.25	0	0	0
$\lvert\bar{u}\rvert^4\bar{u}$	**91**	6.5	2.5	0	0	0
$\bar{u}_3$	**100**	0	0	0	0	0
$\bar{u}_5$	**100**	0	0	0	0	0
β_3 regime	β_1	β_2	β_3	β_4	β_5	β_6
$\lvert\bar{u}\rvert^2\bar{u}$	2.5	0	**61.75**	18	17.5	0.25
$\lvert\bar{u}\rvert^4\bar{u}$	5.5	0	**38**	34.5	21.75	0.25
$\bar{u}_3$	0	0	**100**	0	0	0
$\bar{u}_5$	0	0	**100**	0	0	0
β_5 regime	β_1	β_2	β_3	β_4	β_5	β_6
$\lvert\bar{u}\rvert^2\bar{u}$	5.25	0.75	7.5	5	**62**	19.5
$\lvert\bar{u}\rvert^4\bar{u}$	6.75	2	6.25	2.5	**61.25**	21.25
$\bar{u}_3$	0	0	0	0	**100**	0
$\bar{u}_5$	0	0	0	0	**100**	0

Table 2.4: Classification accuracy with noisy measurements ($\sigma = 0.2$) using 400 realizations in the β_1, β_3 and β_5 regimes. The accuracy of classification for the correct regime is denoted by the bold numbers, whereas the other percentages denote to what extent and where misclassifications occur. The accuracy of the classification schemes are evaluated using linear measurements ($\bar{u}$ in (2.18)) with the cubic and quintic libraries illustrated in Figs. 2.6 and 2.7. Also shown are classification results using point measurements of the nonlinearity ($\bar{\mathbf{u}}_3$ and $\bar{\mathbf{u}}_5$ in 2.19). Point measurements of the nonlinearity, if possible, offer significant accuracy improvement and robustness to noise.

Chapter 3

ONLINE INTERPOLATION POINT REFINEMENT FOR REDUCED ORDER MODELS USING A GENETIC ALGORITHM

3.1 Reduced Order Modeling

In our analysis, we consider a parametrized, high-dimensional system of nonlinear differential equations that arises, for example, from the finite-difference discretization of a partial differential equation. In the formulation proposed, the linear and nonlinear terms for the state vector $\mathbf{u}(t)$ are separated:

$$\frac{d\mathbf{u}(t)}{dt} = L\mathbf{u}(t) + N(\mathbf{u}(t), \mu), \tag{3.1}$$

where $\mathbf{u}(t) = [u_1(t)\ u_2(t)\ \cdots\ u_n(t)]^T \in \mathbb{R}^n$ and $n \gg 1$. Typically, $u_j(t) = u(x_j, t)$ is the value of the field of interest discretized at the spatial location x_j. The linear part of the dynamics is given by the linear operator $L \in \mathbb{R}^{n\times n}$ and the nonlinear terms are in the vector $N(\mathbf{u}(t)) = [N_1(\mathbf{u}(t), \mu) \quad N_2(\mathbf{u}(t), \mu) \quad \cdots \quad N_n(\mathbf{u}(t), \mu)]^T \in \mathbb{R}^n$. The nonlinear function is evaluated component-wise at the n spatial grid points used for discretization. Note that we have assumed, without loss of generality, that the parametric dependence μ is in the nonlinear term.

Typical discretization schemes for achieving a prescribed spatial resolution and accuracy require the number of discretization points n to be very large, resulting in a high-dimensional state vector. For sufficiently complicated problems where significant spatial refinement is required and/or higher spatial dimension problems (2D or 3D computations, for instance) can potentially lead to a computationally intractable problem where ROMs are necessary. The POD-Galerkin method [45, 11] is a principled dimensionality-reduction scheme that approximates the function $\mathbf{u}(t)$ with rank-r-optimal basis functions where $r \ll n$. These

optimal basis functions are computed from a singular value decomposition of a time series of snapshots of the nonlinear dynamical system (3.1). Given the snapshots of the state variable $\mathbf{u}(t_j)$ at $j = 1, 2, \cdots, p$ times, the snapshot matrix $\mathbf{X} = [\mathbf{u}(t_1)\ \mathbf{u}(t_2)\ \cdots\ \mathbf{u}(t_p)] \in \mathbb{R}^{n\times p}$ is constructed and the SVD of $\mathbf{X}$ is computed

$$\mathbf{X} = \mathbf{\Phi}_r \mathbf{\Sigma} \mathbf{W}^* . \tag{3.2}$$

The r-dimensional basis for optimally approximating $\mathbf{u}(t)$ is given by the first r columns of matrix $\mathbf{\Phi}$, denoted by $\mathbf{\Phi}_r$. The POD-Galerkin approximation is then computed from the following decomposition

$$\mathbf{u}(t) \approx \mathbf{\Phi}_r \mathbf{a}(t) \tag{3.3}$$

where $\mathbf{a}(t) \in \mathbb{R}^r$ is the time-dependent coefficient vector and $r \ll n$. Note that such an expansion implicitly assumes a separation of variables between time and space. Plugging this modal expansion into the governing equation (3.1) and applying orthogonality (multiplying by $\mathbf{\Phi}_r^T$) gives the dimensionally reduced evolution

$$\frac{d\mathbf{a}(t)}{dt} = \mathbf{\Phi}_r^T L \mathbf{\Phi}_r \mathbf{a}(t) + \mathbf{\Phi}_r^T N(\mathbf{\Phi}_r \mathbf{a}(t), \mu) \tag{3.4}$$

where it is noted that $\mathbf{\Phi}_r^T \mathbf{\Phi}_r = \mathbf{I}$, the $r \times r$ identity matrix.

By solving this system of much smaller dimension, the solution of a high-dimensional nonlinear dynamical system can be approximated with optimal (in an ℓ_2 sense) basis functions. Of critical importance is evaluating the nonlinear terms in an efficient way using the gappy POD or DEIM mathematical architecture. Otherwise, the evaluation of the nonlinear terms still requires calculation of functions and inner products with the original dimension n. In certain cases, such as for the quadratic nonlinearity in the Navier-Stokes equations, the nonlinear terms can be computed once in an offline manner. However, parametrized systems generally require repeated evaluation of the nonlinear terms as the POD modes change with μ. Equation (3.4) provides the starting point of the ROM architecture. It also illustrates one of the two critical innovations of ROMs: a principled dimensionality reduction.

3.2 Gappy Sampling and Discrete Empirical Interpolation Method

In and of itself, the dimensionality reduction provided by the POD method does not ensure that the reduced model (3.4) remains low-dimensional. This is simply due to the evaluation of the nonlinear term $\mathbf{\Phi}_r^T N(\mathbf{\Phi}_r \mathbf{a}(t), \mu)$. Specifically, the nonlinearity forces the evaluation of the nonlinear function and inner products at every time step when advancing (3.4) with a time-stepping scheme. Such inner products require $O(n)$ computation, keeping the ROM fixed in the higher dimensional space. In contrast, the linear term $\mathbf{\Phi}_r^T L \mathbf{\Phi}_r$ can be computed once, either at the beginning of the simulation, or in an offline stage. This product of matrices yields a matrix of size $r \times r$, thus requiring only $O(r)$ computations to update the linear portion of the evolution.

To avoid the costly computations associated with approximating $\mathbf{N} = N(\mathbf{\Phi}_r \mathbf{a}(t))$, we compute an approximation to the nonlinearity through projection and interpolation instead of evaluating it directly. A considerable reduction in complexity is achieved by the DEIM, for instance, because evaluating the approximate nonlinear term does not require a prolongation of the reduced state variables back to the original high dimensional state approximation required to evaluate the nonlinearity in the POD approximation. The DEIM therefore improves the efficiency of the POD approximation and achieves a complexity reduction of the nonlinear term with a complexity proportional to the number of reduced variables. The DEIM constructs these specially selected interpolation indices that specify an interpolation-based projection to provide a nearly ℓ_2 optimal subspace approximation to the nonlinear term without the expense of orthogonal projection [29].

3.2.1 Sparse (Gappy) Sampling

To better understand how the computation of the nonlinearity through projection and interpolation, we consider taking m measurements ($m > r$ with $O(m) \sim O(r)$) of the full state vector $\mathbf{u}$. Specifically, only $m \ll n$ measurements are required for reconstruction, allowing

us to define the sparse representation variable $\tilde{\mathbf{u}} \in \mathbb{R}^m$

$$\tilde{\mathbf{u}} = \mathbf{P}\mathbf{u} \tag{3.5}$$

where the measurement matrix $\mathbf{P} \in \mathbb{R}^{m \times n}$ specifies m measurement locations of the full state $\mathbf{u} \in \mathbb{R}^n$. As an example, the measurement matrix might take the form

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & \cdots & & & & & \cdots & 0 \\ 0 & \cdots & 0 & 1 & 0 & \cdots & & \cdots & 0 \\ 0 & \cdots & & & \cdots & 0 & 1 & 0 \cdots & 0 \\ \vdots & 0 & & & \cdots & 0 & 0 & 1 \cdots & \vdots \\ 0 & \cdots & & & \cdots & 0 & 0 & 0 \cdots & 1 \end{bmatrix} \tag{3.6}$$

where measurement locations take on the value of unity and the matrix elements are zero elsewhere. The matrix $\mathbf{P}$ defines a projection onto an m-dimensional measurement space $\tilde{\mathbf{u}}$ that is used to approximate $\mathbf{u}$.

The sparse sampling of (3.5) forms the basis of the *Gappy POD* method introduced by Everson and Sirovich [37]. In their example of eigenface reconstruction, they used a small number of measurements, or gappy data, to reconstruct an entire face. This serves as the basis for approximating the nonlinear inner products in (3.4) and overcoming the computational complexity of the POD reduction. The measurement matrix $\mathbf{P}$ allows for an approximation of the state vector $\mathbf{u}$ from m measurements. The approximation is given by substituting (3.3) into (3.5):

$$\tilde{\mathbf{u}} \approx \mathbf{P} \sum_{j=1}^{m} \tilde{a}_j \phi_j \tag{3.7}$$

where the coefficients $\tilde{a}_j$ minimize the error in approximation

$$\|\tilde{\mathbf{u}} - \mathbf{P}\mathbf{u}\|. \tag{3.8}$$

The goal is to determine the $\tilde{a}_j$ despite the fact that taking inner products of (3.7) can no longer be performed. Specifically, the vector $\tilde{\mathbf{u}}$ is of length m whereas the POD modes are of length n. Implied in this sparse sampling is that the modes $\phi_j(x)$ are in general not

orthogonal over the m-dimensional support of $\tilde{\mathbf{u}}$, which is denoted as $s[\tilde{\mathbf{u}}]$. Specifically, the following two relationships hold

$$M_{jk} = (\phi_j, \phi_k) = \delta_{jk} \tag{3.9a}$$

$$M_{jk} = (\phi_j, \phi_k)_{s[\tilde{\mathbf{u}}]} \neq 0 \quad \text{for all } j, k \tag{3.9b}$$

where M_{jk} are the entries of the Hermitian matrix $\mathbf{M}$ and δ_{jk} is the Kronecker delta function. The fact that the POD modes are not orthogonal on the support $s[\tilde{\mathbf{u}}]$ leads us to consider alternatives for evaluating the vector $\tilde{\mathbf{a}}$.

As demonstrated by Everson and Sirovich [37], the $\tilde{a}_j$ is determined by a least-square fit error algorithm. Specifically, we project the full state vector $\mathbf{u}$ onto the support space and determine the vector $\tilde{\mathbf{a}}$ with the equation:

$$\mathbf{M}\tilde{\mathbf{a}} = \mathbf{f} \tag{3.10}$$

where the elements of the matrix $\mathbf{M}$ are given by (3.9b) and the components of the vector f_k are given by $f_j = (\mathbf{u}, \psi_j)_{s[\tilde{\mathbf{u}}]}$. The pseudo-inverse for determining $\tilde{\mathbf{a}}$ is a least-square fit algorithm. Note that in the event the measurement space is sufficiently dense, or as the support space is the entire space, then $\mathbf{M} = \mathbf{I}$ and $\tilde{\mathbf{a}} \to \mathbf{a}$, thus implying the eigenvalues of $\mathbf{M}$ approach unity as the number of measurements become dense. Once the vector $\tilde{\mathbf{a}}$ is determined, then a reconstruction of the solution can be performed using

$$\mathbf{u}(x, t) \approx \mathbf{\Phi}\tilde{\mathbf{a}}\,. \tag{3.11}$$

It only remains to consider the efficacy of the measurement matrix $\mathbf{P}$. Originally, random measurements were proposed [37]. However, the ROMs community quickly developed principled techniques based upon, for example, minimization of the condition number of $\mathbf{M}$ [85], selection of minima or maxima of POD modes [87], and/or greedy algorithms of EIM/DEIM [10, 29]. Thus m measurement locations were judiciously chosen for the task of accurately interpolating the nonlinear terms in the ROM. This type of sparsity has been commonly used throughout the ROMs community.

3.2.2 DEIM Algorithm

The DEIM algorithm constructs two low-rank spaces through the SVD: one for the full system using the snapshots matrix $\mathbf{X}$, and a second using the snapshot matrix composed of samples of the nonlinearity alone. Thus a low-rank representation of the nonlinearity is given by

$$\mathbf{N} = [N(\mathbf{u}_1)\ N(\mathbf{u}_2)\ \cdots\ N(\mathbf{u}_p)] = \mathbf{\Xi}\mathbf{\Sigma}_N\mathbf{W}_N^* \tag{3.12}$$

where the matrix $\mathbf{\Xi}$ contains the optimal (in an ℓ_2 sense) basis set for spanning the nonlinearity. Specifically, we consider the rank-m basis set $\mathbf{\Xi}_m = [\boldsymbol{\xi}_1\ \boldsymbol{\xi}_2\ \cdots\ \boldsymbol{\xi}_m]$ that approximates the nonlinear function ($m \ll n$ and $m \sim r$). The approximation to the nonlinearity $\mathbf{N}$ in this SVD basis set is given by:

$$\mathbf{N} \approx \mathbf{\Xi}_m\mathbf{c}(t) \tag{3.13}$$

where $\mathbf{c}(t)$ is similar to $\mathbf{a}(t)$ in (3.3). Since this is a highly overdetermined system, a suitable vector $\mathbf{c}(t)$ can be found by selecting only m rows of the system. The DEIM algorithm provides a greedy search algorithm for selecting an appropriate m rows. Although not guaranteed to be optimal, in practice the row selection tends to provide sampling points that are accurate for reconstruction of the full state.

The DEIM algorithm begins by considering the vectors $\mathbf{e}_{\gamma_j} \in \mathbf{R}^n$ which are the γ_j-th column of the n dimensional identity matrix. We can then construct the projection matrix $\mathbf{P} = [\mathbf{e}_{\gamma_1}\ \mathbf{e}_{\gamma_2}\ \cdots\ \mathbf{e}_{\gamma_m}]$ which is chosen so that $\mathbf{P}^T\mathbf{\Xi}_m$ is nonsingular. Then $\mathbf{c}(t)$ is uniquely defined from

$$\mathbf{P}^T\mathbf{N} = \mathbf{P}^T\mathbf{\Xi}_m\mathbf{c}(t), \tag{3.14}$$

and thus,

$$\mathbf{N} \approx \mathbf{\Xi}_m(\mathbf{P}^T\mathbf{\Xi}_m)^{-1}\mathbf{P}^T\mathbf{N}. \tag{3.15}$$

The tremendous advantage of this result for nonlinear model reduction is that the term $\mathbf{P}^T\mathbf{N}$ requires evaluation of nonlinearity only at m indices, where $m \ll n$. The DEIM further proposes a principled method for choosing the basis vectors $\boldsymbol{\xi}_j$ and indices γ_j. The

Table 3.1: DEIM algorithm for finding approximation basis for the nonlinearity and interpolation indices.

DEIM algorithm	
Basis	
• construct snapshot matrix	$\mathbf{X} = [\mathbf{u}(t_1)\ \mathbf{u}(t_2)\ \cdots\ \mathbf{u}(t_p)]$
• construct nonlinear snapshot matrix	$\mathbf{N} = [N(\mathbf{u}(t_1))\ N(\mathbf{u}(t_2))\ \cdots\ N(\mathbf{u}(t_p))]$
• singular value decomposition of $\mathbf{N}$	$\mathbf{N} = \boldsymbol{\Xi}\boldsymbol{\Sigma}_N\mathbf{W}_N^*$
• rank-m approximating basis	$\boldsymbol{\Xi}_m = [\boldsymbol{\xi}_1\ \boldsymbol{\xi}_2\ \cdots\ \boldsymbol{\xi}_m]$
Interpolation Indices (Iteration Loop)	
• choose the first index (initialization)	$[\rho, \gamma_1] = \max\lvert\boldsymbol{\xi}_1\rvert$
• approximate $\boldsymbol{\xi}_j$ by $\boldsymbol{\xi}_1, ..., \boldsymbol{\xi}_{j-1}$ at indices $\gamma_1, ..., \gamma_{j-1}$	Solve for $\mathbf{c}$: $\mathbf{P}^T\boldsymbol{\xi}_j = \mathbf{P}^T\boldsymbol{\Xi}_{j-1}\mathbf{c}$ with $\mathbf{P} = [\mathbf{e}_{\gamma_1}\ \cdots\ \mathbf{e}_{\gamma_{j-1}}]$
• select γ_j and loop ($j = 2, 3, ..., m$)	$[\rho, \gamma_j] = \max\lvert\boldsymbol{\xi}_j - \boldsymbol{\Xi}_{j-1}\mathbf{c}\rvert$

DEIM algorithm, which is based upon a greedy-like search, is detailed in [29] and further demonstrated in Table 3.1.

3.2.3 Application to Library Learning for parametrized ROMs

The DEIM algorithm is highly effective for determining sampling (sensor) locations. Such sensors can be used with sparse representation and compressive sensing to (i) identify dynamical regimes, (ii) reconstruct the full state of the system, and (iii) provide an efficient nonlinear model reduction and POD-Galerkin prediction for the future state. Given the

parametrized nature of the evolution equation (3.4), we use the concept of library building which arises in machine learning from leveraging low-rank "features" from data. In the ROM community, it has recently become an issue of intense investigation. Indeed, a variety of recent works [78, 16, 12, 72, 4, 30, 70, 69, 68, 50] have produced libraries of ROM models that can be selected and/or interpolated through measurement and classification. Before these more formal techniques based upon machine learning were developed, it was already realized that parameter domains could be decomposed into subdomains and a local ROM/POD computed in each subdomain. Patera *et al.* [36] used a partitioning based on a binary tree whereas Amsallem *et al.* [1] used a Voronoi Tessellation of the domain. Such methods were closely related to the work of Du and Gunzburger [35] where the data snapshots were partitioned into subsets and multiple reduced bases computed. Thus the concept of library building is well established and intuitively appealing for parametrized systems.

We capitalize on these recent innovations and build optimal interpolation locations from multiple dynamics states [78]. However, the focus of this work is on computing, in an online fashion, nearly optimal sparse sensor locations from interpolation points found to work across all the libraries in an offline stage. The offline stage uses the DEIM architecture as this method gives good starting points for the interpolation. The genetic algorithm we propose then improves upon the interpolated points by a quick search of nearby interpolation points. It is the pre-computed library structure and interpolation points that allow the genetic algorithm to work with only a short search.

3.3 Genetic Algorithm for Improved Interpolation

The background Secs. 2 and 3 provide the mathematical framework for the innovations of this paper. Up to this point, the DEIM architecture for parametrized PDEs [78] provides good interpolation points for the ROM method. Our goal is to make the interpolation points optimal or nearly so. Unfortunately, non-convex problems such as this are extremely difficult to optimize, leading to the consideration of genetic algorithms, which are a subset of evolutionary algorithms, for determining near optimal interpolation points.

The genetic algorithm principal is quite simple: given a set of feasible trial solutions (either constrained or unconstrained), an objective (fitness) function is evaluated. The idea is to keep those solutions that give the minimal value of the objective function and mutate them in order to try and do even better. Mutation in our context involves randomly shifting the locations of the interpolation points. Beneficial mutations that give a better minimization, such as good classification and minimal reconstruction error, are kept while those that perform poorly are discarded. The process is repeated through a number of iterations, or *generations*, with the idea that better and better fitness function values are generated via the mutation process.

More precisely, the genetic algorithm can be framed as the constrained optimization problem with the objective function

$$\min \|\tilde{\mathbf{u}} - \mathbf{P}\mathbf{u}\|_2 \quad \text{subject to correct classification} \tag{3.16}$$

where $\mathbf{P}$ is a measurement matrix used for interpolation. Suppose that m mutations, as illustrated in Fig. 3.1, are given for the matrix $\mathbf{P}$ so that

$$j^{\text{th}} \text{ guess is } \mathbf{P}_j\,. \tag{3.17}$$

Thus m solutions are evaluated and compared with each other in order to see which of the solutions generate the smallest objective function since our goal is to minimize it. We can order the guesses so that the first $p < m$ gives the smallest values of $f(\mathbf{P})$. Arranging our data, we then have

$$\begin{aligned} \text{keep} \quad & \mathbf{P}_j \ j = 1, 2, \cdots, p \\ \text{discard} \quad & \mathbf{P}_j \ j = p+1, p+2, \cdots, m\,. \end{aligned} \tag{3.18}$$

Since the first p solutions are the best, these are kept in the next generation. In addition,we now generate $m-p$ new trial solutions that are randomly mutated from the p best solutions. This process is repeated through a finite number of iterations M with the hope that convergence to the optimal, or near-optimal, solution is achieved. Table 3.2 shows the algorithm

Table 3.2: DEIM algorithm for finding approximation basis for the nonlinearity and interpolation indices.

Genetic search algorithm	
• construct initial measurement matrix [78]	$\mathbf{P}$
• perturb measurements and classify	$\mathbf{P} \rightarrow \mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3, \cdots, \mathbf{P}_m$
• keep matrices with correct classification	$\mathbf{P}_k, \mathbf{P}_j, \mathbf{P}_\ell, \cdots$
• save ten best measurement matrices	$\mathbf{P} \rightarrow \mathbf{P}_1, \mathbf{P}_2, \cdots, \mathbf{P}_{10}$
• repeat steps for M generations	
• randomly choose one of ten best $\mathbf{P}$ and repeat	

structure particular to our application. In our simulations, $m = 100$ mutations are produced and $p = 10$ are kept for further mutation. The number of generations is not fixed, but we find that even with $M = 3$, significant improvement in reconstruction error can be achieved.

As we will show, the algorithm provides an efficient and highly effective method for optimizing the interpolation locations, even in a potentially online fashion. A disadvantage of the method is that there are no theorems guaranteeing that the iterations will converge to the optimal solution, and there are many reasons the genetic search can fail. However, we are using it here in a very specific fashion. Specifically, our initial measurement matrix $\mathbf{P}$ is already quite good for classification and reconstruction purposes [78]. Thus the algorithm starts close to the optimal solution. The goal is then to further refine the interpolation points so as to potentially cut down on the reconstruction and classification error. The limited scope of the algorithm mitigates many of the standard pitfalls of the genetic algorithm.

3.4 Model Problems

Two models help illustrate the principles and success of the genetic search algorithm coupled with DEIM. In the first example, only three interpolation points are necessary for classification and reconstruction [78]. Moreover, for this problem, a brute force search optimization can be performed to rigorously identify the best possible interpolation points. This allows us to compare the method advocated to a ground truth model. In the second example, the classical problem of flow around a cylinder is considered. This model has been ubiquitous in the ROMs community for demonstrating new dimensionality-reduction techniques.

3.4.1 Cubic-Quintic Ginzburg-Landau Equation

The Ginzburg-Landau (GL) equation is on of the canonical models of applied mathematics and mathematical physics as it manifests a wide range dynamical behaviors [31]. It is widely used in the study of pattern forming systems, bifurcation theory and dynamical systems. Its appeal stems from its widespread use in the sciences: modeling phenomena as diverse as condensed matter physics to biological waves. The particular variant considered here is the cubic-quintic GL with fourth-order diffusion [54]:

$$iU_t + \left(\frac{1}{2} - i\tau\right) U_{xx} - i\kappa U_{xxxx} + (1 - i\mu)\,|U|^2 U + (\nu - i\epsilon)|U|^4 U - i\gamma U = 0, \tag{3.19}$$

where $U(x,t)$ is a complex valued function of space and time. Under discretization of the spatial variable, $U(x,t)$ becomes a vector $\mathbf{u}$ with n components, i.e. $\mathbf{u}_j(t) = U(x_j,t)$ with $j = 1, 2, \cdots n$.

An efficient and exponentially accurate numerical solution to (3.19) can be found using standard spectral methods [55]. Specifically, the equation is solved by Fourier transforming in the spatial dimension and then time-stepping with an adaptive 4th-order Runge-Kutta method. The extent of the spatial domain is $x \in [-20, 20]$ with $n = 1024$ discretized points. Importantly, in what follows the interpolation indices are dictated by their position away from the center of the computational domain. The center of the domain is at x_0 which is

	τ	κ	μ	ν	ϵ	γ	description
β_1	-0.3	-0.05	1.45	0	-0.1	-0.5	3-hump, localized
β_2	-0.3	-0.05	1.4	0	-0.1	-0.5	localized, side lobes
β_3	0.08	0	0.66	-0.1	-0.1	-0.1	breather
β_4	0.125	0	1	-0.6	-0.1	-0.1	exploding soliton
β_5	0.08	-0.05	0.6	-0.1	-0.1	-0.1	fat soliton
β_6	0.08	-0.05	0.5	-0.1	-0.1	-0.1	dissipative soliton

Table 3.3: Values of the parameters from equation (3.19) that lead to six distinct dynamical regimes. To exemplify our algorithm, the first, third and fifth regimes will be discussed in this paper.

the 513th point in the domain. The interpolation indices demonstrated are relative to this center point.

To generate a variety of dynamical regimes, the parameters of the cubic-quintic GL are tuned to a variety of unique dynamical regimes. The unique parameter regime considered are denoted by the parameter $\beta = (\tau, \kappa, \mu, \nu, \epsilon, \gamma)$ which indicates the specific values chosen. Table 3.3 shows six different parameter regimes that have unique low-dimensional attractors as described in the table. It has been shown in previous work that only three interpolation points are necessary for classification of the dynamical state, state reconstruction and future state prediction [78, 16]. This previous work also explored how to construct the sampling matrix $\mathbf{P}$ from the DEIM algorithm and its multiple dynamical state.

We will execute the genetic algorithm outlined in Table 3.2 for improving the sampling matrix $\mathbf{P}$ initially determined from the algorithm in [78]. Before doing so, we consider a brute force search of the best possible three measurement locations based upon their ability to classify the correct dynamical regime and minimize reconstruction error. Although generally this is an *np*-hard problem, the limited number of sensors and small number of potential locations for sensors allow us to do an exhaustive search for the best interpolation locations.

The brute force search first selects all indices triplets that correctly classify the dynamical regimes in the absence of noisy measurements. From this subset, white noise is added to the measurements and 400 rounds of classification are performed. Only the measurement triplets giving above 95% accuracy for the classification of each dynamical regime are retained. The retained triplets are then sorted by the reconstruction error. Figure 3.2 shows the triplet interpolation points retained from the exhaustive search with the classification criteria specified and the position of the interpolation points along with the reconstruction error. The DEIM algorithm proposed in [78] produces interpolation points with reconstruction errors nearly an order of magnitude larger than those produced from the exhaustive search. Our objective is to use a genetic algorithm to reduce our error by this order of magnitude and produce interpolation points consistent with some of the best interpolation points displayed in Fig. 3.2.

The brute for search produces a number of interpolation triplets whose reconstruction accuracy are quite similar. Clearly displayed in the graph is the clustering of the interpolation points around critical spatial regions. A histogram of the first (blue), second (magenta) and third (green) interpolation points is shown in Fig. 3.3(a). The first two interpolation points have a narrow distribution around the 4th-8th interpolation points and 12th-16th interpolation points respectively. The third interpolation point is more diffusely spread across a spatial region with improvements demonstrated for interpolation points further to the right in Fig. 3.2(a). This histogram provides critical information about sensor and interpolation point locations. Of note is the fact that the DEIM algorithm always picks the maximum of the first POD mode as an interpolation location. This would correspond to a measurement at $x = 0$. However, none of the candidate triplets retained from a brute force search consider this interpolation point to be important. In fact, the interpolation points starting from the second iteration of the DEIM algorithm are what seem to be important according to the brute force search. This leads us to conjecture that we should initially use the triplet pair from the 2nd-4th DEIM points rather than the 1st-3rd DEIM points. We call these the DEIM+1 interpolation points as we shift our measurement indices to the start

after the first iteration of DEIM.

The genetic algorithm search can now be enacted from both the DEIM locations computed in [78] and the DEIM+1 locations suggested by the exhaustive search. Figure 3.3(b) shows the convergence of the genetic search optimization procedure starting from both these initial measurement matrices $\mathbf{P}$. It should be noted that the DEIM+1 initially begins with approximately half the error of the standard DEIM algorithm, suggesting it should be used as a default. In this specific scenario, both initial measurement matrices are modified and converge to the near-optimal solution within only 3-5 generations of the search. This is a promising result since the mutations and generations are straightforward to compute and can potentially be done in an online fashion. The benefit from this approach is a reduction of the error by nearly an order of magnitude, making it an attractive scheme.

To finish our analysis, we compare the DEIM architecture against some classic gappy POD and DEIM methods. Figure 3.4 gives an algorithmic comparison of the interpolation point selection of various techniques against the proposed method and the ground truth optimal solution obtained by exhaustive search. Both the reconstruction error and classification accuracy are important in selecting the interpolation indices, and both are represented in panels (b) and (c) of Fig. 3.4. Importantly, the method proposed here, which starts with the DEIM+1 points and does a quick genetic algorithm search produces nearly results that are almost equivalent to the exhaustive search. This is an impressive result given the efficiency of the genetic search and online computation possibilities. And even if one is not interested in executing the genetic search, the DEIM+1 points used for $\mathbf{P}$ provide nearly double the performance (in terms of accuracy) versus DEIM.

3.4.2 Flow Around a Cylinder

The previous example provides an excellent proof-of-concept given that we could compute a ground truth optimal solution. The results suggest that we should start with the DEIM+1 measurement matrix $\mathbf{P}$ and execute the genetic algorithm from there. We apply this method on the classic problem of flow around a cylinder. This problem is also well understood and

has already been the subject of studies concerning sparse spatial measurements [12, 83, 52, 51, 14]. Specifically, it is known that for low to moderate Reynolds numbers, the dynamics is spatially low-dimensional and POD approaches have been successful in quantifying the dynamics [83, 32, 61, 40, 58]. The Reynolds number, Re, plays the role of the bifurcation parameter μ in (3.1), i.e. it is a parametrized dynamical system.

The data we consider comes from numerical simulations of the incompressible Navier-Stokes equation:

$$\frac{\partial u}{\partial t} + u \cdot \nabla u + \nabla p - \frac{1}{Re}\nabla^2 u = 0 \tag{3.20a}$$

$$\nabla \cdot u = 0 \tag{3.20b}$$

where $u(x, y, t) \in \mathbb{R}^2$ represents the 2D velocity, and $p(x, y, t) \in \mathbb{R}^2$ the corresponding pressure field. The boundary condition are as follows: (i) Constant flow of $u = (1, 0)^T$ at $x = -15$, i.e., the entry of the channel, (ii) Constant pressure of $p = 0$ at $x = 25$, i.e., the end of the channel, and (iii) Neumann boundary conditions, i.e. $\frac{\partial u}{\partial \mathbf{n}} = 0$ on the boundary of the channel and the cylinder (centered at $(x, y) = (0, 0)$ and of radius unity).

We consider the fluid flow for Reynolds number $Re = 40, 150, 300, 1000$ and perform an SVD on the data matrix in order to extract POD modes. The rapid decay of singular values allows us to use a small number of POD modes to describe the fluid flow and build local ROMs. The POD modes retained for each Reynolds number is shown in Fig. 3.5. These modes are projected on cylindrical coordinates to better demonstrate the structure of the pressure field generated on the cylinder.

The POD modes can be used to construct a DEIM+1 interpolation matrix $\mathbf{P}$ illustrated in Fig. 3.1. The DEIM+1 interpolation points already provide a good set of interpolation points for classification and reconstruction of the solution, beating standard DEIM by nearly a factor of two. However, the genetic algorithm advocated in this work can be used to adjust the interpolation points and achieve both better classification performance and improved reconstructions. In the cubic-quintic GL equation, the error was improved by nearly an order of magnitude over the standard DEIM approach. For the flow around the cylinder,

the error is also improved from the DEIM+1 algorithm, quickly reducing the error with $M = 2$ generations and converging to the nearly optimal interpolation points within $M = 10$ generations. Given the limited number of interpolation points, the genetic search can be computed in an online fashion even for this two-dimensional fluid flow problem.

Figure 3.6 is a composite figure showing the pressure field evolution in time along a the cylinder. The heat map shows the dominant, low-dimensional pattern of activity that is used for generating POD modes. Overlaid on this are the best sensor/interpolation locations at each generation of the genetic algorithm scheme for 10 interpolation points over 7 generations of the search. Note the placement of the interpolation points around the cylinder. Specifically, as the number of generations increases, the interpolation points move to better sampling positions, reducing the error in the ROM. The convergence of the error across 10 generations of the algorithm is shown in Fig. 3.7 along with the final placement of the interpolation points. The near optimal interpolation points are not trivially found. Overall, the DEIM+1 architecture with genetic algorithm search reduces the error by anywhere between a factor of two and an order of magnitude, making it attractive for online error reduction and ROM performance enhancement.

3.5 Conclusions

ROMs are enabled by two critical steps: (i) the construction of a low-rank subspace where the dynamics can be accurately projected, and (ii) a sparse sampling method that allows for an interpolation-based projection to provide a nearly ℓ_2 optimal subspace approximation to the nonlinear term without the expense of orthogonal projection. Innovations to improve these two aims can improve the outlook of scientific computing methods for modern, high-dimensional simulations that are rapidly approaching exascale levels. These methods also hold promise for real-time control of complex systems, such as turbulence [17]. This work has focused on improving the sparse sampling method commonly used in the literature. In partnership with the DEIM infrastructure, a genetic algorithm was demonstrated to determine nearly optimal sampling locations for producing a subspace approximation of the nonlinear

term without the expense of orthogonal projection. The algorithm can be executed in a potentially online manner, improving the error by up to an order-of-magnitude in the examples demonstrated here. For a fixed number of interpolation points m, the first $m+1$ DEIM interpolation points are computed and the first point is discarded. This DEIM+1 sampling matrix alone can reduce the error by a factor of two before starting the genetic algorithm search.

In general, genetic algorithms are not ideal for optimization since they rarely have guarantees on convergence and have many potential pitfalls. In our case, the DEIM+1 starting point for the interpolation point selection algorithm is already close to the true optimum. Thus the genetic algorithm is not searching blindly in a high-dimensional fitness space. Rather, the algorithm aims to simply make small adjustments and refinements to the sampling matrix in order to maximize the performance of the nonlinear interpolation approximation. In this scenario, many of the commonly observed genetic algorithm failures are of little concern. The method is shown to reduce the error by a substantial amount within only one or two generations, thus making it attractive for implementation in real large-scale simulations where accuracy of the solution may have significant impact on the total computational cost. In comparison to many other sparse sampling strategies used in the literature, it out performs them by a significant amount both in terms of accuracy and ability to classify the dynamical regime. Indeed, the algorithm refines the sampling matrix $\mathbf{P}$ to be nearly optimal.

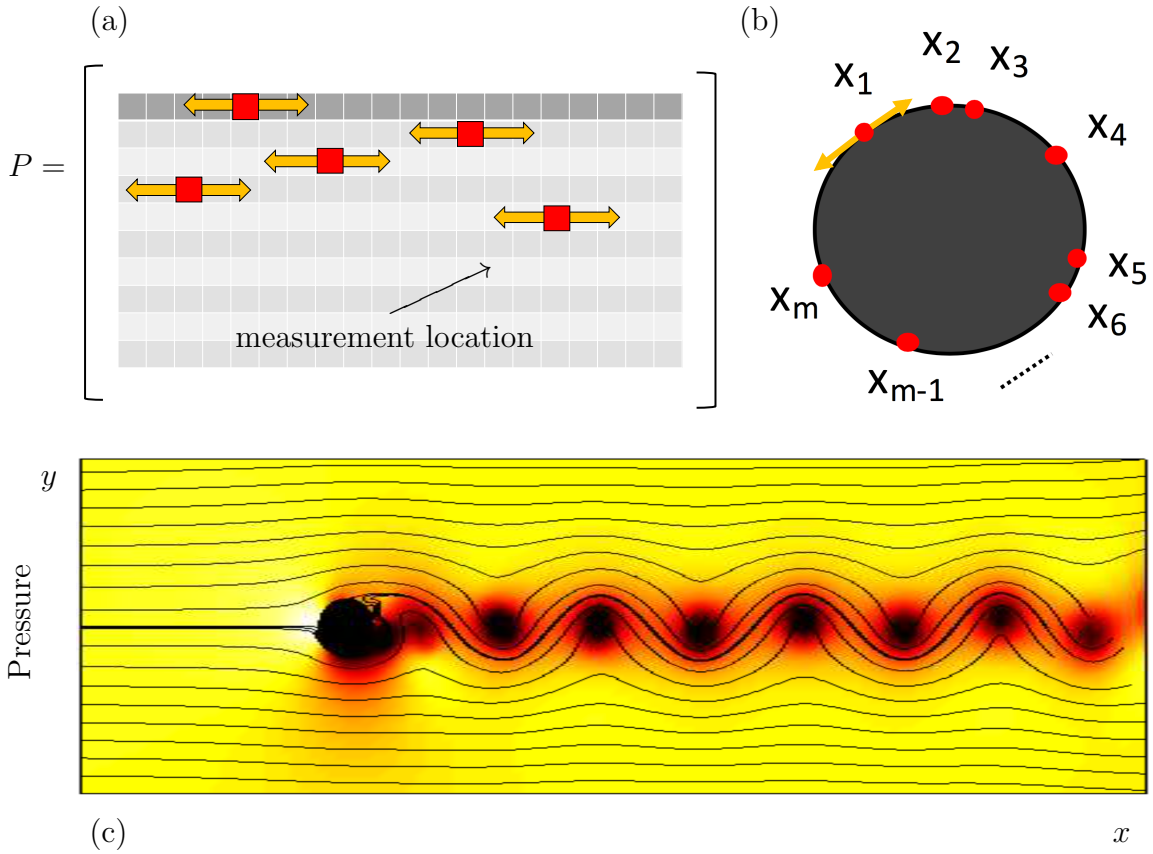


Figure 3.1: Illustration of the genetic algorithm search for optimal sparse sampling locations. (a) The measurement matrix P is constructed from the DEIM algorithm across libraries of POD modes [78]. Given that the matrix P has already been demonstrated to produce good interpolation points, the interpolation indices are shifted to nearby locations in a genetic algorithm search for the best interpolation configuration. (b) The algorithm has a practical physical interpretation for the example of flow around a cylinder where the sensor locations are shifted for best performance. (c) The near-optimal interpolation points found from the genetic algorithm are used to both classify the dynamic regime, in this example it discovers the Reynolds number, and to reconstruct the full pressure field (on and off the cylinder) with minimal residual [12].

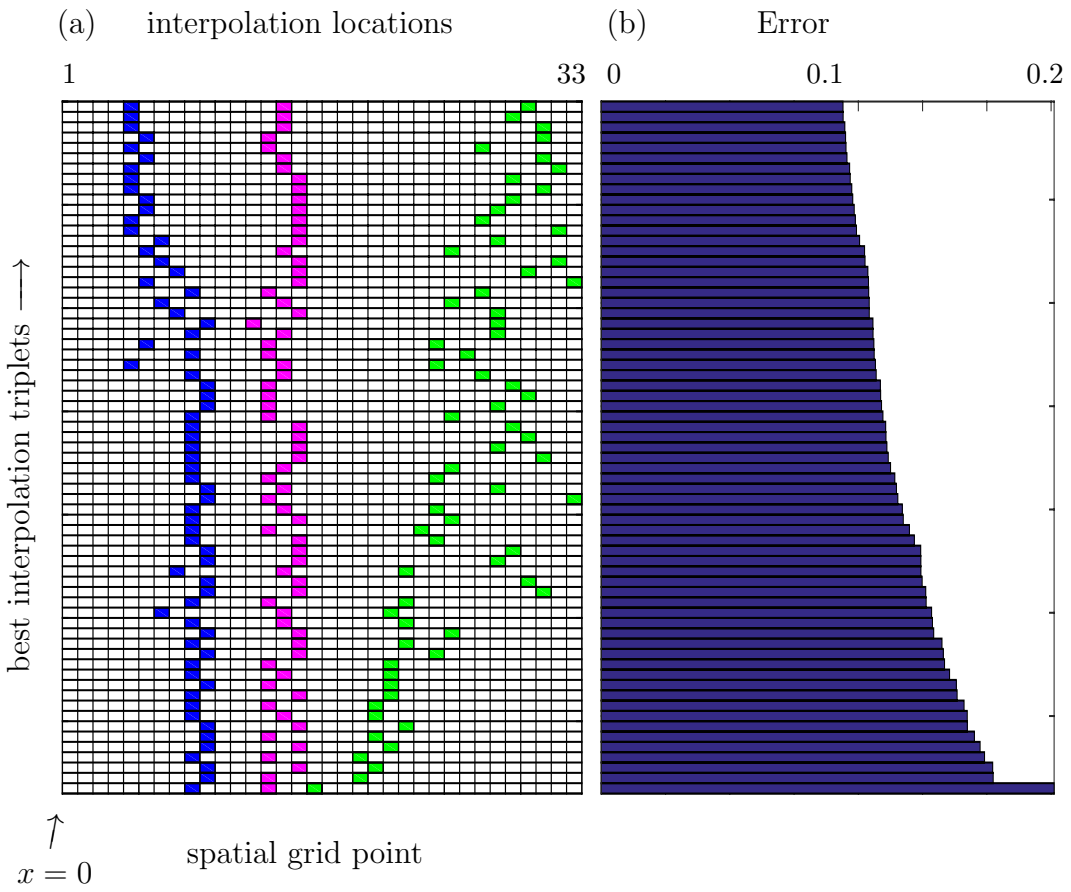


Figure 3.2: Results of an exhaustive brute force search for selecting the best three interpolation point triplets that correctly classify the dynamical regimes in the absence of noisy measurements. From this subset, white noise is added to the measurements and 400 rounds of classification are performed. Only the measurement triplets giving above 95% accuracy for the classification of each dynamical regime are retained. The retained triplets are then sorted by the reconstruction error as shown in (a). The corresponding error is shown in (b).

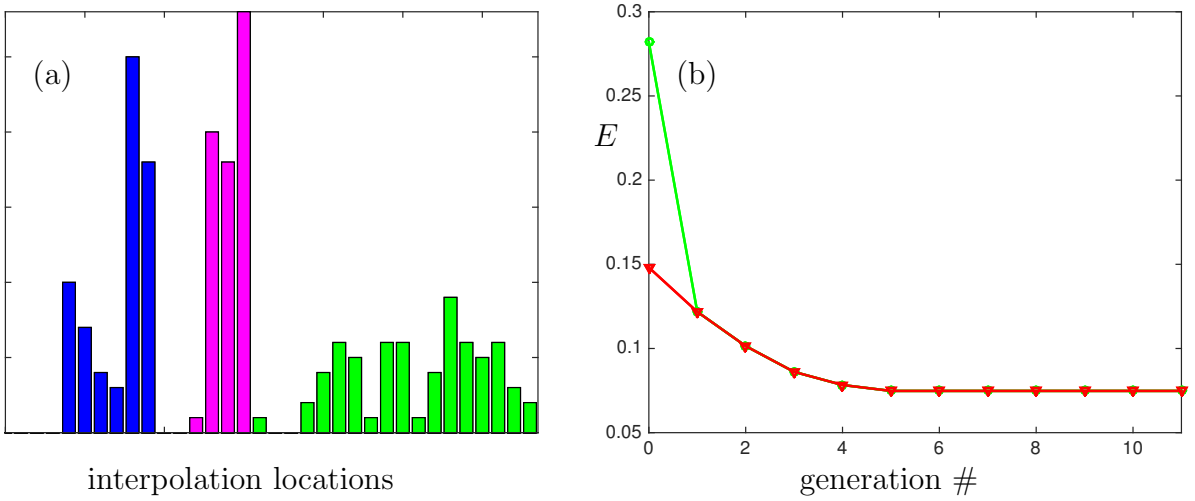


Figure 3.3: (a) Histogram of the first (blue), second (magenta) and third (green) interpolation points found from the exhaustive optimization search algorithm in Fig. 3.2. The first two interpolation points are highly localized near key spatial locations. Note that the histograms demonstrate that the first interpolation point selected from the standard DEIM algorithm is not an optimal point. Instead, for this case we can consider the 2nd-4th point instead, shifting the selection of optimal points by one iteration and resulting in the DEIM+1 algorithm, i.e. we generate the first 4 interpolation points from DEIM and drop the first. (b) The genetic algorithm is executed starting form the sparse sampling matrix $\mathbf{P}$ of DEIM and DEIM+1 showing that within 2-5 generations nearly optimal interpolation points are found in regards to the error E.

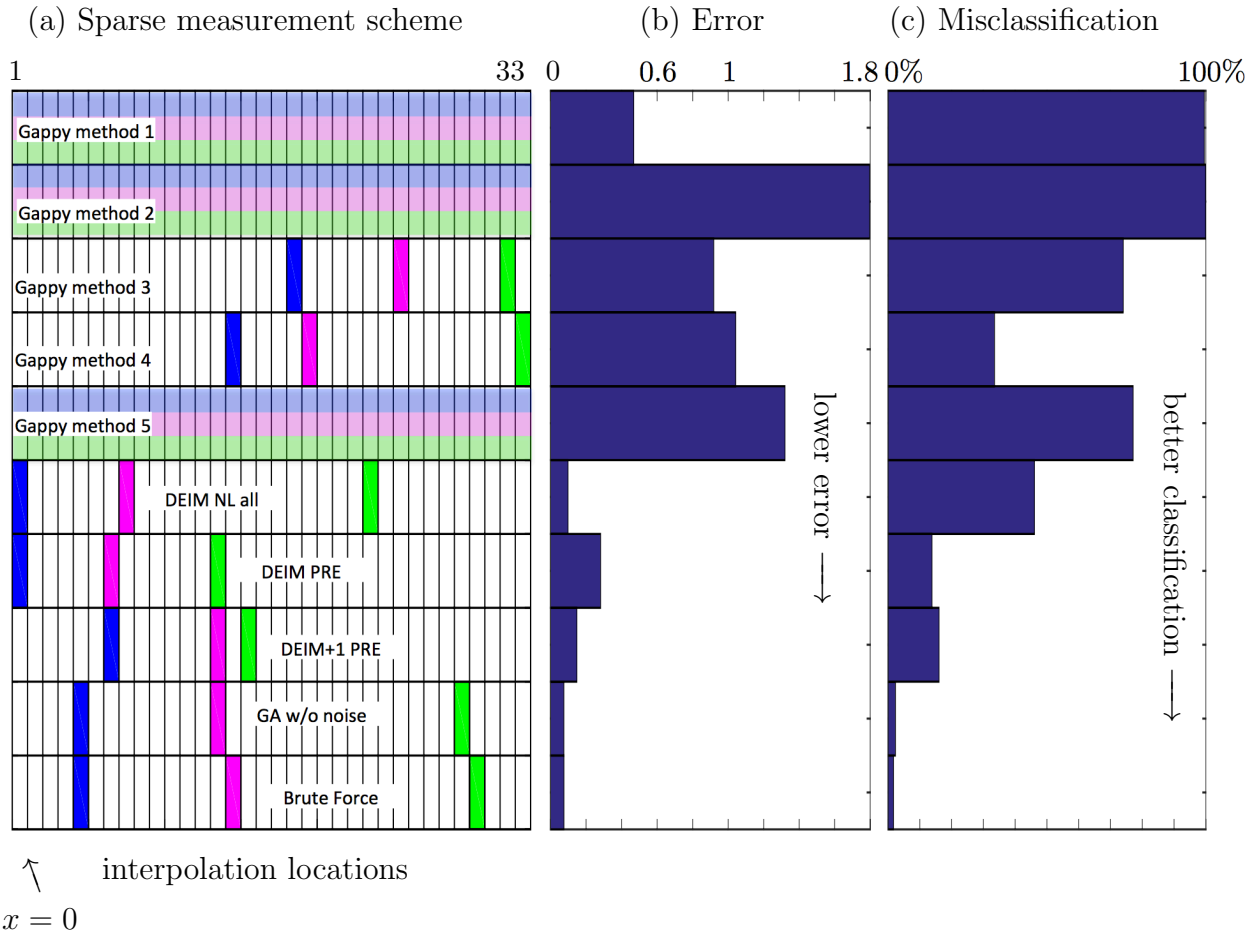


Figure 3.4: (a) Comparison of various sparse sampling strategies, including various gappy POD and DEIM methods, against the optimal sparse sampling solution determined by exhaustive search. The first 33 indices of the discretized PDE are shown where the first index corresponds to $x = 0$. The first, second and third measurement locations are denoted by the blue, magenta and green bars respectively. The color bars that span the index locations represents random measurement locations which can be potentially at any of the indices. The accompanying (b) error and (c) misclassification scores are also given. In contrast to many of the other methods, the genetic algorithm proposed produces results that are almost identical to the true optimal solution, making it a viable method for online enhancement of ROMs. The various sparse selection methods are as follows: (i) Gappy method 1: random selection of three indices from interpolation range 1-33 (where the histograms in Fig. 3.3 suggest the measurements should occur), (ii) Gappy method 2: random selection from all possible interpolation points on the domain, (iii) Gappy method 3: condition number minimization routine for three interpolation points [85], (iv) Gappy method 4: same as Gappy method 3 but with ten interpolation points (i.e. it is now full rank), (v) Gappy method 5: selection of interpolation points from maxima and minima of POD modes [87], (vi) DEIM NL all: DEIM algorithm applied jointly to all the nonlinear terms of all dynamical regimes, (vii) DEIM PRE: algorithm developed in [78], (viii) DEIM+1 PRE: use the algorithm in DEIM PRE but discard the first DEIM point and select from the 2nd-4th DEIM points, (ix) GA: genetic algorithm advocated here, (x) Brute force: optimal solution from exhaustive search.

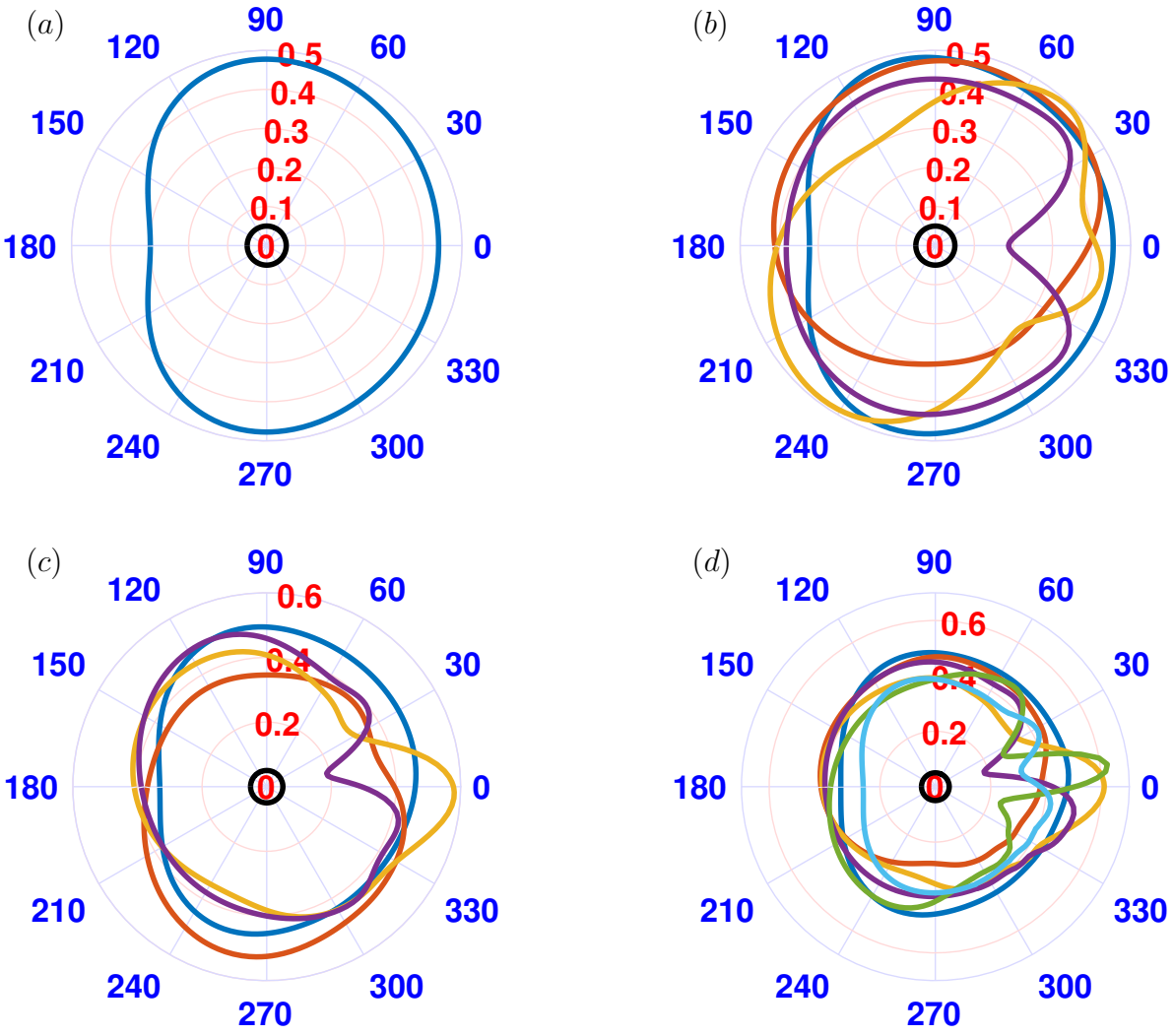


Figure 3.5: Dominant POD modes for Reynolds numbers (a) 40, (b) 150, (c) 300 and (d) 1000. The POD modes are plotted in cylindrical coordinates to illustrate the pressure field generated on the cylinder. For low Reynolds numbers ($Re = 40$), a single dominant POD mode exists. As the Reynolds number increases, more exotic mode structures, which are asymmetric, are manifested. The blue labels are the degrees around the cylinder while the red labels are the amplitudes of the POD pressure modes. The first mode is in blue, followed by red, gold, purple, green and cyan.

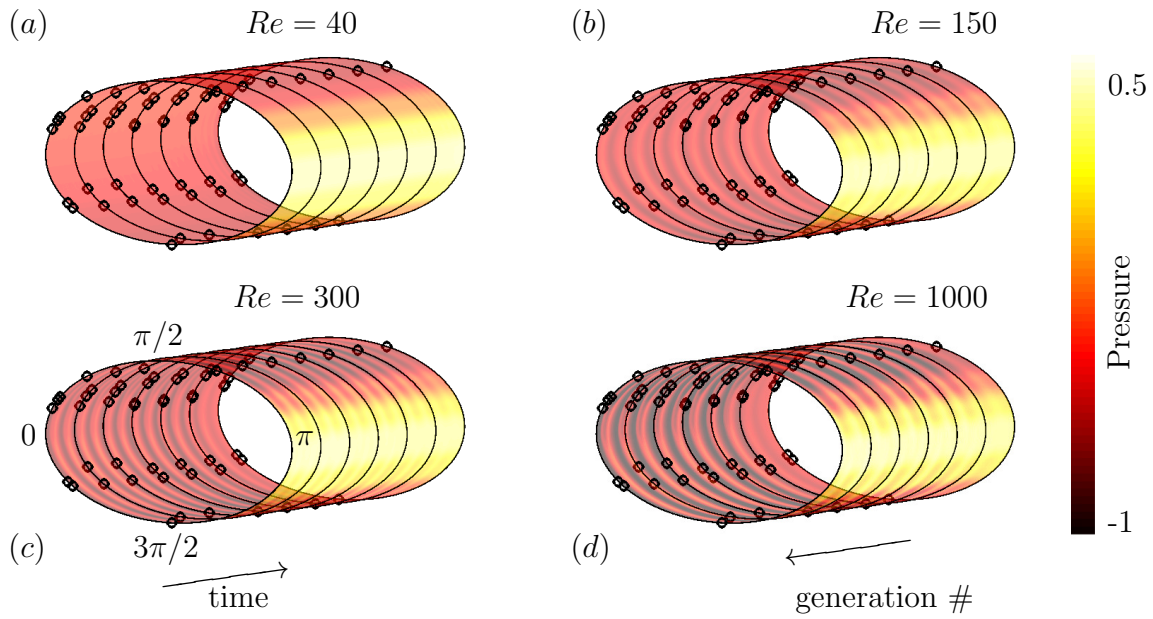


Figure 3.6: The heat map on the cylinder shows the dominant, low-dimensional pattern of activity that is used for generating POD modes for Reynolds number (a) 40, (b) 150, (c) 300 and (d) 1000. Overlaid on the heat map are the best sensor/interpolation locations (circles) at each generation of the genetic algorithm scheme for $m = 10$ interpolation points over $M = 7$ generations of the search. The interpolation locations for each generation in the genetic algorithm start at the back and move forward.

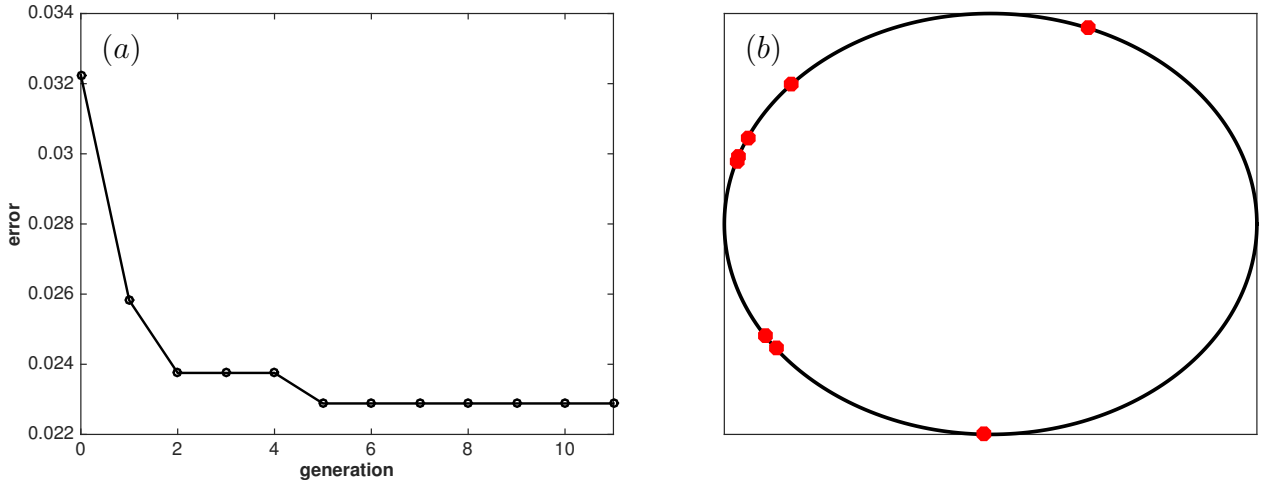


Figure 3.7: (a) Error reduction and convergence of the genetic algorithm from an initial DEIM+1 interpolation matrix $\mathbf{P}$. Significant error reduction is accomplished in a single generation. A total of $M = 10$ generations are shown, illustrating the convergence to the nearly optimal solution. (b) The final position of the interpolation points (red) is shown for 10 interpolation points on the cylinder.

Chapter 4

STRUCTURE-PRESERVING NONLINEAR MODEL REDUCTION FOR FINITE-VOLUME MODELS OF CONSERVATION LAWS

4.1 Reduced order models: Galerkin, least-squares Petrov-Galerkin and GNAT

Suppose we are given a partial differential equation (PDE) describing a problem of computational fluid dynamics and an initial condition:

$$\frac{\partial u}{\partial t}(x,t) = F(u,x,t),$$
$$u(0) = u^0,$$

where $t > 0$ is time, $x \in \mathbf{D}$ and F is a nonlinear function. Applying, for example, the finite volume discretization [38, 57] method to this computational fluid dynamics equation, we get a system of ordinary differential equations (ODEs) and a set of initial conditions.

$$\frac{d\mathbf{u}}{dt}(t) = \mathbf{F}(\mathbf{u},t), \tag{4.1}$$

$$\mathbf{u}(0) = \mathbf{u}^0, \tag{4.2}$$

where $\mathbf{u} \in \mathbb{R}^n$ is now the fluid state vector representing the conserved fluid quantities, n is the dimension of the semi-discretization, $\mathbf{u}^0 \in \mathbb{R}^n$ is the initial condition. Note that components of $\mathbf{u}$ represent the mass, moments in x, y, z directions and energy of the model at mesh cell centers. For high fidelity models the number of discretization points n can be very large leading to intensive large scale computations. To reduce the dimension of the system one can apply the Galerkin method as follows. Compute a low dimensional approximating basis $\mathbf{\Phi}_r$ using, for instance, the proper orthogonal decomposition (POD). Note that in practice, the dimension of the basis r is much smaller then the dimension of the system n. This step is

followed by approximation of the solution using the pre-computed basis: $\mathbf{u}(t) \approx \mathbf{u}^0 + \mathbf{\Phi}_r \mathbf{a}(t)$. Substituting this into the equation (4.1)-(4.2) we get

$$\frac{d\mathbf{a}}{dt}(t) = \mathbf{\Phi}_r^T \mathbf{F}(\mathbf{u}^0 + \mathbf{\Phi}_r \mathbf{a}(t), t),$$
$$\mathbf{a}(0) = 0.$$

Thus, finding the solution of n dimensional system is reduced to finding the solution of r dimensional system, where $r \ll n$. This system now can be solved using a time integrator method such as Runge-Kutta scheme.

Instead of first applying the projection method and then doing time discretization, it is possible to first apply a time integrator scheme followed by projection. Petrov-Galerkin [71] and GNAT [24] are instances of such reduced order models. Applying, for example, Runge-Kutta scheme to the system (4.1), reduces it to a sequence of n_t nonlinear systems of equations, where n_t is the number of time steps carried out. Consider one instance of such a system:

$$\mathbf{R}(\mathbf{u}) = 0 \tag{4.3}$$

where $\mathbf{R} : \mathbb{R}^n \to \mathbb{R}^n$ is the nonlinear residual function. The fluid state vector $\mathbf{u}$ is implicitly defined by the above equation. We will refer to this model as full order model (FOM). Equation (4.3) can be solved using Gauss-Newton's method in the following way: pick an initial guess $\mathbf{u}^0$ and
for $k = 1, 2, ..., K$:

1. $\mathbf{J}^k \Delta \mathbf{u}^k = -\mathbf{R}^p$

2. $\mathbf{u}^k = \mathbf{u}^{k-1} + \alpha \Delta \mathbf{u}^k$

where

$$\mathbf{J}^k = \frac{\partial \mathbf{R}}{\partial \mathbf{u}} \left(\mathbf{u}^k \right) \tag{4.4}$$

is the $n \times n$ Jacobian of the function $\mathbf{R}$ and α is the step size determined by a line search algorithm. We now use the least-squares Petrov-Galerkin method to solve the equation (4.3). As in the Galerkin projection method, we precompute the approximation basis $\mathbf{\Phi}_r \in \mathbb{R}^{n \times r}$ and look for the solution of the form $\mathbf{u}(t) \approx \mathbf{u}^0 + \mathbf{\Phi}_r \mathbf{a}(t)$, where $\mathbf{a} \in \mathbb{R}^p$ represents generalized coordinates. Substituting this into the equation (4.3) we get

$$\mathbf{R}(\mathbf{u}^0 + \mathbf{\Phi}_r \mathbf{a}) = 0$$

which is an overdetermined system of nonlinear equations. This system can be solved, for example, by minimizing the 2-norm of the function:

$$\min_{\mathbf{a} \in \mathbb{R}^p} ||\mathbf{R}(\mathbf{u}_0 + \mathbf{\Phi}_r \mathbf{a})||_2^2. \tag{4.5}$$

which is the least-squares Petrov-Galerkin method. One of the most common ways of solving this optimization problem is Gauss-Newton method. It is done as follows:

$$\Delta \mathbf{a}^k = \arg\min_{v \in \mathbb{R}^k} \|\mathbf{R}^k + \mathbf{J}^k \mathbf{\Phi}_r v\|, \tag{4.6}$$

$$\mathbf{a}^k = \mathbf{a}^{k-1} + \alpha_k \Delta \mathbf{a}^k, \tag{4.7}$$

$$\mathbf{u}^k = \mathbf{u}^0 + \mathbf{\Phi}_r \mathbf{a}^k. \tag{4.8}$$

The minimization problem can be solved by writing down the normal equation of (4.6). We reduced solving an n dimensional system to solving an r dimensional system, however this still can be computationally intensive due to nonlinear terms.

If we instead use subsamples of the solution we can significantly simplify the computation. Suppose $\mathbf{P} \in \mathbb{R}^{\mathbf{r} \times \mathbf{n}}$ is the sampling matrix consisting of some columns of the identity matrix. This matrix indicates locations where the subsamples will be taken at the online stage and used for quick computations. In this case the minimization problem becomes

$$\min_{\mathbf{a} \in \mathbb{R}^p} ||\mathbf{P}\mathbf{R}(\mathbf{u_0} + \mathbf{\Phi_r} \mathbf{a})||_2^2. \tag{4.9}$$

with Gauss-Newton iteration being:

$$\Delta \mathbf{a}^k = \arg\min_{\mathbf{b}\in\mathbb{R}^k} \|\mathbf{P}\mathbf{R}^{\mathbf{k}} + \mathbf{P}\mathbf{J}^{\mathbf{k}}\mathbf{\Phi}_{\mathbf{r}}\mathbf{b}\|,$$
$$\mathbf{a}^k = \mathbf{a}^{k-1} + \alpha_k \Delta \mathbf{a}^k,$$
$$\mathbf{u}^k = \mathbf{u}^0 + \mathbf{\Phi}_r \mathbf{a}^k.$$

This naive approach of simply subsampling the residual will most likely fail to work. Instead, the GNAT method tries to choose locations that will approximate the residual and its Jacobian well. This method precomputes an orthogonal reduced basis $\mathbf{\Psi}$ for the residual and replaces $\mathbf{R}$ in the equation (4.5) with $\mathbf{\Psi}(\mathbf{P}\mathbf{\Psi})^{+}\mathbf{\Psi}\mathbf{R}$, where "+" is the Moore-Penrose pseudoinverse. The problem then becomes:

$$\min_{\mathbf{a}\in\mathbb{R}^p} ||\mathbf{\Psi}(\mathbf{P}\mathbf{\Psi})^{+}\mathbf{\Psi}\mathbf{R}(\mathbf{u_0} + \mathbf{\Phi_r}\mathbf{a})||_2^2 \tag{4.10}$$

and the corresponding Gauss-Newton approximation

$$\Delta \mathbf{a}^k = \arg\min_{v\in\mathbb{R}^k} \|\mathbf{\Psi}(\mathbf{P}\mathbf{\Psi})^{+}\mathbf{\Psi}\mathbf{R}^{\mathbf{k}} + \mathbf{\Psi}(\mathbf{P}\mathbf{\Psi})^{+}\mathbf{\Psi}\mathbf{J}^{\mathbf{k}}\mathbf{\Phi}_{\mathbf{r}}\mathbf{v}\|$$
$$\mathbf{a}^k = \mathbf{a}^{k-1} + \alpha_k \Delta \mathbf{a}^k$$
$$\mathbf{u}^k = \mathbf{u}^0 + \mathbf{\Phi}_r \mathbf{a}^k$$

The node selecting algorithm and more details on GNAT method can be found in [24].

4.2 New Approach for Structure Preservation

Conservation laws in CFD equations are important for computing accurate results. The above mentioned methods might not preserve this important property of the model. Note that the ROM methods mentioned above solve problems with nonlinear least-squares approach, hence, it is possible to convert them into a constrained minimization problem and require the solution to satisfy the conservation laws on the full domain $\mathbf{D}$. The conservation of the mass of the fluid in the domain $\mathbf{D}$ is described by the following equation:

$$\frac{\partial}{\partial t}\int_{\mathbf{D}} \rho(x,t)dx + \int_{\mathbf{\Gamma}} \rho(x,\tau)\sum_{i=1}^{i=3} u_i(x,\tau)n_i(x,\tau)dxd\tau = \int_{\mathbf{D}} \mathbf{Q}_\rho(x,t)dx,$$

where $\mathbf{\Gamma}$ is the boundary of $\mathbf{D}$, $\mathbf{u} = [u_1, u_2, u_3]$ is the velocity of the fluid in 3D, ρ is the density function, $\mathbf{Q}_\rho$ is the force term and $\mathbf{n} = [n_1, n_2, n_3]$ is the normal component of the velocity. Further in the notes the Einstein notation will be used and the summation sign will be skipped:

$$\frac{\partial}{\partial t}\int_{\mathbf{D}} \rho(x,t)dx + \int_{\mathbf{\Gamma}} \rho(x,\tau)u_i(x,\tau)n_i(x,\tau)dxd\tau = \int_{\mathbf{D}} \mathbf{Q}_\rho(x,t)dx, \tag{4.11}$$

Similarly, the conservation of the moments and energy are described by the following equations:

$$\frac{\partial}{\partial t}\int_{\mathbf{D}} \rho(x,t)u_k(x,t)dx + \int_{\mathbf{\Gamma}} u_k(x,\tau)\rho(x,\tau)u_i(x,\tau)n_i(x,\tau)dxd\tau = \int_{\mathbf{D}} \mathbf{Q}_{\rho u}(x,t)dx, \quad k = 1,2,3 \tag{4.12}$$

$$\frac{\partial}{\partial t}\int_{\mathbf{D}} e(x,t)dx + \int_{\mathbf{\Gamma}} e(x,\tau)n_i(x,\tau)u_i(x,\tau)dxd\tau = \int_{\mathbf{D}} \mathbf{Q}_e(x,t)dx, \tag{4.13}$$

Thus, the constrained optimization problem has the following form:

$$\min_{\mathbf{u}\in\mathbb{R}^n} ||\mathbf{R}(\mathbf{u}^0 + \mathbf{\Phi}_r\mathbf{a})||_2^2 \qquad \text{subject to conservation laws (4.11)-(4.13).}$$

Note that in the 3D case we have only five constraints where conservation laws are required to hold on the full domain $\mathbf{D}$. However, taking any partition of the domain $\mathbf{D}$, we can require the conservation laws to hold on each subdomain. This will increase the number of constraints imposed to the problem. In other words, suppose $\mathbf{D} = \bigcup_{j=1}^{m} \mathbf{D}_j$, where $\text{int}(\mathbf{D}_i) \cap \text{int}(\mathbf{D}_j) = \emptyset,\ i \neq j$. Denote the boundary of each subset $\mathbf{D}_j$ by $\mathbf{\Gamma}_j$. Thus, overall we will have $n_c = 5m$ constraints:

$$\frac{\partial}{\partial t}\int_{\mathbf{D_j}} \rho(x,t)dx + \int_{\mathbf{\Gamma_j}} \rho(x,\tau)u_i(x,\tau)n_i(x,\tau)dxd\tau = \int_{\mathbf{D_j}} \mathbf{Q}_\rho(x,t)dx, \tag{4.14}$$

$$\frac{\partial}{\partial t}\int_{\mathbf{D_j}} \rho(x,t)u_k(x,t)dx + \int_{\mathbf{\Gamma_j}} u_k(x,\tau)\rho(x,\tau)u_i(x,\tau)n_i(x,\tau)dxd\tau = \int_{\mathbf{D_j}} \mathbf{Q}_{\rho u}(x,t)dx, \quad k = 1,2,3 \tag{4.15}$$

$$\frac{\partial}{\partial t}\int_{\mathbf{D_j}} e(x,t)dx + \int_{\mathbf{\Gamma_j}} e(x,\tau)n_i(x,\tau)u_i(x,\tau)dxd\tau = \int_{\mathbf{D_j}} \mathbf{Q}_e(x,t)dx, \tag{4.16}$$

Now the optimization problem has more constraints:

$$\min_{\mathbf{a}\in\mathbb{R}^n} ||\mathbf{R}(\mathbf{u}_0 + \mathbf{\Phi}_r\mathbf{a})||_2^2 \text{ subject to } n_c \text{ number of conservation laws (4.14)- (4.16).} \tag{4.17}$$

Note, there are three different possibilities for n_c:

1. $n_c < r$; underdetermined system of equations for constraints; need to use regularized optimization to solve for $\mathbf{a}$.

2. $n_c = r$; determined system of equations for constraints; unique solution for $\mathbf{a}$

3. $n_c > r$; overdetermined system of equations for constraints

In the first scenario one way of solving the optimization problem is to use a method mentioned in the paper by R. Zimmerman, A. Vendl and G. Stefan [88]. They call this method the constrained least-squares (LSQ)-ROM. The authors demonstrate that a constrained nonlinear least-squares problem can be solved almost as efficiently as the unconstrained problem. They perform a constrained Gauss-Newton iteration very efficiently using so-called Schur complement on block matrices. To make the algorithm faster, the line search step was omitted and the step size was set to constant value of 1. A slight modification of the proposed algorithm is given in table 4.1 which is adopted for our case. In general, one could use also sequential quadratic programming to solve the constrained optimization problem. However, in that case the least-squares structure would not be used.

In the second case $\mathbf{a}$ can be uniquely defined from the equations representing constraints and in the last case we can use the least squares method on the same system of equations and find $\mathbf{a}$.

Table 4.1: Algorithm to solve a constrained least-squares problem.

For $k = 0 : k_{\max}$ do the following	
Compute the Jacobian of residual, $\mathbf{J}_{\mathbf{R}}^k$	$\mathbf{J}_{\mathbf{R}}^k = \left(\frac{\partial \mathbf{R}}{\partial \mathbf{a}_1} \left(\mathbf{a}^k \right), ..., \frac{\partial \mathbf{R}}{\partial \mathbf{a}_r} \left(\mathbf{a}^k \right) \right)$
Compute the Jacobian of constraints, $\mathbf{J}_{\mathbf{g}}^k$	$\mathbf{J}_{\mathbf{g}}^k = \left(\frac{\partial \mathbf{g}}{\partial \mathbf{a}_1} \left(\mathbf{a}^k \right), ..., \frac{\partial \mathbf{g}}{\partial \mathbf{a}_r} \left(\mathbf{a}^k \right) \right)$
Compute the following matrix and vector	$\mathbf{H} = {\mathbf{J}_{\mathbf{g}}^k}^T \mathbf{J}_{\mathbf{g}}^k \in \mathbb{R}^{r \times r}$, $\mathbf{h} = \mathbf{J}_{\mathbf{g}}^k \mathbf{R}^k \in \mathbb{R}^r$
Find $\mathcal{P}$	$\mathbf{H}\mathcal{P} = \mathbf{J}_{\mathbf{g}}^k$, $\mathcal{P} \in \mathbb{R}^{r \times n_c}$
Find $\mathbf{x}$	$\mathbf{H}\mathbf{x} = \mathbf{h}$
Compute the following matrices	$\mathbf{S} = -(\mathbf{J}_{\mathbf{g}}^k \mathcal{P})^{-1} \in \mathbb{R}^{n_c \times n_c}, \quad \mathbf{Q} = \mathcal{P}\mathbf{S} \in \mathbb{R}^{r \times n_c}$
Compute search direction	$\Delta \mathbf{a}^k = \mathbf{Q}\mathbf{g}^k - (\mathbf{x} - \mathbf{Q}\mathbf{J}_{\mathbf{g}}^k) \in \mathbb{R}^r$
If converged, break	if $\|\Delta \mathbf{a}^k\| < \epsilon \left(\|\mathbf{a}^k\| + \epsilon \right)$: break
Update $\mathbf{a}$	$\mathbf{a}^{k+1} = \mathbf{a}^k + \Delta \mathbf{a}^k$
Go to the next step	$k = k + 1$

4.3 Model Problem

The method proposed in the previous section was tested on both the steady and unsteady quasi-1-D Euler equation [49] . However, this work discusses only the steady state flow. More details for unsteady state problem can be found in [49]. This equation describes one dimensional flow in a nozzle with small cross sectional area and expresses the conservation of the mass, momentum and total energy. In case of supersonic flow the equation can describe shock waves.

$$\frac{\partial U}{\partial t} + \frac{1}{S} \frac{\partial FS}{\partial x} = Q,$$

where

$$U = \begin{bmatrix} \rho \\ \rho u \\ e \end{bmatrix}, \quad F = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ (e+p)u \end{bmatrix}, \quad Q = \begin{bmatrix} 0 \\ \frac{p}{S}\frac{\partial A}{\partial x} \\ 0 \end{bmatrix},$$

$$p = (\gamma - 1)\rho\epsilon, \quad \epsilon = \frac{e}{p} - \frac{1}{2}u^2, \quad S = S(x).$$

The variables in the equation have the following meanings: ρ, u and e are the density, velocity and energy of a system correspondingly. Quantities p, ϵ and γ represent the pressure, the internal (thermal) heat and a ratio of specific heats respectively. S is a function of x and describes the cross sectional area of the channel. The finite volume discretization is applied to this equation and the flux differencing of splitting of Roe [77] is used to construct the interface flux.

The problem was solved with several methods and the results were compared. Here are the description of the methods used:

Method 1 (FOM). The full order model is considered and solved using the forward Euler time scheme and Gauss-Newton's method. During this stage the state vector, residual and its Jacobian are stored.

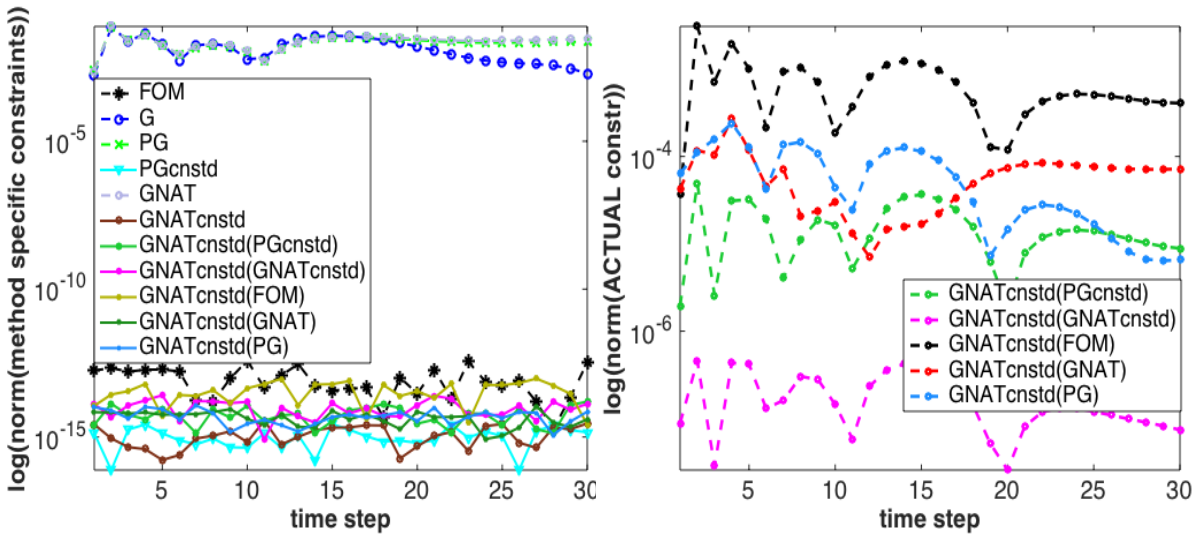
Method 2 (G). Using the stored values build a low rank POD basis and considers a reduced order model generated from the Galerkin projection. Solve it with forward Euler discretization and Gauss-Newton's method. Again the state vector, residual and the Jacobian are stored.

Method 3 (PG). The same as Method 2 except solved with the least-squares Petrov-Galerkin method.

Method 4 (PG_constrained). Solve the residual minimization problem subject to conservation laws on the full domain (or on its partition).

Method 5 (GNAT). Use GNAT algorithm with snapshots from method 3.

Method 6 (GNAT_constrained). Solve the constraint optimization problem with actual constraints that are computed as follows: at each step reconstruct the full state from



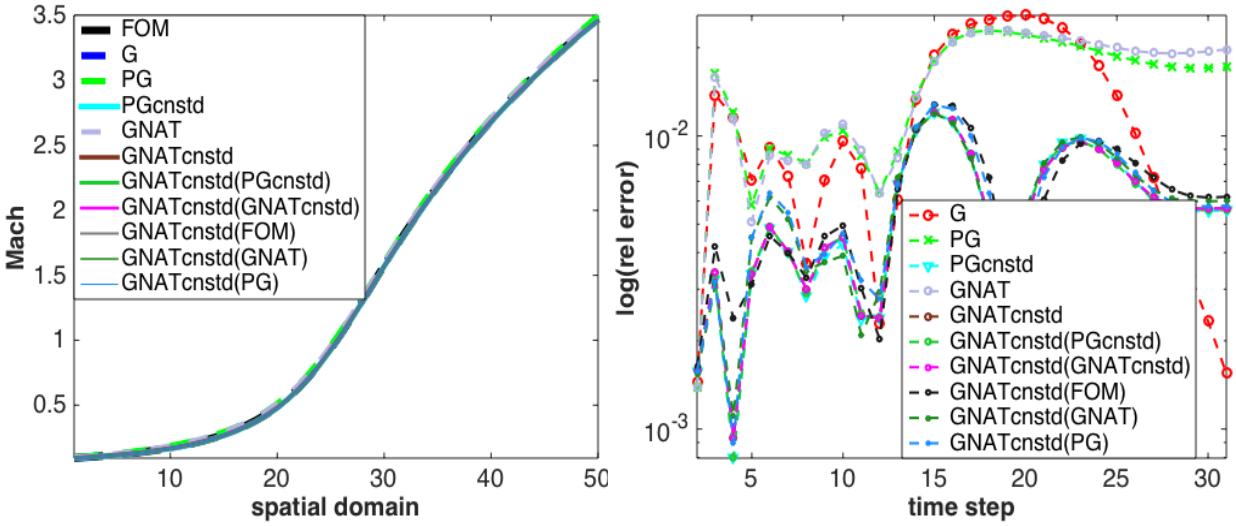
(a) (b)

Figure 4.1: (a) Method specific constraints. (b) Norm of constraints when solving with GNAT method and approximated constraints.

generalized coordinates and compute the constraints using the full state vector like in Method 4.

Method 7 (GNAT_constrained_method). Instead of doing expensive reconstruction and costly computations afterwards as in Method 6, use the values of the state vector at sample points to approximate the constraints. This can be done by precomputing a low-rank approximation basis by using, for example, the POD at an offline stage. The snapshot vectors could be taken from Method 1 or any of the methods Method 3 - Method 6.

The log of the norm of the constraints at each time step for each of the methods listed above is illustrated in Fig. 4.1 (a). Clearly, the previously known methods violate the most important property of the model while the new methods with imposed constraints preserves it. Even though is seems that after hyper-reduction the norm of the constraints is slightly



(a) (b)

Figure 4.2: (a) Mach number at the final time step. (b) Relative error with respect to FOM solution.

worse, note that these plots are on log scale and the values are close to machine epsilon. Further, Fig. 4.1 (b) shows the value of actual constraints when solving the problem with approximated constraints (Method 7). Again the values are logarithmically scaled so it is clear that actual constraint are still small and the conservation laws hold on the full domain. Different snapshots used for the Method 7 are indicated in the parenthesis. The best result was achieved by using snapshots from Method 6 for building the basis.

Next, the Mach number for all methods at the final time step is shown in Fig. 4.2. It is a dimensionless quantity equal to the ratio of the flow velocity past the boundary to the local speed of sound. The Mach number is mainly used to measure how close can the flow be treated as incompressible flow. The results of all methods look very similar to each other including the Mach number from FOM simulation. In order to compare Methods 2-7 we also

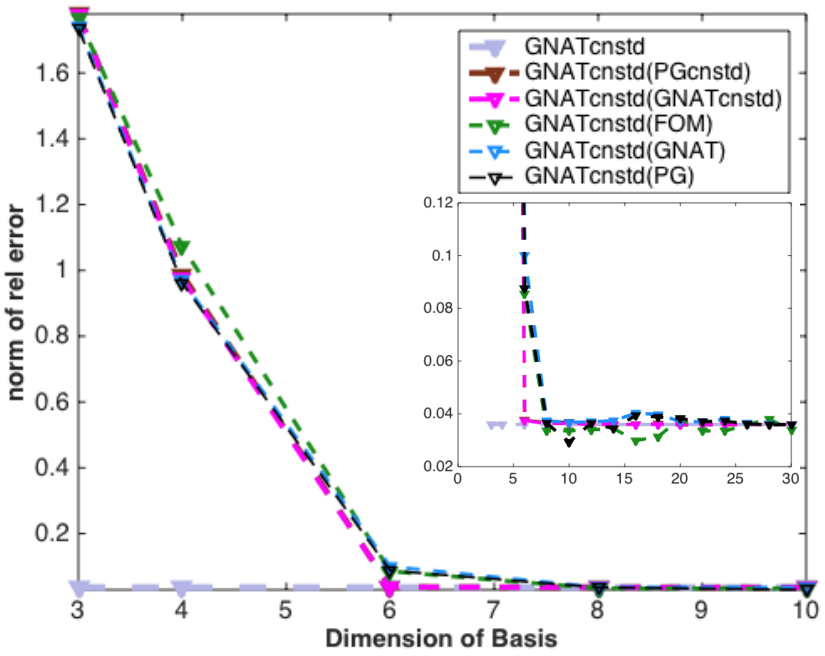


Figure 4.3: Relative error for each method when approximating with different number of basis vectors.

computed the relative error at each time step:

$$E^j_{\text{rel}} = \frac{\|\mathbf{u}^j_{FOM} - \mathbf{u}^j_{\text{Method k}}\|_2}{\|\mathbf{u}^j_{\text{FOM}}\|}$$

where the superscript j denotes the j-th time step and Method k is one of the methods described above. Fig. 4.2 (b) illustrates the logarithm of the relative error from different methods. It is easy to see that the proposed method behaves comparably better than all other methods used before.

The explore how the number of basis vectors in Method 7 affects the results we changed the dimension of the basis. Only three basis vectors capture over 95% of the energy and four vectors are enough to capture 99%. The results of this experiment are shown in Fig. 4.3. Only six basis vectors (capture 99.9% of energy of the system) are enough to achieve a reasonable accuracy.

To examine the effect of multiple domains we conducted several experiments. We varied

the number of subdomains and hence increased the number of constraints. Fig. 4.4 (a) and (b) show results of multiple domains when only $r = 5$ basis vectors for the flux reconstruction are kept. The first figure is the norm of the constraints for each case, whereas the second one is the norm of the relative error $E_{\text{rel}} = [E^1_{\text{rel}}, ..., E^{n_t}_{\text{rel}}]$. On the other hand, the norm of the constraints and norm of the relative error at each time step are shown in Fig. 4.5. One can see that the norm of the relative error is growing with the number of subdomains. The norm of actual constraints also does not decrease with the number of domains. This means we did not get the result we were expecting to get. The conjecture is that we need a balance between the number of constraints and unknown variables, i.e. degrees of freedom. A larger number of constraints make the constraints dominate. By increasing the number of constraints the problem reduces to the original problem on a coarse grid. Currently, research [49] is in progress which attempts to explain this phenomenon in detail and tries to resolve this issue. The same quantities were computed for a basis with $r = 15$ vectors and are shown in Fig. 4.6-4.7.

4.4 Error Bounds

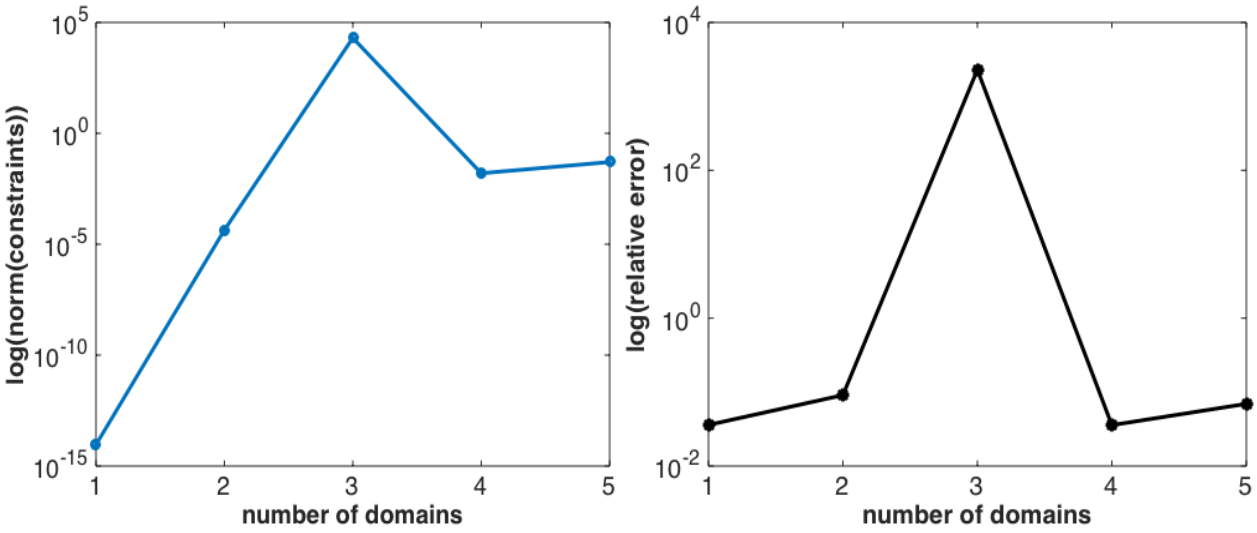
To prove that the method preserves the structure,we bound the quantities that are forced to be conserved. We conducted the following analysis. Recall that in 3D the conservation laws on the subdomains $D_1, ..., D_m$ have the form:

$$\frac{\partial}{\partial t}\int_{\mathbf{D_j}} \rho(x,t)dx + \int_{\mathbf{\Gamma_j}} \rho(x,\tau)u_i(x,\tau)n_i(x,\tau)dxd\tau = \int_{\mathbf{D_j}} \mathbf{Q}_\rho(x,t)dx$$

$$\frac{\partial}{\partial t}\int_{\mathbf{D_j}} \rho(x,t)u_k(x,t)dx + \int_{\mathbf{\Gamma_j}} u_k(x,\tau)\rho(x,\tau)u_i(x,\tau)n_i(x,\tau)dxd\tau = \int_{\mathbf{D_j}} \mathbf{Q}_{\rho u}(x,t)dx, \quad k = 1,2,3$$

$$\frac{\partial}{\partial t}\int_{\mathbf{D_j}} e(x,t)dx + \int_{\mathbf{\Gamma_j}} e(x,\tau)n_i(x,\tau)u_i(x,\tau)dxd\tau = \int_{\mathbf{D_j}} \mathbf{Q}_e(x,t)dx,$$

where Γ_j is the boundary of D_j.



(a) (b)

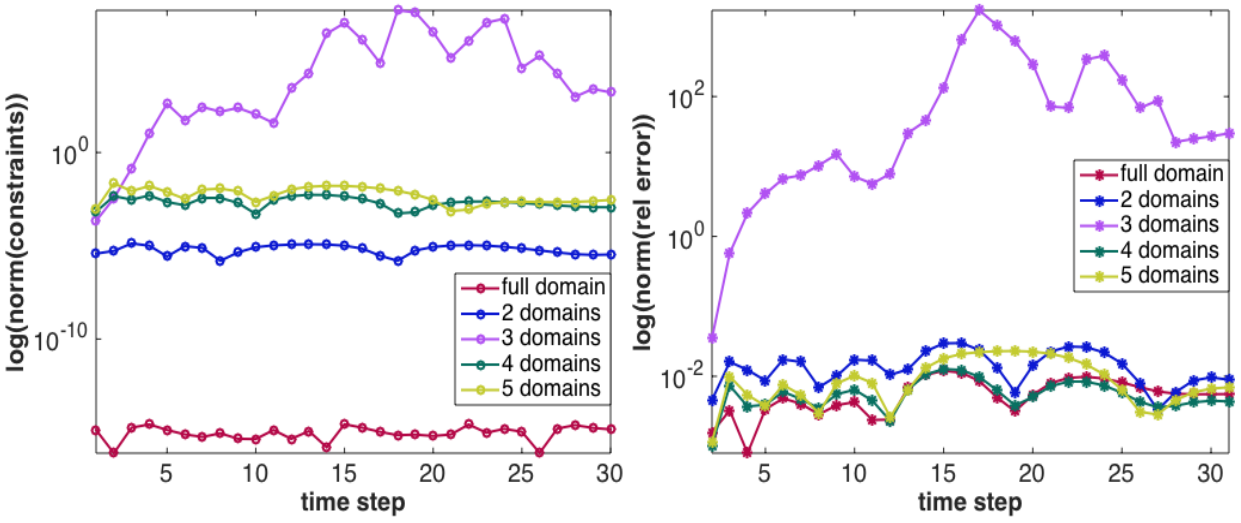
Figure 4.4: (a) Norm of actual constraints for GNAT_constrained method for each multi-domain case, $r = 5$. (b) Norm of the relative error for PG_constrained method for each multi-domain case, $r = 5$.

Discretizing in space with n cells and combining these three equations for each subdomain, we get

$$\mathbf{E}\frac{d\mathbf{u}}{dt} = \mathbf{F}(\mathbf{u}, t),$$

where $\mathbf{E} \in \mathbb{R}^{m \times n_c}$ is the matrix whose (i, j)-th entry is non-zero if corresponding state j is in the i-th subdomain and the value is equal to the volume of the cell (m is the number of subdomains and n_c is a total number of degrees of freedom; for example, $n_c = 3m$ for 1D problems and $n_c = 5m$ for 3D problems), $\mathbf{u} \in \mathbb{R}^{n_c}$ represents ρ, ρu_1 and e in 1D problems. To solve this numerically we need to discretize also in time. A linear k-step scheme leads to the following general formula:

$$\alpha_0^n \mathbf{E}\mathbf{u}^n - \Delta t \beta_0^n \mathbf{F}(\mathbf{u}^n, t^n) + \sum_{j=1}^{k} \alpha_j^n \mathbf{E}\mathbf{u}^{n-j} - \Delta t \sum_{j=1}^{k} \beta_j^n \mathbf{F}(\mathbf{u}^{n-j}, t^n) = 0.$$



(a) (b)

Figure 4.5: (a)Norm of the relative error for PG_constrained method for multi-domain case at each time step, $r = 5$. (b) Norm of constraints at each time step.

Applying the GNAT method with approximated constraints and denoting the approximation of the solution by $\tilde{\mathbf{u}}^n$ and the gappy approximation of $\mathbf{F}$ by $\tilde{\mathbf{F}} = \mathbb{P}\mathbf{F}$, where $\mathbb{P}$ is the gappy projection operator, we get

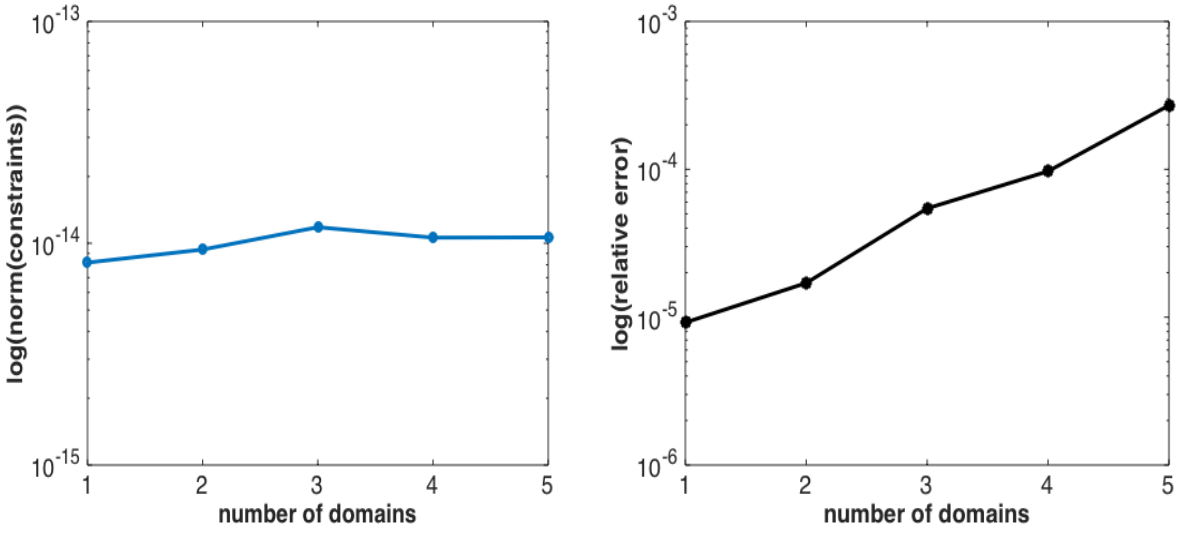
$$\alpha_0^n \mathbf{E}\tilde{\mathbf{u}}^n - \Delta t \beta_0^n \tilde{\mathbf{F}}(\tilde{\mathbf{u}}^n) + \sum_{j=1}^{k} \alpha_j^n \mathbf{E}\tilde{\mathbf{u}}^{n-j} - \Delta t \sum_{j=1}^{k} \beta_j^n \tilde{\mathbf{F}}(\tilde{\mathbf{u}}^{n-j}) = 0. \tag{4.18}$$

For simplicity we skipped the second argument of function $\tilde{\mathbf{F}}$. For the real constraint $\mathbf{u}_\star^n$ we get the following formula:

$$\alpha_0^n \mathbf{E}\mathbf{u}_\star^n - \Delta t \beta_0^n \mathbf{F}(\mathbf{u}_\star^n) + \sum_{j=1}^{k} \alpha_j^n \mathbf{E}\mathbf{u}_\star^{n-j} - \Delta t \sum_{j=1}^{k} \beta_j^n \mathbf{F}(\mathbf{u}_\star^{n-j}) = 0. \tag{4.19}$$

Theorem 4.4.1. *If* $\mathbf{F}$ *is Lipschitz with constant* $\mathbf{M}$ *for each of the conserved quantities:*

$$\|\mathbf{F}(y) - \mathbf{F}(x)\| \leq \mathbf{M}\|\mathbf{E}(y - x)\|$$



(a) (b)

Figure 4.6: (a)Norm of the relative error for PG_constrained method for each multi-domain case, $r = 15$. (b) Norm of actual constraints for GNAT_constrained method for each multi-domain case, $r = 15$.

and $0 < |\alpha_0^n| - \Delta t \mathbf{M} |\beta_0^n|$*, then the following holds:*

$$\|\mathbf{E}(\mathbf{u}_\star^n - \tilde{\mathbf{u}}^n)\| \leq \sum_{j=0}^{k} \epsilon_j^n \| (\mathbb{I} - \mathbb{P})\, \mathbf{F}(\tilde{\mathbf{u}}^{n-j})\| + \sum_{j=1}^{k} \gamma_j^n \|\mathbf{E}(\mathbf{u}_\star^{n-j} - \tilde{\mathbf{u}}^{n-j})\|.$$

for some constants ϵ_j^n *and* γ_j^n*.*

Proof. Subtracting the equation (4.18) from the equation (4.19) and using the triangle inequality for the norms, we get:

$$|\alpha_0^n| \|\mathbf{E}(\mathbf{u}_\star^n - \tilde{\mathbf{u}}^n)\| \leq \Delta t |\beta_0^n| \|\mathbf{F}(\mathbf{u}_\star^n) - \tilde{\mathbf{F}}(\tilde{\mathbf{u}}^n)\| + \sum_{j=1}^{k} |\alpha_j^n| \|\mathbf{E}(\mathbf{u}_\star^{n-j} - \tilde{\mathbf{u}}^{n-j})\|$$
$$+ \Delta t \sum_{j=1}^{k} |\beta_j^n| \|\mathbf{F}(\mathbf{u}_\star^{n-j}) - \tilde{\mathbf{F}}(\tilde{\mathbf{u}}^{n-j})\|$$

Adding and subtracting $\mathbf{F}(\tilde{\mathbf{u}}^n)$ or $\mathbf{F}(\tilde{\mathbf{u}}^{n-j})$ inside the norm, replacing $\tilde{\mathbf{F}}$ by $\mathbb{P}\mathbf{F}$ and again

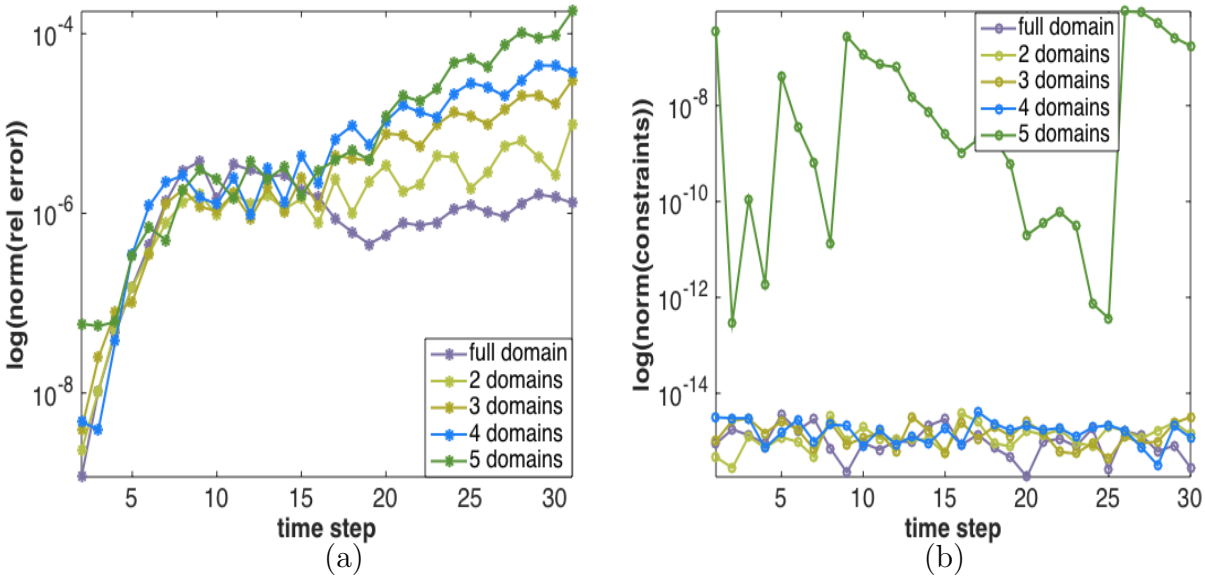


Figure 4.7: (a)Norm of the relative error for PG_constrained method for multi-domain case at each time step, $r = 15$. (b) Norm of actual constraints for GNAT_constrained method at each time step, $r = 15$.

using triangle inequality, we get

$$
\begin{aligned}
|\alpha_0^n| \|\mathbf{E}(\mathbf{u}_\star^n - \tilde{\mathbf{u}}^n)\| \leq \Delta t |\beta_0^n| \left(\|\mathbf{F}(\mathbf{u}_\star^n) - \mathbf{F}(\tilde{\mathbf{u}}^n)\| + \|\mathbf{F}(\tilde{\mathbf{u}}^n) - \mathbb{P}\mathbf{F}(\tilde{\mathbf{u}}^n)\| \right) \\
+ \Delta t \sum_{j=1}^{k} |\beta_j^n| \left(\|\mathbf{F}(\mathbf{u}_\star^{n-j}) - \mathbf{F}(\tilde{\mathbf{u}}^{n-j})\| + \|\mathbf{F}(\tilde{\mathbf{u}}^{n-j}) - \mathbb{P}\mathbf{F}(\tilde{\mathbf{u}}^{n-j})\| \right) \\
+ \sum_{j=1}^{k} |\alpha_j^n| \|\mathbf{E}(\mathbf{u}_\star^{n-j} - \tilde{\mathbf{u}}^{n-j})\|.
\end{aligned}
$$

Since $\mathbf{F}$ is Lipschitz, i.e. $\|\mathbf{F}(y) - \mathbf{F}(x)\| \leq \mathbf{M}\|\mathbf{E}(y - x)\|$, therefore,

$$
\begin{aligned}
\left(|\alpha_0^n| - \mathbf{M}\Delta t |\beta_0^n|\right) \|\mathbf{E}(\mathbf{u}_\star^n - \tilde{\mathbf{u}}^n)\| \leq \Delta t |\beta_0^n| \| \left(\mathbb{I} - \mathbb{P}\right) \mathbf{F}(\tilde{\mathbf{u}}^n)\| + \Delta t \sum_{j=1}^{k} |\beta_j^n| \| \left(\mathbb{I} - \mathbb{P}\right) \mathbf{F}(\tilde{\mathbf{u}}^{n-j})\| \\
+ \sum_{j=1}^{k} \left(|\alpha_j^n| + \mathbf{M}\Delta t |\beta_j^n|\right) \|\mathbf{E}(\mathbf{u}_\star^{n-j} - \tilde{\mathbf{u}}^{n-j})\|.
\end{aligned}
$$

Note that we get a recursive formula. Denote $h^n := |\alpha_0^n| - \mathbf{M}\Delta t|\beta_0^n|$. By assumption of the theorem $h^n > 0$.

Define also $\epsilon_j^n = \dfrac{\beta_j^n \Delta t}{h^n}$ and $\gamma_j^n = \dfrac{|\alpha_j^n| + \mathbf{M}\Delta t|\beta_j^n|}{h^n}$. Using the above recursive formula and these definitions, we get

$$\|\mathbf{E}(\mathbf{u}_\star^n - \tilde{\mathbf{u}}^n)\| \leq \sum_{j=0}^{k} \epsilon_j^n \|\left(\mathbb{I} - \mathbb{P}\right)\mathbf{F}(\tilde{\mathbf{u}}^{n-j})\| + \sum_{j=1}^{k} \gamma_j^n \|\mathbf{E}(\mathbf{u}_\star^{n-j} - \tilde{\mathbf{u}}^{n-j})\|.$$

This completes the proof. □

Chapter 5

OUTLOOK AND CONCLUSION

Even though many fast algorithms and improved mathematical models have been developed during the last decade, as well as new computer hardware that have made the computations more efficient, many computations still remain intensive due to high-fidelity requirements. Reduced order models were developed to make these computations easier. These models try to find the simplest mathematical representation of systems that captures the dominant behavior. Reduced order models have been widely used for a broad range of applications including pattern recognition, computational fluid dynamics [3, 27], Micro-Electro-Mechanical Systems [63], etc. Despite the fact that ROMs significantly reduced the computational cost, it is still expensive to compute solutions mainly because of nonlinear terms. There are several approaches designed to make the computation of nonlinear terms less costly, including EIM [10], DEIM [29] and GNAT [27, 24] algorithms. These algorithms assume that the parameters in a system are fixed. This work considers dynamical systems with parameters that can change their values after some amount of time, i.e. parametrized systems. To make the computations efficient, chapter 2 of this work proposes to build libraries for nonlinearities during the offline stage. This will help to enact the DEIM algorithm and identify spatial sampling points that allow for a fast reconstruction of the nonlinear terms. It is demonstrated that the POD and DIEM methods combined can lead to (i) correct identification of parameter regimes of the dynamical system, (ii) full state reconstruction and (iii) future state prediction of the nonlinear dynamical system. All these tasks are accomplished in an efficient and low-dimensional way leading to significant speedups in computation. Specifically, POD is used to construct the linear and nonlinear libraries representing the dynamics of the system. Then sparse sensor locations capable to

correctly classify the dynamical regime and efficiently reconstructing the solution are found using DEIM algorithm. This method is effective only for those nonlinear dynamical systems that have low-dimensional attractors.

Chapter 3 improves the sparse sampling locations from chapter 2 using a genetic algorithm combined with the DEIM points. It is illustrated that this method achieves nearly optimal locations for reconstruction of nonlinear terms and correct classification. This could be done in an online manner and lead to nearly an order of magnitude improvement in reconstruction error. As initial locations for the genetic algorithm, the points from chapter 2 are taken. Another important fact to note is that for placing m sensors, sometimes it is useful to take DEIM with $m+1$ points and then discard the first point. For the Ginzburg-Landau equation this reduced the error by factor of two. Applying the genetic algorithm on these points can significantly increase the convergence of the algorithm. In general, genetic algorithms have many disadvantages and converge rarely. However, starting close to the optimal solution helps the algorithm to be efficient. DIEM or DEIM+1 initial locations are shown to be reasonable locations for sensor placement and using the genetic algorithm on these locations make it simply force small adjustments in order to maximize the accuracy of approximating the nonlinear terms. It is shown that within only couple of generations the error is reduced substantially. This method out performs many other sparse sampling methods known in the literature in terms of accuracy and ability to classify.

Unfortunately, reduced order models lack robustness and in many applications, particularly in computational fluid dynamics, the ROMs fail to preserve the structure of solution. A commonly used method for solving PDEs that arise from computational fluid dynamics problems are finite volume methods. After discretizing the space into small cells, the method requires the conservation laws to hold on these tiny domains. However, small numerical errors can accumulate and violate the conservation laws on the full spacial domain leading to erroneous solutions. This can be overcome by imposing constraints to the system as demonstrated in chapter 4, and solving a constrained least squares problem. It is shown that this method preserves the structure, and by combining it with GNAT, the hyper-reduction

technique can out perform commonly used methods. This method rapidly reduces the computational time and gives a desired accuracy of the solution. It is worth noting that the number of constraints can be increased by dividing the full spatial domain into subdomains and forcing the conservation laws to hold on these subdomains. In this work, the increase in number of constraints did not prove to be as good as was expected due to imbalance between unknown quantities and number of constraints. However, current research by K. Carlberg and Y. Choi provides a fix this pitfall [49].

BIBLIOGRAPHY

[1] D. Amsallem, J. Cortial, and C. Farhat. *On demand CFD-based aeroelastic predictions using a database of reduced-order bases and models*. AIAA Conference, 2009.

[2] D. Amsallem, M. J. Zahr, and K. Washabaugh. Fast local reduced basis updates for the efficient reduction of nonlinear systems with hyper-reduction. *Advances in Computational Mathematics DOI*, 10(1007):9409–0, February 2015.

[3] David Amsallem and Charbel Farhat. Interpolation method for adapting reduced-order models and application to aeroelasticity. *AIAA journal*, 46(7):1803–1813, 2008.

[4] David Amsallem, Radek Tezaur, and Charbel Farhat. Real-time solution of computational problems using databases of parametric linear reduced-order models with arbitrary underlying meshes. *arXiv preprint arXiv:1506.07153*, 2015.

[5] Patricia Astrid. Fast reduced order modeling technique for large scale ltv systems. In *American Control Conference, 2004. Proceedings of the 2004*, volume 1, pages 762–767. IEEE, 2004.

[6] Patricia Astrid. Fast reduced order modeling technique for large scale ltv systems. In *American Control Conference, 2004. Proceedings of the 2004*, volume 1, pages 762–767. IEEE, 2004.

[7] Z. Bai, T. Wimalajeewa, Z. Berger, G. Wang, M. Glauser, and P. K. Varshney. Low-dimensional approach for reconstruction of airfoil data via compressive sensing. *AIAA Journal*, 53:920–933, 2014.

[8] R. G. Baraniuk. Compressive sensing. *IEEE Signal Processing Magazine*, 24(4):118–120, 2007.

[9] R. G. Baraniuk, V. Cevher, M. F. Duarte, and C. Hegde. Model-based compressive sensing. *IEEE Transactions on Information Theory*, 56(4):1982–2001, 2010.

[10] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera. An 'empirical interpolation' method: Application to efficient reduced-basis discretization of partial differential equations. *C. R. Math. Acad. Sci. Paris*, 339, 2004.

[11] P. Benner, S. Gugercin, and K. Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Review*, 2015.

[12] I. Bright, G. Lin, and J. N. Kutz. Compressive sensing and machine learning strategies for characterizing the flow around a cylinder with limited pressure measurements. *Physics of Fluids*, 25(1271):27102–15, 2013.

[13] B. W. Brunton, S. L. Brunton, J. L. Proctor, and J. N. Kutz. *Optimal sensor placement and enhanced sparsity for classification.* ArXiv e-prints, 2014.

[14] Bingni W Brunton, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Optimal sensor placement and enhanced sparsity for classification. *arXiv preprint arXiv:1310.4217*, 2013.

[15] S. L. Brunton, J. L. Proctor, and J. N. Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, pages 3932–3937, 2016.

[16] 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. App. Dyn. Sys.*, 13:1716–1732, 2014.

[17] Steven L Brunton and Bernd R Noack. Closed-loop turbulence control: Progress and challenges. *Applied Mechanics Reviews*, 67(5):050801, 2015.

[18] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Compressive sampling and dynamic mode decomposition. *arXiv preprint arXiv:1312.5186*, 2013.

[19] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

[20] E. J. Candes and T. Tao. Decoding by linear programming. *Information Theory, IEEE Transactions on*, 51, 2005.

[21] E. J. Candès and T. Tao. Near optimal signal recovery from random projections: Universal encoding strategies. *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006.

[22] Emmanuel J Candès et al. Compressive sampling. In *Proceedings of the international congress of mathematicians*, volume 3, pages 1433–1452. Madrid, Spain, 2006.

[23] Emmanuel J Candes, Justin K Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on pure and applied mathematics*, 59(8):1207–1223, 2006.

[24] 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. *Journal of Computational Physics*, 242:623–647, 2013.

[25] Kevin Carlberg, Matthew Barone, and Harbir Antil. Galerkin v. discrete-optimal projection in nonlinear model reduction. *arXiv preprint arXiv:1504.03749*, 2015.

[26] Kevin Carlberg, Charbel Bou-Mosleh, and Charbel Farhat. Efficient non-linear model reduction via a least-squares petrov–galerkin projection and compressive tensor approximations. *International Journal for Numerical Methods in Engineering*, 86(2):155–181, 2011.

[27] Kevin Carlberg, Julien Cortial, David Amsallem, Matthew Zahr, and Charbel Farhat. The gnat nonlinear model reduction method and its application to fluid dynamics problems. In *6th AIAA Theoretical Fluid Mechanics Conference, Honolulu, Hawaii, June*, volume 2730, pages 2011–3112, 2011.

[28] Kevin Carlberg, Ray Tuminaro, and Paul Boggs. Efficient structure-preserving model reduction for nonlinear mechanical systems with application to structural dynamics. *preprint, Sandia National Laboratories, Livermore, CA*, 94551, 2012.

[29] S. Chaturantabut and D. Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM J. SCI. COMPUT.*, 32:2737–2764, 2010.

[30] Y. Choi, D. Amsallem, and C. Farhat. Gradient-based constrained optimization using a database of linear reduced-order models. *arXiv:*, 1506.07849, 2015.

[31] M. Cross and P. Hohenberg. Pattern formation out of equilibrium. *Reviews of Modern Physics*, 65:851–1112, 1993.

[32] A. E. Deane, I. G. Kevrekidis, G. E. Karniadakis, and S. A. Orszag. Low-dimensional models for complex geometry flows: Application to grooved channels and circular cylinders. *Phys. Fluids*, 3, 1991.

[33] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[34] D. L. Donoho. For most large underdetermined systems of linear equations the minimal 1-norm solution is also the sparsest solution. *Communications on pure and applied mathematics*, 59:797–829, 2006.

[35] Qiang Du and Max Gunzburger. Model reduction by proper orthogonal decomposition coupled with centroidal voronoi tessellation. In *Proc. Fluids Engineering Division Summer Meeting, FEDSM2002-31051, ASME*, 2002.

[36] J. L. Eftang, A. T. Patera, and E. M. Rnquist. *An hp certified reduced-basis method for parameterized elliptic PDEs*. Siam SISC, 2010.

[37] R. Everson and L. Sirovich. Karhunen-loéve procedure for gappy data. *J. Opt. Soc. Am. A*, 12:1657–1664, 1995.

[38] Robert Eymard, Thierry Gallouët, and Raphaèle Herbin. Finite volume methods. *Handbook of numerical analysis*, 7:713–1018, 2000.

[39] J. E. Fowler. Compressive-projection principal component analysis. *IEEE Transactions on Image Processing*, 18(10):2230–2242, 2009.

[40] B. Galletti, C. H. Bruneau, and L. Zannetti. Low-order modelling of laminar flow regimes past a confined square cylinder. *J. Fluid Mech.*, 503:161–170, 2004.

[41] S. Ganguli and H. Sompolinsky. Compressed sensing, sparsity, and dimensionality in neuronal information processing and data analysis. *Annual Review of Neuroscience*, 35:485–508, 2012.

[42] M. Gavish and D. L. Donoho. *The optimal hard threshold for singular values is* $4/\sqrt{3}$. ArXiv e-prints, 2014.

[43] A. C. Gilbert, J. Y. Park, and M. B. Wakin. *Sketched SVD: Recovering spectral features from compressive measurements*. ArXiv e-prints, 2012.

[44] J. I. Gold and M. N. Shadlen. Representation of a perceptual decision in developing oculomotor commands. *Nature*, 404:390–394, 1999.

[45] P. J. Holmes, J. L. Lumley, G. Berkooz, and C. W. Rowley. *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge Monographs in Mechanics. Cambridge University Press, 2012.

[46] H. Hotelling. Analysis of a complex of statistical variables into principal components. *J. Educ. Psychol*, 24:417–441, September 1933.

[47] H. Hotelling. Analysis of a complex of statistical variables into principal components. *J. Educ. Psychol*, 24:498–520, October 1933.

[48] L. M. Jones, A. Fontanini, B. F. Sadacca, P. Miller, and D. B. Katz. Natural stimuli evoke dynamic sequences of states in sensory cortical ensembles. *Proc. Natl. Acad. Sci.*, 104, 2007.

[49] Y. Choi K. Carlberg, S. Sargsyan. Structure-preserving nonlinear model reduction for finite-volume models of conservation laws. in preparation, 2016.

[50] E. Kaiser, B. R. Noack, L. Cordier, A. Spohn, M. Segond, M. Abel, G. Daviller, J. Osth, S. Krajnovic, and R. K. Niven. Cluster-based reduced-order modelling of a mixing layer. *J. Fluid Mech.*, 754:365–414, 2014.

[51] K. Kaspers, L. Mathelin, and H. Abou-Kandil. *A machine learning approach for constrained sensor placement*. submitted, 2015.

[52] K. Kaspers, L. Mathelin, and H. Abou-Kandil. *A statistical learning strategy for flow field estimation from wall-mounted sensors*. submitted, 2015.

[53] J. Kunert, E. Shlizerman, and J. N. Kutz. Low-dimensional functionality of complex network dynamics: Neuro-sensory integration in the caenorhabditis elegans connectome. *Phys. Rev. E*, 89, 2014.

[54] J. N. Kutz. Mode-locked soliton lasers. *SIAM Rev.*, 48:629–678, 2006.

[55] J. N. Kutz. *Data-Driven Modeling & Scientific Computation: Methods for Complex Systems & Big Data*. Oxford University Press, 2013.

[56] G. Laurent, M. Stopfer, R. W. Friedrich, M. I. Rabinovich, A. Volkovskii, and H. D. I. Abarbanel. Odor encoding as an active, dynamical process: Experiments, computation and theory. *Annu. Rev. Neurosci.*, 24:263 – 297, 2001.

[57] Randall J LeVeque. *Finite volume methods for hyperbolic problems*, volume 31. Cambridge university press, 2002.

[58] E. Liberge and A. Hamdouni. Reduced order modelling method via proper orthogonal decomposition (pod) for flow around an oscillating cylinder. *J. Fluids Struc.*, 26:292–311, 2010.

[59] Edward N Lorenz. Empirical orthogonal functions and statistical weather prediction. Technical report, Massachusetts Institute of Technology, Department of Meteorology, 1956.

[60] J. L. Lumley. *Stochastic Tools in Turbulence*. Academic Press, 1970.

[61] X. Ma and G. E. Karniadakis. A low-dimensional model for simulating three-dimensional cylinder flow. *J. Fluid Mech.*, 458:181–190, 2002.

[62] A. Mackey, H. Schaeffer, and S. Osher. On the compressive spectral method. *UCLA CAM Report*, pages 14–33, 2014.

[63] Ali H Nayfeh, Mohammad I Younis, and Eihab M Abdel-Rahman. Reduced-order models for mems applications. *Nonlinear dynamics*, 41(1-3):211–236, 2005.

[64] N. C. Nguyen, A. T. Patera, and J. Peraire. A "best points" interpolation method for efficient approximation of parametrized functions. *Int. J. Num. Methods Eng.*, 73:521–543, 2008.

[65] B. Olshausen and D. Field. Sparse coding of sensory inputs. *Current Opinion in Neurobiology*, 14:481– 487, 2004.

[66] Vidvuds Ozoliņš, Rongjie Lai, Russel Caflisch, and Stanley Osher. Compressed modes for variational problems in mathematics and physics. *Proceedings of the National Academy of Sciences*, 110(46):18368–18373, 2013.

[67] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(7):559–572, 1901.

[68] B. Peherstorfer and K. Willcox. Detecting and adapting to parameter changes for reduced models of dynamic data-driven application systems. *Procedia Computer Science*, 51:2553–2562, 2015.

[69] B. Peherstorfer and K. Willcox. Dynamic data-driven reduced-order models. *Computer Methods in Applied Mechanics and Engineering,"*, 291:21–41, 2015.

[70] B. Peherstorfer and K. Willcox. *Online Adaptive Model Reduction for Nonlinear Systems via Low-Rank Updates*. SIAM Journal on Scientific Computing, to appear, 2015.

[71] GI Petrov. Application of the method of galerkin to a problem involving the stationary flow of a viscous fluid. *Prikl. Matem. Mekh*, 4(3), 1940.

[72] J. L. Proctor, S. L. Brunton, B. W. Brunton, and J. N. Kutz. Exploiting sparsity and equation-free architectures in complex systems. *European Journal of Physics*, 223:2665–2684, 2014.

[73] H. Qi and S. M. Hughes. Invariance of principal components under low-dimensional random projection of the data. *IEEE International Conference on Image Processing*, October 2012.

[74] A. Quarteroni and G. Rozza, editors. *Reduced Order Methods for Modeling and Computational Reduction*. Springer, 2014.

[75] M. Rabinovich, R. Huerta, P. Varona, and V. S. Afraimovich. Transient cognitive dynamics, metastability, and decision making. *PLoS Comput. Biol.*, 4, 2008.

[76] M. Rabinovich, A. Volkovskii, P. Lecanda, R. Huerta, H. D. I. Abarbanel, and G. Laurent. Dynamical encoding by networks of competing neuron groups: Winnerless competition. *Phys. Rev. Lettl.*, 87, 2001.

[77] Philip L Roe. Approximate riemann solvers, parameter vectors, and difference schemes. *Journal of computational physics*, 43(2):357–372, 1981.

[78] Syuzanna Sargsyan, Steven L Brunton, and J Nathan Kutz. Nonlinear model reduction for dynamical systems using sparse sensor locations from learned libraries. *Physical Review E*, 92(3):033304, 2015.

[79] H. Schaeffer, R. Caflisch, C. D. Hauck, and S. Osher. Sparse dynamics for partial differential equations. *Proceedings of the National Academy of Sciences USA*, pages 6634–6639, 2013.

[80] Hayden Schaeffer, Russel Caflisch, Cory D Hauck, and Stanley Osher. Sparse dynamics for partial differential equations. *Proceedings of the National Academy of Sciences*, 110(17):6634–6639, 2013.

[81] E. Shlizerman, J. Riffell, and J. N. Kutz. Data-driven inference of network connectivity for modeling the dynamics of neural codes in the insect antennal lobe. *Front. Neuro.*, 18:1–15, 2014.

[82] N. Trefethen and D. Bau Iii. *Numerical Linear Algebra*. SIAM, Philadelphia, 1997.

[83] D. Venturi and G. E. Karniadakis. Gappy data and reconstruction procedures for flow past cylinder. *J. Fluid Mech.*, 519:315–336, 2004.

[84] W. X. Wang, R. Yang, Y. C. Lai, V. Kovanis, and C. Grebogi. Predicting catastrophes in nonlinear dynamical systems by compressive sensing. *Physical Review Letters*, 106, 2011.

[85] K. Willcox. Unsteady flow sensing and estimation via the gappy proper orthogonal decomposition. *Computers and Fluids*, 35:208–226, 2006.

[86] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(2):210–227, 2009.

[87] B. Yildirim, C. Chryssostomidis, and G. E. Karniadakis. Efficient sensor placement for ocean measurements using low-dimensional concepts. *Ocean Modeling*, 273:160–173, 2009.

[88] Ralf Zimmermann, Alexander Vendl, and Stefan Görtz. Reduced-order modeling of steady flows subject to aerodynamic constraints. *AIAA journal*, 52(2):255–266, 2014.