**Robust and Scalable Projection-based Reduced-order Models for Simulations of Reacting Flows**

by

Christopher R. Wentland

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Aerospace Engineering)
in the University of Michigan
2023

Doctoral Committee:

Professor Karthik Duraisamy, Co-Chair
Assistant Professor Cheng Huang, Co-Chair
Associate Professor Jesse Capecelatro
Professor Krzysztof Fidkowski

Christopher R. Wentland

chriswen@umich.edu

ORCID iD: 0000-0002-8500-569X

# Acknowledgements

It has taken a village to complete this thesis, from friends and family who encouraged me at every point, to labmates and department colleagues who offered advice and camaraderie, and mentors who shaped my research skills. All of these people deserve recognition, so I won't skimp on acknowledgements.

My advisor, Professor Karthik Duraisamy, has been an incredible mentor to me from the moment I walked into FXB. He has been encouraging of my interests and goals, celebratory of my successes, and focused on my growth as a researcher. His curiosity in math and engineering is infectious, and I hope I can emulate a fraction of his passion in my career. He took a chance on a student with almost no research background, and I cannot thank him enough for that.

Prof. Cheng Huang, also an amazing advisor, has been my guide through the thorny world of combustion and a patient recipient of countless questions regarding GEMS. I can't count all the times I popped up at his desk to ask him about theory or code. He has given me so much support in the research we've worked on together, and his development of the MP-LSVT method made this thesis possible. He has been in turns an honest critic, a sympathetic ear, and a great friend.

The two other members of my committee, Prof. Krzysztof Fidkowski and Prof. Jesse Capecelatro, have been immensely generous in reviewing this dissertation. Their feedback throughout the editing and defense process exposed me to new perspectives from which to understand my research and present it to others. Their comments on this thesis have helped shape it into a work that better reflects the nuances of combustion, CFD, and data science.

My labmates past and present have been a constant source of learning, collaboration,

and friendship. I have been lucky to count Achu, Adam, Anand, Aniruddhe, Christian, Daisuke, James, Jiayang (David), Niloy, Noah, Sahil, Shaowu, and Yaser as my coworkers. The FXB 2000 crew, Bernardo, Elnaz, Jasmin, and Malhar, have been wonderful company, with a lot of laughs even while fighting over the thermostat. My former FXB 2006 (a.k.a. "The Hellhole") crew, namely Ayoub, Danny, Eric, and Vishal took me under their collective wing and guided me through my turbulent first years at UM. Without them I most certainly would not have made it through to candidacy, and they made working in a windowless office not only bearable, but fun. Last but most certainly not least, all of the work in this thesis has been assisted by Nick in one way or another. From calming me down after I failed my first combustion exam, to writing PLATFORM, to answering every HPC question I had, to keeping the lab social scene alive with board game nights, he helped keep me sane and steady through five and a half long years. A thesis can't have a co-author, but he's just about the closest thing.

The many UM Aerospace Engineering grad students that I've had the fortune of befriending over the years have been great companions on our shared journey. The largest contingent, from the MDO lab, including Alex, Ali, Anil, Ben, Eytan, Galen, Hannah, Josh, Kleb, Marco, Sabet, and Saja made life a million times more fun over lunches, ultimate frisbee, triathlon training, parties, and company in FXB. Among the other labs, Anthony, Christina, Elliot, Kevin, Miles, Nima, Prince, Ral, and Sebastian regularly brightened my day just seeing each other and chatting around FXB. The wise sages of AERO past, including Fabian, Jacob, John, Kaelan, Krystal, Logan, Ryan, Sam, and Shamsheer were all friends and role models alike, welcoming me to the department and showing me the ropes. From my cohort of PhD students, I'm particularly thankful to Andy, Gary, Pawel, Shivam, and Supraj for helping me through my first year of classes and prelims. I'm the last one to get out, but I couldn't have even made it here without you guys. At the end, I was grateful to co-work with Neil, who kept me accountable as we wrote our respective theses over long hours in the robotics building.

I am entirely indebted to the numerous staff members with UM ARC, the various DOD HPC centers, FXB facilities, and the AERO department for keeping a roof over

my head and the computers running. I'm particularly grateful to David at ERDC for his endless patience and willingness to help at a moment's notice, and to Ruthie for making FXB more cheerful and keeping the grad program running smoothly.

Before coming to Michigan, I was lucky to have several mentors who took the time to teach me about research before I really knew what it entailed. Dr. Mark Nutt and Dr. Casey Trail took me on at Argonne National Lab, even though I knew nothing about nuclear waste, and guided me step by step through my first real research projects. Prof. Yildiz Bayazitoglu, beyond teaching me heat transfer, gave me an opportunity to learn the craft as a teaching assistant and generously helped me apply to grad schools. Prof. Tayfun Tezduyar was the first person to teach me CFD, and patiently sat through hours of my naive questions on the topic, giving advice on applying to grad school and securing my first CFD-related research position. Prof. Makoto Yamamoto (and his entire group, as well), in turn, was kind enough to take me into his lab and give me an incredible experience working and living in Japan.

My Benet friends, Dave, Jack, Sam, and Tarik, were my saving grace during the pandemic. Reconnecting through our weekly video calls gave me something to look forward to and kept my spirits up when the world was going to pot. Being stuck in a tiny apartment wasn't so bad while reminiscing about the dumb stuff we used to get up to.

My friends from Rice supported me all through undergrad and continue to do so even today, especially Annie, Farish, Isaac, Madhuri, Olivia, and Sanjiv. Late nights struggling through PDE, vibrations, and mechanical design homework were easier with such great people around me. I am so thankful for them continuing to cheer me on through my PhD even though we're spread out around the country. I also thank the crazy people I worked with in Eclipse, particularly Andrew, Cole, Elijah, Morgen, Jeremy, Josh, and Sam for inspiring a love of rockets and space in me. Also from Rice, the post-doc who told me I couldn't make it through a PhD program failed to discourage me, and instead made me want to prove them otherwise. This petty spite helped push me through the darker moments of grad school.

Of course, my mom Sheila, my dad Rob, and my sister Kelly have been my rock since day one. They pulled me up from my lowest moments, and are always cheering the loudest when I succeed. Keeping me fed before deadlines, calling to check in and remind me they love me no matter what, and driving to Ann Arbor just to spend time with me, they've kept me going when I didn't think I could go on any longer. Getting to see them more regularly after moving back to the Midwest was a blessing, and I'm already sad to have moved so far away.

Finally, I thank my fiancée, Lauren, for being the light of my life. She suffered through the rollercoaster of my PhD for over four years, putting up with late nights and irritable moods with nothing by love and support. I can say with absolutely certainty that I could not have completed this PhD program without her endless compassion and care. Lauren, living life with you has been a dream-come-true, and I cannot wait to see where the future takes us.

# Table of Contents

# List of Tables

# List of Figures

# Abstract

This thesis investigates the development and application of *projection-based reduced-order models* (PROMs) to mitigate the exorbitant computational cost of high-fidelity numerical simulations of complex systems. Traditionally, PROMs operate by learning a low-dimensional representation of the system state from a small amount of high-fidelity simulation data, projecting the governing equations onto a low-dimensional subspace, and evolving the resulting system on the low-dimensional manifold inexpensively. Although PROMs have been applied in industrial problems, successes are largely restricted to linear and elliptic/parabolic systems such as those describing solid mechanics, heat transfer, and diffusion-dominated flows. For advection-dominated and highly non-linear flows, classical PROMs are found to be deficient in reliably generating robust and accurate *predictions* of flows featuring multi-scale and multi-physics phenomena. Further, PROMs of non-linear systems require *hyper-reduction* methods to achieve significant computational cost savings, and such approaches have yet to be rigorously investigated in stiff and chaotic flow problems.

The methods developed in this thesis are motivated by and applied to complex reacting flows, with a particular emphasis on rocket combustion. Despite decades of research in the design of rocket combustors, their development often requires long and expensive experimental test campaigns to ensure operational performance and safety. High-fidelity numerical simulation of high-pressure turbulent combustion requires enormous computational power which is not affordable in industrial design settings. This work advances the construction of accurate, robust, and scalable PROMs for this challenging class of problems.

This work evolves from the recent model-form preserving least-squares with variable

transformation (MP-LSVT) method, which derives the ROM using a least-squares procedure, and simulates the dynamics with respect to an alternative state representation. This approach exhibits greatly improved accuracy and stability over classical PROM methods for reacting flow simulations. The method is derived here, and a number of critical nuances – which are often not well-documented in the literature – are detailed at length, including data preparation, extensions to non-linear manifolds, least-squares weighting, hyper-reduction regression basis and sample mesh construction, and online model adaptation. These techniques are then applied to a number of challenging multi-scale and reacting flow systems.

First, an open-source framework for implementing novel ROM approaches for 1D reacting flows, named the Prototyping Environment for Reacting Flow Order Reduction Methods (PERFORM), is outlined. Developed exclusively by the author, this package is used to conduct a critical examination of several novel neural network ROM approaches is conducted for a model premixed flame case. This approach exhibits utility in enabling accurate representations of flows characterized by sharp gradients and propagating waves. Further, non-intrusive neural network ROM approaches (i.e. those which do not require access to the numerical solver) are shown to greatly outperform comparable classical intrusive PROM methods. However, analysis of the cost of training these neural network models reveals that they are hardly an efficient solution compared to equivalent linear approximations.

Moving to more practical flows, scalable hyper-reduced PROMs are developed within a massively parallel compressible reacting flow solver, and demonstrated for a 2D transonic flow over an open cavity, a 3D single-element rocket combustor, and a 3D nine-element rocket combustor. The effects of the sample mesh and hyper-reduction approximation dimension on PROM performance, computational cost savings, and load balancing is probed at length. Recent algorithms for selecting sample points are shown to generate accurate models, while some methods used in the classical PROM literature are shown to generate unstable solutions. Over three orders of magnitude computational costs savings, while retaining simulation accuracy, are realized for the single-element rocket

combustor case. Further, the nine-element rocket combustor experiment represents the largest and most physically-complex system investigated to date, involving extreme stiffness and nearly 250 million degrees of freedom. However, the ultimate goal of PROMs is truly generalizable, predictive models. To this end, analyses are conducting for a recent adaptive PROM approach, revealing that future-state and parametric predictions are achievable for very long time horizons. Much work remains to be done to make such adaptive PROMs robust and efficient for such large-scale and extremely complex systems.

Finally, best practices for the development and application of PROMs are documented. Insight is provided on centering and scaling high fidelity data snapshots. Additionally, simple least-squares-based residual weighting approaches are shown to promote long-time PROM accuracy for problems exhibiting extreme scale disparities. These guidelines will hopefully inform future PROM practitioners and help mitigate costly trial-and-error efforts. In summary, this work shows that novel projection-based reduced-order models offer an attractive means to leverage an ever-growing ecosystem of numerical and experimental data to generate accurate and low-cost solutions.

# Chapter 1

# Introduction

The founding motivation for this work stems from a need to better predict combustion instabilities in rocket combustion, and from there expanded to explore more fundamental issues in applying data-driven order-reduction techniques to multi-scale and reacting fluid flows. Although work in this thesis approximates the multi-phase nature of liquid-propellant rocket combustion with pure gas-phase combustion, it is critical to understand the context of modern rocket engine design and the future trajectory of modeling efforts in this field. This discussion begins with a brief primer on rocket engines and the destructive combustion instabilities often encountered in their design. General issues of extreme computational cost in simulating reacting flows are then addressed, along with contemporary attempts to alleviate such costs with data-driven models.

## 1.1 Historical Context and Motivation

### 1.1.1 Combustion Instabilities in Rocket Engines

In the 21st century, the ability to launch objects into space is critical to the function of national governments, corporations, and scientific research organizations alike. Space is now increasingly accessible to all countries and inseparable from the everyday lives of people around the world. The impact of space vehicles, probes, and satellites is experienced in both mundane tasks and life-critical operations. Communication satellites provide data transmission for cell phones, and GPS satellites provide real-time location

services. Cameras pointed at the Earth from satellites help track storms and wildfires, and observe dangers to the environment as glaciers recede and rainforests are cut. Space stations provide a testing ground for life to thrive beyond Earth, and enable discoveries in medicine, agriculture, and engineering to improve life on Earth. Space telescopes peer into the depths of the universe to help better understand the cosmic soup of galaxies, suns, and planets. These extra-planetary objects accomplish incredible things, and getting them into space requires similarly incredible feats of engineering.

Launching these massive objects (many satellites weigh over one ton) into space is accomplished exclusively by chemical rocket propulsion. No other practical means of propulsion is able to generate the force required to lift payloads hundreds of miles and reach velocities greater than 15,000 miles per hour. Rocket engines accomplish this by reacting chemicals which generate high-temperature and high-pressure gases in a combustion chamber, forcing these gases through a nozzle which accelerates them to supersonic speeds, and then ejecting the gases behind the engine. This ejection of high-speed gas imparts enormous force, or thrust, on the rocket: the nine Merlin 1D engines of the SpaceX Falcon rocket generate 1.71 million pounds-force (7.686 MN) of thrust [1]. Figure 1.1 illustrates these elements of the thrust chamber. While air-breathing jet engines, like those which power commercial aircraft or military fighter planes, are capable of generating moderate thrust levels (the General Electric GEnx-1B76 can generate over 76 thousand pounds-force, or 339 kN [2]), they ultimately suffer from an unfortunate lack of air in space.

Chemical rocket propulsion comes in two dominant forms: solid and liquid propellants. Solid propellants are composed of combustible reactants cured into a solid fuel grain. When fired, it sustains a highly-energetic reaction converting solid fuel into hot gases until the grain is consumed. Most solid rocket motors can be stored at room temperature for long periods of time, can be ignited instantly, and require few to no auxiliary systems to operate. As such, they play a significant role in rocket-propelled munitions such as rocket artillery and missiles. However, solid propellant reactions are extremely difficult to control, as combustion cannot be stopped once it begins. Further, the reaction of solid

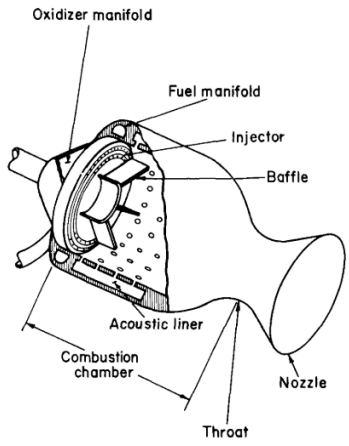



**Figure 1.1:** Cross section of liquid bipropellant thrust chamber (NASA [5]).

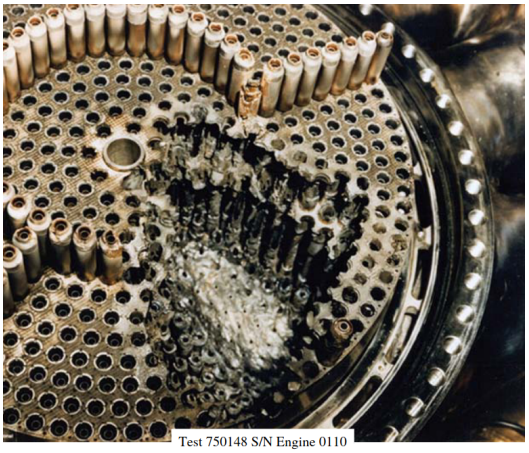



**Figure 1.2:** Failure of Space Shuttle main injector assembly (Goetz and Monk, NASA [6]).

propellants tends to be far less efficient than reactions between many liquid or gaseous propellants, achieving lower specific impulse values [3].

Due to the limitations of solid propellants, liquid propellants are, by far, the preferred method to power space launch vehicles. They enable much more fine-tuned thrust control: liquid engines have been designed to shut down and restart multiple times (e.g., the Apollo Lunar Module reaction control system), and the thrust can be actively throttled to adjust the rocket's trajectory (e.g., SpaceX Falcon purported to throttle down to 57% of maximum thrust [1]). There are two primary variants of liquid propellants: bipropellants and monopropellants. Bipropellants involve a liquid oxidizer (e.g. oxygen, dinitrogen tetroxide) and liquid fuel (e.g., hydrogen, kerosene, monomethylhydrazine), which are mixed and atomized into a gas before combustion. Monopropellants involve a single propellant (e.g., hydrazine) which decomposes in a highly exothermic reaction when it comes in contact with a catalyst. The selection of liquid propellants largely depends on the engine's mission (e.g., launch, station-keeping, orientation control) and a balance of combustion efficiency (liquid hydrogen and oxygen is the most efficient), cost (kerosene is cheaper than hydrogen to manufacture and store [4]), and safety risk (monomethylhydrazine is highly toxic). In this thesis, only systems of non-hypergolic bipropellants are examined.

Despite many benefits, the design, manufacturing, and operation of liquid-propellant rocket engines are not without major complications. They require complex pump sys-

tems to pressurize the propellants and intricate injection systems to mix and atomize the liquid to promote efficient combustion. Further, some propellants must be stored at temperatures below -400 °F (-240 °C) [7]. Even leveraging extensive engineering design practices honed over decades of rocket engine development programs, producing a successful engine is a dangerous process requiring frequent testing and rigorous certification. One design flaw which has troubled engine designers for decades is the possibility of *combustion instabilities*.

Combustion instability broadly refers to organized vibrations in the liquid propellants, combustion gases, or solid structure which can degrade engine performance, and potentially damage or complete destroy the engine. There are three general categories of such instabilities: propellant feed instabilities ("chugging"), intermediate instabilities ("buzzing"), and resonant combustion ("screaming"). As these colloquial names may imply, these are listed in order of increasing frequency of the vibration and relative danger to the integrity of the engine. Chugging instabilities are low-frequency, low-amplitude oscillations generated by irregular injection of propellants, leading to the accumulation of unburnt reactants and subsequent explosive reaction. Buzzing instabilities often result from the intermediate-frequency coupling of acoustic waves in the reacting gases and the structural components of the engine assembly. Chugging and buzzing are not always a threat to the engine's integrity, but leads to decreased, irregular performance.

The final instability, screaming, refers to a highly destructive feedback mechanism between acoustic waves in the combustion chamber and the unsteady heat release caused by the reaction of the propellants. The reaction generates heat, which raises the pressure of the reacting gases, which in turn encourages a more vigorous reaction, which generates more heat, and so on. Depending on the geometry of the combustion chamber, the composition, temperature, and flow rate of the propellants, and the ability of the engine's structure to dissipate acoustic energy, this feedback loop may amplify the acoustic waves to enormous amplitudes. The intensity of this process is illustrated by CH* chemiluminescence images captured by Orth *et al.* [8], displaying the immense difference in heat release in a rocket combustor experiencing stable versus unstable combustion. These large

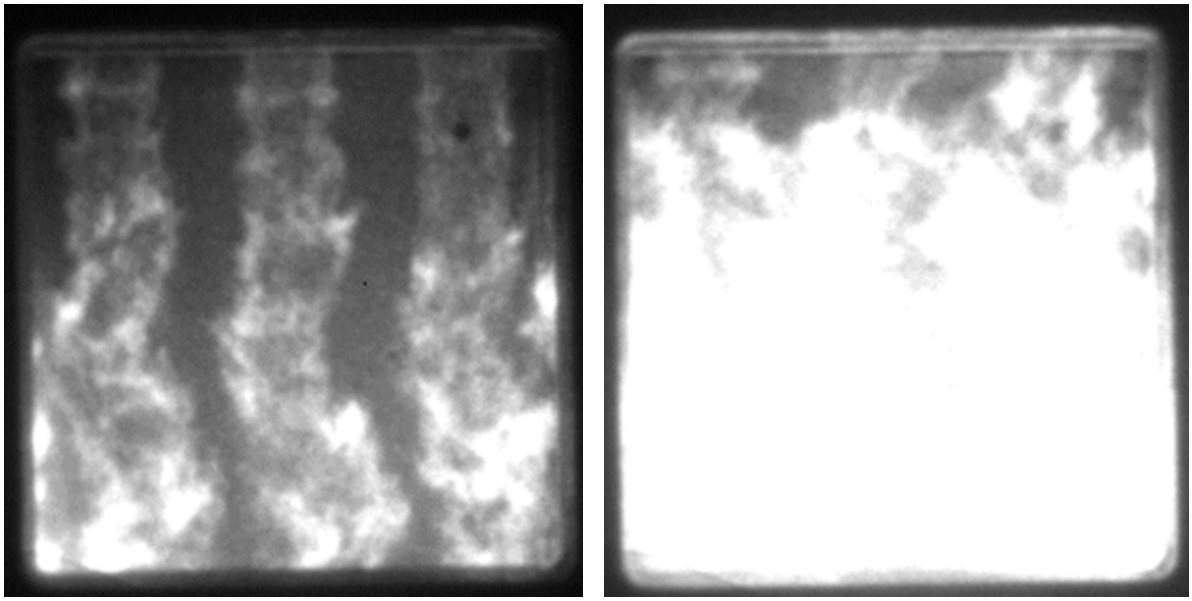


**Figure 1.3:** CH* chemiluminescence photos of multi-element rocket combustor in stable combustion (left) and unstable combustion (right) (reproduced with permission from [8]).

pressure waves and the accompanying high heat-release rates may shake the engine apart, or burn through the combustion chamber walls or propellant injection mechanisms. An example is shown in Fig. 1.2, where vibrations in the Space Shuttle main engine cracked oxidizer injectors and burned through the injector face, to catastrophic effect. Unlike chugging and buzzing, which can often be accommodated for or easily fixed, screaming is nearly impossible to predict and there exist no sure methods of preventing it. Many possible mechanisms have been proposed as the cause of high-frequency combustion instabilities, but the extreme complexity of the system (multi-phase, high-pressure, turbulent combustion) has eluded any strong consensus on the topic.

Screaming has been catalogued in engine development dating back to the 1950's in the American Thor and Atlas missile programs, and continued to plague development programs throughout the prolific USA-USSR Space Race era and beyond. A classic example of an extreme engineering challenge caused by combustion instabilities is the development of the Rocketdyne F-1 engine, which to this date remains the most powerful American liquid-propellant engine to launch. In the early stages of its development, almost every single engine test ended abruptly with the onset of dangerous combustion instabilities. What proceeded was the long, arduous Project First program, assembled to eliminate sustained combustion instabilities, lasting from October 1962 until the flight certification

of the propellant injector in November 1965 [9]. The program amounted to iterative modifications to propellant spray patterns, propellant injector port arrangements, and patterns of metal baffles between sections of the injector plate. Illustrations of such baffles can be seen in Fig. 1.1. Many of these tests failed to eliminate combustion instabilities, and Project First ultimately accounted for approximately 2,000 of the ∼3,000 full-scale engine tests conducted for the F-1 development program [10]. Although the project was ultimately a success, it was an extremely labor- and time-intensive effort. To this day, similar iterative, *ad hoc* approaches remain the preferred method of eliminating combustion instabilities. Although advances have been made in the construction of acoustic dampers [11], they are not a guaranteed solution.

A number of analytical methods have been formulated in attempts to predict the onset of sustained high-frequency combustion instabilities and avoid costly experimental campaigns. However, due to the extremely non-linear physics of coupled combustion and fluid flow dynamics, paired with the complex geometry and propellant injection configurations in rocket engines, these methods often rely on linearizations or oversimplifications of the rocket engine physics. They are often incapable of incorporating the effects of possible damping mechanisms such as baffles and acoustic liners, or possible driving mechanisms such as irregular distribution of propellants [12]. Further, those methods which do yield accurate predictions of instabilities are usually valid for relatively small-amplitude acoustic waves or unable to predict a limit cycle [13]. Although significant effort has been taken to develop accurate models for gas turbine combustors in recent decades (e.g., [14]), the contemporary analytical combustion instability theory for rocket combustors is comparably lacking.

In the modern era, in which private companies vie for a lucrative heavy-launch market and NASA takes a relative back seat, publicly-available data on engine development programs are scarce. However, there have been occasional hints that combustion instabilities continue to haunt engine design programs well into the 2020's [15]. With the understanding that analytical methods of predicting instability fall short in their generalizability, and extensive test campaigns may incur excessive costs, the use of numerical simulations

in the modeling of reacting fluid flows is now discussed, with an emphasis on applications to rocket combustors.

### 1.1.2 Simulation of Reacting Fluid Flows

Computer simulations of physical processes have existed since the World War II era, during which the development of early computers and researchers' access to them expanded greatly in pursuit of the atomic bomb. Broadly, the purpose of numerical simulations is to approximately reconstruct a physical process which has no tractable analytical solution and is either impossible or cost-prohibitive to experimentally measure. Rocket engines are ideal candidates for simulation due to their physical complexity and high cost of manufacturing and testing. This process is broken into its two major components: the modeling of fluid flows and the modeling of chemical reactions.

Numerical fluid modeling, or computational fluid dynamics (CFD), is based in the modeling of the Navier–Stokes equations (detailed in Sec. 2.1), a set of partial differential equations which dictate the conservation of mass, momentum, and energy of a fluid flow. Except in special circumstances where a system cannot be treated as a continuum (such as nanofluidics and rarefied flows), the Navier–Stokes equations are a reliable description of fluid flows. However, analytical solutions for these equations are limited to extremely simple scenarios, such as Poiselle pipe flow or the Taylor–Greene vortex, where elements of the Navier–Stokes equations are neglected or the system exhibits some symmetry. In all other scenarios, certainly in any application of practical significance, analytical solutions are impossible and numerical methods must be used to obtain approximate solutions. In general, these methods amount to discretizing the spatial domain via methods such as finite-difference, finite-volume, finite-element, or discontinuous Galerkin methods, and marching the system forward in time with temporal discretization methods. These simulations have been successfully employed in the design and analysis of spacecraft [16], automobiles [17], pumps [18], and wind turbines [19], among many other applications. However, these models generally grow rapidly in cost with their spatial and temporal resolution. Many important physical phenomena, such as turbulence, require extremely

small spatio-temporal resolutions to capture accurately. This necessitates the use of powerful supercomputers, which are inaccessible to all but select government and academic researchers, to compute direct numerical simulation (DNS) models which resolve all characteristic spatio-temporal scales. Most CFD practitioners must instead rely on low-fidelity approximations, such as Reynolds-averaged Navier–Stokes (RANS) models or large eddy simulations (LES) [20, 21]. These models describe the effect of unresolved physics on the resolved system in order to obtain a computationally-tractable solution. While inexpensive relative to DNS, these models often fail to accurately recreate important fluid behavior such as flow separation [22] or correctly predict near-wall flow statistics [23]. Such low-cost models are useful as a first-order approximation, but they cannot match DNS in its accuracy and generalizability.

Chemical reaction modeling is also a well-studied field. Combustion, or any exothermic reaction of a fuel (often a hydrocarbon) with an oxidizer (often oxygen), is of particular interest due to its importance in generating electricity in gas turbines, mechanical power in piston and jet engines, and thrust in rocket engines. Chemical reactions are, at their core, interactions between individual molecules governed by the principles of quantum mechanics. There is an enormous number of molecules in practical chemical reaction systems (there are $1.882 \times 10^{22}$ molecules in just one gram of oxygen), and it is impossible to model their individual behaviors. As such, statistical or empirical models are used to describe the bulk behavior of chemicals in a mixture. Many detailed chemical reaction mechanisms have been developed which may include dozens of molecules and hundreds of reactions (e.g., the 52 species, 325 reaction GRI-Mech 3.0 mechanism for methane combustion [24]). These are, in general, not computationally tractable for simulations of large reacting systems. Reduced mechanisms seek to condense these complex processes with fewer chemical species and reactions, often using automated processes to generate accurate approximations with fewer than 20 species and reactions (e.g., [25]), but sometimes fail to match experiments under certain conditions (e.g., very high temperature or equivalence ratio [26], large hydrocarbons [27]).

The combination of these two fields, reacting turbulent flows, is less well-understood

than its component fields for a variety of reasons. The characteristic spatio-temporal scales of chemical reactions are smaller even than those of turbulence, placing well-resolved simulations for all but fairly simple canonical problems well out of reach of modern supercomputers. For reference, recent well-resolved simulations of a rectangular premixed flame, running on 3,072 GPUs with advanced combusting flow software, still requires over $\sim$4 seconds to compute a single time step [28]. Additionally, the highly non-linear reaction source terms produce extremely stiff systems, making robust numerical solutions challenging [29]. Finally, relevant reacting flows often involve complex physics such liquid sprays [30], radiative heat transfer [31], and solid soot particulates [32]. Accurately modeling combustion in the presence of turbulence has been a particular difficulty for decades, but is extensively researched thanks to the importance of turbulence in efficient, reliable, and low-emission combustion [33]. Many approximate models, including flamelet/progress variable [34], thickened flame [35], and transported PDF models [36], have been successfully applied to many practical combusting flow systems. However, many of these models struggle to accurately predict important features (such as local ignition and extinction [37]) in all flow configurations (such as partially-premixed flames [38]).

The above paragraphs illustrate two important points. First, the well-resolved simulations of combusting flows are extremely computationally expensive, largely driven by small, disparate, coupled spatio-temporal scales of turbulence and reactions, as well as by complex and stiff chemical kinetics. Second, although low-cost models such as RANS/LES, reduced chemical mechanisms, or flamelet models are constantly improving, they are often not generalizable to all circumstances and may not achieve significant cost reduction to be industrially-viable. Although advances in high-performance computing hardware and software continue to open new possibilities of bigger and faster computational physics, high-fidelity simulation of the complex multi-scale processes observed in rocket combustors remain oppressively expensive. Certainly, in the context of engineering design, in which many parameters of the combustor geometry, operating conditions, and propellant composition must be iteratively changed, a simulation runtime measured in

weeks or months on massive supercomputers cannot be justified. In such a situation, performing an experiment is likely equally or less expensive, and generates much more useful operational data. This has stimulated inquiries into a third path, which seeks to learn learn low-cost surrogate models of complex systems from a small number of experiments or high-fidelity simulations. Such approaches can be broadly categorized as *data-driven modeling*.

## 1.2 Data-driven Reduced-order Modeling

In recent years, the term "data-driven" has become a veritable buzzword in many fields, including medicine, finance, agriculture, logistics, and engineering. Although designs and applications vary widely, data-driven methods generally follow the same core process:

1. Gather real-world information about an extremely complex system which humans cannot parse efficiently.

2. Feed this data into an algorithm which learns to map the input data to meaningful output targets which are easily parsed by humans.

3. Use the model to make useful predictions on new, unseen data.

Such models are crucial in situations where a process cannot be readily queried (patients cannot be forced to enroll in a clinical trial), or the process must be queried many times (personalized ads must be tailored for millions of search engine users). The design of rocket combustors exhibits both challenges in some of their most extreme forms. Experimental test articles are costly to manufacture and hazardous to test. Further, rocket engines must be tested repeatedly to certify performance and safety standards. Developing reliable, approximate models that can replace (some of) these experiments has the potential to accelerate rocket design programs and lower their cost significantly. This idea reflects a broader effort in computational physics which seeks to address so-called *many-query problems*, in which a model must be evaluated many times to ensure the targeted outcome is confidently estimated. Such problems include optimization, uncertainty

quantification, rare event prediction, and control, all crucial element in developing robust solutions to modern engineering problems. However, the cost of existing lower-fidelity physics models, such as single-phase models for turbulent spray combustion, is often still prohibitively high and precludes their use in many-query operations. This motivates the use of data-driven modeling to discover new, inexpensive models which can be queried at a fraction of the cost of standard models.

Data-driven modeling is not new. Linear regression has been used extensively for over two centuries, and many statistical models still used regularly in modern times (e.g., k-means clustering, principal component analysis) were formulated over 50 years ago [39]. What has changed in recent years is the exponential growth in computing power, data storage, and data collection mechanisms available to data scientists, as well as increased access to publicly available datasets made possible by the Internet. This same phenomenon has occurred in engineering research, where huge experimental and DNS/LES datasets can be easily exchanged, standardized data formats allow for quick data ingestion, and open-source and hardware/software interoperability programs permit rapid verification and validation of numerical experiments. This wealth of high-fidelity data has enabled a rapid expansion of research in methods which learn low-cost models of complex physical processes, often referred to as *reduced-order models*, (ROMs) where "order" refers to the dimensionality of the problem.

The phrase reduced-order model (sometimes *reduced models* or *model order reduction*) has taken on two distinct meanings in the fluid mechanics and combustion literature. The first type of ROM refers to models which are derived by simplifying complex physics with physically-meaningful (but lower-cost) approximations, or by fitting compact analytical equations to empirically-observed patterns. Examples of the former (which are not necessarily data-driven) for general fluid flows include axisymmetric models of three-dimensional systems, while the latter includes polynomial models of gas thermodynamic and transport properties [40]. Such approaches are prevalent throughout the combustion community. Methods of generating chemical reaction mechanism often rely on sensitivity analyses to remove candidate species and reactions [41] or assume reaction process order-

ing [42]. The analysis of combustion instabilities often represents the combustor as an assembly of acoustic elements which transmit acoustics according to transfer functions; modeling the flame transfer function is extremely difficult, but enables vast simplifications of the system [43]. Such acoustic models may be paired with the Euler equations to provide a low-cost approximation of the non-linear combusting gas dynamics [44]. The immensely complex process of two-phase combustion in the presence of turbulence involves a host of modeling approximations in representing liquid spray breakup, dispersion, mixing, evaporation, and reaction, often derived from empirical observations [45]. For such ROMs, the result of order reduction (relative to, perhaps, DNS or molecular reaction models) is a secondary consequence of approximating physical processes. In this sense, this type of model is henceforth referred to instead as a *reduced-physics model*. While these play a critical role in developing computationally-tractable models, they are not the focus of this work.

The second type of ROM, central to this work, refers to models which learn a mapping from a low-dimensional representation of the system state to the full-dimensional state, and provide a means of evolving this low-dimensional representation in time. Figure 1.4 illustrates a simple example of this process, showing the evolution of a reduced-order state in two dimensions and the corresponding full-order state in three dimensions. At their core, these methods attempt to recreate the behavior of the full-dimensional system, not that of a physically-simplified surrogate. The low-dimensional state is often a mathematical construct without particular physical significance. In this case, order reduction is a direct consequence of the mapping from low- to high-dimensional states. As such, *reduced-order model* seems more appropriately applied to these models, and the acronym ROM is used to refer to them exclusively in this thesis.

All ROMs begin by defining the mapping from a low-dimensional reduced space to the high-dimensional physical space. This mapping may be categorized broadly as linear or non-linear. Linear methods reconstruct the high-dimensional state from a linear combination of a small number of basis vectors. The scalar coefficients of this linear combination represent the low-dimensional state. Prominent methods for computing the

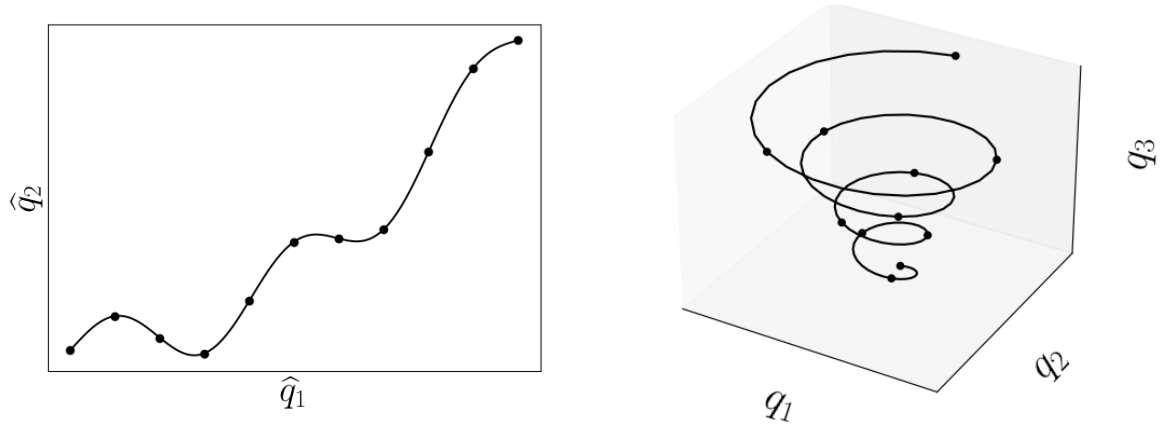



**Figure 1.4:** Illustration of state representation and time evolution in reduced-order space (left) and corresponding full-order state (right).

basis vectors from data include proper orthogonal decomposition [46], balanced truncation [47], and the reduced-basis method (RBM) [48]. Non-linear methods, on the other hand, formulate the mapping as an arbitrarily non-linear function of the low-dimensional state. Examples include autoencoders [49], kernel principal component analysis [50], and generative topographic mappings [51]. In practice, linear and non-linear mapping methods differ in their ease of calculation and expressiveness. Linear methods benefit from closed-form solutions, and the mapping from the reduced to full-order space amounts to simple linear algebra operations. However, the linear representation often results in high approximation error when applied to very non-linear solution manifolds. For example, similar to the Gibbs phenomenon of Fourier series, linear representations of sharp gradients exhibit "ringing" artifacts. Non-linear methods, on the other hand, may estimate such non-linearities much more accurately, though they rarely guarantee any measure of optimality and may require costly training procedures.

After the mapping has been computed, a method of evolving the low-dimensional state in time must be selected. The Koopman operator [52], which can be approximated by dynamic mode decomposition [53], is a linear operator which simply advances the state forward in time. Operator inference [54], alternatively, learns a quadratic form of the reduced-order state time evolution via a least-squares minimization problem. A host of neural network approaches also propose to model the dynamics of the reduced state using, for example, recurrent neural networks [55], temporal convolutional networks [56], and

adversarial networks [57]. Finally, of particular interest to this work, is the *projection-based* reduced-order model.

### 1.2.1 Projection-based Reduced-order Models

Unlike the methods described above, projection-based ROMs (PROMs) do not discard the original governing equations which generated the high-fidelity data. Instead, they project the system onto a low-dimensional space and evaluate the resulting reduced set of equations as you might the original high-dimensional ODE. In theory, the solution to this low-dimensional system is far less expensive than that of the full-order system. The two dominant projection methods are Galerkin projection and least-squares Petrov–Galerkin (LSPG) projection. These methods will be described in greater detail in Chapter 3, but a brief overview is provided here to highlight existing shortcomings in the state-of-the-art.

Galerkin PROMs have a rich history in the study of coherent structures in turbulence, dating back to work in turbulent boundary layers by Aubry *et al.* [58] and progressing to more complex geometries through the 1990's (e.g., [59, 60, 61, 62]). However, as Galerkin PROMs were applied to more complex fluid flow systems, they were discovered to suffer from a variety of stability and accuracy issues [63]. Despite careful modifications targeted at PROMs for compressible flows [64, 65], these stability issues arguably stem from the core fact that Galerkin projection is the $\ell^2$-optimal projection of the time-continuous governing ODE, while in practice the PROM is solved in the time-discrete O$\Delta$E setting [66].

In the early 2010's, LSPG projection [67, 68] was proposed as a discrete-optimal alternative to Galerkin projection. It has since been well-documented to generate far more stable and accurate PROMs compared to Galerkin projection for canonical 2D and 3D fluid flow problems [69]. Although broader adoption of LSPG PROMs has been slow, it has performed exceptionally well for several of perhaps the largest and most challenging aerodynamics problems to date [70].

In the late 2010's, early work began in applying Galerkin and LSPG PROMs to reacting flow problems. Preliminary results indicated that these PROMs struggled with

the stiff chemical kinetics and steep gradients which are characteristic of combusting flows [71, 72]. While *ad hoc* solutions of temperature limiters [73] and chemical species limiters [74] alleviated these difficulties somewhat, the general robustness of these PROMs remained poor. It was not until very recently that the model-form preserving least-squares with variable transformation (MP-LSVT) [75] was proposed as an alternative to LSPG, exhibiting greatly improved performance for PROMs of reacting flows.

Several key limitations remain to be solved for PROMs of general non-linear problems, namely computational efficiency and predictivity. As will be discussed in Chapter 4, PROMs of non-linear systems require a hyper-reduction method to limit the cost of evaluating non-linear terms in the governing ODE. Very little research has been conducted to compare the performance of various hyper-reduction approaches beyond studies of small one-dimensional systems [76]. Further, prior work on PROMs of large-scale systems typically do not explore performance beyond training data sets, i.e. predictive performance. Recent work implies that online adaptation of the PROM basis and hyper-reduction sampling scheme provide vastly improved predictive performance [77, 78], though these methods have yet to be tested for high-dimensional, multi-scale systems.

## 1.3 Objectives and Contributions

As discussed above, there is a relative paucity of investigations of PROMs for extremely large-scale, highly non-linear, multi-scale and multi-physics flow problems of some engineering significance. This thesis helps to fill this gap, detailing many of the unique challenges of PROMS for reacting flow problems, exposing the limitations of the state-of-the-art projection-based ROMs, and proposing some solutions for these problems and the methods by which they can be achieved in a compute- and memory-scalable fashion. In summary, the contributions by the author detailed in this thesis are as follows:

1. Development of PERFORM, an open-source project with which the reduced-order modeling community can prototype and test new methods for a series of benchmark problems which are significantly more challenging and informative than standard

ROM toy problems. Its use in pushing the capabilities of modern ROM methods is evaluated for an acoustically-forced 1D premixed flame, investigating novel deep autoencoder and recurrent neural network ROM methods. These results, enabled by PERFORM, give insight into both the exceptional ability of neural networks in representing advection-dominated reacting flows and the excessive cost of training the same neural networks.

2. A baseline assessment of state-of-the-art linear projection-based models for several complex multi-scale systems, showing that the prevailing state-of-the-art is ineffective. Detailed analyses of the MP-LSVT method demonstrate its superior performance for problems of greater complexity than any previously tested with the same approach.

3. Application of projection-based reduced-order modeling for a 3D multi-injector rocket combustor with nearly 250 million degrees of freedom, wherein combustion is modeled via a 12-species and 38-reaction. This represents the first study (to the author's knowledge) of a problem of this size and physical complexity, involving a self-excited transverse combustion instability, coupled injector dynamics, and exceedingly stiff reaction kinetics. Although this system is still a vast simplification of real combustor physics, this effort is a major step forward towards demonstrating the viability of PROM methods for industrial-scale systems.

4. Implementation of a pre- and post-processing toolchain for hyper-reduced PROMs within the open-source linear algebra toolkit PLATFORM [79]. This enables distributed solutions for costly greedy algorithms which are required to generate stable and accurate hyper-reduced PROMs, and is designed for easy expansion with novel sampling methods. Further, these additions allow for rapid generation of full-dimensional field data from hyper-reduced PROM outputs for error measurements and visualization. This end-to-end process assists in applying hyper-reduced PROMs to extremely high-dimensional systems.

5. Development of a truly memory- and compute-scalable PROM hyper-reduction im-

plementation in a massively-parallel combustion CFD laboratory code, enabling PROMs which do not scale with the original high-fidelity model dimension. With this design, true computational cost savings can be realized, and low-cost, low-memory simulations of extremely large combusting flow systems may be computed on laptops or personal workstations.

6. Demonstrated scalability of hyper-reduction for PROMs of practical engineering systems, including 2D transonic cavity flow and two 3D rocket combustors. Over three orders of magnitude computational savings are realized, decreasing the cost from $\mathcal{O}(10{,}000)$ CPU-hours to $\mathcal{O}(10)$ CPU-hours. This appears to be the first rigorous comparison of classical sparse sampling algorithms and more recent algorithms for systems of this size and complexity, revealing stark differences in load-balancing, accuracy, and computational cost savings. Insights into other critical parameters which define the hyper-reduced PROM problem are also made, aiding future investigations into PROMs of multi-scale, multi-physics systems.

7. Optimization of inefficient basis and hyper-reduction sample mesh adaptation algorithms in the aforementioned massively-parallel combustion CFD code. This enables the execution of truly predictive PROMs for large-scale systems which were previously unattainable due to excessive memory consumption and inter-process communication overhead. This approach is demonstrated for a 3D single-element rocket combustor, for which accurate models of unseen physics beyond the training dataset are computed with dynamic trial spaces and sample meshes.

8. A collected discussion and analysis of several important steps in developing effective PROMs for large-scale complex problems, including data preparation (centering and normalization), residual weighting for residual-minimization PROMs, limiters/clipping functions for preventing non-physical solutions, and variable transformations for PROMs of extremely stiff systems. These topics are often ignored, neglected, or glossed over in the literature, leaving future researchers to waste time learning these simple tricks of the trade. Especially for large-scale multi-

scale systems characterized by propagating waves, sharp gradients, and disparate dimensional scales, these methods are shown to be crucial elements of the PROM toolchain, and findings here provide best practices for practitioners.

Take care to note, however, that this thesis does *not* incorporate many of the complex physical phenomena present in realistic liquid-propellant rocket combustion. Due to computational constraints, none of the "high-fidelity" models can be considered well-resolved in space or time, as they do not incorporate liquid-phase transport or reactions, and all reaction mechanisms are greatly reduced. These results are not presented as accurate reflections of rocket combustion, and highlight a common criticism of data-driven methods for complex systems: if the underlying numerical model does not reflect reality, what is the use of data-driven approaches which generate unrealistic solutions? This is a somewhat shortsighted view, ignoring the progressive nature of scientific research. Data-driven models cannot instantaneously advance from toy problems to industrial-scale systems, but rather require incremental application to more difficult problems and adjustments as obstacles appear.

It is in this frame of mind that this thesis should be viewed instead as one of many steps towards the application of PROMs to realistic rocket combustion simulations. This body of work thus demonstrates that projection-based ROMs of non-linear systems can be a practical and effective means of generating low-cost solutions to extremely challenging problems in engineering. While there remains much work to be done to make these models truly generalizable and viable for industrial applications, they pose an encouraging route of investigation for alleviating the burdensome cost of extensive experimental campaigns and massive high-fidelity simulations in engineering design and analysis.

## 1.4 Organization and Notation

The organization of this thesis is as follows. Chapter 2 details the governing equations, thermodynamic and transport models, chemical reaction models, and spatio-temporal discretization methods used to obtain results exhibited in later chapters. Chapter 3

discusses relevant literature on projection-based reduced-order modeling techniques and derives those methods which are investigated in later chapter, including in-depth discussions on the MP-LSVT method. Chapter 4 outlines methods for constructing compute- and memory-scalable hyper-reduced PROMs. Chapter 5 outlines PERFORM, the open-source ROM development testbed, and novel results comparing linear and non-linear ROMs for an acoustically-forced, freely-propagating model premixed flame. Chapter 6 exhibits numerical experiments of PROMs for three complex multi-scale problems: 2D transonic flow over an open cavity, a 3D truncated single-element rocket combustor, and a nine-element laboratory rocket combustor. This last experiment represents the largest (and perhaps most complex) test case for PROMs to date, to the best of the author's knowledge. Chapter 7 onlines sample mesh and basis adaptation approaches for achieving PROM generalizability, and provides future-state predictive results for the single-element rocket combustor cases previously examined. For the last numerical experiments, Chapter 8 examines how various data preparation approaches, residual weighting methods, and localized temperature limiters affect the accuracy and stability of PROMs for reacting flows. Finally, Chapter 9 summarizes the results and meaningful conclusions of this research, and proposes several important future research directions.

Some universal notation is used throughout this thesis. Scalars and scalar-valued functions are denoted by lowercase, italicized Latin or Greek letters (e.g., $a$, $\psi$). Vectors and vector-valued functions are denoted by lowercase, bolded Latin or Greek letters (e.g., $\mathbf{a}$, $\boldsymbol{\psi}$). Matrices and matrix-valued functions are denoted by uppercase, bolded Latin or Greek letters (e.g., $\mathbf{A}$, $\boldsymbol{\Psi}$). Vector spaces and manifolds (and sets, occasionally) are defined using uppercase, calligraphic Latin letters (e.g., $\mathcal{A}$, $\mathcal{U}$). The "=" sign denotes simple equality, while ":=" denotes a definition. Functions are denoted by a Latin or Greek letter, or common mathematical function abbreviation, followed by parentheses enclosing the function's arguments (e.g., $a(\cdot)$, $\mathbf{a}(\cdot)$, $\mathbf{A}(\cdot)$, $\exp(\cdot)$). Brackets are used to group elements of mathematical equations (e.g., $a + [b + c]^2$), define vectors or matrices (e.g., $\mathbf{a} := [a_1,\ a_2,\ a_3]$), or specify molar concentration (always written as $[X_l]$). The purpose of brackets in a given context should be fairly self-evident. Braces are used to

define sets (e.g., $\mathcal{A} := \{a_1,\ a_2,\ a_3\}$). Single vertical bars indicate the absolute value operation (e.g., $|a|$), and double vertical bars indicate the vector or (induced) matrix norm (e.g., $\|\mathbf{a}\|_2$). Superscripts are generally reserved for exponentiation (e.g., $a^2$, $b^c$), indication of the time step associated with a variable (e.g., $a^n := a(\mathrm{t}^n)$), or transpose ($\mathbf{A}^\top$), inverse ($\mathbf{A}^{-1}$), and pseudoinverse ($\mathbf{A}^+$) operations, for which the context should be self-explanatory. Subscripts are used for numerous indicators including summation or set indices (e.g., $a_i\ \forall\ i \in \{1,\ 2,\ 3\}$), physical value descriptors (e.g., $c_p$, $h_0$), and variable descriptors (e.g., $\mathbf{q}_c$ vs. $\mathbf{q}_p$, $\mathbf{J}_{r,c}$ vs. $\mathbf{J}_{r,p}$). Diacritics are context-dependent, though the macron (e.g., $\overline{\mathbf{a}}$) is generally associated with constant values, the tilde (e.g., $\widetilde{\mathbf{a}}$) is generally associated with full-dimensional ROM quantities, and the circumflex (e.g., $\widehat{\mathbf{a}}$) is generally associated with low-dimensional ROM quantities.

# Chapter 2

# Modeling Combusting Flows

In this chapter, the governing equations and models which describe reacting fluid flows are described, along with the numerical methods by which these equations are discretized in space and time for results presented in this thesis. With the exception of those generated by PERFORM and presented in Chapter 5, all results are computed using the General Equations and Mesh Solver (GEMS). This solver was originally developed by Li at the University of Tennessee [80] and later by Li and Xia at Purdue University [81]. Over nearly two decades, GEMS has been used extensively to model a wide variety of complex combusting flow systems [82, 83, 84]. More recently, it has been expanded to include projection-based reduced-order modeling utilities, which will be described in greater detail in Chapter 3. Although PERFORM uses a similar formulation to GEMS, discrepancies will be explicitly described in Chapter 5.

## 2.1 Governing Equations

Fluid flows that can be treated as a continuum are governed by the unsteady Navier–Stokes equations. They describe the conservation of mass, momentum, energy, and transported scalars in an arbitrary control volume. Neglecting body forces, these equations are given by the PDE

$$\frac{\partial \mathbf{q}_c}{\partial \mathrm{t}} + \nabla \cdot (\mathbf{f} - \mathbf{f}_\nu) = \mathbf{s}, \tag{2.1}$$

where $\mathbf{q}_c$ is the conservative state, $\mathbf{f} := [\mathbf{f}_x,\ \mathbf{f}_y,\ \mathbf{f}_z]$ and $\mathbf{f}_\nu := [\mathbf{f}_{\nu,x},\ \mathbf{f}_{\nu,y},\ \mathbf{f}_{\nu,z}]$ are the inviscid and viscous flux terms in each spatial direction, respectively, and $\mathbf{s}$ are source

terms. These terms are given as

$$\mathbf{q}_c := \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho h^0 - p \\ \hdashline \rho Y_l \\ \hdashline \rho Z \\ \rho C \end{bmatrix}, \ \mathbf{f}_i := \begin{bmatrix} \rho u_i \\ \rho u u_i + \delta_{xi} p \\ \rho v u_i + \delta_{yi} p \\ \rho w u_i + \delta_{zi} p \\ \rho h^0 u_i \\ \hdashline \rho Y_l u_i \\ \hdashline \rho Z u_i \\ \rho C u_i \end{bmatrix}, \ \mathbf{f}_{\nu,i} := \begin{bmatrix} 0 \\ \tau_{ix} \\ \tau_{iy} \\ \tau_{iz} \\ -q_i + \sum_j u_j \tau_{ij} \\ \hdashline \rho V_{l,i} Y_l \\ \hdashline \rho D_Z \frac{\partial Z}{\partial x_i} \\ \rho D_C \frac{\partial C}{\partial x_i} \end{bmatrix}, \quad \mathbf{s} := \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \hdashline \dot{\omega}_l \\ \hdashline 0 \\ \dot{\omega}_C \end{bmatrix}. \tag{2.2}$$

Horizontal dashed lines separate additional equations which are included for different chemical transport and reaction models. The first five equations (the continuity equation, three momentum equations, and the energy equation) generally describe three-dimensional fluid flows. The $z$-momentum equation, $z$-velocity, and gradients in the $z$-direction are neglected in two-dimensional flows. The $y$-momentum equation, $y$-velocity, and gradients in the $y$-direction are also neglected in one-dimensional flows. The relevant terms are as follows: $\rho$ is the density, $u_i \in \{u,\ v,\ w\}$ is the velocity in each spatial direction, $p$ is the static pressure, and $\delta_{ij}$ is the Kronecker delta. The stagnation enthalpy is given by

$$h^0 := \frac{1}{2} \sum_{i}^{\{x,y,z\}} u_i^2 + \sum_{l=1}^{N_Y} h_l Y_l, \tag{2.3}$$

and the viscous shear stress is given by

$$\tau_{ij} := \mu \left[ \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \delta_{ij} \frac{2}{3} \sum_{k=1}^{3} \frac{\partial u_k}{\partial x_k} \right]. \tag{2.4}$$

Neglecting radiation and Dufour effects, the heat flux is given by

$$q_i := -\lambda \frac{\partial T}{\partial x_i} - \rho \sum_{l=1}^{N_Y} V_{l,i} Y_l h_l. \tag{2.5}$$

In Eqs. 2.2 and 2.5, the diffusion velocity product of the $l$th species in the $i$th spatial direction, $V_{l,i}Y_l$, is approximated as

$$V_{l,i}Y_l \approx D_{lM}\frac{\partial Y_l}{\partial x_i}, \tag{2.6}$$

where mass diffusion by pressure gradients, body forces, and temperature gradients have been neglected.

Calculation of thermodynamic properties (the species enthalpy $h_l$) and transport properties (the dynamic viscosity $\mu$, thermal conductivity $\lambda$, and species mass diffusivity $D_{lM}$) depend on the gas and transport models used in the simulation. These are described in greater detail in Section 2.1.1.

The sixth line of terms in Eq. 2.2 describes the scalar transport equation of the $l$th chemical species, for $l \in \{1, \ldots, N_Y - 1\}$, where $N_Y$ is the number of chemical species to be modeled. Here, $Y_l$ is the mass fraction of the $l$th species in the mixture, and $D_{lM}$ is the mass diffusivity of the $l$th species into the mixture. The reaction source term, $\dot{\omega}_l$, also described as the production rate, for the $l$th species is given by the specified reaction model; the laminar finite-rate reaction model used in this thesis is described in Section 2.2.1. Note that only $N_Y - 1$ chemical transport equations are solved. The mass fraction of the $N_Y$th species can, by definition of the mass fraction, be computed from

$$Y_{N_Y} = 1 - \sum_{l=1}^{N_Y-1} Y_l. \tag{2.7}$$

The seventh and eighth line of terms in Eq. 2.2 describe the scalar transport of the fuel mixture fraction $Z$ and progress variable $C$. These equations arise from the flamelet/progress variable (FPV) model for reacting flows [34], which is described in greater detail in Section 2.2.2. The classic FPV model replaces the energy and species transport equations with these two equations, and uses a pre-computed lookup table which maps the fuel mixture fraction and progress variable to the temperature and species mass fractions. As will be described later, the FPV model implemented in GEMS does not eliminate the energy equation, and hence conserves energy while incurring additional

computational cost.

As Eq. 2.1 is not closed (there are more unknowns than equations), an equation of state is provided which relates several quantities explicitly. For all results shown here, the system is treated as a mixture of ideal gases, and the ideal gas law is given by.

$$p = \rho R T, \tag{2.8}$$

where $T$ is the temperature, and $R$ is the mixture specific gas constant.

Throughout this thesis, the set of "primitive variables" is frequently referred to, and is given by

$$\mathbf{q}_p := \left[\begin{array}{ccccc:c:cc} p & u & v & w & T & Y_l & Z & C \end{array}\right]^\top. \tag{2.9}$$

These quantities have special use in that they can be easily interpreted in engineering practice, are used to tabulate empirical fit models, and are easily used to compute secondary quantities such as total pressure and heat transfer rates. Additionally, these variables have several numerical qualities that benefit the construction of robust and accurate reduced-order models, which will be detailed later. The conservative variables, while important in that they have the useful property of conservation, are less immediately useful in an engineering context.

### 2.1.1 Gas Models

GEMS is equipped with several models to compute thermodynamic and transport properties of gases, each with varying levels of accuracy in different pressure and temperature ranges. As this work primarily investigates systems of multi-species mixtures, the methods used for computing mixture quantities which are universal for all models are detailed here. To begin, the mixture enthalpy and entropy are computed simply as

$$h = \sum_{l=1}^{N_Y} Y_l h_l, \quad s = \sum_{l=1}^{N_Y} Y_l s_l. \tag{2.10}$$

The mixture dynamic viscosity is given by Wilke's mixing law [85],

$$\mu = 2\sqrt{2}\sum_{l=1}^{N_Y}\frac{X_l\mu_l}{\phi_l}, \tag{2.11}$$

where $X_l$ is the mole fraction of the $l$th species. The denominator term is given by

$$\phi_l = \sum_{m=1}^{N_Y} X_m\left[1+\left[\frac{\mu_l}{\mu_m}\right]^{1/2}\left[\frac{M_m}{M_l}\right]^{1/4}\right]^2\left[1+\frac{M_l}{M_m}\right]^{-1/2}, \tag{2.12}$$

where $M_l$ is the molecular mass of the $l$th species. Finally, the mixture thermal conductivity is given by Mathur *et al.* [86] (attributed to Burgoyne and Weinberg [87]),

$$\lambda = \frac{1}{2}\left[\sum_{l=1}^{N_Y} X_l\lambda_l + \left[\sum_{l=1}^{N_Y}\frac{X_l}{\lambda_l}\right]^{-1}\right]. \tag{2.13}$$

**Calorically-perfect Gas with Simplified Transport Properties**

The one-dimensional transient flame case (Section 5.2) and the 2D transonic cavity flow (Section 6.1) utilize the calorically-perfect gas (CPG) model with approximate, analytical models for transport properties. The CPG model makes the assumption that the heat capacity at constant pressure of the $l$th species, $c_{p,l}$, is constant, i.e. $c_{p,l}(T) = c_{p,l}$. The species enthalpy is thus computed simply as

$$h_l = h_l^\circ + c_{p,l}T, \tag{2.14}$$

where $h_l^\circ$ is the standard enthalpy of formation for the $l$th species. Similarly, entropy of the $l$th species is computed from the analytical relationship

$$s_l = c_{p,l}\ln\left(\frac{T}{278.0\text{ K}}\right) - R_l\ln\left(\frac{p}{101,325\text{ Pa}}\right), \tag{2.15}$$

where $R_l$ is the specific gas constant of the $l$th species. The dynamic viscosity of the $l$th species, on the other hand, is computed from the empirical Sutherland's law [88], given

by

$$\mu_l = \mu_{\text{ref},l} \left[ \frac{T}{T_{\text{ref},l}} \right]^{3/2} \left[ \frac{T_{\text{ref},l} + S_l}{T + S_l} \right]. \tag{2.16}$$

Here, $\mu_{\text{ref},l}$ is the experimentally-measured dynamic viscosity at temperature $T_{\text{ref},l}$, and $S_l$ is an independent empirical parameter for the $l$th species. With the dynamic viscosity in hand, the thermal conductivity of the $l$th species may be computed from the definition of the Prandtl number, rearranged as

$$\lambda_l = \frac{\mu_l c_{p,l}}{\text{Pr}_l}. \tag{2.17}$$

The Prandtl number of the $l$th species is tabulated experimentally. Similarly, the mass diffusivity of the $l$th species into the mixture can be computed from the definition of the Schmidt number, rearranged as

$$D_{lM} = \frac{\mu_l}{\rho \text{Sc}_l}. \tag{2.18}$$

Again, the Schmidt number of the $l$th species is tabulated experimentally.

**Thermally-perfect Gas with Empirical Fit Transport Properties**

At high temperatures, especially in the temperature ranges of 1,000 - 3,000 K normally experienced in rocket combustors, the assumption that the specific heat capacity of a fluid is constant with temperature is not accurate. The general formulation for the species enthalpy is thus given by

$$h_l = h_l^{\ominus} + \int_{T^{\ominus}}^{T} c_{p,l}(T) \mathrm{d}T. \tag{2.19}$$

In this work, the empirical fits of McBride, Gordon, and Reno [40] are used to compute thermodynamic quantities which incorporate this temperature dependence. The species specific heat capacity at constant pressure, species enthalpy, and species entropy are given by the model forms

$$\frac{c_{p,l}}{R_l} = a_{1,l} + a_{2,l}T + a_{3,l}T^2 + a_{4,l}T^3 + a_{5,l}T^4, \tag{2.20}$$

$$\frac{h_l}{R_l T} = a_{1,l} + a_{2,l}\frac{T}{2} + a_{3,l}\frac{T^2}{3} + a_{4,l}\frac{T^3}{4} + a_{5,l}\frac{T^4}{5} + \frac{a_{6,l}}{T}, \tag{2.21}$$

$$\frac{s_l}{R_l} = a_{1,l}\ln T + a_{2,l}T + a_{3,l}\frac{T^2}{2} + a_{4,l}\frac{T^3}{3} + a_{5,l}\frac{T^4}{4} + a_{7,l}, \tag{2.22}$$

where $a_{i,l}$, $i \in \{1 \dots 7\}$ are tabulated for the $l$th species.

Two different empirical models for transport properties are available in GEMS. The first, which is used in the single-element combustor simulations presented in Section 6.2, computes the species dynamic viscosities and thermal conductivities from the NASA Lewis Research Center model [89], given by,

$$\mu_l = \exp\left(b_{1,l}\ln T + \frac{b_{2,l}}{T} + \frac{b_{3,l}}{T^2} + b_{4,l}\right) \times 10^{-7}, \tag{2.23}$$

$$\lambda_l = \exp\left(c_{1,l}\ln T + \frac{c_{2,l}}{T} + \frac{c_{3,l}}{T^2} + c_{4,l}\right) \times 10^{-4}. \tag{2.24}$$

The scalar terms $b_{i,l}$, $c_{i,l}$, $i \in \{1 \dots 4\}$ are tabulated for the $l$th species. The scaling factors convert dynamic viscosity from micropoise to kg/m-s, and thermal conductivity from $\mu$W/cm-K to W/m-K.

The mass diffusivity of the $l$th species into the mixture is given by Curtiss and Hirschfelder [90] as

$$D_{lM} = \frac{1 - X_l}{\sum_{m \neq l} \frac{X_m}{D_{l,m}}}. \tag{2.25}$$

The binary diffusion coefficient between the $l$th and $m$th species is modeled via Chapman–Enskog theory (from [91])

$$D_{l,m} = \frac{0.0266}{p\left[\frac{\sigma_l + \sigma_m}{2}\right]^2 \Omega} \sqrt{T^3\left[\frac{1}{M_l} + \frac{1}{M_m}\right]}, \tag{2.26}$$

where $\sigma_l$ is the collision diameter of the $l$th species in Angstroms. Self diffusion is ignored ($D_{l,m} = 0 \ \forall \ l = m$). Note that the factor 0.0266 differs from that given in [91], as the diffusion coefficient is computed here in $\mathrm{m}^2/\mathrm{s}$ (instead of $\mathrm{cm}^2/\mathrm{s}$) and pressure is computed in pascals (instead of atmospheres). The diffusion collision integral, $\Omega$, is computed from

the relations of Neufeld *et al.* [92]

$$\Omega := \frac{d_1}{\exp\left(d_2 T^*\right)} + \frac{d_3}{\exp\left(d_4 T^*\right)} + \frac{d_5}{\exp\left(d_6 T^*\right)} + \frac{d_7}{\exp\left(d_8 T^*\right)}, \tag{2.27}$$

where the scalar terms $d_i$, $i \in \{1 \dots 8\}$ are empirically determined, and universal for every species pairing. The reduced temperature $T^*$ is computed as

$$T^* := T \left[\frac{k_B}{\sqrt{\epsilon_l \epsilon_m}}\right], \tag{2.28}$$

where $k_B$ is the Boltzmann constant, and $\epsilon_l$ is the tabulated Lennard–Jones energy of the $l$th species.

The second transport property calculation method is drawn from the TRANSPORT library [93], and used in simulations of the nine-element rocket combustor in Section 6.3. This model consists of third-order polynomials of the form,

$$\mu_l = \exp\left(\sum_{i=1}^{4} b_{i,l} \left[\ln T\right]^{i-1}\right) \times 10^{-1}, \tag{2.29}$$

$$\lambda_l = \exp\left(\sum_{i=1}^{4} c_{i,l} \left[\ln T\right]^{i-1}\right) \times 10^{-5}, \tag{2.30}$$

$$D_{l,m} = \left[\frac{101,325 \text{ Pa}}{p}\right] \exp\left(\sum_{i=1}^{4} d_{i,lm} \left[\ln T\right]^{i-1}\right) \times 10^{-4}. \tag{2.31}$$

Again, the scalar terms $b_{i,l}$, $c_{i,l}$, and $d_{i,l}$, $i \in \{1 \dots 4\}$ are empirically determined for the $l$th species (and pairing of the $l$th and $m$th species for binary diffusion coefficients). The scaling factors convert dynamic viscosity from poise to kg/m-s, thermal conductivity from erg/s-cm-K to W/m-K, and mass diffusivity from $\text{cm}^2/\text{s}$ to $\text{m}^2/\text{s}$. The diffusion of the $l$th species into the mixture is again computed by Eq. 2.25.

### 2.1.2 Subgrid-scale Models

In general, simulations of practical combustion systems will always be under-resolved due to the extremely small characteristic spatio-temporal scales of reacting flows. To

reduce the high cost of resolving these scales, large eddy simulations filter the governing equations into resolved (large-scale) and unresolved (small-scale) motions. The resulting set of equations can be solved for the resolved scales, with the exception of the subgrid stresses. As the subgrid stresses cannot be directly computed (they are a function of unresolved motions), they must be modeled. A broader discussion of turbulence and subgrid scale modeling can be found in [94].

For all GEMS simulations presented here, with the exception of the nine-element combustor discussed in Section 6.3, no explicit filter or subgrid scale model is used. This is not to imply that these are well-resolved DNS models, but can rather be viewed as *implicit LES* (ILES) simulations. In ILES, it is assumed that the numerical dissipation generated by the spatial discretization scheme accounts for the dissipative effects of the unresolved small-scale turbulence. As a result, no subgrid scale model is required. This idea dates back to early observations on monotone convection algorithms [95], and has since been successfully applied to a variety of turbulent flows [96, 97, 98]. A comprehensive overview of ILES can be found in [99]. In GEMS, ILES is applied by using a second-order accurate finite volume scheme, guaranteeing monotonicity with a gradient limiter (detailed in Section 2.3). No artificial dissipation is incorporated in the model. In unpublished studies, GEMS has been shown to produce excellent predictions of experimental flow statistics in a turbulent ship airwake.

In simulations of the nine-element combustor in Section 6.3, GEMS employs the eddy viscosity $\sigma$-model of Nicoud *et al.* [23]. Eddy viscosity models account for the dissipative effects of unresolved turbulence by adding an eddy viscosity term, computing the viscous stress tensor as

$$\tau_{ij} := [\mu + \mu_t] \left[ \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \delta_{ij} \frac{2}{3} \sum_{k=1}^{3} \frac{\partial u_k}{\partial x_k} \right]. \tag{2.32}$$

The $\sigma$-model computes the eddy viscosity $\mu_t$ as

$$\mu_t := \rho [C_\sigma \Delta]^2 D_\sigma, \tag{2.33}$$

where $C_\sigma$ is an empirically-determined constant (in this work, $C_\sigma = 1.4$), and $\Delta$ is the

subgrid characteristic length scale (here, one-third the cell volume). The differential operator $D_\sigma$ is computed as

$$D_\sigma := \frac{\sigma_3 \left[\sigma_1 - \sigma_2\right] \left[\sigma_2 - \sigma_3\right]}{\sigma_1^2}, \tag{2.34}$$

where $\sigma_1$, $\sigma_2$, and $\sigma_3$ are the singular values of the velocity gradient tensor. This eddy viscosity has the benefits of being a locally-defined positive quantity, decays with the cube of distance from a wall, and vanishes in two-dimensional flow. Further, unlike turbulence models such as the Spalart–Allmaras or $k$–$\omega$ turbulence models, the $\sigma$-model involves no additional transport equations.

## 2.2 Reaction Models

Two reaction models are used to simulate chemical reactions in this thesis: the laminar finite rate model and the flamelet/progress variable model. In both models, the $r$th chemical reaction of a mechanism is described with the general form

$$\sum_{l=1}^{N_Y} \nu'_{l,r} \chi_l \rightleftharpoons \sum_{l=1}^{N_Y} \nu''_{l,r} \chi_l, \tag{2.35}$$

where $\nu'_{l,r}$ and $\nu''_{l,r}$ are the stoichiometric coefficients of the $l$th species as reactants and products of the $r$th reaction, respectively. The $l$th chemical species is denoted by the symbol $\chi_l$. A simple example of this format is the reaction of carbon monoxide and hydroperoxyl, or

$$\mathrm{CO} + \mathrm{HO_2} \rightleftharpoons \mathrm{OH^-} + \mathrm{CO_2}. \tag{2.36}$$

The stoichiometry of this reaction is written in Table 2.1.

### 2.2.1 Finite Rate Reactions

The laminar finite rate chemistry model is used to compute results for the one-dimensional model premixed flame in Chapter 5 and the nine-element combustor in Section 6.3. In

| $l$ | $\chi_l$ | $\nu'_l$ | $\nu''_l$ |
|---|---|---|---|
| 1 | CO | 1 | 0 |
| 2 | $HO_2$ | 1 | 0 |
| 3 | $OH^-$ | 0 | 1 |
| 4 | $CO_2$ | 0 | 1 |

**Table 2.1:** Stoichiometry of example reaction, Eq. 2.36.

this model, the production rate of the $l$th species (in kg/m$^3$-s) is given by the relationship

$$\dot{\omega}_l = M_l \sum_{r=1}^{N_r} \left[\nu''_{l,r} - \nu'_{l,r}\right] w_r, \tag{2.37}$$

where $N_r$ is the total number of reactions in the mechanism. The rate-of-progress of the $r$th reaction is computed as

$$w_r = k_{F,r} \prod_{l=1}^{N_Y} [X_l]^{\nu'_{l,r}} - k_{R,r} \prod_{l=1}^{N_Y} [X_l]^{\nu''_{l,r}}, \tag{2.38}$$

where $k_{F,r}$ and $k_{R,r}$ are the forward and reverse reaction rates, respectively. Here, $[X_l]$ is the molar concentration of the $l$th species. The forward reaction rate (or the rate at which reactants are converted to products) is computed as an Arrhenius rate, given by the general form

$$k_{F,r} = A_r T^{b_r} \exp\left(\frac{-E_{a,r}}{R_u T}\right), \tag{2.39}$$

where $A_r$ is the pre-exponential factor, $b_r$ is the temperature exponent, and $E_{a,r}$ is the activation energy of the $r$th reaction. These constant factors are tabulated for each reaction in the mechanism, generally fit to match experimental results. Next, chemical reactions are assumed to progress at a much smaller time scale than that of transport phenomena, and thus any chemical reactions are assumed to be in local equilibrium. The reverse reaction rate can then be computed as

$$k_{R,r} = \frac{k_{F,r}}{k_{C,r}}, \tag{2.40}$$

where $k_{C,r}$ is the equilibrium constant for the $r$th reaction. The equilibrium constant is computed by

$$k_{C,r} = \exp\left(-\sum_{l=1}^{N_Y}\left[\nu''_{l,r} - \nu'_{l,r}\right] g_l\right)\left[\frac{101,325\ \text{Pa}}{TR_u}\right]^{\sum_{l=1}^{N_Y}\left[\nu''_{l,r}-\nu'_{l,r}\right]}, \tag{2.41}$$

where $g_l$ is the Gibbs free energy of the $l$th species in the mixture, given by

$$g_l = \frac{h_l}{R_l T} - \frac{s_l}{R_l}. \tag{2.42}$$

After the species production rates (Eq. 2.37) are computed, they are substituted into the corresponding scalar transport equations for each chemical species, as outlined in the sixth equation in Eq. 2.2.

For an *irreversible* reaction, it is assumed that the reaction only proceeds in the forward direction, i.e. $k_{R,r} = 0$. Irreversible reactions are denoted by a single rightward arrow, such as

$$\mathrm{CH_3 + O \rightarrow H + H_2 + CO}. \tag{2.43}$$

While no reaction is truly irreversible, in some cases the reverse reaction is so unlikely (relative to the forward reaction) that it may be safely ignored. This greatly simplifies the calculation of the reaction rate-of-progress (Eq. 2.38). The one-dimensional model premixed flame in Chapter 5 utilizes a single irreversible reaction, and some reactions of the mechanism used for the nine-element combustor in Section 6.3 are treated as irreversible.

Note that the reaction mechanism used in simulations of the nine-element combustor studied in Section 6.3 includes reactions involving third-body effects in the low-pressure limit. These corrections are either of the Lindemann–Hinshelwood form [100] or Troe form [101].

### 2.2.2 Flamelet/Progress Variable Model

In order to capture complex phenomena in turbulent combustion (such as local extinction and ignition), particularly for complex hydrocarbon fuels, chemical mechanisms for finite rate reaction models often account for dozens of species and hundreds of reactions. Further, as mentioned previously, the characteristic spatio-temporal scales of turbulent flames are extremely small. The cost of accurately modeling these processes during CFD calculations can be extremely computationally expensive.

Laminar flamelet modeling, originally introduced by Peters [102], attempts to simplify chemical transport and reaction modeling for turbulent non-premixed flames by treating the flame as an ensemble of localized flamelets. This operates by introducing the fuel mixture fraction,

$$Z := \frac{\nu Y_{\mathrm{f}} - Y_{\mathrm{ox}} + Y_{\mathrm{ox}}^{0}}{\nu Y_{\mathrm{f}}^{0} + Y_{\mathrm{ox}}^{0}}, \tag{2.44}$$

where $Y_{\mathrm{f}}$ and $Y_{\mathrm{ox}}$ are the local fuel and oxidizer mass fractions, respectively, $Y_{\mathrm{f}}^{0}$ is the mass fraction of fuel in the fuel stream, and $Y_{\mathrm{ox}}^{0}$ is the mass fraction of oxidizer in the oxidizer stream. The stoichiometric mass ratio is defined as

$$\nu := \frac{\nu'_{\mathrm{ox}} M_{\mathrm{ox}}}{\nu'_{\mathrm{f}} M_{\mathrm{f}}}. \tag{2.45}$$

The mixture fraction acts as an independent coordinate that varies from 0 (in the pure oxidizer stream) to 1 (in the pure fuel stream) normal to the flame surface defined by $Z = Z_{\mathrm{st}}$, the stoichiometric mixture fraction (i.e. where $\nu Y_{\mathrm{f}} = Y_{\mathrm{ox}}$). The transport equation for the fuel mass fraction is given as the seventh equation in Eq. 2.2. The species mass fraction fields are related to the fuel mixture fraction field by the steady flamelet equations

$$-\rho\chi \frac{\mathrm{d}^2 Y_l}{\mathrm{d}Z^2} = \dot{\omega}_l, \tag{2.46}$$

where the scalar dissipation rate, in $\mathrm{s}^{-1}$, is given by

$$\chi = 2D_Z \left[\nabla Z \cdot \nabla Z\right]. \tag{2.47}$$

In the original flamelet model formulation, a library of solutions to Eq. 2.46 is precomputed from one-dimensional counterflow diffusion flame simulations (e.g., via FlameMaster [103]). This provides a unique mapping from the local fuel mixture fraction and dissipation rate to individual mass fractions, i.e. $Y_l = Y_l(Z,\ \chi)$. As a result, the mixture fraction transport equation need only be accounted for, replacing the species transport equations.

The original flamelet model, however, does not explicitly model any chemical reaction effects. In order to account for this, an additional tracking variable, the *progress variable* $C$, was introduced by Charles Pierce [34]. This takes the form

$$C := \sum_{m=1}^{N_{\text{prog}}} Y_{l_m}, \tag{2.48}$$

where $N_{\text{prog}}$ is the total number of species mass fractions which make up the progress variable. In general, the constituent mass fractions are chosen to be those of reaction products or late-stage intermediates, as they are an indicator of reaction progress and provide a unique mapping to all chemical states (in combination with the mixture fraction). For example, the simulations of the truncated single-element combustor presented in Section 6.2 use carbon dioxide, carbon monoxide, and hydrogen. The precomputed flamelet library thus provides unique mappings,

$$Y_l = Y_l(Z,\ C), \quad \dot{\omega}_C = \dot{\omega}_C(Z,\ C). \tag{2.49}$$

The transport equation for the progress variable is given as the eighth equation in Eq. 2.2. In summary, the FPV model introduces two scalar transport equations, and precomputes a library of steady flamelet solutions which provide a unique mapping from these two scalars to the species mass fraction fields and progress variable rate of production. Especially when large, complex reaction mechanisms are used, this greatly reduces the computational cost of simulating non-premixed combusting flow systems.

Note that, traditionally, the FPV model also provides unique mappings for the temperature, density, and transport/thermodynamic quantities as part of the flamelet tabu-

lation (e.g., $T = T(Z,\ C)$, $\mu = \mu(Z,\ C)$). GEMS does not include these mappings, and instead solves the energy conservation equation directly. While this approach ensures conservation of energy, it also incurs significant computational costs due to the additional conservation equation and the direct calculation of transport and thermodynamic properties. Note that when a filter is applied for LES, an additional transport equation must be solved for the mean mixture fraction variance ($\widetilde{Z''^2}$). The mean mixture fraction variance is also used in the flamelet library mapping (i.e. $Y_l = Y_l(\widetilde{Z},\ \widetilde{Z''^2},\ \widetilde{C})$). Recall that all results in this thesis do not compute any explicit filtering, hence the mean mixture fraction variance is not accounted for here.

## 2.3 Numerical Discretization

The means by which the Navier–Stokes equations (Eq. 2.1) are discretized in space and time is described, as well as the methods by which the resulting linear system is solved. Discussion is restricted to methods implemented in GEMS; although similar methods are used in PERFORM, some differences are noted in Chapter 5.

The spatial domain is discretized by an unstructured, cell-centered, second-order accurate finite-volume scheme. Inviscid fluxes are computed by Roe's method [104]. Gradients are computed using the formulation of Mitchell [105], whereby nodal quantities are computed as the average of surrounding cell-centered values weighted by the method of Rausch *et al.* [106]. Monotonicity is preserved by the gradient limiter of Barth and Jespersen [107], modified to compute the gradient limiting factor based on the initial, unconstrained face-reconstructed states. A ghost cell scheme is used to enforce boundary conditions.

All full-order model results are discretized in time using backward differentiation formulae with dual time-stepping [108]. Grouping the discretized fluxes and source term into the general non-linear function $\mathbf{f}(\cdot)$, this begins from the form

$$\mathbf{J}_{c,p}\frac{\partial \mathbf{q}_p}{\partial \tau} + \frac{\partial \mathbf{q}_c}{\partial \mathrm{t}} = \mathbf{f}\left(\mathbf{q}_c,\ \mathrm{t}\right), \tag{2.50}$$

where $\mathbf{J}_{c,p} := \partial \mathbf{q}_c / \partial \mathbf{q}_p$, and $\tau$ is a fictitious *dual time*. Discretizing the physical time derivative with the second-order backwards differentiation formula, and the pseudo-time derivative by first-order finite-difference, this arrives at

$$\mathbf{J}_{c,p} \frac{\mathbf{q}_p^k - \mathbf{q}_p^{k-1}}{\Delta\tau} + \frac{3\mathbf{q}_c^k - 4\mathbf{q}_c^{n-1} + \mathbf{q}_c^{n-2}}{2\Delta \mathrm{t}} = \mathbf{f}\left(\mathbf{q}_c^k,\ \mathrm{t}\right). \tag{2.51}$$

The superscript $n$ indicates the $n$th physical time step of the simulation, and $k$ indicates the $k$th subiteration of the iterative solver at the current time step. The physical time step is given by $\Delta\mathrm{t}$, and the pseudo-time step is given by $\Delta\tau$. The initial guess for the iterative state is given as $\mathbf{q}_c^{k=1} = \mathbf{q}_c^{n-1}$. Treating the conservative state as a function of the primitive state (i.e. $\mathbf{q}_c = \mathbf{q}_c(\mathbf{q}_p)$), linearizing the equations about $\mathbf{q}_p^{k-1}$, and rearranging terms arrives at the linear system

$$\left[\left[\frac{\Delta \mathrm{t}}{\Delta\tau} + \frac{3}{2}\right]\mathbf{J}_{c,p} - \Delta \mathrm{t} \frac{\partial \mathbf{f}\left(\mathbf{q}_c^{k-1}\right)}{\partial \mathbf{q}_p}\right]\left[\mathbf{q}_p^k - \mathbf{q}_p^{k-1}\right] = -\frac{1}{2}\left[3\mathbf{q}_c^{k-1} - 4\mathbf{q}_c^{n-1} + \mathbf{q}_c^{n-2}\right] + \Delta \mathrm{t}\mathbf{f}\left(\mathbf{q}_c^{k-1}\right). \tag{2.52}$$

Equation 2.52 is then solved using the line Gauss–Sidel approach detailed by MacCormack [109], with one forward and one backward sweep in each coordinate direction for each subiteration. Upon convergence of dual time-stepping, the iterative solution is assigned to the solution at the next physical time step, i.e. $\mathbf{q}_p^n \leftarrow \mathbf{q}_p^k$. Note that with sufficient convergence, the pseudo-time derivative vanishes, recovering the physical residual.

As can be seen from the $\Delta\mathrm{t}/\Delta\tau$ term in Eq. 2.52, this dual-time formulation has the effect of weighting the block diagonal of the left-hand side matrix. Smaller values of $\Delta\tau$ increase this effect, and larger $\Delta\tau$ diminishes it. Further, using $\mathbf{q}_p = \mathbf{q}_c$ in the limit $\Delta\tau \to \infty$ recovers the standard Newton's method. Using a pseudo-time step size close to or smaller than $\Delta\mathrm{t}$ thus has the effect of increasing the block diagonal dominance of the system. This has the effect of improving the conditioning of the system, thereby increasing the robustness of the unsteady solution and improving the convergence of the line Gauss–Sidel solver.

# Chapter 3

# Projection-based Reduced-order Modeling

An overview of classical projection-based reduced-order model (PROM) methods is now described, along with derivations for the recent MP-LSVT method. For the following chapters, discussion is generalized to PROMs of any time-dependent, non-linear, hyperbolic conservation equations, given by the semi-discrete residual

$$\frac{\mathrm{d}\mathbf{q}_c}{\mathrm{dt}} - \mathbf{f}\left(\mathbf{q}_c,\ \mathrm{t}\right) = \mathbf{0}, \quad \mathbf{q}_c^0 = \mathbf{q}_c\left(\mathrm{t}^0\right). \tag{3.1}$$

Here, $\mathbf{q}_c : \mathbb{R}_{\geq 0} \to \mathbb{R}^N$ is the conserved state, $\mathrm{t} \in \mathbb{R}_{\geq 0}$ is the physical time, and $\mathbf{f} : \mathbb{R}^N \times \mathbb{R}_{\geq 0} \to \mathbb{R}^N$ is a function which is non-linear in the conservative state $\mathbf{q}_c$. In the context of the Navier–Stokes equations, this non-linear function constitutes the spatially-discretized fluxes, sources, body forces, and boundary conditions, and the number of degrees of freedom $N$ may be $\mathcal{O}\left(1 \times 10^6 - 1 \times 10^9\right)$ for simulations of practical engineering systems. The fully-discrete non-linear residual $\mathbf{r} : \mathbb{R}^N \times \mathbb{R}_{\geq 0} \to \mathbb{R}^N$ (after Eq. 3.1 has been temporally discretized) is further defined as

$$\mathbf{r}\left(\mathbf{q}_c^n,\ \mathrm{t}^n\right) := \dot{\mathbf{q}}_c^n - \mathbf{f}\left(\mathbf{q}_c^n,\ \mathrm{t}^n\right) = \mathbf{0}, \tag{3.2}$$

where $n \in \mathbb{N}_0$ is the discrete time step index at time $\mathrm{t}^n$, and $\dot{\mathbf{q}}_c^n$ is the temporal discretization operator (e.g. forward Euler, BDF).

Reduced-order models have a long history of successful applications in linear and elliptic systems. The reader is directed to the review paper by Benner *et al.* [110] for a discussion of PROMs for linear parametric dynamical systems, and the text by Hesthaven

*et al.* [111] for a complete background on the reduced-basis method for parametrized differential equations. These methods are notable for their extremely well-defined error measures, which enable rigorous applications in uncertainty quantification and control systems. However, such measures do not extend to general non-linear, hyperbolic systems; error bounds for such systems are usually ill-defined and impossible to compute *a priori*. As all systems investigated in this thesis are non-linear hyperbolic systems, neither PROMs for linear systems nor error bounds for PROMs are discussed here.

## 3.1 Trial Space Selection

Before constructing a projection-based ROM, a low-dimensional representation of the system state must first be constructed. The simplest method of doing so is constructing the state as a linear combination of a small number of basis vectors, represented by

$$\begin{aligned} \mathbf{q}_c\left(\mathrm{t}\right) \approx \widetilde{\mathbf{q}}_c\left(\mathrm{t}\right) &:= \overline{\mathbf{q}}_c + \sum_{i=1}^{N_c} h_{c,i}\mathbf{u}_{c,i}\widehat{q}_{c,i}(\mathrm{t}), \\ &:= \overline{\mathbf{q}}_c + \mathbf{H}_c\mathbf{U}_c\widehat{\mathbf{q}}_c(\mathrm{t}), \end{aligned} \tag{3.3}$$

where $\overline{\mathbf{q}}_c \in \mathbb{R}^N$ is a constant translation vector, $\mathbf{U}_c := [\mathbf{u}_{c,1},\ \ldots\ ,\ \mathbf{u}_{c,N_c}] \in \mathbb{R}^{N\times N_c}$ is the *trial basis*, and $\widehat{\mathbf{q}}_c := [\widehat{q}_{c,1}(\mathrm{t}),\ \ldots\ ,\ \widehat{q}_{c,N_c}(\mathrm{t})] : \mathbb{R}_{\geq 0} \to \mathbb{R}^{N_c}$ is the *latent state* (alternatively, *modal coefficients* or *generalized coordinates*) vector. The constant diagonal matrix $\mathbf{H}_c := diag\left(h_{c,1},\ \ldots\ ,\ h_{c,N}\right) \in \mathbb{R}^{N\times N}$ scales the conservative state variables.

The trial basis spans the affine *trial space*,

$$\widetilde{\mathcal{U}}_c := \overline{\mathbf{q}}_c + Range\left(\mathbf{U}_c\right). \tag{3.4}$$

It is in this subspace that the approximate solution exists, i.e. $\widetilde{\mathbf{q}}_c : \mathbb{R}_{\geq 0} \to \widetilde{\mathcal{U}}_c$. In reduced-order modeling, choosing $N_c \ll N$ generates a compact representation of the state and achieve significant order reduction. The question now becomes *how* to select an appropriate trial space that generates a reasonable approximation of the true state with the lowest dimension $N_c$ possible. For parametrized elliptic and parabolic governing systems,

the reduced basis method computes the approximate solution as a linear combination of solution realizations sampled over the parameter space. For convection-dominated hyperbolic systems, however, the proper orthogonal decomposition is a more appropriate choice.

### 3.1.1 Proper Orthogonal Decomposition

The proper orthogonal decomposition (POD) has a long history in a variety of fields under different names, such as the linear Karhunen–Loève approximation in statistics, principal component analysis in data analysis and machine learning, or the Eckart–Young–Mirsky theorem in mathematics. These methods ultimately amount to computing the $\ell^2$-optimal projector of a dataset onto an $N_c$-dimensional subspace. This is formalized by the least-squares problem

$$\mathbf{U}_c = \underset{\mathbf{A}\in\mathbb{R}^{N\times N_c}}{argmin} \left\|\mathbf{Q}'_c - \mathbf{A}\mathbf{A}^\top\mathbf{Q}'_c\right\|_{\mathrm{F}}, \tag{3.5}$$

where the *data snapshot matrix* is defined as

$$\mathbf{Q}'_c = \left[\mathbf{q}'_c\left(\mathrm{t}^0\right),\ \mathbf{q}'_c\left(\mathrm{t}^1\right),\ \ldots\ ,\ \mathbf{q}'_c\left(\mathrm{T}\right)\right] \in \mathbb{R}^{N\times N_\mathrm{T}}, \tag{3.6}$$

and the scaled, unsteady component of the solution is given as

$$\mathbf{q}'_c\left(\mathrm{t}\right) = \mathbf{H}_c^{-1}\left[\mathbf{q}_c\left(\mathrm{t}\right) - \overline{\mathbf{q}}_c\right]. \tag{3.7}$$

The collection of data snapshots defines the *data-driven* nature of this process: a small number of high-fidelity simulations are computed, during which $N_\mathrm{T}$ snapshots $\mathbf{q}_c\left(\mathrm{t}^i\right)$ of the conservative fields are saved to disk. These snapshots are centered about $\overline{\mathbf{q}}_c$, scaled by $\mathbf{H}_c^{-1}$, aggregated into the snapshot matrix (Eq. 3.6), and the optimal linear projection of this dataset onto an $N_c$-dimensional subspace is computed. Measurements of the quality of this projection is discussed in Section 3.4.3.

The solution of the least-squares problem in Eq. 3.5 has a convenient analytical solution via the singular value decomposition (SVD). Operating under the assumption that

the number of snapshots $N_\mathrm{T}$ is less than the number of degrees of freedom $N$ (true for almost all practical applications), the thin SVD of the data snapshot matrix is given by the form

$$\mathbf{Q}'_c = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top, \tag{3.8}$$

where $\mathbf{U} \in \mathbb{R}^{N\times N_\mathrm{T}}$ and $\mathbf{V} \in \mathbb{R}^{N_\mathrm{T}\times N_\mathrm{T}}$ are the left and right singular vectors of $\mathbf{Q}'_c$, respectively. The diagonal matrix of singular values $\boldsymbol{\Sigma} = diag(\sigma_1,\ \sigma_2,\ \ldots\ ,\ \sigma_{N_\mathrm{T}}) \in \mathbb{R}^{N_\mathrm{T}\times N_\mathrm{T}}$ are ordered such that $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{N_\mathrm{T}} \geq 0$. The left and right singular vectors are orthonormal, i.e., $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ and $\mathbf{V}^T\mathbf{V} = \mathbf{I}$. The solution to Eq. 3.5, and hence the trial basis $\mathbf{U}_c$, is computed by extracting the first $N_c$ columns of $\mathbf{U}$, corresponding to the $N_c$ largest singular values. These singular vectors account for the greatest variance in the dataset for any selection of $N_c$ singular vectors.

Algorithms for computing the SVD for extremely large datasets are intensely researched due to the general importance of the SVD in a variety of linear algebra problems. Popular linear algebra packages normally use a process of Householder reflections and QR decompositions, which can be difficult to implement natively in distributed-memory systems. Alternatively, there exist two simpler methods: the method of snapshots, and randomized SVD. All POD bases used to compute PROMs in this thesis utilize the former method, which are detailed here briefly. The method of snapshots, pioneered by Sirovich [112], begins by recognizing that the right singular vectors of $\mathbf{Q}'_c$ are the same as the right eigenvectors of $\mathbf{Q}'^\top_c\mathbf{Q}'_c \in \mathbb{R}^{N_\mathrm{T}\times N_\mathrm{T}}$, whose eigenvalues are also the square of the singular values of $\mathbf{Q}'_c$. This can be written as

$$\mathbf{Q}'^\top_c\mathbf{Q}'_c\mathbf{V} = \boldsymbol{\Sigma}^2\mathbf{V}. \tag{3.9}$$

As generally $N_\mathrm{T} \sim \mathcal{O}\,(100\text{–}1,000)$, and $\mathbf{Q}'^\top_c\mathbf{Q}'_c$ is symmetric positive definite, this eigenvalue problem can be trivially solved using standard serial routines, such as those supplied by MATLAB, NumPy/SciPy (Python), LAPACK (Fortran), or Eigen (C++). Leveraging the fact that the right singular vectors are orthonormal, the SVD (Eq. 3.8) can be

rearranged to compute the left singular vectors as

$$\mathbf{U} = \mathbf{Q}_c' \mathbf{V} \boldsymbol{\Sigma}^{-1}. \tag{3.10}$$

This approach is notably less memory-intensive than standard SVD routines, and is fairly trivial to implement in a parallel, distributed-memory environment (requiring only a parallel inner product, a serial eigensolve, and local matrix-vector products).

However, even the method of snapshots can be prohibitively memory- and compute-intensive. *Randomized* linear algebra attempts to alleviate much of the computational burden by approximating the SVD by decomposing the action of the data matrix on a small random matrix [113]. The number of random samples (columns of the random matrix) increases the accuracy and cost of this approximate solution. Although the randomized SVD is not used for any results in this paper, it would be negligent to not mention its utility for PROMs of large-scale problems. For example, the work by McQuarrie *et al.* [114] uses the randomized SVD for decompositions of 2D single-element rocket injector simulation data.

### 3.1.2 Non-linear Autoencoders

Up until this point, discussion has been limited to linear representations of the solution, of the form given by Eq. 3.3. The accuracy of this linear approximation to the subset of solution snapshots is often discussed in terms of the Kolmogorov $n$-width [115]. Note that the letter $n$ used here has no relation to the time step index used throughout this thesis, and this notation is only used for consistency with the literature. The Kolmogorov $n$-width is formally defined as

$$d_n(\mathcal{A}) = \inf_{\mathcal{V}_n} \sup_{x \in \mathcal{A}} \inf_{y \in \mathcal{V}_n} \|x - y\| . \tag{3.11}$$

This measures a distance between an *optimal* $n$-dimensional subspace $\mathcal{V}_n$ and a subset $\mathcal{A}$ of a normed linear space. Those solution subsets $\mathcal{A}$ for which increasing $n$ leads to rapidly-improving approximations are said to have a *quickly-decaying* $n$-width. Conversely, those

solution subsets which require large $n$ to achieve sufficiently accurate approximations are said to have a *slowly-decaying* $n$-width. Although computing exact bounds for $n$-widths is restricted to very specific solution sets, it has been empirically observed that fluid flow fields characterized by convection phenomena and sharp gradients (e.g., shocks or flames) exhibit poor convergence of approximation error via linear representations.

To overcome this inherent challenge, a class of techniques referred to as *non-linear manifold* methods seeks more general and expressive representations of the solution subset. As discussed briefly in Section 1.2, kernel principal component analysis [50] and generative topographic mappings [51] are just two traditional methods. In the past two decades, however, with increased access to open-source machine learning libraries, HPC resources, and large datasets, neural networks have become extremely relevant to the dimension-reduction research community.

Note that the following discussion of non-linear autoencoders derives directly from the work by Lee and Carlberg on non-linear manifold projection-based reduced-order models [116]. This seminal paper built a strong theoretical foundation for the application of neural network autoencoders in projection-based ROMs, and its terminology and notation are extremely intuitive. As such, although the results in Chapter 5 expands on this work and provides an honest evaluation of the cost/benefit tradeoff in neural network PROMs for reacting flows, the credit for developing the method lies solely with Lee and Carlberg. With that said, a very brief primer on neural networks and their relation to dimension-reduction follows.

**Feedforward Neural Networks**

From a mathematical point of view, a feedforward neural network can be considered as the successive composition of arbitrary non-linear functions which ingest a vector of input data $\mathbf{x} \in \mathbb{R}^{N_I}$ and maps it to a target vector $\mathbf{y} \in \mathbb{R}^{N_O}$. This process thus takes the form

$$\mathbf{y} = \boldsymbol{\varphi}\left(\mathbf{x},\ \boldsymbol{\Theta}\right), \tag{3.12}$$

$$\boldsymbol{\varphi}\left(\mathbf{x},\ \boldsymbol{\Theta}\right) := \boldsymbol{\varphi}^{N_L}\left(\cdot,\ \boldsymbol{\Theta}^{N_L}\right) \circ \boldsymbol{\varphi}^{N_L-1}\left(\cdot,\ \boldsymbol{\Theta}^{N_L-1}\right) \circ \ldots \circ \boldsymbol{\varphi}^{2}\left(\cdot,\ \boldsymbol{\Theta}^{2}\right) \circ \boldsymbol{\varphi}^{1}\left(\mathbf{x},\ \boldsymbol{\Theta}^{1}\right). \tag{3.13}$$

Each successive function $\boldsymbol{\varphi}^i : \mathbb{R}^{N_i} \to \mathbb{R}^{N_{i+1}}$ is referred to as the $i$th *layer* of the neural network, and $\boldsymbol{\Theta} := \{\boldsymbol{\Theta}^1, \ \ldots \ , \ \boldsymbol{\Theta}^{N_L}\}$ are the parameters which define the layers. The input of the first layer $\boldsymbol{\varphi}^1$ is of dimension $\mathbb{R}^{N_I}$, and the output of the last layer $\boldsymbol{\varphi}^{N_L}$ is of dimension $\mathbb{R}^{N_O}$. All inner input/output dimensions are arbitrary.

As a demonstrative example, one of the simplest forms of $\boldsymbol{\varphi}$ is the fully-connected layer, which is given by

$$\boldsymbol{\varphi}_{\mathrm{FC}}(\mathbf{x}, \boldsymbol{\Theta}) := \boldsymbol{\sigma}\left(\mathbf{W}\mathbf{x} + \mathbf{b}\right). \tag{3.14}$$

The layer parameters are given as the layer weights $\mathbf{W} \in \mathbb{R}^{N_{i+1} \times N_i}$ and biases $\mathbf{b} \in \mathbb{R}^{N_{i+1}}$. The function $\boldsymbol{\sigma} : \mathbb{R}^{N_{i+1}} \to \mathbb{R}^{N_{i+1}}$ is a user-specified *activation function*, which is typically chosen to be a non-linear function to supply the non-linear representative power of neural networks. Popular activation functions include sigmoid, tanh, ReLU, and swish functions.

Given a training dataset of matching pairs of input vectors and corresponding "correct" output vectors (e.g., a picture of a cat and the label "cat"), the parameters $\boldsymbol{\Theta}^i$ dictate the accuracy of the neural network's attempt to map the input data to the output data. These parameters are computed through an iterative training procedure by which training input is processed by the neural network, a *loss function* $c : \mathbb{R}^{N_O} \times \mathbb{R}^{N_O} \to \mathbb{R}$ measures how well the network's output matches the correct output vector, and the parameters are updated according to the loss function. This update to the network parameters is typically computed by *gradient descent via backpropagation*, whereby the contribution of a parameter to the loss $c\left(\mathbf{y}, \ \boldsymbol{\varphi}\left(\mathbf{x}, \ \boldsymbol{\Theta}\right)\right)$ determines how it is modified. The mechanics of gradient-based optimization are not described here; the reader is directed to the review by Ruder for further details [117].

Suffice to say, however, this iterative procedure of measuring the "correctness" of the network and incrementally adjusting the parameters can be an incredibly slow and computationally expensive process. Modern GPU and TPU computing architectures have enabled massive acceleration of this training process, but state-of-the-art neural networks have commensurately ballooned in the number of trainable parameters, reaching hundreds of billions [118] to trillions of parameters [119] in recent years. Further, gradient descent methods are not guaranteed to find a global optimum, and avoiding inaccurate

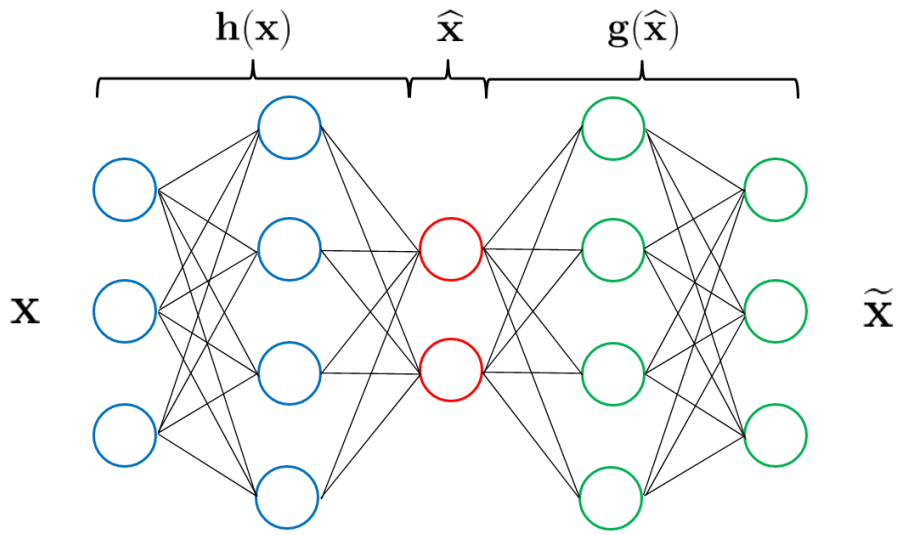



**Figure 3.1:** Simplified autoencoder composed of encoder network $\mathbf{h}(\cdot)$, latent state $\widehat{\mathbf{x}}$, and decoder network $\mathbf{g}(\cdot)$, given input $\mathbf{x}$.

local optima is an ongoing area of research. Ultimately, this training cost and lack of analytical optimality is the necessary cost of seeking a general, non-linear representation of extremely complex datasets. Autoencoders, a specific neural network architecture which enables non-linear reduced-order models, are now detailed.

### Autoencoder Trial Manifolds

Autoencoders describe a specific class of neural networks which are composed of an input network (the *encoder*) which maps the input data to a low-dimensional vector (the *latent state*), followed by an output network (the *decoder*) which attempts to map the latent state back to the input data. That is, the input data and the target output are the same, and the autoencoder learns to compress the data and retrieve them from the low-dimensional representation. In this sense, an autoencoder is trained to learn the identity operation, mapping input data to themselves through a low-dimensional bottleneck. A visual representation of a simple autoencoder is shown in Fig. 3.1. The bottleneck defines the dimension-reduction aspect of autoencoders: the full state may be represented by a low-dimensional state and extracted by the decoder network. However, unlike linear dimension-reduction as provided by POD in Section 3.1.1, neural networks enable arbitrarily non-linear representations of functions with the potential to efficiently approximate solution subsets with slowly-decaying Kolmogorov $n$-widths, such as those observed in

convection-dominated fluid flows. For dimension-reduction of fluid flow fields, the training data are the snapshots of the conservative or primitive states. An autoencoder for this data can thus be formalized by the representation

$$\widetilde{\mathbf{q}}_c = \mathbf{g}\left(\cdot,\ \boldsymbol{\Theta}^g\right) \circ \mathbf{h}\left(\mathbf{q}_c,\ \boldsymbol{\Theta}^h\right), \tag{3.15}$$

where $\mathbf{h} : \mathbb{R}^N \to \mathbb{R}^{N_c}$ is the encoder network, and $\mathbf{g} : \mathbb{R}^{N_c} \to \mathbb{R}^N$ is the decoder network. Note that the encoder function $\mathbf{h}$ is implied to be preceded by a centering and scaling operation, similar to that given in Eq. 3.7, and the decoder $\mathbf{g}$ is implied to be proceeded by the reverse operation, i.e.

$$\mathbf{h}\left(\mathbf{q}_c\right) := \boldsymbol{\varphi}_{h,c}\left(\mathbf{H}_c^{-1}\left[\mathbf{q}_c - \overline{\mathbf{q}}_c\right],\ \boldsymbol{\Theta}^h\right), \tag{3.16}$$

$$\mathbf{g}\left(\widehat{\mathbf{q}}_c\right) := \overline{\mathbf{q}}_c + \mathbf{H}_c \boldsymbol{\varphi}_{g,c}\left(\widehat{\mathbf{q}}_c,\ \boldsymbol{\Theta}^g\right). \tag{3.17}$$

Here, $\boldsymbol{\varphi}_{h,c} : \mathbb{R}^N \to \mathbb{R}^{N_c}$ and $\boldsymbol{\varphi}_{g,c} : \mathbb{R}^{N_c} \to \mathbb{R}^N$ are solely the neural network components of the encoder and decoder, respectively. This ensures that the network is trained on standardized data, as was the case for computing the POD basis in Section 3.1.1.

After the autoencoder network is trained as previously detailed, the decoder alone may be extracted, and the physical state may then be approximated as

$$\mathbf{q}_c\left(\mathrm{t}\right) \approx \widetilde{\mathbf{q}}_c(\mathrm{t}) := \mathbf{g}\left(\widehat{\mathbf{q}}_c\left(\mathrm{t}\right)\right). \tag{3.18}$$

Note the similarity of Eqs. 3.17 and 3.18 to the linear representation in Eq. 3.3. Indeed, the above equations are a more general form of such a low-dimensional representation, and devolves to the linear case when the neural network is replaced by a linear operator, i.e. $\boldsymbol{\varphi}_{g,c}\left(\mathbf{x}\right) := \mathbf{U}_c\mathbf{x}$. Of course, neural networks benefit from supplying arbitrary non-linear representations, and the approximate solution is thus defined on a *non-linear manifold* $\widetilde{\mathcal{U}}_c := \{\mathbf{g}\left(\widehat{\mathbf{x}}\right) \mid \widehat{\mathbf{x}} \in \mathbb{R}^{N_c}\}$. In theory, this non-linear manifold has the potential to much more accurately model the true solution subset than a linear subspace for a fixed latent dimension $N_c$. In Chapter 5, it is shown that this is indeed the case given a sufficiently

expressive neural network architecture, particularly for solution subsets characterized by propagating waves and sharp gradients.

The general representation of the approximate state given by Eq. 3.18 is retained in deriving PROMs in Sections 3.2–3.3. Where appropriate, simplifications are provided for the case of a linear trial space.

## 3.2 Classical Projection-based ROMs

Inserting the approximation $\mathbf{q}_c \approx \widetilde{\mathbf{q}}_c = \mathbf{g}\left(\widehat{\mathbf{q}}_c\right)$ into the governing ODE (Eq. 3.1) results in the system

$$\frac{\mathrm{d}\mathbf{g}\left(\widehat{\mathbf{q}}_c\right)}{\mathrm{dt}} - \mathbf{f}\left(\widetilde{\mathbf{q}}_c,\ \mathrm{t}\right) = \mathbf{0}, \quad \widetilde{\mathbf{q}}_c^0 = \widetilde{\mathbf{q}}_c\left(\mathrm{t}^0\right). \tag{3.19}$$

Employing the chain rule, noting that the trial space translation vector and scaling matrix are constant, and scaling the ODE via the diagonal matrix $\mathbf{H}_r^{-1} \in \mathbb{R}^{N\times N}$ to standardize the semi-discrete residual, this system can be modified into

$$\mathbf{H}_r^{-1}\left[\mathbf{H}_c\mathbf{J}_{\varphi,c}\frac{\mathrm{d}\widehat{\mathbf{q}}_c}{\mathrm{dt}} - \mathbf{f}\left(\widetilde{\mathbf{q}}_c,\ \mathrm{t}\right)\right] = \mathbf{0}, \tag{3.20}$$

where the Jacobian $\mathbf{J}_{\varphi,c} := \partial\boldsymbol{\varphi}_{g,c}\left(\widehat{\mathbf{q}}_c\right)/\partial\widehat{\mathbf{q}}_c : \mathbb{R}^{N_c} \to \mathbb{R}^{N\times N_c}$. For a linear representation, the Jacobian is simply $\mathbf{J}_{\varphi,c} = \mathbf{U}_c$. The purpose of the residual scaling will be made apparent later.

This has clearly resulted in no practical dimension reduction, as Eqs 3.19 and 3.20 are still $N$-dimensional systems. To accomplish dimension reduction, a *projection* operation, the namesake of projection-based reduced-order models, is required. This entails choosing a *test basis* $\mathbf{W}_c \in \mathbb{R}^{N\times N_c}$ by which the approximate ODE is projected as

$$\mathbf{W}_c^\top\mathbf{H}_r^{-1}\left[\mathbf{H}_c\mathbf{J}_{\varphi,c}\frac{\mathrm{d}\widehat{\mathbf{q}}_c}{\mathrm{dt}} - \mathbf{f}\left(\widetilde{\mathbf{q}}_c,\ \mathrm{t}\right)\right] = \mathbf{0}. \tag{3.21}$$

Assuming $\mathbf{W}_c^\top\mathbf{H}_r^{-1}\mathbf{H}_c\mathbf{J}_{\varphi,c} \in \mathbb{R}^{N_c\times N_c}$ is invertible, rearranging terms arrives at

$$\frac{\mathrm{d}\widehat{\mathbf{q}}_c}{\mathrm{dt}} = \left[\mathbf{W}_c^\top\mathbf{H}_r^{-1}\mathbf{H}_c\mathbf{J}_{\varphi,c}\right]^{-1}\mathbf{W}_c^\top\mathbf{H}_r^{-1}\mathbf{f}\left(\widetilde{\mathbf{q}}_c,\ \mathrm{t}\right). \tag{3.22}$$

Finally, Eq. 3.22 is a $N_c$-dimensional ODE which can be marched forward in time with respect to the latent state $\widehat{\mathbf{q}}_c$. A natural choice is $N_c \ll N$, and thus this time integration process is theoretically cheaper than that of the FOM. This may not always be the case above a certain value of $N_c$, as the inversion (and reduction across processes for parallel applications) of the dense matrix $\mathbf{W}_c^\top \mathbf{H}_r^{-1} \mathbf{H}_c \mathbf{J}_{\varphi,c}$ may be more computationally expensive than the solution of the FOM's sparse linear system arising from Eq. 2.52. Further, the evaluation of the non-linear terms $\mathbf{f}(\cdot,\ \mathrm{t})$ still depends on the full-order state $\widetilde{\mathbf{q}}_c$. These and other computational barriers will be addressed in more detail in Chapter 4.

The question now becomes how the projecting test basis is chosen. In the following sections, classical projection methods are detailed, namely Galerkin projection and least-squares Petrov–Galerkin projection. In Section 3.3 the recent model-form preserving least-squares with variable transformation method is derived, and its potential advantages over classical projection methods are discussed.

### 3.2.1 Galerkin Projection

Galerkin projection can be defined as the projection of the full-order ROM ODE onto the space tangent to the unscaled trial manifold, i.e.

$$\mathbf{W}_c\left(\widehat{\mathbf{q}}_c\right) := \mathbf{J}_{\varphi,c}\left(\widehat{\mathbf{q}}_c\right). \tag{3.23}$$

Substituting this into Eq. 3.22 arrives at

$$\frac{\mathrm{d}\widehat{\mathbf{q}}_c}{\mathrm{dt}} = \left[\mathbf{J}_{\varphi,c}^\top \mathbf{H}_r^{-1} \mathbf{H}_c \mathbf{J}_{\varphi,c}\right]^{-1} \mathbf{J}_{\varphi,c}^\top \mathbf{H}_r^{-1} \mathbf{f}\left(\widetilde{\mathbf{q}}_c,\ \mathrm{t}\right). \tag{3.24}$$

As detailed extensively by Lee and Carlberg [116], when employing a non-linear manifold trial space, the solution of Eq. 3.24 with an implicit time integrator via Newton's method requires the calculation of the third-order tensor $\partial \mathbf{J}_{\varphi,c} / \partial \widehat{\mathbf{q}}_c$. This is generally computational intractable, and neglecting this term amounts to a quasi-Newton method with an approximate residual Jacobian.

For a linear representation, the test basis is simply $\mathbf{W}_c = \mathbf{U}_c$, leading to

$$\frac{\mathrm{d}\widehat{\mathbf{q}}_c}{\mathrm{dt}} = \left[\mathbf{U}_c^\top \mathbf{H}_r^{-1} \mathbf{H}_c \mathbf{U}_c\right]^{-1} \mathbf{U}_c^\top \mathbf{H}_r^{-1} \mathbf{f}\left(\widetilde{\mathbf{q}}_c,\ \mathrm{t}\right). \tag{3.25}$$

In the specific case of $\mathbf{H}_c = \mathbf{H}_r = \mathbf{I}$, recalling that $\mathbf{U}_c$ is orthonormal, this simplifies the ODE greatly to the more familiar Galerkin PROM form

$$\frac{\mathrm{d}\widehat{\mathbf{q}}_c}{\mathrm{dt}} = \mathbf{U}_c^\top \mathbf{f}\left(\widetilde{\mathbf{q}}_c,\ \mathrm{t}\right). \tag{3.26}$$

As discussed in Section 1.2, Galerkin PROMs have a long history of applications to simulations of fluid flows (e.g., [58, 61, 62]). However, they have been observed to exhibit unacceptable stability and accuracy issues for more complex systems. Although the Galerkin PROM can be shown to be continuous-optimal in the sense that it minimizes the $\ell^2$-norm of the semi-discrete ROM residual [69], in reality the fully-discrete ROM residual is of more practical concern. Further, there is no expectation that the projection of the non-linear function $\mathbf{f}\left(\cdot,\ \mathrm{t}\right)$ onto the space tangent to the trial space is accurate, as the conservative state space and the space $\{\mathbf{f}\left(\mathbf{y},\ \mathrm{t}\right) \mid \mathbf{y} \in \mathbb{R}^N\}$ are entirely different. This fact motivates the least-squares Petrov–Galerkin projection method.

### 3.2.2 Least-squares Petrov–Galerkin Projection

The least-squares Petrov–Galerkin (LSPG) projection method, formulated and refined by Carlberg and coworkers [67, 68, 69], is motivated by minimizing the *fully-discrete ROM residual*, in contrast to the semi-discrete residual minimized by Galerkin projection. The fully-discrete ROM residual $\mathbf{r} : \mathbb{R}^{N_c} \to \mathbb{R}^N$ is given as

$$\mathbf{r}\left(\widehat{\mathbf{q}}_c^n\right) := \dot{\widetilde{\mathbf{q}}}_c^n - \mathbf{f}\left(\mathbf{g}\left(\widehat{\mathbf{q}}_c^n\right),\ \mathrm{t}\right) = \mathbf{0}, \tag{3.27}$$

where $\dot{\widetilde{\mathbf{q}}}_c$ is the temporal discretization operator (e.g. forward Euler, BDF) of the approximate state $\widetilde{\mathbf{q}}_c = \mathbf{g}\left(\widehat{\mathbf{q}}_c\right)$ at the $n$th time step. Computing the least-squares minimization

of the scaled fully-discrete residual takes the form,

$$\widehat{\mathbf{q}}_c^n = \underset{\mathbf{y}\in\mathbb{R}^{N_c}}{argmin} \ \left\|\mathbf{H}_r^{-1}\mathbf{r}\left(\mathbf{y}\right)\right\|_2, \tag{3.28}$$

where the purpose of scaling the residual by $\mathbf{H}_r^{-1}$ is made apparent. This has the effect of ensuring that, with careful calculation of $\mathbf{H}_r$, the various elements of the residual contribute similarly to the solution of the non-linear least squares problem. This is particularly important for systems which exhibit extreme scale disparities (as is often the case in high-pressure reacting systems) and solvers which cannot readily be non-dimensionalized (such as compressible combustion solvers). The residual weighting is sometimes couched in terms of a weighted norm, written alternatively as

$$\widehat{\mathbf{q}}_c^n = \underset{\mathbf{y}\in\mathbb{R}^{N_c}}{argmin} \ \left\|\mathbf{r}\left(\mathbf{y}\right)\right\|_{\mathbf{H}_r^{-1}}, \tag{3.29}$$

where the weighted norm indicates $\|\mathbf{r}\|_{\mathbf{H}_r^{-1}} := \sqrt{\mathbf{r}^\top\mathbf{H}_r^{-2}\mathbf{r}}$. A detailed exploration of the idea of alternative norms is provided by Parish and Rizzi [120]. This concept is also tangentially related to the work by Lindsay *et al.* [121] on PROM preconditioning, which showed that even a simple Jacobi preconditioner results in drastic improvements in PROM stability and accuracy relative to PROMs of the equivalent unscaled, dimensional system.

The solution of Eq. 3.28 is usually accomplished iteratively by linearizing the residual at the $n$th time step about the solution at the $k$th subiteration, given by the process

$$\delta\widehat{\mathbf{q}}_c^k = \underset{\mathbf{y}\in\mathbb{R}^{N_c}}{argmin} \ \left\|\mathbf{H}_r^{-1}\left(\widehat{\mathbf{J}}_{r,c}^k\mathbf{y} - \mathbf{r}\left(\widehat{\mathbf{q}}_c^k\right)\right)\right\|_2, \tag{3.30}$$

$$\widehat{\mathbf{q}}_c^{k+1} = \widehat{\mathbf{q}}_c^k + \alpha\left[\delta\widehat{\mathbf{q}}_c^k\right], \tag{3.31}$$

where the ROM residual Jacobian is defined as $\widehat{\mathbf{J}}_{r,c}^k := \partial\mathbf{r}\left(\widehat{\mathbf{q}}_c^k\right)/\partial\widehat{\mathbf{q}}_c : \mathbb{R}^{N_c} \to \mathbb{R}^{N\times N_c}$. The constant factor $\alpha \in \mathbb{R}_+$ is the step size, which is unity for all PROMs in this thesis; alternative methods may adapt $\alpha := \alpha^{n,k}$ to control iterative convergence. Further, the least-squares problem may be computed in terms of the full-order Jacobian $\mathbf{J}_{r,c}^k :=$

$\partial \mathbf{r}\left(\widehat{\mathbf{q}}_c^k\right)/\partial \mathbf{q}_c : \mathbb{R}^{N_c} \to \mathbb{R}^{N\times N}$ via the chain rule,

$$\delta\widehat{\mathbf{q}}_c^k = \underset{\mathbf{y}\in\mathbb{R}^{N_c}}{argmin} \ \left\|\mathbf{H}_r^{-1}\left[\mathbf{J}_{r,c}^k\mathbf{H}_c\mathbf{J}_{\varphi,c}^k\mathbf{y} - \mathbf{r}\left(\widehat{\mathbf{q}}_c^k\right)\right]\right\|_2, \tag{3.32}$$

where the decoder Jacobian $\mathbf{J}_{\varphi,c}^k := \partial\boldsymbol{\varphi}_{g,c}\left(\widehat{\mathbf{q}}_c^k\right)/\partial\widehat{\mathbf{q}}_c$ is now specified in the time-discrete setting. In the case of a linear trial space, $\mathbf{J}_{\varphi,c}^k = \mathbf{U}_c$, and this simplifies to

$$\delta\widehat{\mathbf{q}}_c^k = \underset{\mathbf{y}\in\mathbb{R}^{N_c}}{argmin} \ \left\|\mathbf{H}_r^{-1}\left[\mathbf{J}_{r,c}^k\mathbf{H}_c\mathbf{U}_c\mathbf{y} - \mathbf{r}\left(\widehat{\mathbf{q}}_c^k\right)\right]\right\|_2. \tag{3.33}$$

This particular form is useful inasmuch as the full-order residual Jacobian $\mathbf{J}_{r,c}^k$ is normally already available if a given FOM solver contains implicit time integration capabilities. The form given in Eq. 3.30 may be simpler if automatic differentiation tools are available in the solver. In either case, upon sufficient convergence of the solution, the iterative solution is set to the solution at the next physical time step, i.e. $\widehat{\mathbf{q}}_c^n \leftarrow \widehat{\mathbf{q}}_c^k$, and the process is repeated for proceeding time steps.

The simplest means of computing the solution to Eq. 3.30 is the formation of the normal equations, given by,

$$\left[\mathbf{H}_r^{-1}\widehat{\mathbf{J}}_{r,c}^k\right]^\top\mathbf{H}_r^{-1}\widehat{\mathbf{J}}_{r,c}^k\delta\widehat{\mathbf{q}}_c^k = -\left[\mathbf{H}_r^{-1}\widehat{\mathbf{J}}_{r,c}^k\right]^\top\mathbf{H}_r^{-1}\mathbf{r}\left(\widehat{\mathbf{q}}_c^k\right). \tag{3.34}$$

This formulation reveals the meaning of "Petrov–Galerkin" in LSPG: a Petrov–Galerkin projection is any projection in which the test space is *not* the same as the space tangent to the trial manifold (as is the case for Galerkin projection). In Eq. 3.34, expanding terms using the chain rule arrives at

$$\left[\mathbf{H}_r^{-1}\mathbf{J}_{r,c}^k\mathbf{H}_c\mathbf{J}_{\varphi,c}^k\right]^\top\mathbf{H}_r^{-1}\mathbf{J}_{r,c}^k\mathbf{H}_c\mathbf{J}_{\varphi,c}^k\delta\widehat{\mathbf{q}}_c^k = -\left[\mathbf{H}_r^{-1}\mathbf{J}_{r,c}^k\mathbf{H}_c\mathbf{J}_{\varphi,c}^k\right]^\top\mathbf{H}_r^{-1}\mathbf{r}\left(\widehat{\mathbf{q}}_c^k\right). \tag{3.35}$$

This form can be interpreted as a Petrov–Galerkin projection of the Gauss–Newton solution of Eq. 3.21 with a test basis

$$\mathbf{W}_c^k := \mathbf{H}_r^{-1}\mathbf{J}_{r,c}^k\mathbf{H}_c\mathbf{J}_{\varphi,c}^k. \tag{3.36}$$

Even with a linear state representation ($\mathbf{J}^k_{\varphi,c} = \mathbf{U}_c$), the test basis is time-variant, unlike Galerkin projection with a linear state representation. This poses a significant computational cost increase over Galerkin projection. For a non-linear state representation, the cost difference is less extreme, as the decoder Jacobian is dependent on the latent state, i.e. $\mathbf{J}^k_{\varphi,c} := \mathbf{J}_{\varphi,c}\left(\widehat{\mathbf{q}}^k_c\right)$, and the test basis is time-variant for Galerkin and LSPG projection alike.

Forming the complete normal equations is a computationally expensive process. A simple alternative involves the QR decomposition of the residual Jacobian,

$$\widehat{\mathbf{J}}^k_{r,c} = \mathbf{Q}\mathbf{R}, \tag{3.37}$$

where $\mathbf{Q} := [\mathbf{Q}_{N_c},\ \mathbf{Q}_{N-N_c}] \in \mathbb{R}^{N\times N}$ has orthogonal columns, composed of $\mathbf{Q}_{N_c} \in \mathbb{R}^{N\times N_c}$ and $\mathbf{Q}_{N-N_c} \in \mathbb{R}^{N\times N-N_c}$. The matrix $\mathbf{R} = [\mathbf{R}_{N_c},\ \mathbf{0}]^\top \in \mathbb{R}^{N\times N_c}$ is composed of upper triangular $\mathbf{R}_{N_c} \in \mathbb{R}^{N_c\times N_c}$ and a zero matrix $\mathbf{0} \in \mathbb{R}^{N-N_c\times N_c}$. Note that the matrix $\mathbf{R}$ here is not the same as that used for residual scaling, but this notation is used temporarily for the sake of consistency with the literature (and general familiarity with the method). The latent state update is solved from

$$\mathbf{R}_{N_c}\delta\widehat{\mathbf{q}}^k_c = [\mathbf{Q}_{N_c}]^\top \mathbf{r}\left(\widehat{\mathbf{q}}^k_c\right). \tag{3.38}$$

The upper triangular solve is significantly less expensive than the dense solve in Eq. 3.34, and does not require the dense matrix-matrix multiplication $\left[\mathbf{H}_r^{-1}\mathbf{J}^k_{r,c}\right]^\top \mathbf{H}_r^{-1}\mathbf{J}^k_{r,c}$.

Least-squares Petrov–Galerkin projection has been shown to generate vastly more accurate and stable PROMs compared to Galerkin PROMs. The first publications to outline LSPG computed a robust PROM simulation of a three-dimensional Ahmed body [67, 68], and later reported excellent results for transonic flow over a two-dimensional open cavity [69]. Continued work with LSPG has produced stable and accurate PROMs of three-dimensional separated flow over an airfoil at high angle-of-attack [122] and flow around a three-dimensional model of an F-16 aircraft [70]. The success of LSPG has been attributed to the fact that it minimizes the fully-discrete ROM residual [66] which

is, in practice, the set of equations being solved by the PROM. This has the effect of avoiding the inaccurate projection of the non-linear terms $\mathbf{f}(\cdot,\ \mathrm{t})$ onto the space tangent to the trial space (as seen in Eq. 3.24), and instead projects the fully-discrete residual via the space tangent to the fully-discrete residual (as seen in Eq. 3.34).

However, early applications of both Galerkin and LSPG PROMs to advection-dominated combustion problems indicated that both methods were insufficient in generating robust and accurate PROMs for such systems [72, 73]. Both methods frequently resulted in large deviations in the expected heat release, and often produced non-physical states such as negative temperature or pressure values. The latter phenomenon is largely due to ringing artifacts arising from the linear representation of sharp temperature and species gradients. Limiter methods were proposed to prevent such non-physical phenomena [73, 74], but are generally considered *ad hoc* solutions. More generally, these issues are often attributed to the loss of conservation in projecting the conservative equations onto a subspace which is purely a data-driven construction. Efforts have been made to enforce conservation over subsets of the computational domain [123, 124], which have seen success for simple 1D flow problems. For reacting flow problems, the failure of classical PROM methods is also linked to persistently stiff PROMs which have difficulty converging due to the high degree of non-linearity induced by chemical source terms, which are extremely sensitive to small changes in chemical composition and temperature.

## 3.3 Model-form Preserving Least-squares with Variable Transformation

The recent model-form preserving least-squares with variable transformation (MP-LSVT), developed by Huang *et al.* [75], seeks to resolve the difficulties of constructing robust and accurate PROMs for multi-scale, multi-physics problems for which Galerkin and LSPG PROMs fail. This method is founded on a very simple modification of LSPG: solve the ROM residual least-squares problem with respect to the latent representation of an alternative (but physically-complete) *target state* $\mathbf{q}_p : \mathbb{R}_{\geq 0} \to \mathbb{R}^N$. That is, construct a

low-dimensional representation of the target state, which for a linear representation is written as

$$\begin{aligned}\mathbf{q}_p\left(\mathrm{t}\right) \approx \widetilde{\mathbf{q}}_p\left(\mathrm{t}\right) &:= \overline{\mathbf{q}}_p + \sum_{i=1}^{N_p} h_{p,i}\mathbf{u}_{p,i}\widehat{q}_{p,i}(\mathrm{t}), \\ &:= \overline{\mathbf{q}}_p + \mathbf{H}_p\mathbf{U}_p\widehat{\mathbf{q}}_p(\mathrm{t}).\end{aligned} \tag{3.39}$$

Similar to the conservative variable formulation, $\overline{\mathbf{q}}_p \in \mathbb{R}^N$ is a constant translation vector, $\mathbf{U}_p := \left[\mathbf{u}_{p,1},\ \ldots\ ,\ \mathbf{u}_{p,N_p}\right] \in \mathbb{R}^{N\times N_p}$ is the target state trial basis, and $\widehat{\mathbf{q}}_p := \left[\widehat{q}_{p,1}(\mathrm{t}),\ \ldots\ ,\ \widehat{q}_{p,N_p}(\mathrm{t})\right] : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^{N_p}$ is the target latent state vector. The constant matrix $\mathbf{H}_p := \mathrm{diag}\left(h_{p,1},\ \ldots\ ,\ h_{p,N}\right) \in \mathbb{R}^{N\times N}$ scales the target state variables. For an autoencoder non-linear manifold representation, the encoder/decoder form is given by

$$\mathbf{h}\left(\mathbf{q}_p\right) := \boldsymbol{\varphi}_{h,p}\left(\mathbf{H}_p^{-1}\left[\mathbf{q}_p - \overline{\mathbf{q}}_p\right], \boldsymbol{\Theta}^h\right), \tag{3.40}$$

$$\mathbf{g}\left(\widehat{\mathbf{q}}_p\right) := \overline{\mathbf{q}}_p + \mathbf{H}_p\boldsymbol{\varphi}_{g,p}\left(\widehat{\mathbf{q}}_p, \boldsymbol{\Theta}^g\right). \tag{3.41}$$

In either case, the trial basis or autoencoder is computed from snapshots of the target state collected from a small number of high-fidelity simulations. These are aggregated as

$$\mathbf{Q}'_p = \left[\mathbf{q}'_p\left(\mathrm{t}^0\right),\ \mathbf{q}'_p\left(\mathrm{t}^1\right),\ \ldots\ ,\ \mathbf{q}'_p\left(\mathrm{T}\right)\right] \in \mathbb{R}^{N\times N_\mathrm{T}}, \tag{3.42}$$

where the scaled, unsteady component of the solution is defined as

$$\mathbf{q}'_p\left(\mathrm{t}\right) = \mathbf{H}_p^{-1}\left[\mathbf{q}_p\left(\mathrm{t}\right) - \overline{\mathbf{q}}_p\right]. \tag{3.43}$$

The trial space and non-linear trial manifold are defined, respectively, by

$$\widetilde{\mathcal{U}}_p := \overline{\mathbf{q}}_p + \mathit{Range}\left(\mathbf{U}_p\right), \qquad \text{(linear)} \tag{3.44}$$

$$\widetilde{\mathcal{U}}_p := \left\{\mathbf{g}\left(\widehat{\mathbf{x}}\right) \mid \widehat{\mathbf{x}} \in \mathbb{R}^{N_p}\right\}. \qquad \text{(non-linear)} \tag{3.45}$$

With this low-dimensional representation in hand, the conservative state can then be

treated as a function of the target state,

$$\mathbf{q}_c := \mathbf{q}_c \left( \mathbf{g} \left( \widehat{\mathbf{q}}_p \right) \right), \tag{3.46}$$

and the fully-discrete ROM residual can be reframed as

$$\mathbf{r} \left( \widehat{\mathbf{q}}_p^n \right) := \dot{\widetilde{\mathbf{q}}}_c(\widehat{\mathbf{q}}_p^n) - \mathbf{f} \left( \mathbf{g} \left( \widehat{\mathbf{q}}_p^n \right), \, \mathrm{t} \right) = \mathbf{0}. \tag{3.47}$$

Again, $\dot{\widetilde{\mathbf{q}}}_c(\widehat{\mathbf{q}}_p^n)$ is the temporal discretization operator for the full-order conservative state, treated now as a function of the target latent state. Similarly, the non-linear functions associated with the spatial discretization $\mathbf{f}(\cdot, \, \mathrm{t})$ is treated as a function of the target state. The least-squares minimization of the ROM residual thus takes the form

$$\widehat{\mathbf{q}}_p^n = \underset{\mathbf{y} \in \mathbb{R}^{N_p}}{argmin} \; \left\| \mathbf{H}_r^{-1} \mathbf{r} \left( \mathbf{y} \right) \right\|_2. \tag{3.48}$$

The scaling of the ROM residual by $\mathbf{H}_r^{-1}$ remains the same, as the model equations have not changed. The primary difference is that the solution is computed as a minimization with respect to the target latent state, rather than the conservative latent state. Solution via Gauss–Newton is thus given by

$$\delta \widehat{\mathbf{q}}_p^k = \underset{\mathbf{y} \in \mathbb{R}^{N_p}}{argmin} \; \left\| \mathbf{H}_r^{-1} \left[ \widehat{\mathbf{J}}_{r,p}^k \mathbf{y} - \mathbf{r} \left( \widehat{\mathbf{q}}_p^k \right) \right] \right\|_2, \tag{3.49}$$

$$\widehat{\mathbf{q}}_p^{k+1} = \widehat{\mathbf{q}}_p^k + \alpha \left[ \delta \widehat{\mathbf{q}}_p^k \right], \tag{3.50}$$

where the ROM residual Jacobian is now computed with respect to the target latent state, defined as $\widehat{\mathbf{J}}_{r,p}^n := \partial \mathbf{r} \left( \widehat{\mathbf{q}}_p^k \right) / \partial \widehat{\mathbf{q}}_p : \mathbb{R}^{N_p} \to \mathbb{R}^{N \times N_p}$. This Jacobian can similarly be decomposed as

$$\widehat{\mathbf{J}}_{r,p}^k := \frac{\partial \mathbf{r} \left( \widehat{\mathbf{q}}_p^k \right)}{\partial \widetilde{\mathbf{q}}_p} \frac{\partial \widetilde{\mathbf{q}}_p^k}{\partial \widehat{\mathbf{q}}_p}, \tag{3.51}$$

$$= \mathbf{J}_{r,p}^k \mathbf{H}_p \mathbf{J}_{\varphi,p}^k, \tag{3.52}$$

where $\mathbf{J}^k_{r,p} := \partial \mathbf{r}\left(\widehat{\mathbf{q}}^k_p\right)/\partial \widetilde{\mathbf{q}}_p : \mathbb{R}^{N_p} \to \mathbb{R}^{N \times N}$ and $\mathbf{J}^k_{\varphi,p} := \partial \boldsymbol{\varphi}_{g,p}\left(\widehat{\mathbf{q}}_p\right)/\partial \widehat{\mathbf{q}}_p : \mathbb{R}^{N_p} \to \mathbb{R}^{N \times N_p}$.

The solution of Eq. 3.49 via the normal equations again leads to a Petrov–Galerkin projection of the form,

$$\left[\mathbf{W}^k_p\right]^\top \mathbf{W}^k_p \delta \widehat{\mathbf{q}}^k_p = -\left[\mathbf{W}^k_p\right]^\top \mathbf{H}_r^{-1} \mathbf{r}\left(\widehat{\mathbf{q}}^k_p\right), \tag{3.53}$$

where the test basis is computed as

$$\mathbf{W}^k_p := \mathbf{H}_r^{-1} \mathbf{J}^k_{r,p} \mathbf{H}_p \mathbf{J}^k_{\varphi,p}. \tag{3.54}$$

Note that throughout this section, the subscript $p$ has been used to indicate quantities associated with the target state. As noted in Eq. 2.9, this notation is often used to refer to the primitive state, which is given by the variables

$$\mathbf{q}_p := \left[\begin{array}{ccccc:c:cc} p & u & v & w & T & Y_l & Z & C \end{array}\right]^\top. \tag{3.55}$$

Indeed, all work utilizing the MP-LSVT method [75, 125, 126] to date has used the primitive variables as the target state. This choice has empirically generated extremely accurate PROMs for complex reacting flow systems, including two- and three-dimensional single-element rocket combustors, and a two-dimensional multi-element rocket combustor. In direct comparisons to Galerkin and LSPG PROMs, MP-LSVT vastly improves the stability and convergence of these systems. Analysis of the conditioning of the linearized least-squares solve indicates that MP-LSVT using the primitive state produces systems with much lower condition numbers than those generated by Galerkin and LSPG. The exact reason for this improved conditioning is poorly understood beyond observed trends in numerical experiments.

The primitive variables are of particular interest in that they are immediately useful to engineers in analyzing the performance of the system, and many secondary quantities can be easily computed from them. However, the generality of the MP-LSVT invites investigations of other sets of target variables. It remains to be seen whether other

variable sets may generate further improvements in linear solve conditioning, or enable more accurate predictions of important physical quantities such as heat release.

## 3.4 Offline Calculations

In this section and later chapters, a distinctions is drawn between two steps of the PROM evaluation: the *offline* and *online* stages. The offline stage refers to any computations which are either necessary to prepare the PROM evaluation (e.g., calculating the trial basis, hyper-reduction basis and sampling) or evaluate the FOM dataset (e.g., calculating projection error, as described below). The online stage refers to any computations that occur during the evaluation of the unsteady PROM solution. In theory, offline steps should require far less computational expense than FOM or PROM evaluations, and hence are generally not considered part of the overall computational budget. This is usually the case, though as will be noted in Chapter 5, this is often not the case for the training of autoencoders for non-linear manifold PROMs. Withholding discussion of neural network training cost for the moment, several important offline computations are now described.

### 3.4.1 Processing Large Datasets

Throughout this thesis, datasets measuring hundreds of gigabytes in size must be processed. The algorithms for computing POD bases or determining sparse sampling points, which involve large matrix-matrix products, QR factorizations, SVDs, or eigendecompositions, require significant additional workspace memory. Such computations are certainly impossible to fit in the memory of desktop workstations (usually 32-64 GB), and are often too large even for HPC node memory (usually 128-256 GB). Further, the efficient evaluation of large-scale linear algebra operations, such as those for computing the SVD or QR decomposition, requires parallel computations to complete them in a reasonable amount of wall clock time. Loading large datasets from a cluster file system to node memory and distributing portions of the data to each process present additional bottlenecks. As

such, a distributed-memory, parallel processing framework is required to implement these algorithms. However, modern computational frameworks such as ScaLAPACK [127] and SLATE [128] often require intimate knowledge of library-specific memory layouts and computational pathways, making them unwieldy in rapid prototyping settings.

The Parallel Linear Algebra Tool FOr Reduced Modeling (PLATFORM) [79], developed by Nicholas Arnold-Medabalimi at the University of Michigan, provides a user-friendly framework for handling distributed and scalable I/O and linear algebra. The toolbox provides general utilities to efficiently load large datasets into node memory, distributed among processes in optimal block-cyclic ordering. The datasets can then be processed using simple interfaces to functions from the PBLAS/ScaLAPACK libraries, such as the rank-revealing QR factorization (PDGEQPF), least-squares solve (PDGELS), or SVD (PDGESVD). Abstraction of the distributed matrices allow for the rapid prototyping and deployment of new algorithms.

PLATFORM has already been successfully employed for computing decompositions and PROM preprocessing operations for large datasets [125, 129, 130, 131, 132]. All offline algorithms presented here are implemented using PLATFORM. Performance metrics for the datasets examined in this paper are not included, but benchmarks for other large datasets and detailed explanations of the inner workings of PLATFORM are given by Arnold-Medabalimi *et al.* [79].

### 3.4.2 POD Energy

For a linear representation computed by POD, choosing the dimension of the trial bases $\mathbf{U}_c$ or $\mathbf{U}_p$ is not an exact science. Although a given POD basis is $\ell^2$-optimal for a given dimension $N_c$ or $N_p$, it is impossible to know *a priori* how the accuracy of the representation will affect the unsteady solution of a PROM for a general non-linear system. In fact, enrichment of the trial space by increasing the basis dimension is never guaranteed to improve PROM accuracy. Numerical experiments have shown that increasing the trial space dimension may even lead to numerical instabilities [75].

Despite this problem, one must use some heuristics for initial experimentation with

projection-based ROMs, as computing a uniform grid search over the trial space dimension can be cost-prohibitive, especially for large-scale systems. By far the most popular heuristic is the *POD energy*, or conversely the POD residual energy (unity minus the POD energy). The latter measure is often reported as a percentage, given by the calculation

$$\text{POD Residual Energy}, \% = 100 \times \left(1 - \frac{\sum_{i=1}^{N_c} \sigma_i^2}{\sum_{i=1}^{N_\text{T}} \sigma_i^2}\right), \tag{3.56}$$

where $\sigma_i \in \mathbb{R}_{\geq 0}$ are the singular values of the training dataset, computed by the SVD and ordered such that $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{N_c} \geq 0$.

This measure dates back to the origins of POD, when POD was used to decompose and analyze turbulent velocity fields [46]. In that context, the POD energy is an analogue of the kinetic energy contributed by a given trial mode, with lower mode numbers (associated with low-frequency, high-amplitude waveforms) contributing a relatively large portion of the energy, while the high mode numbers (associated with high-frequency, low-amplitude waveforms) contributing relatively little. The extension of POD energy to general flow fields does not always have quite a clear relationship, but helps form a good first guess for how well the trial basis will represent the unsteady flow field.

Projection-based ROM practitioners will generally begin experimentation with trial bases achieving a POD residual energy of approximately 1.0%, 0.1%, and 0.01%. Figure 3.2 displays a sample POD residual energy decay for the two-dimensional transonic cavity flow presented in Chapter 6 for datasets spanning different simulation periods. For the 6 ms case, then, the practitioner might initially experiment with trial bases composed of the leading 19, 64, and 121 modes, corresponding to 1.0%, 0.1%, and 0.01% residual energy.

Such explorations of POD residual energy for datasets of varying sizes have the additional benefit of indicating whether a dataset describes a statistically-stationary flow. That is, when adding additional snapshots to the dataset does not result in a significant change in its spectral content, the dataset can be considered to capture the characteristic physics of the system. As can be seen in Fig. 3.2, while it appears that the lower-

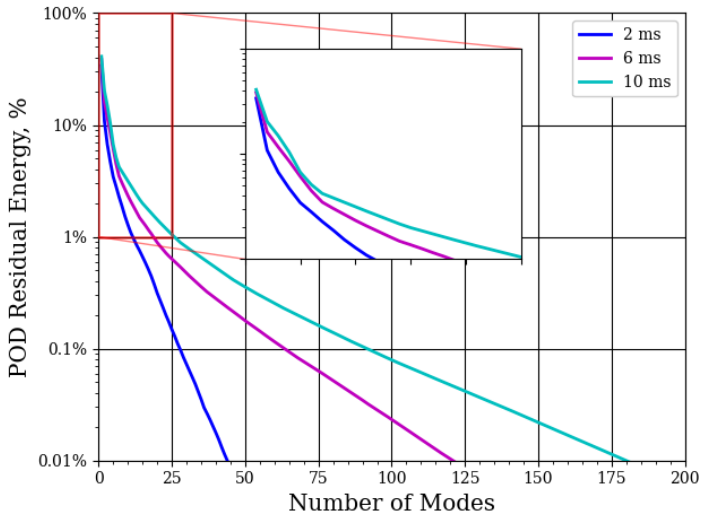



**Figure 3.2:** Example of POD residual energy decay for 2D cavity case, for datasets encompassing 2, 6, and 10 milliseconds of simulation time.

frequency range ($< 25$ modes) has somewhat converged, there is no indication that the higher-frequency spectral content has converged.

### 3.4.3 Projection Error

For results presented in Chapters 5-7, the *projection error* will be frequently used as a diagnostic tool for understanding the quality of a computed trial manifold $\widetilde{\mathcal{U}}$ in modeling the FOM solution data. This error measure, with a linear trial space for the $n$th time instance of the $v$th state variable (density, $x$-momentum, etc.), is given by the formulation

$$\boldsymbol{\epsilon}_v^n = \frac{\left\| \mathbf{q}_v^n - \left[ \overline{\mathbf{q}} + \mathbf{H}\mathbf{U}\mathbf{U}^\top \mathbf{H}^{-1} \left[ \mathbf{q}_v^n - \overline{\mathbf{q}} \right] \right] \right\|_2}{\left\| \mathbf{q}_v^n \right\|_2}. \tag{3.57}$$

Scaling the error for each variable by the norm of the unprojected state ensures reasonable comparisons of error between state variables of drastically different magnitudes (e.g., $\mathcal{O}\left(1\text{e}6\right)$ for pressure and $\mathcal{O}\left(1\right)$ for species mass fractions). For a non-linear trial manifold $\widetilde{\mathcal{U}}$, an equivalent projection operation akin to $\mathbf{U}\mathbf{U}^\top$ does not exist. Instead, the projection error is defined as

$$\boldsymbol{\epsilon}_v^n = \frac{\left\| \mathbf{q}_v^n - \left[ \underset{\mathbf{y} \in \widetilde{\mathcal{U}}}{argmin} \ \left\| \mathbf{q}^n - \mathbf{y} \right\|_2 \right]_v \right\|_2}{\left\| \mathbf{q}_v^n \right\|_2}. \tag{3.58}$$

As $\widetilde{\mathcal{U}}$ is a non-linear manifold defined by the range of the decoder $\mathbf{g}(\cdot)$, the solution of the *argmin* is a non-linear least squares problem. In practice, the `least_squares()` function provided by the SciPy Python `optimize` package (with default tolerances) is used to compute this solution. An initial guess for the input to the decoder is given by the encoding of the state, $\mathbf{h}(\mathbf{q}^n)$. This process effectively finds the point on the non-linear trial manifold which is closest to the data $\mathbf{q}^n$.

Often the time average over the simulation period $[\mathrm{t}^0, \mathrm{T}]$ will be provided, and is computed as

$$\boldsymbol{\epsilon}_v = \frac{1}{N_\mathrm{T}} \sum_{n=1}^{N_\mathrm{T}} \boldsymbol{\epsilon}_v^n. \tag{3.59}$$

Further, the total average projection error across all state variables provides a very broad measure of the trial space quality, and is given by

$$\boldsymbol{\epsilon} = \frac{1}{N_v} \sum_{v=1}^{N_v} \boldsymbol{\epsilon}_v. \tag{3.60}$$

Examples of these error measures are provided for a one-dimensional transient flame simulation, similar to those detailed in Chapter 5, though without acoustic forcing at the outlet. Figures 3.3 and 3.4 display instantaneous snapshots of temperature and reactant mass fraction fields, respectively, and the corresponding projections onto a linear trial space and non-linear manifold (by deep convolutional autoencoder) of dimension $N_p = 3$. Figures 3.5 and 3.6 display the $\ell^2$ error measure over time, as described by Eqs. 3.57 and 3.58.

Projection error is useful in providing an upper bound on the accuracy of the PROM. The unsteady PROM cannot produce a solution more accurate that the projected solution (except by pure coincidence), as the trial space is never an exact representation of the true solution. Further, computing the projection of unseen data (in parametric or future state prediction) provides a measure of the generalizability of the trial manifold. This can help determine whether a PROM will be appropriate for such predictions and perhaps preclude expensive online evaluations which are bound to fail purely due to the unfitness of the trial manifold.

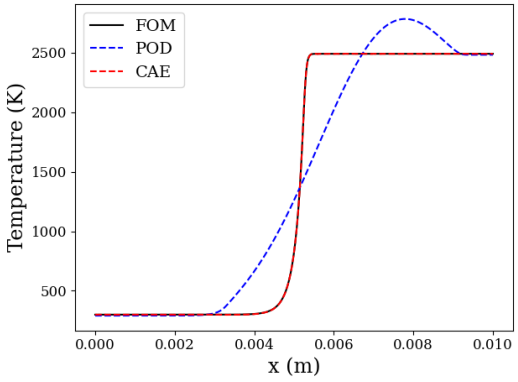



**Figure 3.3:** Instantaneous temperature fields and projections, $N_p = 3$.

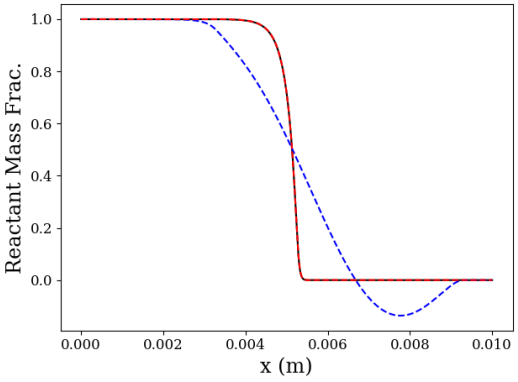



**Figure 3.4:** Instantaneous mass fraction fields and projections, $N_p = 3$.

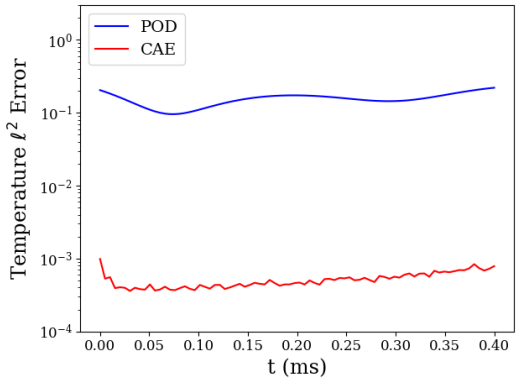



**Figure 3.5:** Unsteady temperature projection error, $N_p = 3$.

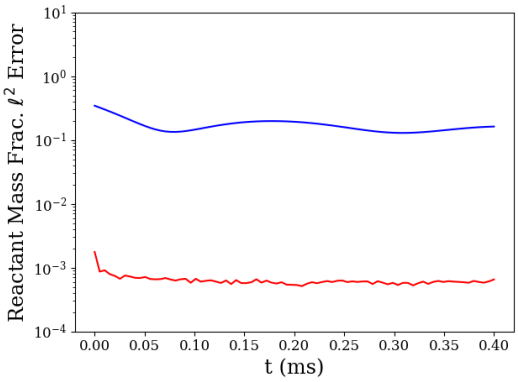



**Figure 3.6:** Unsteady mass fraction projection error, $N_p = 3$.

# Chapter 4

# Accelerated PROMs of Non-linear Systems

So far, the fact that the Galerkin (Eq. 3.24), LSPG (Eq. 3.30), and MP-LSVT (Eq. 3.49) PROMs *will not generate computational cost savings* for large-scale non-linear systems has been pointedly ignored. This deficiency is largely due to the fact that evaluating the non-linear terms $\mathbf{f}(\cdot,\ \mathrm{t})$ arising from fluxes, source terms, body forces, and boundary conditions still scales with the full-order dimension $N$. Linear time-invariant systems do not suffer from this issue; given a system of the form $\mathrm{d}\mathbf{q}_c/\mathrm{dt} - \mathbf{A}\mathbf{q}_c = \mathbf{0}$, $\mathbf{A} \in \mathbb{R}^{N\times N}$, the resulting Galerkin PROM (assuming $\mathbf{H}_c = \mathbf{H}_r = \mathbf{I}$) takes the form

$$\frac{\mathrm{d}\widehat{\mathbf{q}}_c}{\mathrm{dt}} = \mathbf{U}_c^\top \mathbf{A}\mathbf{U}_c\widehat{\mathbf{q}}_c. \tag{4.1}$$

The matrix $\mathbf{U}_c^\top \mathbf{A}\mathbf{U}_c\widehat{\mathbf{q}}_c \in \mathbb{R}^{N_c\times N_c}$ can be precomputed in the offline stage before evaluating the PROM in the online stage. Such a precomputation is impossible for general non-linear systems. The low-dimensional state must first be *lifted* to the full-dimensional state (via $\mathbf{g}(\widehat{\mathbf{q}}_c)$) to evaluate the non-linear terms $\mathbf{f}(\cdot,\ \mathrm{t})$. In the case of Galerkin projection, this term must then be projected onto the space tangent to the trial space before integrating the low-dimensional ODE in time. In the case of LSPG and MP-LSVT PROMs solved via the normal equations, the time-variant test basis must be computed from the residual Jacobian. Despite the fact that the resulting low-dimensional system may be less expensive to temporally integrate, these additional operations often outweigh any cost savings. Particularly for complex multi-physics systems, the evaluation of $\mathbf{f}(\cdot,\ \mathrm{t})$ accounts for the vast majority of the solver cost, and failing to reduce this cost often fails to reduce the

PROM cost below that of the FOM.

To achieve the intended goal of significantly reducing the cost of evaluating the model, the PROM's dependence on the full-order dimension $N$ must be eliminated. Techniques which seek this goal are generally referred to as *hyper-reduction* methods. Most prevalent and mature are so-called *sparse sampling* approaches, which evaluate the governing equations or non-linear terms at a small number of carefully selected degrees of freedom. Solutions which implement this methodology are referred to here as hyper-reduced PROMs (HPROMs). Alternatively, non-hyper-reduced PROMs are referred to as *unsampled* PROMs.

This thesis focuses on sparse-sampling methods for linear subspace PROMs of the *approximate-then-project* type, namely the discrete empirical interpolation method [133] and gappy proper orthogonal decomposition [134]. Work on *project-then-approximate* approaches, such as cubature methods [135, 136] and the energy-conserving sampling and weighting method [137], show promise in enhancing the accuracy of hyper-reduced PROMs. To date, however, these methods have been analyzed almost exclusively for finite-element structural dynamics models. Their application to projection-based PROMs of hyperbolic fluid flow systems is still in its early stages [122], and is not analyzed here. Neural network approaches to hyper-reduction [138] and hyper-reduction for non-linear manifold PROMs [139] have been proposed, but are also not analyzed here.

Before proceeding, a *sampling operator* is defined as $\mathbf{S} := [\mathbf{e}_{s_1},\ \mathbf{e}_{s_2},\ \dots\ ,\ \mathbf{e}_{s_{N_s}}]^\top \in \mathbb{R}^{N_s \times N}$, which is composed of $N_s$ unique canonical unit vectors $\mathbf{e}_s \in \mathbb{R}^N$. For example, given $N = 5$, $N_s = 3$, $\mathbf{S} = [\mathbf{e}_1,\ \mathbf{e}_3,\ \mathbf{e}_4]$, and a vector $\mathbf{y} = [y_1,\ \dots\ ,\ y_5]^\top$, the sampling operation $\mathbf{Sy}$ is computed as

$$\mathbf{Sy} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_3 \\ y_4 \end{bmatrix}. \tag{4.2}$$

The *sample set* is defined as $\mathcal{S} := \{s_1,\ s_2,\ \ldots\ ,\ s_{N_s}\}$, $|\mathcal{S}| = N_s$ containing the indices of those degrees of freedom which are sampled. For the above example, this set would thus be $\mathcal{S} = \{1,\ 3,\ 4\}$.

Such sampling operations are central to each sparse-sampling method described here, and how the sampling indices are selected may have a drastic effect on the accuracy and robustness of the hyper-reduced PROM. Several algorithms for selecting sampling indices are described later in Section 4.5. Discussions of missing point estimation for Galerkin PROMs and collocation for LSPG and MP-LSVT PROMs follow, along with by gappy POD for each PROM method.

## 4.1 Missing Point Estimation for Galerkin PROMs

Hyper-reduction via missing point estimation (MPE) was introduced by Astrid and coworkers, first for PROMs of a linear time-varying 2D heat conduction problem [140] and later for a non-linear model of a glass melt feeder [141]. It aims to reduce the computational expense of non-linear PROMs by sampling the governing semi-discrete ODE, followed by Galerkin projection of the ODE via the sampled trial basis. The first step can be written by applying the sampling operator to the full-order ROM ODE (Eq. 3.20)

$$\mathbf{S}\mathbf{H}_r^{-1}\left[\mathbf{H}_c\mathbf{J}_{\varphi,c}\frac{\mathrm{d}\widehat{\mathbf{q}}_c}{\mathrm{dt}} - \mathbf{f}\left(\widetilde{\mathbf{q}}_c,\ \mathrm{t}\right)\right] = \mathbf{0} \tag{4.3}$$

Projection by $\mathbf{S}\mathbf{J}_{\varphi,c}$ and rearranging terms leads to the equation

$$\frac{\mathrm{d}\widehat{\mathbf{q}}_c}{\mathrm{dt}} = \left[\left[\mathbf{S}\mathbf{J}_{\varphi,c}\right]^\top \mathbf{S}\mathbf{H}_r^{-1}\mathbf{H}_c\mathbf{J}_{\varphi,c}\right]^{-1}\left[\mathbf{S}\mathbf{J}_{\varphi,c}\right]^\top \mathbf{S}\mathbf{H}_r^{-1}\mathbf{f}\left(\widetilde{\mathbf{q}}_c,\ \mathrm{t}\right) \tag{4.4}$$

In this work, hyper-reduction is only performed for PROMs utilizing linear trial spaces, so substituting $\mathbf{J}_{\varphi,c} = \mathbf{U}_c$ leads to the form

$$\frac{\mathrm{d}\widehat{\mathbf{q}}_c}{\mathrm{dt}} = \left[\left[\mathbf{S}\mathbf{U}_c\right]^\top \mathbf{S}\mathbf{H}_r^{-1}\mathbf{H}_c\mathbf{U}_c\right]^{-1}\left[\mathbf{S}\mathbf{U}_c\right]^\top \mathbf{S}\mathbf{H}_r^{-1}\mathbf{f}\left(\widetilde{\mathbf{q}}_c,\ \mathrm{t}\right) \tag{4.5}$$

In the case $\mathbf{H}_c = \mathbf{H}_r = \mathbf{I}$, this simplifies further to

$$\frac{\mathrm{d}\widehat{\mathbf{q}}_c}{\mathrm{dt}} = \left[\left[\mathbf{S}\mathbf{U}_c\right]^\top \mathbf{S}\mathbf{U}_c\right]^{-1} \left[\mathbf{S}\mathbf{U}_c\right]^\top \mathbf{S}\mathbf{f}\left(\widetilde{\mathbf{q}}_c,\ \mathrm{t}\right) \tag{4.6}$$

Note that the matrix inverse term is not equal to the identity matrix, as it did in the unsampled Galerkin PROM in Eq. 3.26, as the matrix $\mathbf{S}\mathbf{U}_c$ has no guarantee of orthonormality. However, the matrix $\left[\left[\mathbf{S}\mathbf{U}_c\right]^\top \mathbf{S}\mathbf{U}_c\right]^{-1} \left[\mathbf{S}\mathbf{U}_c\right]^\top \in \mathbb{R}^{N_c \times N_s}$ (or the scaled equivalent for Eq. 4.5) may be precomputed in the offline stage.

The operation $\mathbf{S}\mathbf{f}\left(\widetilde{\mathbf{q}}_c,\ \mathrm{t}\right)$ is the key to cost reduction via the MPE method, whereby only $N_s$ elements of the non-linear function $\mathbf{f}\left(\cdot,\ \mathrm{t}\right)$ must be evaluated. Similar operations underpin all sparse sampling hyper-reduction methods for PROMs of non-linear systems. In the case $N_s \ll N$, the cost of evaluating the non-linear terms may be drastically reduced. Further, only those mesh elements which are required to compute $\mathbf{S}\mathbf{f}\left(\widetilde{\mathbf{q}}_c,\ \mathrm{t}\right)$ must be allocated, potentially greatly reducing the memory consumption of the hyper-reduced PROM relative to the FOM or unsampled PROM. This *sample mesh* concept will be discussed at greater length in Section 4.6.

A similar method was proposed by Bos *et al.* [142] in the context of systems governed by an explicit state space update $\mathbf{q}_c^n = \mathbf{f}\left(\mathbf{q}_c^{n-1},\ \mathrm{t}\right)$. Their method was proposed at the same time as Astrid [140] and deserves similar credit as it was formulated for general non-linear systems. However, their derivation lacks generality to arbitrary PDEs, and is only equivalent to that of MPE in the specific case of the solution of Eq 4.5 via an explicit time integrator.

## 4.2 Collocation for LSPG and MP-LSVT PROMs

Hyper-reduction of the residual-based PROMs (LSPG and MP-LSVT) by collocation derives from the work by LeGresley [143], which applies a similar method to that proposed by MPE for the cost reduction of general non-linear least-squares problems. Given the

fully-discrete residual $\mathbf{r}$, the least-squares collocation problem is stated as

$$\widehat{\mathbf{q}}_c^n = \underset{\mathbf{y}\in\mathbb{R}^{N_c}}{argmin} \left\| \mathbf{S}\mathbf{H}_r^{-1}\mathbf{r}\left(\mathbf{y}\right)\right\|_2 . \tag{4.7}$$

By this process, the least-squares minimization problem must only be computed based on the $N_s$ sampled elements of the residual. The original work by LeGresley used this procedure exclusively for the solution of steady systems with the goal of accelerating multi-disciplinary optimization problems. The work by Carlberg *et al.* [68] extended this process to LSPG PROMs, although it was presented as an inferior approach compared to the Gauss–Newton with approximated tensors (GNAT) approach proposed in their paper. The GNAT method is detailed in Section 4.3. Collocation hyper-reduction for LSPG and MP-LSVT PROMs is addressed here.

The solution of Eq. 4.7 by Gauss–Newton is given as

$$\delta\widehat{\mathbf{q}}_c^k = \underset{\mathbf{y}\in\mathbb{R}^{N_c}}{argmin} \left\| \mathbf{S}\mathbf{H}_r^{-1}\left[\widehat{\mathbf{J}}_{r,c}^k\mathbf{y} - \mathbf{r}\left(\widehat{\mathbf{q}}_c^k\right)\right]\right\|_2 , \tag{4.8}$$

$$\widehat{\mathbf{q}}_c^{k+1} = \widehat{\mathbf{q}}_c^k + \alpha\left[\delta\widehat{\mathbf{q}}_c^k\right] . \tag{4.9}$$

The normal equations for Eq. 4.8 is thus given by

$$\left[\mathbf{W}_c^k\right]^\top \mathbf{W}_c^k\delta\widehat{\mathbf{q}}_c^k = -\left[\mathbf{W}_c^k\right]^\top \mathbf{S}\mathbf{H}_r^{-1}\mathbf{r}\left(\widehat{\mathbf{q}}_c^k\right) , \tag{4.10}$$

where the test basis is defined as

$$\mathbf{W}_c^k := \mathbf{S}\mathbf{H}_r^{-1}\mathbf{J}_{r,c}^k\mathbf{H}_c\mathbf{J}_{\varphi,c}^k. \tag{4.11}$$

In the case of a linear trial space, the test basis simplifies to

$$\mathbf{W}_c^k := \mathbf{S}\mathbf{H}_r^{-1}\mathbf{J}_{r,c}^k\mathbf{H}_c\mathbf{U}_c. \tag{4.12}$$

In this formulation, note that only $N_s$ elements of the non-linear residual term must by

calculated for $\mathbf{SH}_r^{-1}\mathbf{r}(\cdot)$. Further, only $N_s$ rows of the the residual Jacobian must be calculated for $\mathbf{SH}_r^{-1}\mathbf{J}_{r,c}^k$. As with MPE, this subsampling operation has the capability to drastically reduce the cost of evaluating the LSPG PROM, as the calculation of the non-linear residual and its Jacobian usually dominates the computational cost for solvers of even moderate complexity.

The process for computing the collocated MP-LSVT PROM is, as implied by the unsampled PROM derivation in Section 3.3, extremely similar to that of collocated LSPG. The non-linear least-squares problem is simply reframed to minimize the residual with respect to the target space modal coefficients, i.e.

$$\widehat{\mathbf{q}}_p^n = \underset{\mathbf{y}\in\mathbb{R}^{N_p}}{argmin} \left\|\mathbf{SH}_r^{-1}\mathbf{r}(\mathbf{y})\right\|_2. \tag{4.13}$$

Sparing the reader repetitive derivation details, for a linear trial space the normal equations of the Gauss–Newton solution of Eq. 4.13 is given by

$$\left[\mathbf{W}_p^k\right]^\top \mathbf{W}_p^k \delta\widehat{\mathbf{q}}_p^k = -\left[\mathbf{W}_p^k\right]^\top \mathbf{SH}_r^{-1}\mathbf{r}\left(\widehat{\mathbf{q}}_p^k\right), \tag{4.14}$$

where the test basis is defined as

$$\mathbf{W}_p^k := \mathbf{SH}_r^{-1}\mathbf{J}_{r,p}^k\mathbf{H}_p\mathbf{U}_p. \tag{4.15}$$

Again, note that only $N_s$ rows of the non-linear residual $\mathbf{SH}_r^{-1}\mathbf{r}(\cdot)$ and residual Jacobian $\mathbf{SH}_r^{-1}\mathbf{J}_{r,p}$ must be computed to solve the collocated MP-LSVT PROM.

## 4.3 DEIM and GNAT

Both hyper-reduction methods presented so far, MPE and collocation, have enabled cost reduction of PROMs simply by sampling the original governing PROM equations and computing the solution normally. However, as shown by Carlberg et al. [68], these methods often generate inaccurate or unstable solutions for convection-dominated fluid flow systems. Alternatively, several methods have been proposed which compute data-driven,

approximate reconstructions of non-linear terms based on a small number of samples of the functions. As such, these approaches attempt to incorporate the action of the full-field function, instead of discarding unsampled elements entirely as in MPE or collocation. Because of this, they are sometimes referred to as *function reconstruction* methods.

Two well-established function reconstruction methods are the discrete empirical interpolation method (DEIM) and gappy proper orthogonal decomposition (gappy POD). The two are closely related, but were originally developed for different applications. DEIM, introduced by Chaturantabut and Sorensen [133], is a discrete formulation of the empirical interpolation method (EIM) [144]. It introduces an approximation of non-linear functions of the form,

$$\mathbf{r}(\mathbf{y}) \approx \widetilde{\mathbf{r}}(\mathbf{y}) := \boldsymbol{\Psi} \left[\mathbf{S}\boldsymbol{\Psi}\right]^{-1} \mathbf{S}\mathbf{r}(\mathbf{y}), \tag{4.16}$$

where the residual function $\mathbf{r}$ is used for the sake of notational simplicity, but note that this method extends to approximation of any non-linear function. Indeed, the original work by Chaturantabut and Sorensen [133] computed approximations of the non-linear function $\mathbf{f}(\cdot, \mathrm{t})$ for Galerkin PROMs, as will be discussed in Section 4.3.1. The matrix $\boldsymbol{\Psi} := [\boldsymbol{\psi}_1, \ \boldsymbol{\psi}_2, \ \ldots \ , \ \boldsymbol{\psi}_{N_r}] \in \mathbb{R}^{N \times N_r}$ is an orthonormal basis, usually generated by POD from FOM snapshots of the non-linear function $\mathbf{r}$. It is assumed that the matrix $\mathbf{S}\boldsymbol{\Psi} \in \mathbb{R}^{N_s \times N_r}$ has full rank. Again, in computing $\widetilde{\mathbf{r}}$, the non-linear function $\mathbf{r}$ must only be sampled at $N_s$ degrees of freedom, instead of its full dimension $N$.

Note that the number of DEIM basis modes is equal to the number of sampled degrees of freedom, i.e., $N_s = N_r$. By this formulation, the non-linear function is interpolated exactly at $N_s < N$ degrees of freedom and interpolated approximately at all other degrees of freedom. The error in this approximation is bounded [133] by the inequality

$$\left\|\mathbf{r}(\mathbf{y}) - \widetilde{\mathbf{r}}(\mathbf{y})\right\|_2 \leq \left\|\left[\mathbf{S}\boldsymbol{\Psi}\right]^{-1}\right\|_2 \left\|\left[\mathbf{I} - \boldsymbol{\Psi}\boldsymbol{\Psi}^\top\right] \mathbf{r}(\mathbf{y})\right\|_2, \tag{4.17}$$

where the first norm term on the right-hand side is a measure of the *sampling error*, and the second norm term is the *projection error*. As will be discussed shortly, various methods of selecting the sampling degrees of freedom often attempt to minimize these

sources of error.

Again, DEIM is restricted to the case where $N_s = N_r$. For practical engineering systems, $N_r \ll N$ generally, and such extreme sparse sampling may result in high interpolation error at the unsampled degrees of freedom. Furthermore, Peherstorfer *et al.* [76] show that increasing $N_r$ can lead to an unstable increase in the interpolation error if $N_s = N_r$. The gappy proper orthogonal decomposition (gappy POD) offers a solution to these issues. Gappy POD was originally developed long before the advent of DEIM to approximate full field data from a few sparse samples [134], and has seen extensive use in sparse sensor placement [145, 146] and extensions of missing point estimation [147]. As such, gappy POD has also been used to approximate non-linear functions in dynamical systems. The method takes a very similar approach to DEIM, but relaxes the restriction on the number of sampled degrees of freedom, allowing $N_s \geq N_r$. The approximation becomes a least-squares regression of the form

$$\mathbf{r}(\mathbf{y}) \approx \widetilde{\mathbf{r}}(\mathbf{y}) := \boldsymbol{\Psi} \left[\mathbf{S}\boldsymbol{\Psi}\right]^{+} \mathbf{S}\mathbf{r}(\mathbf{y}), \tag{4.18}$$

where the operation $[\cdot]^{+}$ indicates the Moore–Penrose inverse (or pseudo-inverse). Again, this assumes $\mathbf{S}\boldsymbol{\Psi}$, which is now a rectangular matrix, has full column rank. Similarly to Eq. 4.17, the gappy POD regression error is bounded [76] by

$$\left\|\mathbf{r}(\mathbf{y}) - \widetilde{\mathbf{r}}(\mathbf{y})\right\|_2 \leq \left\|\left[\mathbf{S}\boldsymbol{\Psi}\right]^{+}\right\|_2 \left\|\left[\mathbf{I} - \boldsymbol{\Psi}\boldsymbol{\Psi}^{\top}\right] \mathbf{r}(\mathbf{y})\right\|_2. \tag{4.19}$$

For the remainder of this thesis, discussion of hyper-reduction is restricted to gappy POD, as it is more general than DEIM. The reader is simply reminded that in the specific case of $N_r = N_c$, gappy POD is equivalent to DEIM.

The approximation of non-linear functions by Eq. 4.18 can be applied to PROM formulations to vastly reduce the computational cost of evaluating non-linear terms arising from the discretization of the governing equations. Hyper-reduction of Galerkin PROMs by gappy POD is described first, followed by hyper-reduction of LSPG and MP-LSVT PROMs.

### 4.3.1 Gappy POD for Galerkin PROMs

For Galerkin PROMs, the traditional method of gappy POD approximates the non-linear terms $\mathbf{f}(\cdot,\ \mathrm{t})$ as

$$\mathbf{H}_r^{-1}\widetilde{\mathbf{f}}(\mathbf{q}_c,\ \mathrm{t}) := \boldsymbol{\Psi}\left[\mathbf{S}\boldsymbol{\Psi}\right]^{+}\mathbf{S}\mathbf{H}_r^{-1}\mathbf{f}(\mathbf{q}_c,\ \mathrm{t}). \tag{4.20}$$

Substituting this approximation into the linear subspace Galerkin PROM (Eq. 3.25) results in the hyper-reduced PROM

$$\frac{\mathrm{d}\widehat{\mathbf{q}}_c}{\mathrm{dt}} = \left[\mathbf{U}_c^{\top}\mathbf{H}_r^{-1}\mathbf{H}_c\mathbf{U}_c\right]^{-1}\mathbf{U}_c^{\top}\boldsymbol{\Psi}\left[\mathbf{S}\boldsymbol{\Psi}\right]^{+}\mathbf{S}\mathbf{H}_r^{-1}\mathbf{f}(\widetilde{\mathbf{q}}_c,\ \mathrm{t}). \tag{4.21}$$

The low-dimensional matrix $\left[\mathbf{U}_c^{\top}\mathbf{H}_r^{-1}\mathbf{H}_c\mathbf{U}_c\right]^{-1}\mathbf{U}_c^{\top}\boldsymbol{\Psi}\left[\mathbf{S}\boldsymbol{\Psi}\right]^{+} \in \mathbb{R}^{N_c\times N_s}$ can be pre-computed in the offline stage and loaded from disk prior to evaluating the unsteady PROM. Further simplifying, if $\mathbf{H}_c = \mathbf{H}_r = \mathbf{I}$, the formulation is reduced to

$$\frac{\mathrm{d}\widehat{\mathbf{q}}_c}{\mathrm{dt}} = \mathbf{U}_c^{\top}\boldsymbol{\Psi}\left[\mathbf{S}\boldsymbol{\Psi}\right]^{+}\mathbf{S}\mathbf{f}(\widetilde{\mathbf{q}}_c,\ \mathrm{t}), \tag{4.22}$$

where again, the low-dimensional matrix $\mathbf{U}_c^{\top}\boldsymbol{\Psi}\left[\mathbf{S}\boldsymbol{\Psi}\right]^{+} \in \mathbb{R}^{N_c\times N_s}$ can be precomputed. Gappy POD Galerkin PROMs of this form have been successfully applied to a host of non-linear fluid flow problems [148, 149, 150, 151, 152].

An alternative method of computing the gappy POD Galerkin PROM is posed by Peherstorfer [153], who reframes the fully discrete O$\Delta$E in the form

$$\mathbf{q}_c^{n-1} - \mathbf{f}_{\mathbf{r}}(\mathbf{q}_c^n, \mathrm{t}) = \mathbf{0}, \tag{4.23}$$

where all terms of the time integrator, except for $\mathbf{q}_c^{n-1}$, have been grouped with the non-linear terms $\mathbf{f}(\cdot,\ \mathrm{t})$ into the lumped $\mathbf{f}_{\mathbf{r}}(\cdot,\ \mathrm{t})$. Note that this is more general than the explicit update discussed by Bos *et al.* [142], as it extends to implicit time integrators as well. This form is, at first glance, confusing, as the solution at the next time step, $\mathbf{q}_c^n$, is usually sought as a function of past time steps, not the other way around as in Eq. 4.23. However, this simply reflects the nature of implicit time integrators, in which the non-

linear terms $\mathbf{f}(\cdot,\ \mathrm{t})$ are evaluated at future time steps. For example, the backward Euler (BDF1) time integrator would compute $\mathbf{f_r}(\cdot,\ \mathrm{t})$ as

$$\mathbf{f_r}(\mathbf{q}_c^n,\ \mathrm{t}) := \mathbf{q}_c^n - \Delta \mathrm{t}\mathbf{f}(\mathbf{q}_c^n,\ \mathrm{t}). \quad \text{(BDF1)} \tag{4.24}$$

The approximate non-linear term is then approximated as

$$\mathbf{H}_r^{-1}\mathbf{f_r}(\mathbf{q}_c) \approx \widetilde{\mathbf{f}}_\mathbf{r}(\mathbf{q}_c) := \mathbf{\Psi}\left[\mathbf{S\Psi}\right]^+ \mathbf{S}\mathbf{H}_r^{-1}\mathbf{f_r}(\mathbf{q}_c). \tag{4.25}$$

Substitution into Eq. 4.23 and projection by the tangent trial space leads to an alternative hyper-reduced Galerkin PROM of the form

$$\widehat{\mathbf{q}}_c^{n-1} - \left[\mathbf{J}_{\varphi,c}^\top \mathbf{H}_r^{-1}\mathbf{H}_c\mathbf{J}_{\varphi,c}\right]^{-1}\mathbf{H}_c\mathbf{J}_{\varphi,c}^\top \widetilde{\mathbf{f}}_\mathbf{r}(\widetilde{\mathbf{q}}_c^n) = \mathbf{0}. \tag{4.26}$$

As mentioned previously, the above formulation is fairly non-standard and can be quite confusing for PROM practitioners. However, as will be discussed in Sections 4.4, this formulation lends itself to a simplification in which the regression basis $\mathbf{\Psi}$ can be equated to the solution trial basis $\mathbf{U}_c$ or $\mathbf{U}_p$.

### 4.3.2 GNAT for LSPG and MP-LSVT PROMs

Applying gappy POD to LSPG and MP-LSVT PROMs takes an approach akin to that of collocation. Instead of constructing a least-squares regression for the sampled non-linear function $\mathbf{f}(\cdot,\ \mathrm{t})$, however, the complete fully-discrete residual $\mathbf{r}$ is approximated and substituted into the residual minimization problem given by Eq. 3.28 or 3.48. The procedure is examined for LSPG, which was first formulated by Carlberg and coworkers [67, 68], who referred to the procedure as Gauss–Newton with approximated tensors (GNAT). Substituting the residual approximation into Eq. 3.28 results in the non-linear least-squares problem

$$\widehat{\mathbf{q}}_c^n = \underset{\mathbf{y}\in\mathbb{R}^{N_c}}{argmin}\ \left\|\mathbf{\Psi}\left[\mathbf{S\Psi}\right]^+ \mathbf{S}\mathbf{H}_r^{-1}\mathbf{r}(\mathbf{y})\right\|_2. \tag{4.27}$$

Solving Eq. 4.27 by Gauss–Newton results in the iterative process

$$\delta\widehat{\mathbf{q}}_c^k = \underset{\mathbf{y}\in\mathbb{R}^{N_c}}{argmin} \left\| \mathbf{\Psi} \left[\mathbf{S\Psi}\right]^+ \mathbf{SH}_r^{-1} \left[ \widehat{\mathbf{J}}_{r,c}^k \mathbf{y} - \mathbf{r}\left(\widehat{\mathbf{q}}_c^k\right) \right] \right\|_2, \tag{4.28}$$

$$\widehat{\mathbf{q}}_c^{k+1} = \widehat{\mathbf{q}}_c^k + \alpha \left[\delta\widehat{\mathbf{q}}_c^k\right]. \tag{4.29}$$

Solution of Eq. 4.28 via the normal equations is again given by a Petrov–Galerkin PROM of the form

$$\left[\mathbf{W}_c^k\right]^\top \mathbf{W}_c^k \delta\widehat{\mathbf{q}}_c^k = -\left[\mathbf{W}_c^k\right]^\top \mathbf{\Psi} \left[\mathbf{S\Psi}\right]^+ \mathbf{SH}_r^{-1} \mathbf{r}\left(\widehat{\mathbf{q}}_c^k\right), \tag{4.30}$$

where the test basis is defined as

$$\mathbf{W}_c^k := \mathbf{\Psi} \left[\mathbf{S\Psi}\right]^+ \mathbf{SH}_r^{-1} \mathbf{J}_{r,c}^k \mathbf{H}_c \mathbf{J}_{\varphi,c}^k. \tag{4.31}$$

The original work on GNAT [68] suggests an alternative formulation, whereby separate regression bases are computed for the residual and residual Jacobian, denoted by $\mathbf{\Psi_r} \in \mathbb{R}^{N\times N_r}$ and $\mathbf{\Psi_J} \in \mathbb{R}^{N\times N_J}$ respectively. Equation 4.28 can thus be written as

$$\delta\widehat{\mathbf{q}}_c^k = \underset{\mathbf{y}\in\mathbb{R}^{N_c}}{argmin} \left\| \mathbf{\Psi_J} \left[\mathbf{S\Psi_J}\right]^+ \mathbf{SH}_r^{-1} \widehat{\mathbf{J}}_{r,c}^k \mathbf{y} - \mathbf{\Psi_r} \left[\mathbf{S\Psi_r}\right]^+ \mathbf{SH}_r^{-1} \mathbf{r}\left(\widehat{\mathbf{q}}_c^k\right) \right\|_2. \tag{4.32}$$

In practice, choosing $\mathbf{\Psi_r} \neq \mathbf{\Psi_J}$ has not been observed to generate HPROMs which are drastically better than those generated by choosing $\mathbf{\Psi_r} = \mathbf{\Psi_J}$. The added computational complexity makes it difficult to justify implementing Eq. 4.32. As such, for all results presented in this thesis, $\mathbf{\Psi_r} = \mathbf{\Psi_J}$.

Hyper-reduction of MP-LSVT PROMs via gappy POD follows a similar method, beginning from the fully-discrete residual minimization with respect to the target state latent variables,

$$\widehat{\mathbf{q}}_p^n = \underset{\mathbf{y}\in\mathbb{R}^{N_p}}{argmin} \left\| \mathbf{\Psi} \left[\mathbf{S\Psi}\right]^+ \mathbf{SH}_r^{-1} \mathbf{r}\left(\mathbf{y}\right) \right\|_2. \tag{4.33}$$

Solution via Gauss–Newton and the normal equations results in a similar hyper-reduced

Petrov–Galerkin formulation of

$$\left[\mathbf{W}_p^k\right]^\top \mathbf{W}_p^k \delta\widehat{\mathbf{q}}_p^k = -\left[\mathbf{W}_p^k\right]^\top \boldsymbol{\Psi}\left[\mathbf{S}\boldsymbol{\Psi}\right]^+ \mathbf{S}\mathbf{H}_r^{-1}\mathbf{r}\left(\widehat{\mathbf{q}}_p^k\right), \tag{4.34}$$

with the test basis

$$\mathbf{W}_p^k := \boldsymbol{\Psi}\left[\mathbf{S}\boldsymbol{\Psi}\right]^+ \mathbf{S}\mathbf{H}_r^{-1}\mathbf{J}_{r,p}^k \mathbf{H}_p \mathbf{J}_{\varphi,p}^k. \tag{4.35}$$

Only $[\mathbf{S}\boldsymbol{\Psi}]^+ \in \mathbb{R}^{N_r \times N_s}$ is pre-computed offline. Although the matrix $\boldsymbol{\Psi}[\mathbf{S}\boldsymbol{\Psi}]^+ \in \mathbb{R}^{N \times N_s}$ can be pre-computed offline, a matrix of $N \times N_s$ elements can be prohibitively large for high-dimensional systems and may not be possible to store in memory. Further, substitution of Eq. 4.35 into Eq. 4.34 and recognizing the product $\boldsymbol{\Psi}^\top\boldsymbol{\Psi} = \mathbf{I}$ eliminates the need to precompute $\boldsymbol{\Psi}[\mathbf{S}\boldsymbol{\Psi}]^+$.

As with collocation, note that computing the sampled residual $\mathbf{S}\mathbf{r}(\cdot)$ only requires that the residual be evaluated at $N_s$ degrees of freedom. Further, in Eqs. 4.31 and 4.35 only the corresponding $N_s$ rows of the residual Jacobians $\mathbf{J}_c$ and $\mathbf{J}_p$ need to be computed. Thus, the hyper-reduced PROM is truly independent of the FOM dimension $N$. When $N_s \ll N$ and evaluation of the non-linear terms accounts for a majority of the computational burden, this sparse sampling has the potential to drastically reduce the cost of the hyper-reduced PROM relative to the FOM. Prior work by the author has recorded four orders of magnitude cost reduction for a three-dimensional model rocket combustor [125]. Up to five orders of magnitude cost reduction has been recorded [70], consuming a few core-hours to compute a hyper-reduced LSPG PROM where the equivalent FOM simulation consumed over two thousand core-weeks. Such cost savings bring projection-based PROMs into the realm of possibility for many-query applications.

## 4.4 Regression Basis Calculation

The question of how the regression basis $\boldsymbol{\Psi}$ is computed is now addressed. Although the process generally mirrors that of computing the solution trial bases $\mathbf{U}_c$ or $\mathbf{U}_p$, there are several approaches for computing $\boldsymbol{\Psi}$ which are suggested by the literature and deserve

explanation.

### 4.4.1 Galerkin RHS Approximation

In the case of gappy POD Galerkin PROMs, the two approximations of non-linear terms in the ROM ODE/OΔE have been outlined, but are restated here. The traditional method of gappy POD for Galerkin PROMs [133] computes a regression for the non-linear function $\mathbf{f}(\cdot,\ \mathrm{t})$ of the form

$$\mathbf{H}_f^{-1}\mathbf{f}(\mathbf{q}_c,\ \mathrm{t}) \approx \widetilde{\mathbf{f}}(\mathbf{q}_c, \mathrm{t}) := \boldsymbol{\Psi}\left[\mathbf{S}\boldsymbol{\Psi}\right]^{+}\mathbf{S}\mathbf{H}_f^{-1}\mathbf{f}(\mathbf{q}_c,\ \mathrm{t}) \tag{4.36}$$

In this case, as with computing the solution trial bases, snapshots of the unsteady non-linear term field $\mathbf{f}(\mathbf{q}_c,\ \mathrm{t})$ are collected from FOM simulations. Unlike the process for the solution trial bases, a reference state about which snapshots are centered is not defined. However, disparate scales of the non-linear terms must be accounted for, and some scaling operations should be computed via $\mathbf{H}_f^{-1}$. The snapshot matrix is thus aggregated as

$$\mathbf{F} = \left[\mathbf{H}_f^{-1}\mathbf{f}\left(\mathbf{q}_c\left(\mathrm{t}^0\right),\ \mathrm{t}^0\right),\ \mathbf{H}_f^{-1}\mathbf{f}\left(\mathbf{q}_c\left(\mathrm{t}^1\right),\ \mathrm{t}^1\right),\ \ldots,\ \mathbf{H}_f^{-1}\mathbf{f}\left(\mathbf{q}_c\left(\mathrm{T}\right),\ \mathrm{t}^{\mathrm{T}}\right)\right] \tag{4.37}$$

The regression basis is then computed from the POD of the snapshot matrix $\mathbf{F}$, extracting the leading $N_r$ singular vectors as $\boldsymbol{\Psi}$.

### 4.4.2 State Approximation

The method of Peherstorfer [153], as previously discussed in Section 4.3, reframes the fully-discrete residual as

$$\mathbf{q}_c^{n-1} - \mathbf{f}_{\mathbf{r}}(\mathbf{q}_c^n,\ \mathrm{t}^n) = \mathbf{0} \tag{4.38}$$

Although this formulation may seem non-intuitive, it is only a grouping of terms arising from the implicit time integration of the governing ODE. Interestingly, treating the lumped non-linear term $\mathbf{f}_{\mathbf{r}}(\cdot,\ \mathrm{t})$ as the non-linear function to be approximated makes the

connection

$$\mathbf{q}_c^{n-1} \approx \widetilde{\mathbf{f}}_{\mathbf{r}}\left(\mathbf{q}_c,\ \mathrm{t}\right) := \boldsymbol{\Psi}\left[\mathbf{S}\boldsymbol{\Psi}\right]^{+}\mathbf{S}\mathbf{f}_{\mathbf{r}}\left(\mathbf{q}_c,\ \mathrm{t}\right) \tag{4.39}$$

At first glance, this formulation might suggests that the solution trial basis can be used as the regression basis; that is, $\boldsymbol{\Psi} = \mathbf{U}_c$. However, this is not entirely accurate, as the state representation is given as $\widetilde{\mathbf{q}}_c := \overline{\mathbf{q}}_c + \mathbf{H}_c\mathbf{U}_c\widehat{\mathbf{q}}_c$. Attempting to directly approximate this vector by gappy POD results in

$$\mathbf{q}_c \approx \overline{\mathbf{q}}_c + \mathbf{H}_c\mathbf{U}_c\left[\mathbf{S}\mathbf{H}_c\mathbf{U}_c\right]^{+}\mathbf{S}\left[\mathbf{q}_c - \overline{\mathbf{q}}_c\right] \tag{4.40}$$

which is not appropriate for use in Eq. 4.39, especially in the context of time-variant $\mathbf{U}_c$ or $\mathbf{S}$ (adaptive methods). The governing system in Eq. 4.23 requires some manipulation to achieve a more appropriate gappy POD approximation.

Discussion is restricted to linear multi-step methods, which take the general form

$$\mathbf{q}_c^n + \sum_{i=1}^{s} a_i\mathbf{q}_c^{n-i} + \Delta\mathrm{t}\sum_{i=0}^{s} b_i\mathbf{f}\left(\mathbf{q}_c^{n-i},\ \mathrm{t}^{n-i}\right) = \mathbf{0} \tag{4.41}$$

where $b_i,\ a_i \in \mathbb{R}$ are tabulated coefficients for a given method of order $s$. Note that for a linear multi-step method to be consistent, the coefficients $a_i$ must satisfy

$$\sum_{i=1}^{s} a_i = -1 \tag{4.42}$$

Substituting the linear trial space representation $\mathbf{q}_c := \overline{\mathbf{q}}_c + \mathbf{H}_c\mathbf{U}_c\widehat{\mathbf{q}}_c$ into Eq. 4.41, and recognizing that $\overline{\mathbf{q}}_c + \overline{\mathbf{q}}_c\sum_{i=1}^{s} a_i = \mathbf{0}$ by Eq. 4.42, the fully-discrete residual can be rewritten to match the form in Eq. 4.38 as

$$\mathbf{H}_c\mathbf{U}_c\widehat{\mathbf{q}}_c^{n-1} + \mathbf{H}_c\mathbf{U}_c\left[\frac{1}{a_1}\widehat{\mathbf{q}}_c^{n} + \sum_{i=2}^{s}\frac{a_i}{a_1}\widehat{\mathbf{q}}_c^{n-i}\right] + \Delta\mathrm{t}\sum_{i=0}^{s}\frac{b_i}{a_1}\mathbf{f}\left(\widetilde{\mathbf{q}}_c^{n-i},\ \mathrm{t}^{n-i}\right) = \mathbf{0} \tag{4.43}$$

where the lumped non-linear term is defined as

$$-\mathbf{f}_{\mathbf{r}}\left(\widehat{\mathbf{q}}_c^{n},\ \mathrm{t}^{n}\right) := \mathbf{H}_c\mathbf{U}_c\left[\frac{1}{a_1}\widehat{\mathbf{q}}_c^{n} + \sum_{i=2}^{s}\frac{a_i}{a_1}\widehat{\mathbf{q}}_c^{n-i}\right] + \Delta\mathrm{t}\sum_{i=0}^{s}\frac{b_i}{a_1}\mathbf{f}\left(\widetilde{\mathbf{q}}_c^{n-i},\ \mathrm{t}^{n-i}\right) \tag{4.44}$$

Given the resulting equivalence

$$\mathbf{U}_c \widehat{\mathbf{q}}_c^{n-1} = \mathbf{H}_c^{-1} \mathbf{f}_\mathbf{r} \left( \widehat{\mathbf{q}}_c^n,\ \mathrm{t}^n \right), \tag{4.45}$$

it now appears appropriate to approximate the lumped non-linear term as $\widetilde{\mathbf{f}}_\mathbf{r}(\cdot) := \mathbf{\Psi} \left[ \mathbf{S}\mathbf{\Psi} \right]^+ \mathbf{S}\mathbf{H}_c^{-1} \mathbf{f}_\mathbf{r}(\cdot)$ by choosing $\mathbf{\Psi} = \mathbf{U}_c$. Although this derivation may seem tedious to make this distinction solely in the context of linear multi-step time integrators, recall that the attempt to naively set $\mathbf{\Psi} = \mathbf{U}_c$ for any time integration scheme resulted in the unusable gappy POD approximation given in Eq. 4.40.

This approach is valid for Galerkin, LSPG, and MP-LSVT PROMs alike, as each is derived from the original, conservative governing equations and computes the same full-order fully-discrete ROM residual prior to projection. However, note that the conservative state gappy POD regression applied to MP-LSVT PROMs necessitates the calculation of two distinct bases: $\mathbf{U}_p$ for the target state trial space, and $\mathbf{U}_c$ strictly for the gappy POD regression. For Galerkin and LSPG, the trial basis $\mathbf{U}_c$ is simply reused for the gappy POD regression (or slightly reduced or expanded, for $N_c \neq N_r$).

### 4.4.3 Residual Approximation

For hyper-reduction of LSPG PROMs by GNAT, recall that the Gauss–Newton least-squares minimization problem takes the general form

$$\delta \widehat{\mathbf{q}}_c^k = \underset{\mathbf{y} \in \mathbb{R}^{N_c}}{argmin} \left\| \mathbf{\Psi}_\mathbf{J} \left[ \mathbf{S}\mathbf{\Psi}_\mathbf{J} \right]^+ \mathbf{S}\mathbf{H}_r^{-1} \widehat{\mathbf{J}}_{r,c}^k \mathbf{y} - \mathbf{\Psi}_\mathbf{r} \left[ \mathbf{S}\mathbf{\Psi}_\mathbf{r} \right]^+ \mathbf{S}\mathbf{H}_r^{-1} \mathbf{r} \left( \widehat{\mathbf{q}}_c^k \right) \right\|_2, \tag{4.46}$$

where the gappy POD regression via the bases $\mathbf{\Psi}_\mathbf{r} \in \mathbb{R}^{N \times N_r}$ and $\mathbf{\Psi}_\mathbf{J} \in \mathbb{R}^{N \times N_J}$ seek approximate reconstructions of the residual vector $\mathbf{r}\left( \widehat{\mathbf{q}}_c^k \right)$ and the residual Jacobian $\widehat{\mathbf{J}}_{r,c}^k = \mathbf{J}_{r,c}^k \mathbf{H}_c \mathbf{U}_c$, respectively.

The most obvious (albeit naive) approach for constructing $\mathbf{\Psi}_\mathbf{r}$ and $\mathbf{\Psi}_\mathbf{J}$ is to compute POD bases from snapshots of $\mathbf{r}\left( \widehat{\mathbf{q}}_c \right)$ and $\mathbf{J}_{r,c}^k \mathbf{H}_c \mathbf{U}_c$. However, this approach begs two important questions. First, the residual and its Jacobian are defined for each Newton iteration; should snapshots from all iterations be saved, or one, or a few? Second, saving

$N_c$ snapshots for a given Newton iteration evaluation of $\mathbf{J}_{r,c}^k \mathbf{H}_c \mathbf{U}_c$ quickly causes storage consumption and offline POD basis calculation costs to grow drastically. Is this added cost worthwhile?

In the original work on LSPG and GNAT by Carlberg *et al.* [67], several approaches are suggested. Their "Method 3" indicates the most expensive extreme: compute $\boldsymbol{\Psi}_{\mathbf{J}}$ from snapshots of $\mathbf{r}\left(\widehat{\mathbf{q}}_c^k\right)$ and $\mathbf{J}_{r,c}^k \mathbf{H}_c \mathbf{U}_c$ for *every* Newton iteration, resulting in $N_c + 1$ snapshots for every Newton iteration. This approach is quickly discarded as computationally infeasible, and no work since appears to use this method. As a step down, their "Method 2" computes $\boldsymbol{\Psi}_{\mathbf{J}}$ from snapshots of the action of the residual Jacobian on the Gauss–Newton search direction, i.e. $\mathbf{J}_{r,c}^k \mathbf{H}_c \mathbf{U}_c \delta\widehat{\mathbf{q}}_c^k$. This reduces the number of snapshots saved per Newton iteration to two, and is shown to perform well in the original work by Carlberg *et al.* [67] and later work by Bai and Wang [154]. The final and least computationally expensive method simply sets $\boldsymbol{\Psi}_{\mathbf{J}} = \boldsymbol{\Psi}_{\mathbf{r}}$, where $\boldsymbol{\Psi}_{\mathbf{r}}$ is computed from Newton iteration snapshots of the residual vector $\mathbf{r}(\cdot)$. This method is employed in later work by Carlberg *et al.* [68, 69], and is implied to be used by Lauzon *et al.* [155].

Note that the collection of snapshots for every Newton subiteration can quickly become computationally burdensome, often increasing storage requirements by more than ten times. Other works suggest that only a single snapshot should be saved, as in the work by Grimberg *et al.* [70]. They choose to collect the residual of the first Newton iteration for a given time step, arguing that the residuals for a well-converged Gauss–Newton solve quickly approach machine precision and result in poorly-conditioned POD approximations. The first Newton iteration represents the largest magnitude residuals observed for a given time step, and encourages more accurate gappy POD approximations. Similar notions are explored by Tezaur *et al.* [156] in the context of PROMs of steady-state aerodynamics problems, whereby they find that the approximation is most accurate when computed from residual Jacobians of converged solutions or pre-convergence residuals, while those computed from residuals of the converged solution perform relatively poorly.

In this work, GNAT PROMs using POD bases computed from the residual or residual Jacobian are not examined. It is the author's opinion (informed by many failed numerical

experiments) that this approach requires far too much trial-and-error and manual tuning in selecting appropriate residual snapshots and computing (often poorly-conditioned) gappy POD approximations. Further, the dependence of the residual field convergence on the FOM sparse linear solver is not experienced in the low-dimensional dense PROM solve, and it is not clear if the resulting approximation is appropriate for the entire Gauss–Newton solve.

### 4.4.4 Dual Basis Formulation

In this section, a method is proposed which circumvents several of the issues noted above for the state approximation and fully-discrete residual approximation approaches. As mentioned in passing by several authors [157, 70], it is possible to define gappy POD approximations for individual elements of the fully-discrete residual. In the most fine-grained extreme, this would entail constructing gappy POD bases for the flux, boundary condition, source, body force, and time integrator terms separately. However, this results in exceptional computational complexity and high memory consumption. This work will investigate an approach which constructs two gappy POD approximations: one for the time integrator term, and one for the general non-linear right-hand side term.

Recall that the fully-discrete residual is traditionally written as

$$\mathbf{r}\left(\mathbf{q}_c^n\right) := \dot{\widetilde{\mathbf{q}}}_c^n - \mathbf{f}\left(\mathbf{q}_c^n,\ \mathrm{t}\right) = \mathbf{0}, \tag{4.47}$$

where the $\dot{\widetilde{\mathbf{q}}}_c$ is the discrete time integrator term and $\mathbf{f}\left(\cdot,\ \mathrm{t}\right)$ encompasses all other terms associated with the spatial discretization of the governing equations. These terms can be approximated by gappy POD as

$$\dot{\mathbf{q}}_c \approx \widetilde{\dot{\mathbf{q}}}_c := \mathbf{H}_c \boldsymbol{\Psi}_c \left[\mathbf{S}\boldsymbol{\Psi}_c\right]^+ \mathbf{S}\mathbf{H}_c^{-1} \dot{\mathbf{q}}_c, \tag{4.48}$$

$$\mathbf{f}\left(\mathbf{q}_c,\ \mathrm{t}\right) \approx \widetilde{\mathbf{f}}\left(\mathbf{q}_c\right) := \mathbf{H}_f \boldsymbol{\Psi}_f \left[\mathbf{S}\boldsymbol{\Psi}_f\right]^+ \mathbf{S}\mathbf{H}_f^{-1} \mathbf{f}\left(\mathbf{q}_c\right). \tag{4.49}$$

where $\boldsymbol{\Psi}_c \in \mathbb{R}^{N \times N_c}$ and $\mathbf{H}_c \in \mathbb{R}^{N \times N}$ are the time integrator POD basis and diagonal scaling matrix, respectively. The matrices $\boldsymbol{\Psi}_f \in \mathbb{R}^{N \times N_f}$ and $\mathbf{H}_f \in \mathbb{R}^{N \times N}$ are the POD

basis and diagonal scaling matrix for the RHS function, respectively.

For the sake of brevity, the application of these approximations are only detailed for the MP-LSVT method. Inserting them into the residual-minimization problem and solving by Gauss–Newton arrives at the form

$$\delta\widehat{\mathbf{q}}_p^k = \underset{\mathbf{y}\in\mathbb{R}^{N_p}}{argmin} \left\| \mathbf{H}_r^{-1} \Big[ [\mathbf{H}_c\boldsymbol{\Psi}_c\mathbf{A} - \mathbf{H}_f\boldsymbol{\Psi}_f\mathbf{B}]\mathbf{y} + \mathbf{H}_c\boldsymbol{\Psi}_c\mathbf{a} - \mathbf{H}_f\boldsymbol{\Psi}_f\mathbf{b} \Big] \right\|_2^2, \tag{4.50}$$

where terms have been grouped as

$$\mathbf{A} := [\mathbf{S}\boldsymbol{\Psi}_c]^+ \mathbf{S}\mathbf{H}_c^{-1}\mathbf{J}_{c,p}^k\mathbf{H}_p\mathbf{U}_p \in \mathbb{R}^{N_c\times N_p} \tag{4.51}$$

$$\mathbf{B} := [\mathbf{S}\boldsymbol{\Psi}_f]^+ \mathbf{S}\mathbf{H}_f^{-1}\mathbf{J}_{f,p}^k\mathbf{H}_p\mathbf{U}_p \in \mathbb{R}^{N_f\times N_p} \tag{4.52}$$

$$\mathbf{a} := [\mathbf{S}\boldsymbol{\Psi}_c]^+ \mathbf{S}\mathbf{H}_c^{-1}\dot{\mathbf{q}}_c\left(\widetilde{\mathbf{q}}_p^k\right) \in \mathbb{R}^{N_c} \tag{4.53}$$

$$\mathbf{b} := [\mathbf{S}\boldsymbol{\Psi}_f]^+ \mathbf{S}\mathbf{H}_f^{-1}\mathbf{f}\left(\widetilde{\mathbf{q}}_p^k\right) \in \mathbb{R}^{N_f} \tag{4.54}$$

The Jacobian matrices are defined as $\mathbf{J}_{c,p}^k := \dot{\mathbf{q}}_c\left(\widetilde{\mathbf{q}}_p^k\right)/\mathbf{q}_p^k \in \mathbb{R}^{N\times N}$ and $\mathbf{J}_{f,p} := \mathbf{f}\left(\widetilde{\mathbf{q}}_p^k\right)/\mathbf{q}_p^k \in \mathbb{R}^{N\times N}$. Note that the quantities $[\mathbf{S}\boldsymbol{\Psi}_c]^+ \in \mathbb{R}^{N_c\times N_s}$ and $[\mathbf{S}\boldsymbol{\Psi}_f]^+ \in \mathbb{R}^{N_f\times N_s}$ can be computed in the offline stage. Further, note that the application of the sampling operator $\mathbf{S}$ to the time integrator and RHS terms, as well as their Jacobians, indicates that only $N_s$ rows of these terms must be evaluated.

Solving Eq. 4.50 by its normal form results in the rather convoluted formulation

$$\Big[\mathbf{A}^\top\mathbf{C}\mathbf{A} - \mathbf{A}^\top\mathbf{D}\mathbf{B} - \mathbf{B}^\top\mathbf{D}^\top\mathbf{A} + \mathbf{B}^\top\mathbf{E}\mathbf{B}\Big]\delta\widehat{\mathbf{q}}_p^k = -\Big[\mathbf{A}^\top\mathbf{C}\mathbf{a} - \mathbf{A}^\top\mathbf{D}\mathbf{b} - \mathbf{B}^\top\mathbf{D}^\top\mathbf{a} + \mathbf{B}^\top\mathbf{E}\mathbf{b}\Big] \tag{4.55}$$

where terms have again been grouped as

$$\mathbf{C} := \boldsymbol{\Psi}_c^\top\mathbf{H}_c\mathbf{H}_r^{-2}\mathbf{H}_c\boldsymbol{\Psi}_c \in \mathbb{R}^{N_c\times N_c} \tag{4.56}$$

$$\mathbf{D} := \boldsymbol{\Psi}_c^\top\mathbf{H}_c\mathbf{H}_r^{-2}\mathbf{H}_f\boldsymbol{\Psi}_f \in \mathbb{R}^{N_c\times N_f} \tag{4.57}$$

$$\mathbf{E} := \boldsymbol{\Psi}_f^\top\mathbf{H}_f\mathbf{H}_r^{-2}\mathbf{H}_f\boldsymbol{\Psi}_f \in \mathbb{R}^{N_f\times N_f} \tag{4.58}$$

Matrices $\mathbf{C}$, $\mathbf{D}$, and $\mathbf{E}$ can all be computed in the offline stage.

Compared to the standard gappy POD HPROM formulation, this has the negative effect of requiring one additional global matrix reduction and one additional global vector reduction, along with several additional dense matrix-matrix and matrix-vector multiplications. The cost-benefits tradeoff of this approach is investigated experimentally in Section 6.2.6.

For linear multi-step schemes, recall that the time integrator term takes the general form

$$\tilde{\dot{\mathbf{q}}}_c^n = \mathbf{q}_c^n + \sum_{i=1}^{s} a_i \mathbf{q}_c^{n-i} \tag{4.59}$$

and consistent schemes satisfy

$$\sum_{i=1}^{s} a_i = -1 \tag{4.60}$$

From the previous discussions, it is apparent that the choice $\boldsymbol{\Psi}_c = \mathbf{U}_c$ is a valid choice. The process for computing $\boldsymbol{\Psi}_f$ is more straighforward, following from the POD of scaled snapshots of the RHS function as detailed in Section 4.4.

Although the process has only been described for the MP-LSVT method, this approach is also valid for LSPG projection. For LSPG, this necessitates the construction of two bases, $\boldsymbol{\Psi}_f$ and $\mathbf{U}_c = \boldsymbol{\Psi}_c$ (resized as needed). Again, the MP-LSVT method requires the primitive trial basis $\mathbf{U}_p$ as well, necessitating the storage of three POD bases in total.

## 4.5 Sample Selection

As described above, the computation of the non-linear function regression basis $\boldsymbol{\Psi}$ is somewhat nuanced, but is ultimately derived from the POD of a solution, non-linear terms, or fully-discrete residual (Jacobian) snapshot matrix. Methods for selecting the sampling indices defining the sample set $\mathcal{S}$, and hence the construction of the sampling operator $\mathbf{S}$, are much more varied. The selection of $N_s > N_r$ samples which globally minimize the regression error (Eq. 4.19) is a computationally-intractable problem. As such, almost all methods for selecting sampling points are *greedy* methods, which iteratively select points based on some measure of local optimality at a given iteration. The "best" choice of greedy sampling method is an area of active research. Unfortunately,

very few comparisons between sampling methods have been published, with the exception of the work by Peherstorfer *et al.* [76]. Although extremely insightful, their exploration of hyper-reduced PROMs was limited to one-dimensional steady reaction-diffusion system. Results presented in Chapter 6 seek to expand on their work by studying more complex, unsteady 2D and 3D multi-scale and multi-physics problems.

### 4.5.1 Sampling Criteria

Before discussing specific sampling algorithms, sampling criteria for greedy algorithms must be discussed. In general, at each iteration, the algorithm chooses the next index to include in the sample set $\mathcal{S}$ by evaluating some metric (usually an error measure) and selecting the index which maximizes/minimizes that metric. This process is slightly nuanced for the sampling of non-linear functions describing flow physics characterized by disparate spatio-temporal scales. Up until this point, sampling has only been spoken of in terms of individual *degrees of freedom* (DOFs). That is, sampling indices $s \in \{1, \ \dots \ , \ N\}$ are considered to have no specific connection to any variables or mesh elements. In reality, these indices are important for determining local connectivity of mesh elements and for computing numerical quantities such as fluxes and gradients. A state vector (or a function of the state vector) might be visualized as a two-dimensional array, with the leading dimension corresponding to the number of mesh elements $N_e$ and the trailing dimension corresponding to the number of variables $N_v$ associated with a single mesh element. This can be written as

$$\mathbf{q} \rightarrow \begin{bmatrix} q_{1,1} & \cdots & q_{1,N_v} \\ \vdots & \ddots & \vdots \\ q_{N_e,1} & \cdots & q_{N_e,N_v} \end{bmatrix} \tag{4.61}$$

In this work, the corresponding one-dimensional vector is constructed by flattening Eq. 4.61 along the leading dimension, result in the form

$$\mathbf{q} := [q_{1,1}, \ \dots \ , \ q_{N_e,1}, \ q_{1,2}, \ \dots \ , \ q_{1,N_v-1}, \ q_{1,N_v}, \ \dots \ , \ q_{N_e,N_v}]^\top \tag{4.62}$$

For a given mesh element index $j \in \{1, \ \ldots \ , \ N_e\}$, all DOF indices in $\mathbf{q}$ for the associated state variables can be retrieved as the set $s \in \{j + (k-1)N_e \mid 1 \leq k \leq N_v\}$.

As will be discussed in more detail in Section 4.6, sampling individual degrees of freedom of a non-linear function usually requires that additional, auxiliary degrees of freedom also be sampled in order to correctly compute the approximated non-linear function. The union of the sampling operator and these auxiliary degrees of freedom can thus be defined as $\widetilde{\mathbf{S}} \in \mathbb{R}^{\widetilde{N}_s \times N}$, and the sampled non-linear function is computed as $\mathbf{Sf}\left(\mathbf{q}, \ \mathrm{t}\right) = \mathbf{f}\left(\widetilde{\mathbf{S}}\mathbf{q}, \ \mathrm{t}\right)$. This understanding that additional degrees of freedom must be included in the calculation can influence how greedy sampling metrics are calculated. Three distinct approaches are described below.

1. *Agnostic*: The agnostic approach is the simplest sampling criterion, whereby the greedy metric is computed for every unsampled DOF, and only one DOF index is added to the sample set at every greedy iteration. That one DOF is then excluded from all future greedy metric evaluations. Only after $N_s$ DOFs have been selected for $\mathcal{S}$ are the auxiliary DOFs appended to the full sample set $\widetilde{\mathcal{S}}$. This is the natural procedure for single-variable systems [133], but is often used for multi-variate systems as well [148].

2. *Post-sampling*: The post-sampling method, as with the agnostic approach, computes the greedy sampling metric at every unsampled DOF and selects one index to append to the sample set $\mathcal{S}$. However, post-sampling also appends all DOFs associated with the mesh element corresponding to the sampled DOF. For example, if there are five variables associated with each mesh element, then five DOFs in total will be appended to the sample set at every greedy iteration. All DOFs associated with that mesh element are then excluded from future greedy metric evaluations, and the process continues until $N_s$ indices are included in the sample set. This approach has been used successfully for several multi-variate systems [158, 125].

3. *Comprehensive*: The comprehensive method, as with the previous two methods, evaluates the greedy metric at all unsampled degrees of freedom, but then sums

the greedy metric for all DOFs associated with a given mesh element. That is, a single greedy metric is evaluated for each mesh element. The mesh element which maximizes/minimizes the greedy metric is sampled, corresponding to appending all degrees of freedom associated with the mesh element to the sample set $\mathcal{S}$. Again, this is repeated until all $N_s$ degrees of freedom are selected. This approach has been utilized in the original work on GNAT by Carlberg and coworkers [68, 69].

The post-sampling and comprehensive methods are motivated by the fact that for coupled fluid flow systems, all degrees of freedom associated with a cell must be included in the auxiliary sample set $\widetilde{\mathcal{S}}$ if even one degree of freedom is sampled by a greedy algorithm. In this sense, they are a sunk cost in evaluating $\mathbf{Sf}(\cdot,\ \mathrm{t})$, and assessing their contribution to the greedy sampling metric provides a more complete description. All results presented in this thesis utilize the *comprehensive* approach, except those results in Section 6.2.5 comparing it against the post-sampling approach.

As an aside, prior work [68] has noted that it is important to sample at least one mesh element at each inlet and outlet boundary to facilitate the communication of information from outside of the domain to all sampled cells. Although this consideration is logical, it is not applied in this work. As will be seen later, this omission does not appear to affect the accuracy and robustness of the HROMs presented. Whether this is more generally true or a pleasant coincidence is unknown.

### 4.5.2 Sampling Algorithms

Specific algorithms for selecting samples for hyper-reduction are now discussed. This thesis studies random sampling, eigenvector-based sampling, and two variants of the GNAT sampling algorithm, which are each detailed in turn.

For all sampling algorithms, the first $N_r$ sampled degrees of freedom are selected by the QDEIM procedure proposed by Drmač and Gugercin [159]. In the context of DEIM PROMs ($N_r = N_s$), this has been shown to exhibit tighter error bounds on the resulting interpolant over the traditional DEIM greedy sampling procedure. As such, each algorithm described here must only select $N_s - N_r$ samples. This is largely done to

maintain consistency between algorithms, some of which require an initial set of sampling points and others which do not.

1. *Random Sampling*: By far the simplest method, random sampling randomly selects $N_s - N_r$ degrees of freedom using a random shuffle C++ routine. Peherstorfer *et al.* [76] provide probabilistic analyses to show that randomized oversampling limits the growth in regression error as the regression dimension $N_r$ is increased, even in the presence of additional system noise. Although this method has been observed to generate reasonable reconstructions of unsteady flow fields from a few samples, it often leads to poor results for HROMs of reacting flows [125]. However, the simplicity of this method can be appealing, as it requires no complex computations or distributed-memory software for large datasets. The computational burden of the offline cost is thus strictly limited to the precomputation of $[\mathbf{S}\boldsymbol{\Psi}]^+$, which is universal to all sampling methods presented here.

2. *Eigenvector-based Sampling*: Eigenvector-based sampling seeks to minimize the sampling error term in Eq. 4.19 via a greedy approach. This method is a simplification by Peherstorfer *et al.* [76] of a greedy sampling procedure by Zimmermann and Willcox [147] (Algorithm 4). Some of the motivating theory behind the modified algorithm is reproduced here with permission.

   To select the remaining $N_s - N_r$ points, the method leverages the fact that by nature of the $\ell^2$ norm, the sampling error may be rewritten as

$$\left\|[\mathbf{S}\boldsymbol{\Psi}]^+\right\|_2 = \sigma_{\max}\left([\mathbf{S}\boldsymbol{\Psi}]^+\right) = \frac{1}{\sigma_{\min}\left(\mathbf{S}\boldsymbol{\Psi}\right)}, \tag{4.63}$$

   where $\sigma_{\max}$ and $\sigma_{\min}$ indicate the maximum and minimum singular values of their arguments, respectively. Thus, at the $m$th sampling iteration, the row of $\boldsymbol{\Psi}$ is selected which maximizes the smallest singular value (equivalently, the smallest eigenvalue) of $\mathbf{S}_m\boldsymbol{\Psi} \in \mathbb{R}^{m\times N_r}$ via a symmetric rank-one update. Here, $\mathbf{S}_m \in \mathbb{R}^{N\times m}$ is the selection matrix constructed from the $m$ unit vectors selected up to iteration $m$ by the greedy procedure. It is shown in [76] that the bounds on this update of

the smallest eigenvalue, $\lambda_{N_r}$, from update step $m$ to $m+1$, is given by

$$\lambda_{N_r}^{m+1} \geq \lambda_{N_r}^{m} + \frac{1}{2}\left[g + \left\|\boldsymbol{\psi}'_{+}\right\|_2^2 - \sqrt{\left[g + \left\|\boldsymbol{\psi}'_{+}\right\|_2^2\right]^2 - 4g\left[\left[\mathbf{z}_{N_r}^{m}\right]^\top \boldsymbol{\psi}'_{+}\right]^2}\right], \tag{4.64}$$

where $g = \lambda_{N_r-1}^{m} - \lambda_{N_r}^{m}$, and $\mathbf{z}_{N_r}^{m} \in \mathbb{R}^{N_r}$ is the eigenvector associated with eigenvalue $\lambda_{N_r}^{m}$ (simply the $N_r$th canonical unit vector here). Here, $\boldsymbol{\psi}'_{+} = \mathbf{V}_m^\top \boldsymbol{\psi}_{+}^\top \in \mathbb{R}^{N_r}$ is defined for simplification, where $\mathbf{V}_m$ are the right singular vectors of $\mathbf{S}_m^\top \boldsymbol{\Psi}$, and $\boldsymbol{\psi}_{+} \in \mathbb{R}^{1\times N_r}$ is the row of $\boldsymbol{\Psi}$ to be included at greedy iteration $m$. Previous preprint versions of [76] offer a simplification of the bound in Eq. 4.64, given by

$$\lambda_{N_r}^{m+1} \geq \lambda_{N_r}^{m} + \frac{g\left[\left[\mathbf{z}_{N_r}^{m}\right]^\top \boldsymbol{\psi}'_{+}\right]^2}{g + \left\|\boldsymbol{\psi}'_{+}\right\|_2^2}. \tag{4.65}$$

Maximizing this bound on $\lambda_{N_r}^{m+1}$ is thus equivalent to choosing the basis row $\boldsymbol{\psi}_{+}$ which maximizes $(\mathbf{z}_{N_r}^{m})^\top \boldsymbol{\psi}'_{+}$ at each greedy iteration. Work in this thesis follows the bounds in Eq. 4.65, as it leads to a simpler algorithm which is more efficient when working with high-dimensional data in a distributed-memory setting.

The algorithm for eigenvector-based sampling is provided in Alg. 1.

3. *GNAT Sampling, Variant 1*: The sampling strategy devised by Carlberg *et al.* [68] is an extension of the greedy sampling method originally proposed for DEIM [133]. This method seeks to minimize the error in computing the regression of the basis vectors $\boldsymbol{\psi}_i$ themselves. Following a greedy approach, at each sampling iteration $m$ the regression error is calculated by

$$\boldsymbol{\epsilon}_m = \boldsymbol{\psi}_i - \boldsymbol{\Psi}_j \left[\mathbf{S}_m \boldsymbol{\Psi}_j\right]^{+} \mathbf{S}_m \boldsymbol{\psi}_i, \tag{4.66}$$

where $\boldsymbol{\Psi}_j \in \mathbb{R}^{N\times j}$ contains the leading $j$ basis vectors of $\boldsymbol{\Psi}$. The index $j$ begins at 1 and is incremented by one at every sampling loop, until $j = N_r$. At every iteration, the `ceil` $(N_s/N_r)$ degrees of freedom for which the absolute error $|\boldsymbol{\epsilon}_m|$ is greatest are sampled, and the corresponding canonical unit vectors are appended to

**Algorithm 1** Eigenvector-based sampling

**Input:** Number of desired samples $N_s$, seed sampling set $\mathcal{S}$ and operator $\mathbf{S}$, regression basis $\mathbf{\Psi} \in \mathbb{R}^{N \times N_r}$

$N_{s,e} = \texttt{ceil}\left(N_s / N_v\right)$
$N_{s,m} = \texttt{ceil}\left(|\mathcal{S}| / N_v\right)$
**for** $i = N_{s,m}$ to $N_{s,e}$ **do**
  $\mathbf{U}$, $\mathbf{\Sigma}$, $\mathbf{V} = \texttt{SVD}\left(\mathbf{S\Psi}\right)$
  $\boldsymbol{\epsilon} = \texttt{abs}\left(\mathbf{\Psi v}_{N_r}\right)$
  **for all** $s \in \mathcal{S}$ **do**
    $\epsilon_s = 0$ // Disregard indices already selected
  **end for**
  $\boldsymbol{\epsilon}_e = \mathbf{0} \in \mathbb{R}^{N_e}$
  **for** $j = 1$ to $N_e$ **do**
    **for** $k = 1$ to $N_v$ **do**
      $s = j + (k-1)N_e$
      $\epsilon_{e,j}$ += $\epsilon_s$ // Sum metric for all DOFs of each mesh element
    **end for**
  **end for**
  $j = \texttt{argmax}\left(\boldsymbol{\epsilon}_e\right)$
  **for** $k = 1$ to $N_v$ **do**
    $\mathcal{S} = \mathcal{S} \cup \{j + (k-1)N_e\}$ // Sample all DOFs of sampled mesh element
  **end for**
  $\mathbf{S} = [\mathbf{e}_s]^\top \; \forall \; s \in \mathcal{S}$
**end for**

$\mathbf{S}_m$. This approach has been successfully applied to several practical aerodynamics problems [68, 69].

The algorithm for this variant of GNAT sampling is provided in Alg. 2.

**Algorithm 2** GNAT sampling, variant 1

**Input:** Number of desired samples $N_s$, seed sampling set $\mathcal{S}$ and operator $\mathbf{S}$, regression basis $\boldsymbol{\Psi} \in \mathbb{R}^{N \times N_r}$

$N_{s,e} = \texttt{ceil}\left((N_s - |\mathcal{S}|)/(N_v N_r)\right)$ // Number of mesh elements to sample each iteration

**for** $i = 1$ to $N_r$ **do**

  **if** $i == 1$ **then**

    $\boldsymbol{\epsilon} = \boldsymbol{\psi}_1$

  **else**

    $\mathbf{c} = \underset{\mathbf{y} \in \mathbb{R}^i}{argmin} \left\| \mathbf{S}\left[\boldsymbol{\psi}_1, \ \ldots \ , \ \boldsymbol{\psi}_i\right] \mathbf{y} - \mathbf{S}\boldsymbol{\psi}_{i+1} \right\|$

    $\boldsymbol{\epsilon} = \boldsymbol{\psi}_{i+1} - \left[\boldsymbol{\psi}_1, \ \ldots \ , \ \boldsymbol{\psi}_i\right] \mathbf{c}$

  **end if**

  $\boldsymbol{\epsilon} = \texttt{abs}\left(\boldsymbol{\epsilon}\right)$

  **for all** $s \in \mathcal{S}$ **do**

    $\epsilon_s = 0$ // Disregard indices already selected

  **end for**

  $\boldsymbol{\epsilon}_e = \mathbf{0} \in \mathbb{R}^{N_e}$

  **for** $j = 1$ to $N_e$ **do**

    **for** $k = 1$ to $N_v$ **do**

      $s = j + (k-1)N_e$

      $\epsilon_{e,j}$ += $\epsilon_s$ // Sum metric for all DOFs of each mesh element

    **end for**

  **end for**

  **for** $j = 1$ to $N_{s,e}$ **do**

    $s = \texttt{argmax}\left(\boldsymbol{\epsilon}_e\right)$

    **for** $k = 1$ to $N_v$ **do**

      $\mathcal{S} = \mathcal{S} \cup \{j + (k-1)N_e\}$ // Sample all DOFs of sampled mesh element

    **end for**

    $\boldsymbol{\epsilon}_{e,s} = 0$

    $\mathbf{S} = [\mathbf{e}_s]^\top \ \forall \ s \in \mathcal{S}$

  **end for**

**end for**

4. *GNAT Sampling, Variant 2*: Preprint versions of the work by Peherstorfer *et al.* [76] suggests a variant of the original GNAT sampling procedure. Instead of selecting $\texttt{ceil}\left(N_s/N_r\right)$ sample indices at each greedy iteration, the index $i$ repeatedly loops through the integer interval $\{1, \ \ldots \ , \ N_r\}$, and only *one* sample index, for which the greedy metric in Eq. 4.66 is greatest, is selected at each greedy iteration. This can be thought of as a fine-grained alternative to the original GNAT algorithm,

and theoretically approaches a more optimal selection at the cost of many more evaluations of the greedy metric. This work appears to be the first direct comparison of these two alternative sampling methods.

The algorithm for this variant of GNAT sampling is provided in Alg. 3.

**Algorithm 3** GNAT sampling, variant 2

**Input:** Number of desired samples $N_s$, seed sampling set $\mathcal{S}$ and operator $\mathbf{S}$, regression basis $\mathbf{\Psi} \in \mathbb{R}^{N \times N_r}$

$N_{s,e} = \texttt{ceil}\left(N_s / N_v\right)$
$N_{s,m} = \texttt{ceil}\left(|\mathcal{S}| / N_v\right)$
**for** $i = 1$ to $N_{s,e} - N_{s,m}$ **do**
  **if** $i == 1$ **then**
    $\boldsymbol{\epsilon} = \boldsymbol{\psi}_1$
  **else**
    $b = i \,\%\, N_r$
    $d = \texttt{min}\left(i,\ N_r\right)$
    $\mathbf{c} = \underset{\mathbf{y} \in \mathbb{R}^i}{argmin}\ \|\mathbf{S}\left[\boldsymbol{\psi}_1,\ \ldots\ ,\ \boldsymbol{\psi}_d\right]\mathbf{y} - \mathbf{S}\boldsymbol{\psi}_b\|$
    $\boldsymbol{\epsilon} = \boldsymbol{\psi}_b - \left[\boldsymbol{\psi}_1,\ \ldots\ ,\ \boldsymbol{\psi}_d\right]\mathbf{c}$
  **end if**
  $\boldsymbol{\epsilon} = \texttt{abs}\left(\boldsymbol{\epsilon}\right)$
  **for all** $s \in \mathcal{S}$ **do**
    $\epsilon_s = 0$ // Disregard indices already selected
  **end for**
  $\boldsymbol{\epsilon}_e = \mathbf{0} \in \mathbb{R}^{N_e}$
  **for** $j = 1$ to $N_e$ **do**
    **for** $k = 1$ to $N_v$ **do**
      $s = j + (k-1)N_e$
      $\epsilon_{e,j}$ += $\epsilon_s$ // Sum metric for all DOFs of each mesh element
    **end for**
  **end for**
  $s = \texttt{argmax}\left(\boldsymbol{\epsilon}_e\right)$
  **for** $k = 1$ to $N_v$ **do**
    $\mathcal{S} = \mathcal{S} \cup \{j + (k-1)N_e\}$ // Sample all DOFs of sampled mesh element
  **end for**
  $\mathbf{S} = \left[\mathbf{e}_s\right]^\top \forall\ s \in \mathcal{S}$
**end for**

## 4.6 The Sample Mesh

As mentioned in Section 4.5, sampling individual degrees of freedom of a non-linear function is not equivalent to sampling the state variable arguments associated with those

degrees of freedom, e.g., $\mathbf{S}\mathbf{r}\left(\mathbf{q}_c\right) \neq \mathbf{r}\left(\mathbf{S}\mathbf{q}_c\right)$. Specifically, the evaluation of the non-linear function at a specific degree of freedom often requires access to all state variables at that point in space. For example, solving the continuity equation of the Navier–Stokes equations at a single discrete point in space requires access to both the density and the velocity field. Further, the state at entirely different points in space may also need to be sampled, e.g., to compute fluxes or gradients.

This is visualized in Fig. 4.1 in the context of a cell-centered finite volume discretization with quadrilateral/hexahedral control volumes, with a second-order accurate flux scheme. In this image, the blue cell indicates a control volume where at least one degree of freedom of the non-linear function is to be computed. This type of cell is referred to as a *directly sampled* cell. Red cells, or *flux* cells, are those which share a cell face with directly sampled cells and are required for computing fluxes through the shared faces. Yellow cells are those which are required for computing gradients for second-order state reconstructions at cell faces, or for reconstruction of the state at directly sampled cell vertices. These are referred to as *gradient/vertex* cells. The flux and gradient/vertex cells are referred to as *auxiliary* cells, as they are required for calculating the directly sampled degrees of freedom but are not directly sampled themselves. For two-dimensional simulations, each directly sampled cell is accompanied by four flux cells and eight gradient/vertex cells. For three-dimensional simulations, this increases to six flux cells and 26 gradient/vertex cells per directly sampled cell. These numbers are obviously lower for directly sampled cells near physical domain boundaries.

At each of these cells, the fluid state ($\widetilde{\mathbf{q}}_c$ and/or $\widetilde{\mathbf{q}}_p$), trial basis ($\mathbf{U}_c$ or $\mathbf{U}_p$) and centering state ($\overline{\mathbf{q}}_c$ or $\overline{\mathbf{q}}_p$), must be held in memory. At directly sampled and flux cells, any thermodynamic/transport properties and state gradients must also be held in memory and calculated at every sub-iteration of the solver. This gives a sense of how rapidly the computational cost and memory consumption can increase with increased sampling. Furthermore, one should not expect a linear correlation between the sampling rate and the resulting computational cost of the hyper-reduced PROM.

Note, however, that for $N_s \ll N$, the solution to the HPROMs (Eqs. 4.21, 4.30,

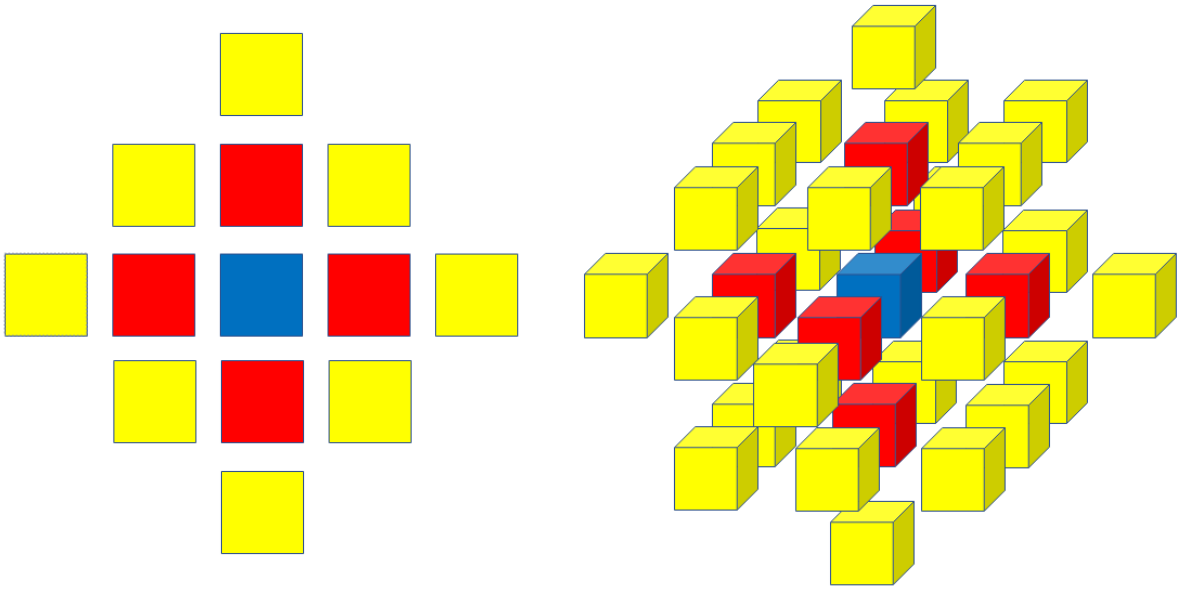


**Figure 4.1:** Sampling schemes for 2nd-order flux scheme in 2D (left) and 3D (right). Blue cells are directly sampled, red are flux cells, and yellow are gradient/vertex cells.

and 4.34) do not require access to the full $N$-dimensional state. The low-dimensional latent state $\widehat{\mathbf{q}}_c$, $\widehat{\mathbf{q}}_p$ can be advanced in time using only sampled degrees of freedom and those auxiliary degrees of freedom required to evaluate $\mathbf{Sr}(\cdot)$. The full-dimensional state $\widetilde{\mathbf{q}}_c$, $\widetilde{\mathbf{q}}_p \in \mathbb{R}^N$ can be reconstructed in the offline stage *after* the hyper-reduced PROM has been computed. This motivates the idea of the *sample mesh* concept discussed by Carlberg *et al.* [68]: only those mesh elements and associated state variables that are strictly required to compute $\mathbf{Sr}(\cdot)$ must be allocated in memory. Those mesh elements, state variables, and additional variables which are not required are simply not allocated. An example of a sample mesh for the transonic cavity flow presented in Section 6.1 is displayed in Fig. 4.2. The sample mesh concept has deep ramifications for the compute- and memory-scalability of PROMs.

The most obvious benefit of the sample mesh is, of course, a drastic decrease in memory allocation. If the sample mesh accounts for 1% of the total mesh, the memory consumption of the hyper-reduced PROM should be roughly 1% of that consumed by the unsampled PROM. For sufficiently small sample meshes, HPROMs may fit into desktop workstation or even laptop computer memory (usually 8-32 GB), eliminating the need for high-capacity HPC node memory (usually 64-256 GB). This would thus enable the

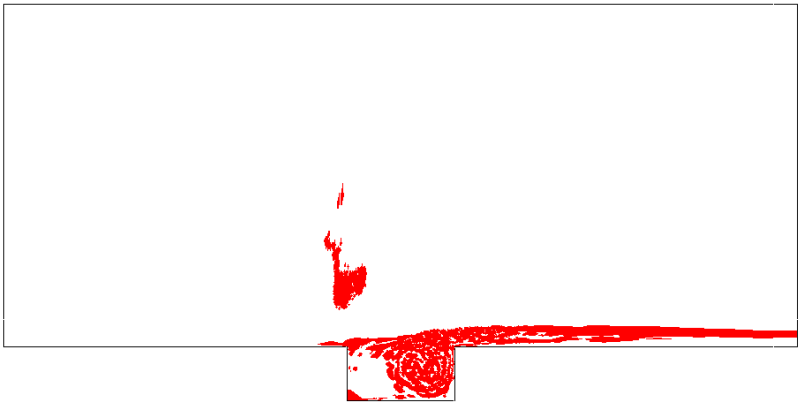


**Figure 4.2:** Example sample mesh for 2D transonic cavity flow, 1% sampling. Red elements indicate cells included in the sample mesh.

use of HPROMs by industrial engineers who do not have access to HPC resources.

The sample mesh often has the additional benefit of improving *load balancing*. In distributed-memory computing, load balancing refers to the equal distribution of the computational load between parallel processing units. This equal distribution is critical to minimizing the amount of time in which processing units are idle, waiting to send or receive data from other units before they can proceed with the calculation. Minimizing the volume and/or frequency of data communications between units is of equal importance, as the latency in transfers between units is orders of magnitude greater than the latency between a computational unit and its on-chip cache or on-node memory. Mesh partitioning software such as METIS [160] attempts to assign an equal number of mesh elements to each processing unit while minimizing the number of communications between units.

In general, a sample mesh naturally incurs fewer inter-process communications simply by nature of containing fewer mesh elements. However, careful consideration of the interaction between sampled cells can assist the mesh partitioning software in finding improved load distributions. In the case of METIS, which treats the unstructured finite volume mesh as an unstructured graph, the cells are treated as graph nodes, and cells which share faces are connected by a graph edge. An edge represents the two-way exchange of information between two graph nodes (referred to as a *point-to-point* communication), but in many instances no information exchange is required between cells in the sample mesh. The most obvious exclusion is to remove all graph nodes (cells) which are not

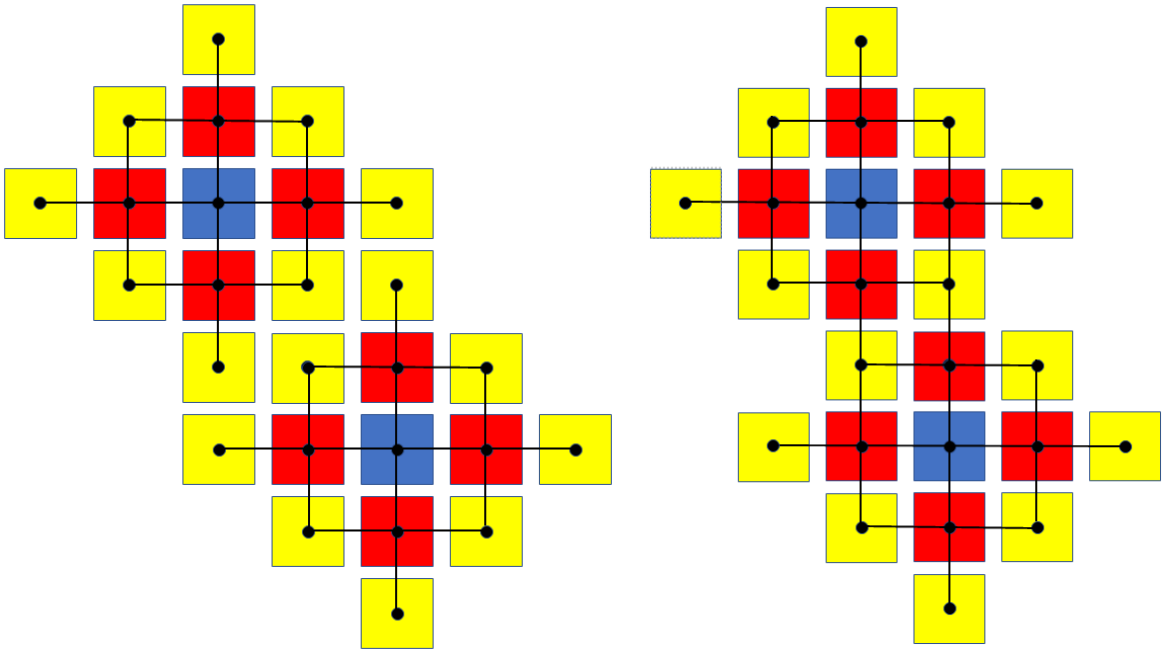


**Figure 4.3:** Mesh (graph) partitioning for load balancing. Some graph edges at finite volume cell faces can be excluded (left), while others are required (right).

sampled. Additionally, any graph edges between two gradient/vertex cells which do not originate from the same directly sampled cell can be eliminated. Figure 4.3 displays a scenario in which edges between cells which share a face may be eliminated, and another scenario in which they may not. Eliminating edges in three spatial dimensions follows a similar process. Eliminating as many graph edges as possible helps METIS to compute an optimal load balancing, as it no longer needs to consider possible edge cuts where edges have been eliminated.

As will be shown in Chapter 6, this approach to sample mesh partitioning can even lead to meshes which require zero MPI communications (besides collective operations). This occurs when sampling produces a number of small, disjoint graphs, i.e., individual graphs distributed in space which are not connected by any edges. On the other hand, fewer large, contiguous graphs arising from clustered sampling tend to result in fewer total nodes and smaller partition sizes, though they generally require more edge cuts to effectively load balance. The sparse sampling strategy can have a significant effect on this balance, as will be discussed in Chapter 6.

# Chapter 5

# Testbed for Data-driven Models of Premixed Flames

To begin, a simple model of one-dimensional premixed combustion is investigated in order to explore the uses and drawbacks of novel neural network ROM approaches, specifically order reduction by deep convolutional autoencoders and system evolution by either non-linear manifold PROMs or recurrent neural networks. The open-source software developed for this work is discussed, along with the mechanics of the neural network ROMs and the respective successes and failures of both linear subspace and non-linear manifold ROMs.

## 5.1 PERFORM: Open-source ROM Development

Before describing the one-dimensional premixed flame model which is examined in this chapter, details are provided for the Prototyping Environment for Reaction Flow Order Reduction Methods (PERFORM) [161], an open-source Python package developed solely by the author for the purpose of studying ROMs for simple reacting flows. This work was motivated by a general dearth of research on ROMs for reacting flows, which may be attributed to the complexity of reacting flow modeling combined with a lack of approachable open-source libraries for combustion CFD. Originally developed for internal use by collaborators, PERFORM is equipped with a self-contained one-dimensional, compressible reacting flow solver which may be queried by a separate ROM framework which is designed for code flexibility and expansion. The package thus serves as a general-purpose testbed for novel ROM methods in both its capacity to generate challenging datasets for

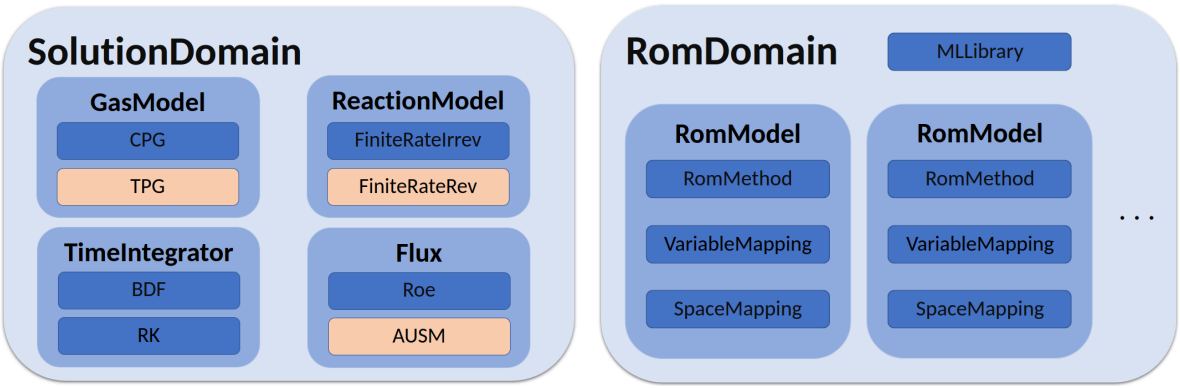



**Figure 5.1:** Simplified PERFORM class hierarchy for flow solver (left) and ROM solver (right). Class instances in blue have been implemented, those in orange are under development.

non-intrusive approaches, as well as its expressive construction which allows for rapid implementation of novel intrusive approaches. The 1D benchmark problems enabled by PERFORM should not be viewed as the terminal stage of ROM development; low-level, massively parallel codes are the ultimate goal of any PROM method. PERFORM aims to minimize ROM *development* and *initial testing* time, rather than computational time. In this sense, PERFORM lowers the barrier to entry for ROM researchers who may have little experience with combustion modeling or low-level programming languages, allowing them to perform research with a challenging, practical class of problems.

The rough structure of PERFORM is visualized in Fig. 5.1. As mentioned previously, the 1D finite-volume flow solver and ROM framework are purposely segregated such that ROM method developers do not need to interact with the flow solver beyond querying functions which calculate, e.g., $\mathbf{f}(\cdot)$, $\mathbf{r}(\cdot)$. The flow solver, packaged under the class `SolutionDomain`, contains generic interfaces for the standard trappings of any finite volume flow solver, including gas models (`GasModel`), chemical reaction models (`ReactionModel`), fluxes (`Flux`), temporal integrators (`TimeIntegrator`), gradient limiters (`Limiter`), and *in situ* data visualizations (`Visualization`). These interfaces are easily expanded beyond PERFORM's current offerings thanks to judicious use of Python's class inheritance and polymorphism.

The overarching reduced-order model solver class, `RomDomain`, is designed to provide maximum flexibility to ROM developers. A given ROM solve may be decomposed into several self-contained `RomModels`, which govern how the low dimensional representation

for a subset of flow variables is evolved in time. For example, one model might target pressure and velocity, while another might target temperature and species mass fractions (linked to the "scalar" ROM concept explored by Zhou [158]). Generic interfaces for specifying target variables (`RomVariableMapping`), defining mappings from the latent space to the trial manifold (`RomSpaceMapping`), and specifying non-standard time evolution methods (`RomTimeStepper`) are similarly simple to expand to meet development needs. Generic interfaces to machine learning libraries such as TensorFlow are also provided.

PERFORM is available as a public repository[1], and comes with several benchmark cases which are ready to run out-of-the-box. These cases include a Sod shock tube, a transient multi-species contact surface (with and without artificial acoustic forcing), a stationary premixed flame (with artificial acoustic forcing), and a transient premixed flame (with and without artificial acoustic forcing). This last case will be analyzed in the following sections. The benchmarks address several of the critical issues facing the broader ROM community, particularly the difficulty of propagating transient flow features beyond the training dataset and making accurate predictions in a complex parameter space. The code has already seen successful use in developing accurate and robust linearized ROMs [162] and investigating true ROM predictivity via basis and hyper-reduction sampling adaptation [78].

## 5.2 Acoustically-forced Transient Flame

The one-dimensional model of an acoustically-forced, freely-propagating premixed flame is now described. Similar constructions have been presented in prior work by the author and collaborators [75, 163]. The spatial domain is one-dimensional, spanning the length $x \in [0.0,\ 1]$ cm, subdivided into 1,024 cells of equal length. As with GEMS, a second-order Roe flux computes the inviscid fluxes, while gradients are computed by a central finite-difference stencil and limited by the face-oriented limiter of Barth and Jespersen.

The chemical system is composed of two fictitious species, a "reactant" species and a "product" species, having the calorically-perfect gas properties given in Table 5.1. In

[1] https://github.com/cwentland0/perform

| Species | MW (g/mol) | $c_{p,l}$ (kJ/kg-K) | $h_l^\circ$ (kJ/kg) | $\mu_l$ (kg/m-s) | $\mathrm{Pr}_l$ | $\mathrm{Sc}_l$ |
|---|---|---|---|---|---|---|
| Reactant | 21.32 | 1.538 | -7,4320 | 7.35e-4 | 0.713 | 0.62 |
| Product | 21.32 | 1.538 | -10,800 | 7.35e-4 | 0.713 | 0.62 |

**Table 5.1:** Constant thermodynamic and transport properties of fictitious species.

this case, the fluid has a constant viscosity, unlike results in Section 6.1 which computes viscosity via Sutherland's law. Note that the only difference between the two species is their enthalpy of formation. The reactant is converted to product by a single irreversible finite-rate reaction, governed by the Arrhenius parameters $A = 2.12 \times 10^{10}$ 1/s, $b = 0$, and $E_a = 2.025 \times 10^8$ kJ/mol.

To begin, the solution is initialized with 100% reactant at 300 K in the region $x \in [0.0,\ 0.2]$ cm, and the remainder of the domain is initialized with product at 2,400 K. The full domain is initialized with a pressure of 1 MPa and zero velocity. An approximate characteristic boundary condition is enforced at the outlet, and a fixed velocity is enforced at the inlet. Using BDF2 integration with a time step of 25 ns, the simulation is started with a velocity of 1 m/s at the inlet, which is manually adjusted until the flame is determined to be suitably "stationary," balancing the bulk advection downstream with the reaction/diffusion upstream. At this point, 10 m/s is added to the velocity throughout the domain, and the primitive state appears as shown in Fig. 5.2. This is the initial condition from which all further FOM simulations are computed. Further, for all further simulations (ROM or FOM), the inlet boundary condition is changed to an approximate characteristic boundary condition.

To build a dataset of parametrically-varied simulations, an artificial pressure forcing is introduced to the outlet boundary condition. This forcing is computed as a sinusoidal perturbation about the outlet mean-flow pressure, of the form

$$p_b = \overline{p}\,[1 + A\ sin\,(2\pi f \mathrm{t})] \tag{5.1}$$

where $\overline{p}$ is the outlet mean-flow pressure (approximately 965 kPa in this case), $A$ is the percentage amplitude, and $f$ is the forcing frequency. For all cases, the amplitude is

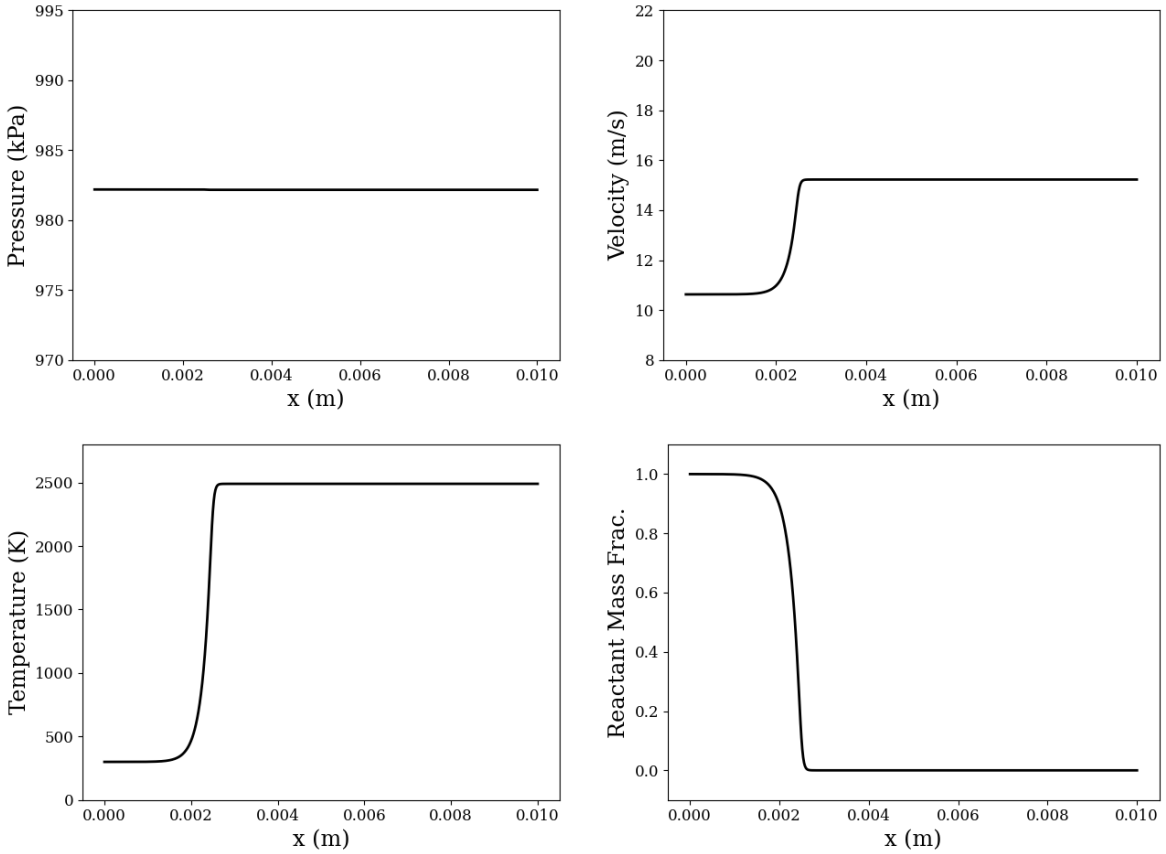



**Figure 5.2:** Initial condition for propagating flame FOM simulations.

| Training | Validation | Testing |
|---|---|---|
| 100 | 118.75 | 87.5 |
| 112.5 | 156.25 | 93.75 |
| 125 | 193.75 | 106.25 |
| 137.5 | | 131.25 |
| 150 | | 143.75 |
| 162.5 | | 168.75 |
| 175 | | 181.25 |
| 187.5 | | 206.25 |
| 200 | | 212.5 |

**Table 5.2:** Training, validation, and testing dataset splits, by forcing frequency (in kHz).

1%. A total of 21 FOM simulations are computed with different forcing frequencies $f$, ranging from 87.5 to 212.5 kHz at intervals of 6.26 kHz. Each simulation is allowed to run for 23,000 iterations, or 575 $\mu$s. Snapshots of the primitive state are collected at every 10 iterations, starting after 3,000 iterations (t = 75$\mu$s). for a total of 2,001 snapshots per FOM simulation (including the initial condition). The data are then split into nine training, three validation, and nine test sets. The first group will be explicitly used to train the data-driven models, the second will be used for neural network training to inform model generalizability during the learning process, and the third will be used to evaluate the predictive capabilities of the resulting models.

Examples of several relevant flow fields are given in Figs. 5.3 and 5.4. Pressure snapshots for several different forcing frequencies are shown in Fig. 5.3, where it is clear that the acoustic behavior is significantly affected by the disparate sound speeds between the hot products and cold reactants, increasing the frequency and amplitude of the signal as it propagates upstream. Figure 5.4 shows two relevant indicators of the flame's propagation downstream, marked by a steep rise in temperature where cold reactant is converted to hot product, and the peak of maximum heat release indicates the precise location of this reaction.

This case presents several interesting challenges for data-driven modeling and PROMs in particular. The system is characterized by three different spatio-temporal scales, namely those associated with the bulk advection, acoustics, and reaction. Traditional approaches to model reduction often struggle to accurately model phenomena such as sharp

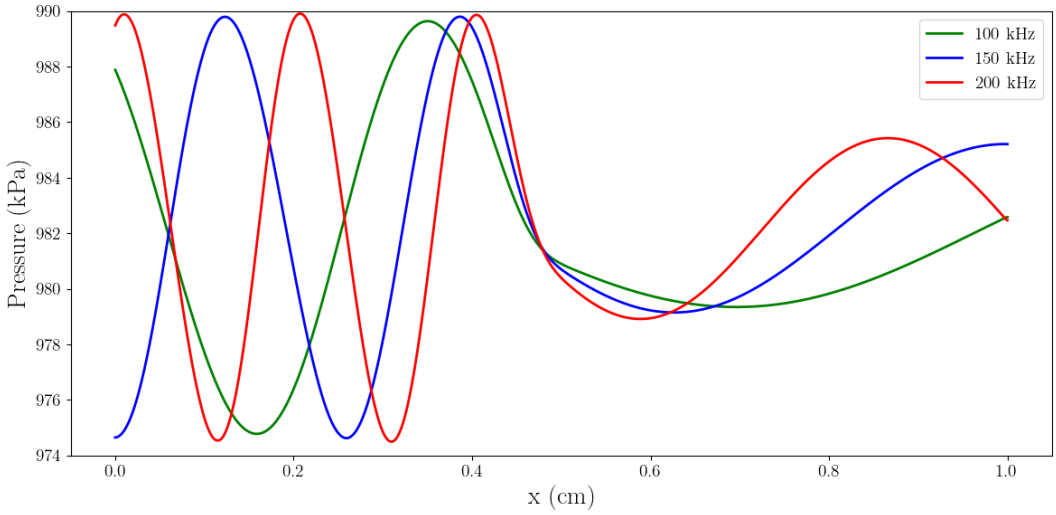



**Figure 5.3:** Forced pressure field examples, $f \in \{100,\ 150,\ 200\}$ kHz.

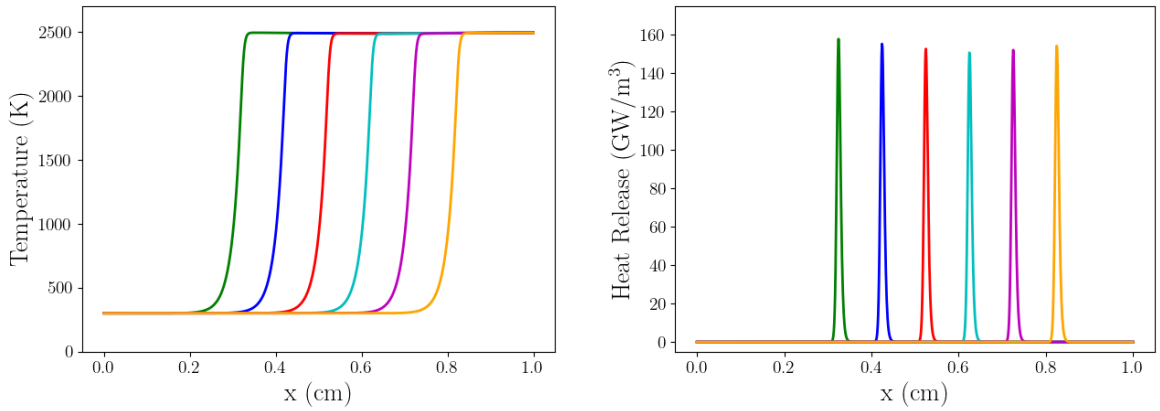



**Figure 5.4:** Flame progression for data collection period t $\in [75,\ 575]$ $\mu$s.

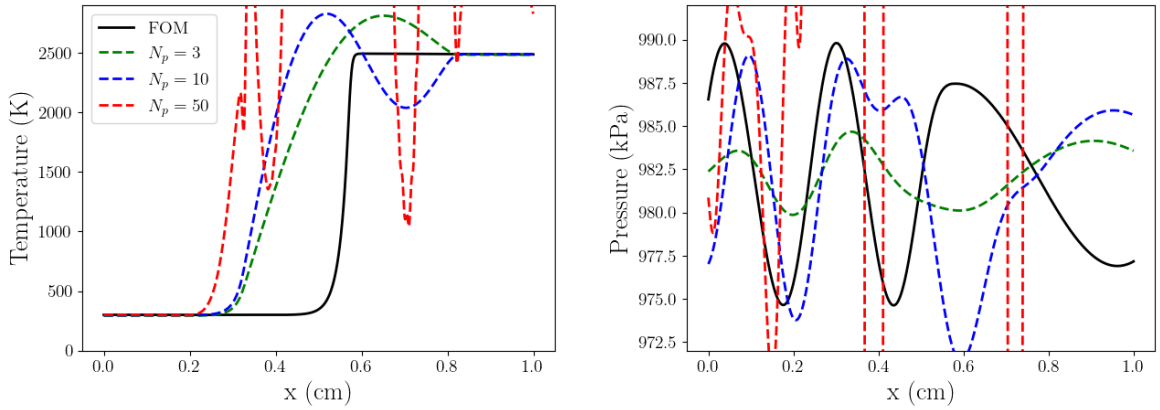



**Figure 5.5:** Intrusive linear subspace MP-LSVT PROM temperature (left) and pressure (right) snapshots, various $N_p$.

gradients, propagating waves, and highly non-linear multi-physics couplings. As will be shown shortly, classical methods (and even some novel machine learning approaches) generally fail to make accurate predictions.

## 5.3 Failure of Intrusive Linear PROMs

To begin, linear subspace MP-LSVT PROMs are tested. The primitive trial basis is computed from the concatenated training datasets (accounting for 18,009 snapshots), centered about the time-average mean of the dataset and normalized by min-max scaling. Details on this centering and scaling operation can be found in Section 8.1. For a given forcing frequency, the PROM is restarted from the corresponding primitive state snapshot at t = 75$\mu$s and allowed to run for 20,000 iterations with a time step of $\Delta$t = 25 ns.

Unfortunately, all linear subspace PROMs performed abysmally for all forcing frequencies. This can be plainly seen in Fig. 5.5, where low trial space dimensions (here, $N_p = 3,\ 10$) results in smeared gradients and decoherence of the pressure signal. At higher trial space dimensions ($N_p = 50$), the solution is entirely unstable, devolving into a meaningless flow field.

The primary reasons for these failures can be observed from the linear projection of the solution onto the trial space. Figure 5.6 shows the projected temperature and pressure fields. For the temperature field, which experiences a strong gradient at the

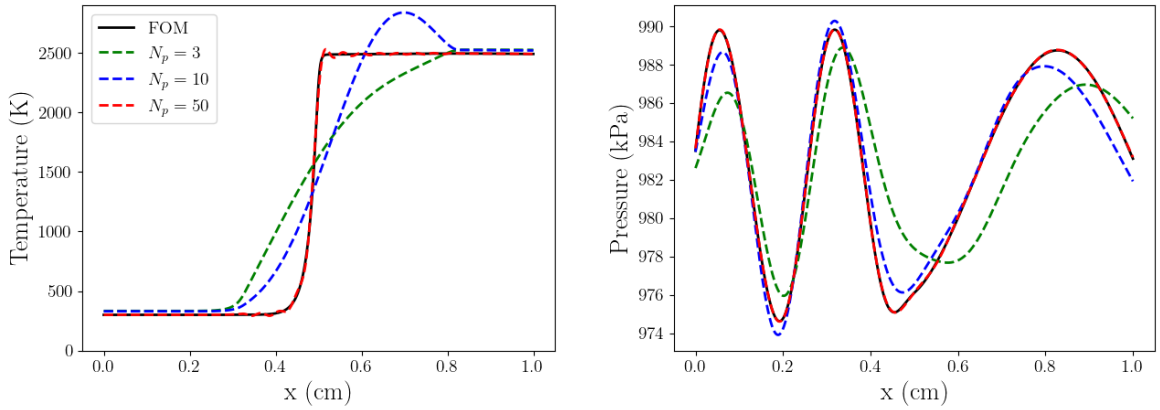



**Figure 5.6:** Linear projections of temperature (left) and pressure (right) fields, $f = 150$ kHz, various $N_p$.

flame, a linear representation will invariably result in significant under- and overshoots in the vicinity of the jump. Increasing the trial space dimension decreases the magnitude of these overshoots but gives rise to high-frequency "ringing" behavior near the flame. This is similar to the Gibbs phenomenon observed in approximating discontinuities with a Fourier series. The accuracy of the projected pressure field is similarly poor, though given the relatively smooth nature of this field, increasing the trial basis resolution converges to an accurate approximation without significant discrepancies.

The poor performance of linear subspace PROMs in modeling advection-dominated flows with strong gradients comes as no surprise, given the previous discussion of slowly-decaying Kolmogorov $n$-widths in in Section 3.1.2. Non-linear alternatives are explored next, in an attempt to alleviate these issues.

## 5.4 Autoencoder Non-linear Manifold PROMs

As proposed by Lee and Carlberg [116], non-linear manifold PROMs have the potential to more accurately represent flows characterized by a slowly-decaying Kolmogorov $n$-width. The theoretical ability of neural networks to approximate arbitrary non-linear functions and the expressiveness of over-parameterized deep neural networks make them appealing candidates for the representation of a non-linear manifold on which to compute PROMs. As outlined in Section 3.1.2, the autoencoder architecture allows for unsupervised learning

| Layer | Type | Output Size |
|---|---|---|
| 1 | Convolution | $512 \times 16$ |
| 2 | Convolution | $256 \times 32$ |
| 3 | Convolution | $128 \times 64$ |
| 4 | Dense | $N_p$ |

**Table 5.3:** Convolutional encoder dimensions. Decoder mirrors encoder with transpose convolutional layers.

of such a mapping directly from flow field data.

Autoencoders are trained using the same datasets described in Table 5.2, though the validation training datasets are used to evaluate the model's performance during training (without participating in actual gradient descent process). The encoder is composed of three convolutional layers, each with a kernel size of 8, a stride size of 2, and same zero padding. These are followed by a fully-connected ("dense") layer which maps the encoder output to the latent dimension $N_p$. The output size (i.e., the number of filters) for the convolutional layers is summarized in Table. 5.3. The decoder mirrors the encoder via transpose convolutional layers. All layers are equipped with the Swish activation function, except for the final decoder output layer, which uses a linear activation. Network weights are initialized with the Glorot (a.k.a. Xavier) uniform distribution, and biases are initialized to zero.

Each network is trained for a maximum of 5,000 epochs, with a batch size of 25 and the mean-squared error loss function. The Adam optimizer with a learning rate of $1 \times 10^{-4}$ is utilized, and early stopping halts training if the validation loss does not improve over 500 epochs. All network construction, training, and evaluation is computed using the TensorFlow library.

The resulting autoencoders display exceptional accuracy in approximating the flame flow fields. Figure 5.7 shows the closest solution on the trial manifold (in the Euclidean norm) for temperature and pressure field snapshots. Even for $N_p = 3$, the representation is nearly perfect. Very close inspection reveals some low-amplitude noise, but larger latent space dimensions appear to entirely eliminate this. At first glance, this would appear to bode well for the resulting non-linear manifold PROMs.

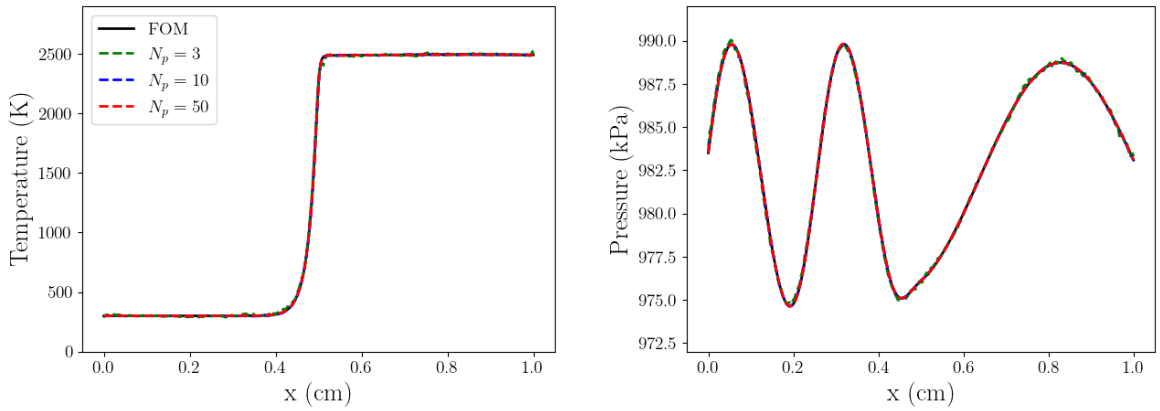



**Figure 5.7:** Approximation of temperature (left) and pressure (right) fields on solution manifold, $f = 150$ kHz, t $= 250\mu$s, various $N_p$.

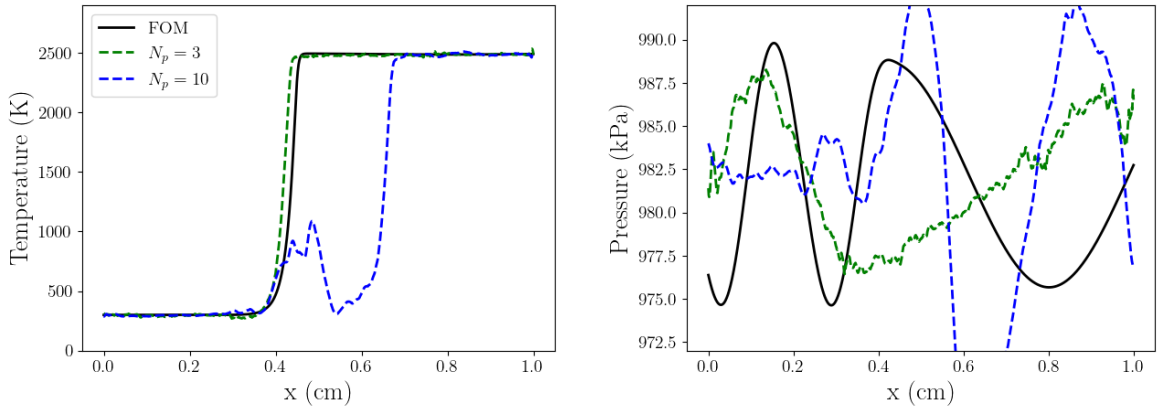



**Figure 5.8:** Intrusive non-linear manifold MP-LSVT PROM snapshots, various $N_p$.

| Model | Runtime (hours) | Cost (FOM runs) |
|---|---|---|
| Online FOM ($\times 1$) | 0.42 | 1 |
| Training, $N_p = 3$ | 5.8 | 13.81 |
| Training, $N_p = 10$ | 4.9 | 11.67 |
| Training, $N_p = 50$ | 5.5 | 13.1 |
| Online CAE PROM, $N_p = 3$ | 7.2 | 17.14 |

**Table 5.4:** Relative offline/online computational costs for CAE PROMs. Non-increasing CAE training costs are due to early stopping.

The same simulation configuration used for the linear subspace MP-LSVT PROMs is repeated here. Unfortunately, the autoencoder non-linear manifold PROMs also perform very poorly. In contrast to the over-smooth solutions generated by the linear trial spaces, the non-linear manifold PROMs suffer from excessive low-amplitude noise in the predicted solution. This is made apparent in Fig. 5.8, revealing that the solution quickly devolves into highly erroneous fields where the approximate solution shown in Fig. 5.7 was so accurate. Ultimately, these non-linear PROMs suffer from an accumulation of error over a large number of time steps. These small errors appear to be amplified by the non-linear nature of the decoder: small deviations in the latent variables may lead to drastic changes in the predicted solution. Whereas linear trial space are relatively limited in their expressiveness, neural networks have the ability to generate both very accurate and very wrong solutions.

Not only is the accuracy of these non-linear manifolds PROMs disappointing, the cost to train and evaluate these models is exorbitant. Table 5.4 summarizes the cost of training and computing the neural network models, revealing that each step accounts for over ten times that of a single FOM simulation. The vast majority of the computational cost for the non-linear manifold PROM lies in calculating the Jacobian of the decoder, which relies on slow automatic differentiation procedures. Opposed to the simple analytical Jacobians possible for linear representations, it is difficult to justify the cost of this method given its experimental performance.

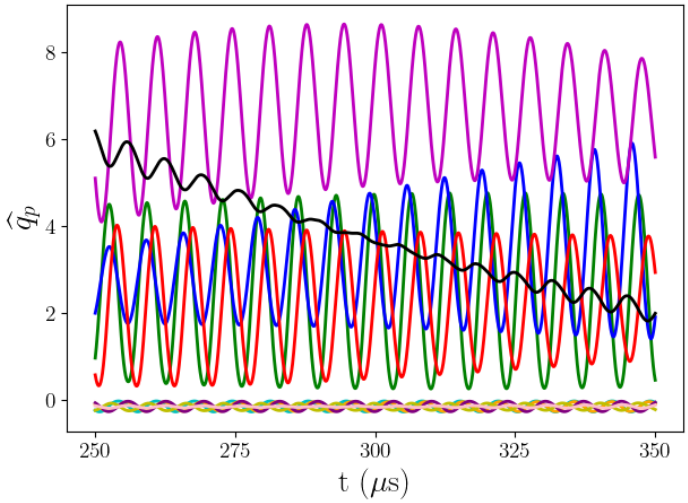



**Figure 5.9:** Encoded latent state trajectories, $f = 150$ kHz, $N_p = 10$.

## 5.5 Recurrent Neural Network ROMs

A number of efforts have been undertaken in applying *non-intrusive* approaches to reduced-order modeling. Such approaches do not require access to or even understanding of the underlying numerical solver, but only the outputs of the solver. Unlike PROMs which directly manipulate the governing equations, non-intrusive approaches are constructed wholly from data and evaluated independently from the numerical solver. Prominent examples include operator inference [54, 164], Koopman learning [131], and a host of neural network approaches [55, 56, 165].

This work follows the method first formulated by Gonzalez and Balajewicz [55], which proposes the use of a long short-term memory (LSTM) network to model the evolution of the latent variables in time. The LSTM is a specific architecture of recurrent neural networks (RNNs), a form of neural network which performs auto-regressive predictions. That is, the network takes as input a series of past observations to make a prediction for the next step in the series, after which point that prediction is fed back into the network as input for the proceeding prediction, and so on. This design makes LSTMs ideal for time-series predictions, as in the case of reduced-order models for dynamical systems. Details on the mathematics of LSTMs (and RNNs more broadly) can be found in the review by Yu *et al.* [166].

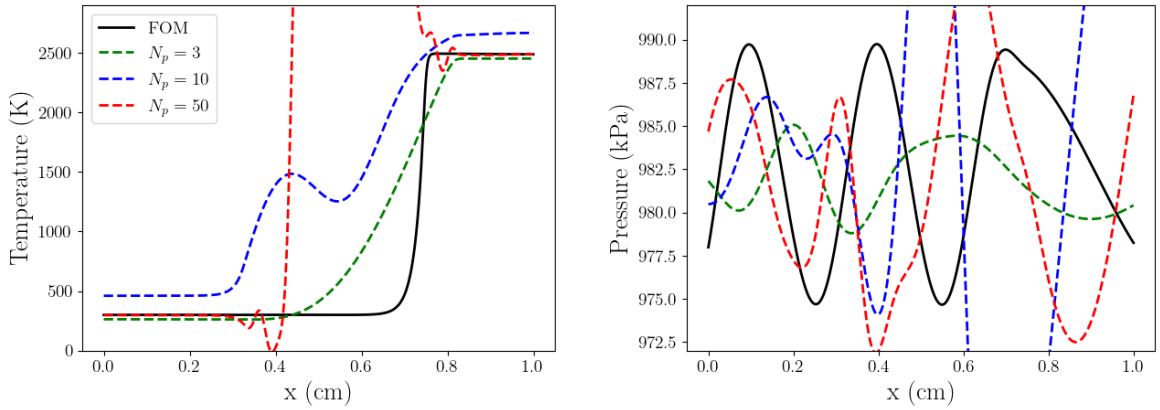



**Figure 5.10:** Online temperature (left) and pressure (right) field predictions for LSTMs trained with POD trajectories, t = 500$\mu$s, $f = 131.25$ kHz, various $N_p$.

For this work, two variants of LSTMs are trained: one trained using the modal coefficient trajectories associated with the linear trial space from Section 5.3, and one trained using the encoded latent variable trajectories associated with the autoencoder from Section 5.4. This effectively compares the approaches of Maulik *et al.* [165] and Gonzalez and Balajewicz [55], respectively.

The LSTM training process is as follows. First, the training data are projected (for the POD representation) or encoded (for the CAE representation). Figure 5.9 displays a representative trajectory of the latent variables for the autoencoder with $N_p = 10$. Then, standalone LSTM networks are trained using contiguous time windows of the training latent variable data. For this work, the LSTMs are constructed from two LSTM layers, followed by a single fully-connected layer that maps to the latent dimension $N_p$. The output size of both LSTM layers is 200, while the lookback window for the network inputs is 50 time steps. The LSTM layers are equipped with tanh activation functions, and the dense layer with a linear activation. The training parameters are the same as for the autoencoders.

Evaluating these non-intrusive ROMs reveals stark contrasts in performance between the POD and CAE representations. Figure 5.10 displays predictions of the temperature and pressure fields near the end of the simulation window, for an unseen forcing frequency $f = 131.25$ kHz. As with the linear subspace PROMs, the ability of the POD modes to approximate the flow field remains extremely poor. At higher latent dimensions, the

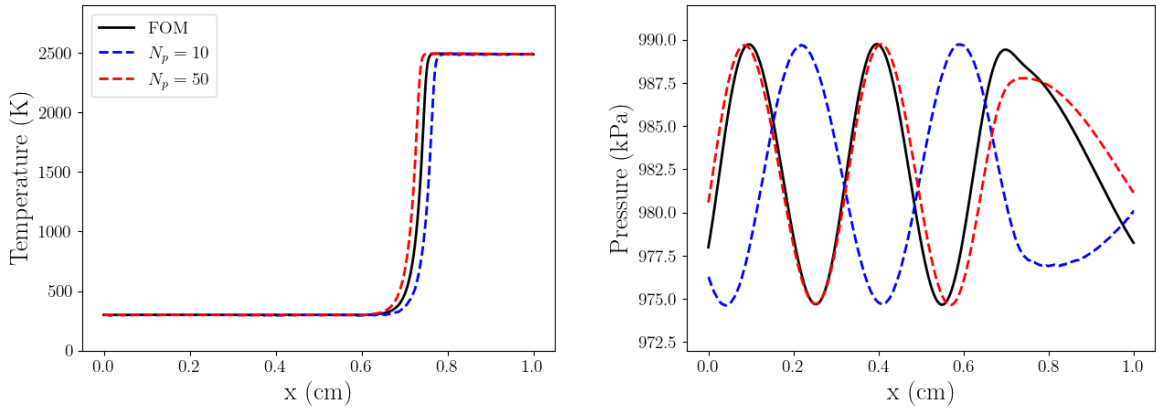
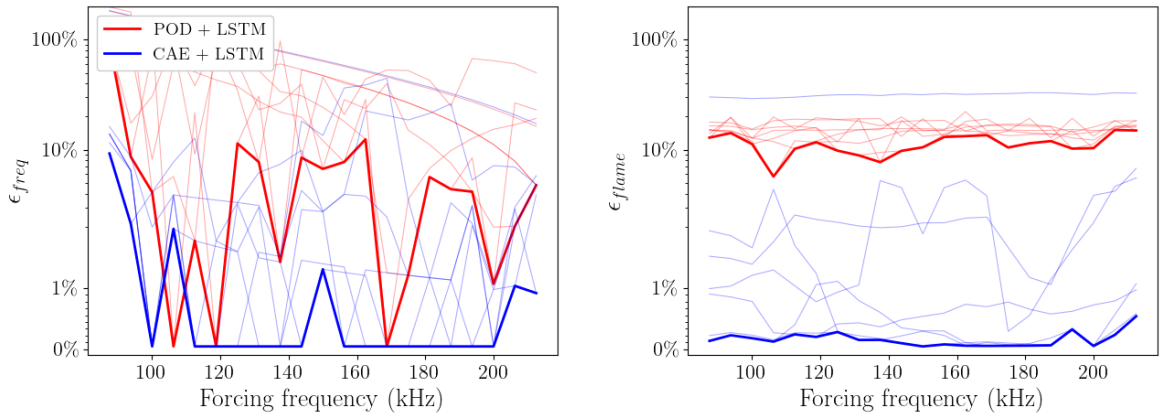



**Figure 5.11:** Online temperature (left) and pressure (right) field predictions for LSTMs trained with CAE trajectories, t = 500$\mu$s, $f = 131.25$ kHz, various $N_p$. Note that $N_p = 3$ was unstable.




**(a)** Upstream $f$.

**(b)** Flame location.

**Figure 5.12:** Aggregate predictions of QoIs across all forcing frequencies, all $N_p$. The best predictions for each model are marked by a bold line.

prediction appears to stagnate entirely. Although the pressure field retains an oscillatory nature, the predicted amplitude and frequency of the signal are wholly incorrect.

The LSTMs equipped with an autoencoder representation, on the other hand, perform exceptionally well. Figure 5.11 shows the same time instance as in Fig. 5.10, but the long-term flow behavior is predicted very well. The temperature profile remains sharp and has progressed at similar speed as in the FOM, and the amplitude and frequency of the pressure signal appear to be preserved (albeit with some phase shift). However, it must be noted that the case for $N_p = 3$ was unstable, unlike that of the POD representation.

Comparing these simulations quantitatively can be slightly challenging, as a minor phase shift in the predicted pressure field can result in enormous $\ell^2$-norm error mea-

surements. Instead, two quantities of interest are measured: the predicted dominant frequency of the pressure signal at $x = 0.25$ cm (computed from the FFT of a point monitor, compared by amplitude), and the time-average accuracy of the flame location, as measured by the point of maximum heat release. Figure 5.12 summarizes the findings across all forcing frequencies and latent dimensions $N_p \in \{3, 5, 10, 20, 50, 100, 200\}$. The best predictions are connected by a bold line, while all others appear as translucent lines. Clearly, the LSTMs with a CAE representation are capable of predicting these QoIs with much higher fidelity than those with a POD representation. In fact, as hinted by the field plots in Fig. 5.10, the frequency predictions of the the POD LSTM may be slightly misleading, as spurious frequencies of varying amplitudes may have been introduced.

## 5.6 Conclusions

This chapter highlights the open-source package PERFORM, developed by the author to assist the community in implementing and applying novel ROM methods for a challenging class of advection-dominated reacting flows. The flexible ROM API minimally coupled with a one-dimensional reacting compressible flow solver enables rapid prototyping and performance assessment for systems beyond the standard toy problems.

The utility of PERFORM is demonstrated by analyzing ROM accuracy in making parametric predictions for an acoustically-forced model premixed flame, specifically investigating traditional linear subspace PROMs, deep autoencoder non-linear manifold PROMs, and non-intrusive LSTM ROMs. Both the linear and non-linear intrusive PROMs exhibit terrible performance, despite the autoencoder's excellent representation of the solution manifold at extremely low latent dimensions. On the other hand, the non-intrusive LSTM ROM displayed remarkable predictive capabilities in modeling both the average flame speed and upstream acoustic content. However, this is only true when the LSTM is trained on latent variable trajectories generated by the neural network autoencoder. Those LSTMs trained on POD modal coefficients failed to accurately represent the transient flame solution, indicative of the general inability to generate a low-dimensional

and linear representation of sharp gradients and propagating waves.

Although similar results have been demonstrated for simpler canonical problems (Burgers' equation, shallow water equations), the ability to quickly implement novel ROM methods like those discussed above may be an invaluable tool for assessing their practical viability. In this sense, PERFORM stands as both a lower barrier to entry for ROM practitioners and a challenge to tackle more difficult modeling problems.

# Chapter 6

# Scalable PROMs for Multi-scale, Multi-physics Flows

For the one-dimensional simulations explored thus far, computational cost is not of particular concern. Such numerical experiments can be evaluated using laptop or desktop computers without much attention to memory and compute scalability. Exercises in hyper-reduction for PROMs of such small dimension are largely academic, as even the cost of the full-order model measures in core-seconds or core-minutes. For more practical two- and three-dimensional systems, however, the computational cost of evaluating the FOM and unsampled PROMs is significantly larger, requiring high-performance computing resources and parallelism across hundreds or thousands of cores. The benefit of hyper-reduction in enabling scalable computations for PROMs is more readily apparent for such high-dimensional non-linear systems.

In this chapter, HPROMs are analyzed for three multi-scale systems: transonic flow over an open cavity, a single-element model rocket combustor, and a multi-element laboratory rocket combustor. The cavity flow system acts as a proof-of-concept for non-reacting flows, while the rocket combustors more deeply explore the challenges in developing robust and scalable HPROMs for reacting flows. Critically, the performance of several gappy POD sampling algorithms are compared, evaluating both the offline cost of the sample selection algorithms and their effects on memory consumption, online computational cost, and accuracy.

Throughout this chapter, results are presented on computational cost savings induced by HPROMs as a *speedup ratio* in terms of core-hour consumption (*core time*). Core time is computed by multiplying the amount of wall time spent running the simulation

by the number of CPU cores used to compute the simulation. Core time metrics provide a more complete measurement of computational cost, though wall time measurements may be useful in assessing performance for time-sensitive applications. The speedup ratio is computed as the ratio of the FOM core time to the PROM core time measurement. For example, if a FOM calculation using ten cores took five hours, and a PROM calculation using two cores took one hour, the core time speedup ratio would be 25. Note that run-time measurements are computed as the *average* time across all processes spent performing significant floating-point operations (*calculation time*) and communicating data between processes via MPI (*MPI time*). I/O timing (e.g., writing field or probe data to disk) is excluded, as it can be a volatile and unreliable measure, depending on hardware limitations and file system usage. There is some inaccuracy in comparing run-times and computational cost from calculation and MPI timings alone, as the overall computing time is not equal to the average across all parallel processes. Nor is it the sum of the maximum calculation time and maximum MPI time, as the process which computes the most floating-point operations may not be the same process which communicates the most data (or waits idly for other processes). However, this approach is as fair a measure as can be reasonably be expected.

All simulations presented in this section are computed on the Cray XC40/50 Onyx cluster maintained by the U.S. Army Engineering Research and Development Center. A single compute node features two Intel E5-2699v4 Broadwell chips (2.8 GHz), each with 22 computational cores for a total of 44 cores per node. Nodes have 121 GB of accessible memory, and are connected by Cray Aries interconnects.

## 6.1 2D Transonic Flow Over an Open Cavity

To begin this section, two-dimensional transonic flow over an open cavity is examined. Flow over open cavities has been studied extensively due to its practical applications in aviation (e.g., bomb or landing gear bays) and the interesting acoustic phenomena it exhibits, particularly the resonant coupling of the cavity leading edge shear layer and

acoustic feedback from the cavity trailing edge. Pioneering experimental work in the mid-1900's such as that by Roshko [167], Krishnamurty [168], and Rossiter [169], numerical modeling work such as that by Colonius *et al.* [170], Rowley *et al.* [171], and Larchevêque *et al.* [172], and more recent experimental efforts such as those by Wagner*et al.* [173] and Casper *et al.* [174] have investigated the effects of cavity dimensions, freestream flow regime, and turbulence on the behavior of this aeroacoustic coupling. This work follows the research conducted by Tezaur *et al.* [175, 157] in investigating PROM performance in modeling two-dimensional flow over a rectangular cavity at a Mach number of 0.6.

### 6.1.1 Full-order Model

The computational domain is modeled as a rectangular cavity set in a flat wall, with the leftmost, rightmost, and topmost boundaries of the domain open to the atmosphere. The geometry is shown in Fig. 6.1. The cavity is $L = 91.71$ mm long, and $D = 45.855$ mm deep ($L/D = 2.0$). The wall extends 290.8 mm (a little over three cavity lengths) both upstream and downstream of the leading and trailing edges of the cavity, respectively, for a total domain length of 673.31 mm. The upper boundary (open to air) is set 290.8 mm from the main wall. No-slip wall boundary conditions are enforced at all walls. A characteristic inlet boundary condition is enforced at the left-most domain boundary, and characteristic outlet boundary conditions are enforced at the topmost and rightmost domain boundaries. These characteristic boundaries allow acoustic waves to exit the domain with minimal reflection. The mesh is composed of 125,000 quadrilateral cells, resulting in a total number of degrees of freedom of $N = 500{,}000$. A red dot in Fig. 6.1 marks the location of a point monitor which will be measured throughout this section. It is placed halfway up the aft wall of the cavity, at $(x,\ y) = (91.71,\ -22.93)$ mm. Note that this full-order model should not be considered a truly accurate representation of flow over an open cavity, and this thesis makes no such claims. Beyond the measurement of the acoustic content and comparison against an empirical model in Fig. 6.6, results presented in this chapter simply aim to model the dynamics present in this approximate simulation.

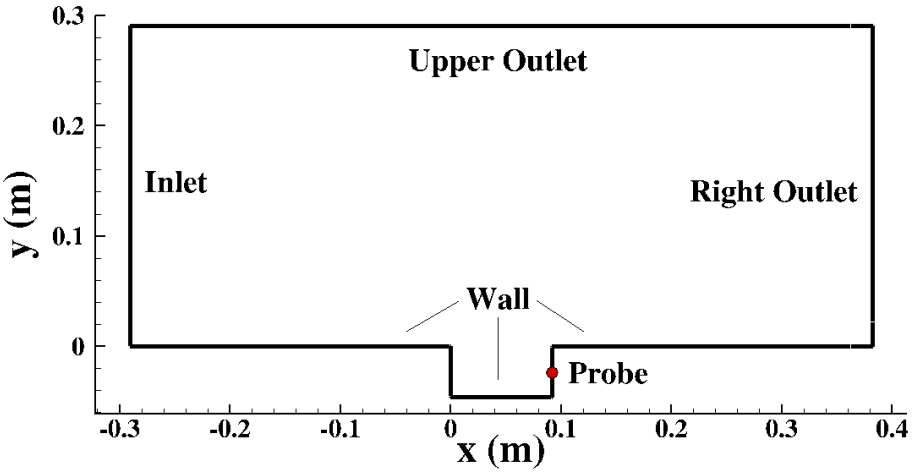



**Figure 6.1:** Cavity flow domain.

| MW (g/mol) | $c_p$ (kJ/kg-K) | $\mu$ (kg/m-s) | Pr | Sc |
|---|---|---|---|---|
| 28.9604 | 1004.84 | 8.46e-7 | 0.72 | 0.62 |

**Table 6.1:** CPG properties of air for cavity flow case.

The working fluid is air, modeled as a calorically-perfect gas with the properties given in Table 6.1. Transport and thermodynamic properties are computed using the simplified analytical forms described in Section 2.1.1. The free-stream velocity is 208.7816 m/s in the $+x$-direction, the pressure is 25 Pa, and the temperature is 300 K. The resulting Mach number of this flow regime is approximately 0.6, and the Reynolds number based on the cavity length is approximately 6,500.

The FOM is initialized with free stream conditions outside the cavity, and the inside of the cavity is initialized with free stream pressure and temperature, and zero velocity. The physical time step for the FOM simulation is $\Delta t_{\text{FOM}} = 1$ $\mu$s. Initial transients are allowed to dissipate and statistically-steady flow is established over 100 ms. After this point, the simulation is continued for 10 ms, during which the state is saved to disk at every physical time step, resulting in 10,001 snapshots (including $\mathbf{q}\,(\text{t} = 100\text{ ms})$). All PROM simulations are restarted from t = 100 ms. Several instantaneous flow field examples are shown in Figs. 6.2–6.4. Particular attention is drawn to the oscillatory pressure field displayed in Fig. 6.2: this emission of pressure waves from the trailing edge of the cavity is the result of the shear layer (originating from leading edge) impinging on

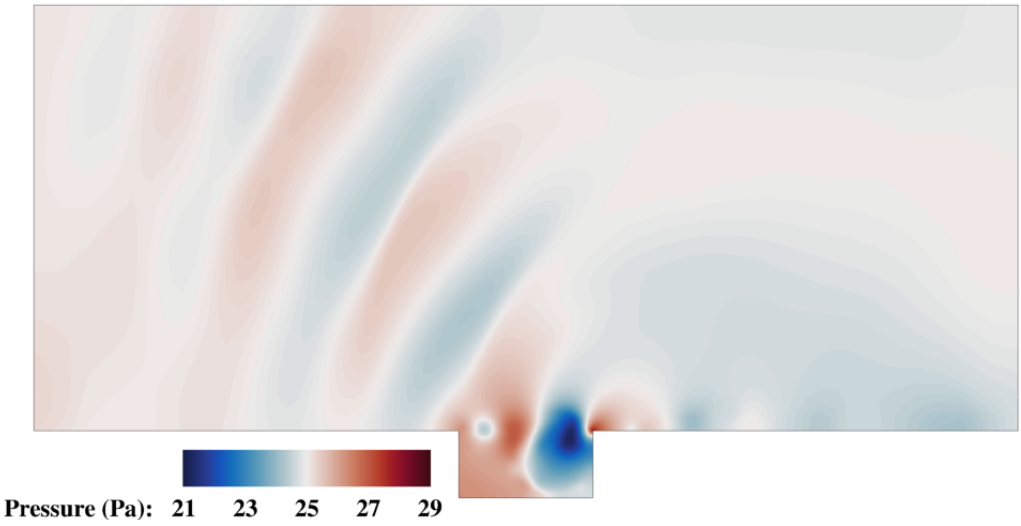



**Figure 6.2:** Cavity pressure field at t = 104 ms.

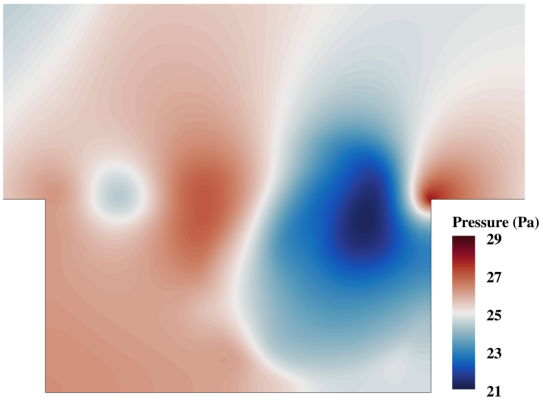



**Figure 6.3:** Cavity pressure field at t = 104 ms, zoomed view.

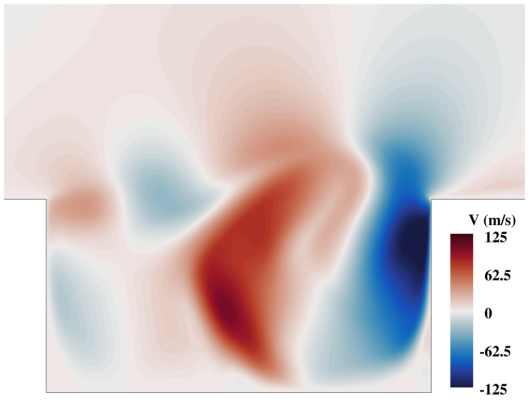



**Figure 6.4:** Cavity $y$-velocity field at t = 104 ms, zoomed view.

the trailing edge. The shear layer rollup (implied by the $y$-velocity field in Fig. 6.4) due to the Helmholtz instability and the pressure perturbation that travels upstream results in several resonant tones. Pressure measurements at the aft wall over the 10 ms data collection period is shown in Fig. 6.5.

For this flow regime and cavity geometry, the formula of Rossiter [169] (with $\alpha = 0.25$, $\kappa = 0.57$) predicts the first three acoustic modes to be $f = \{725.2,\ 1,692.1,\ 2,659.1\}$ Hz. As a rough confirmation of model suitability, 100 ms (t $\in [100,\ 200]$ ms) of pressure data are collected from the aft wall point monitor. The signal is filtered using a low-pass fifth-order Butterworth filter with a critical frequency of 20 kHz, and the power spectral density is computed by Welch's method with a window of 25 ms and 75% window overlap.

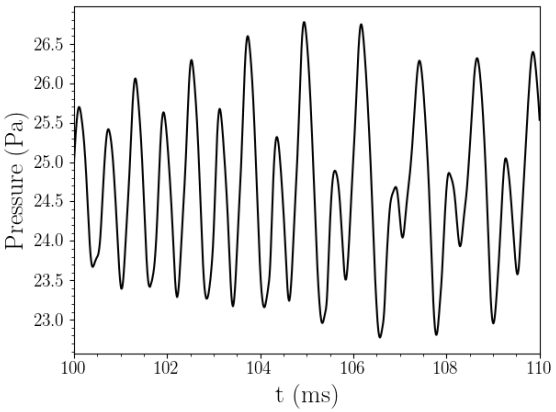



**Figure 6.5:** Pressure probe measurements from aft wall (t ∈ [100, 110] ms).

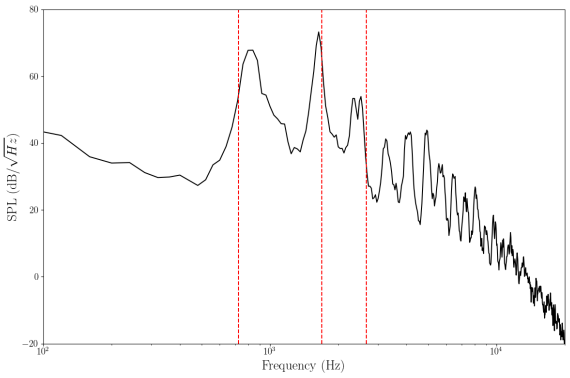



**Figure 6.6:** Sound pressure level of aft wall pressure signal (t ∈ [100, 200] ms). The first three Rossiter frequencies are marked in red.

The resulting sound pressure level is plotted in Fig. 6.6, where the first three predicted Rossiter frequencies are marked in red. Although slightly overpredicting the first mode and underpredicting the third mode, this is a reasonable match in comparing an empirical fit model and a two-dimensional numerical simulation.

The POD trial bases are computed from the 10,001 snapshots of the conservative and primitive variables. The POD residual energy decay is displayed in Fig. 6.7. Achieving 1%, 0.1%, and 0.01% of the conservative state POD residual energy requires 20, 73, and 155 modes respectively. For the primitive state, this increases to 26, 92, and 180 modes respectively. Although this is a fairly simple problem without any reaction phenomena, this slow POD residual energy decay exhibits how traveling waves and large fluctuations in the unsteady flow field can require a large number of trial bases to approximate accurately. Indeed, the primitive and conservative variable projection error plots in Fig. 6.8 indicate that over 125 modes are required to decrease the projection error of velocity and momentum magnitudes below 0.1% relative error.

### 6.1.2 Unsampled PROMs

The performance of unsampled Galerkin, LSPG, and MP-LSVT PROMs is examined first. As mentioned previously, the application of projection-based ROMs alone does not result in computational cost reduction, but still serves as a good performance baseline. If the unsampled PROM performs poorly, the hyper-reduced PROM can hardly be expected

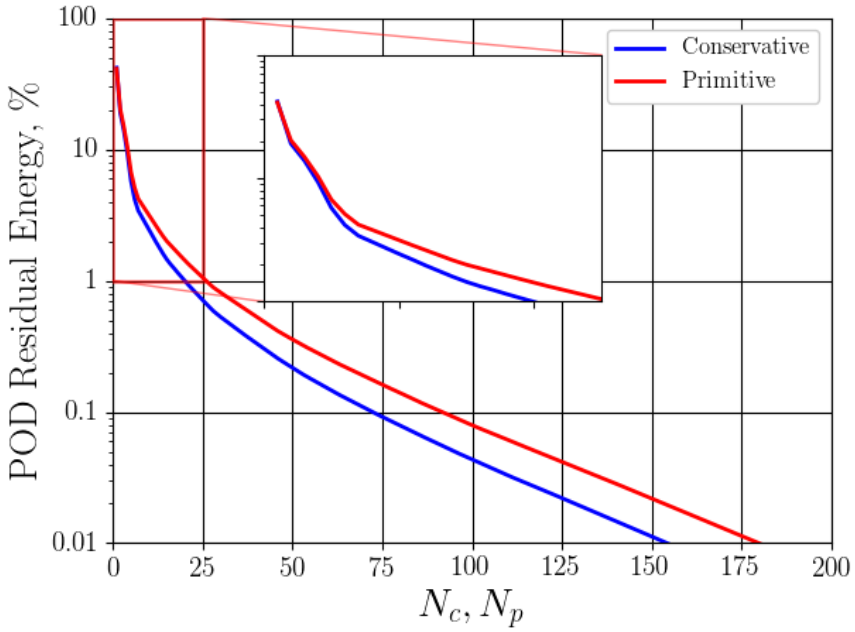
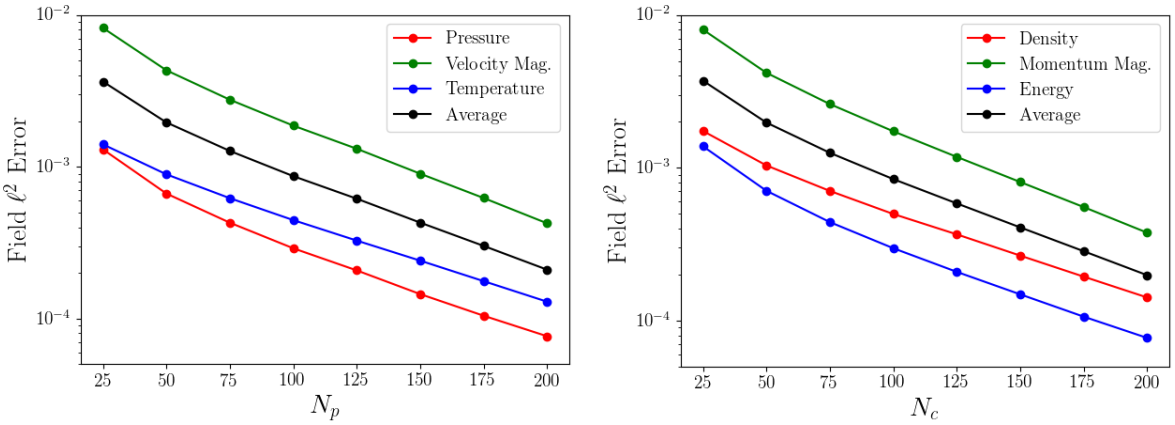
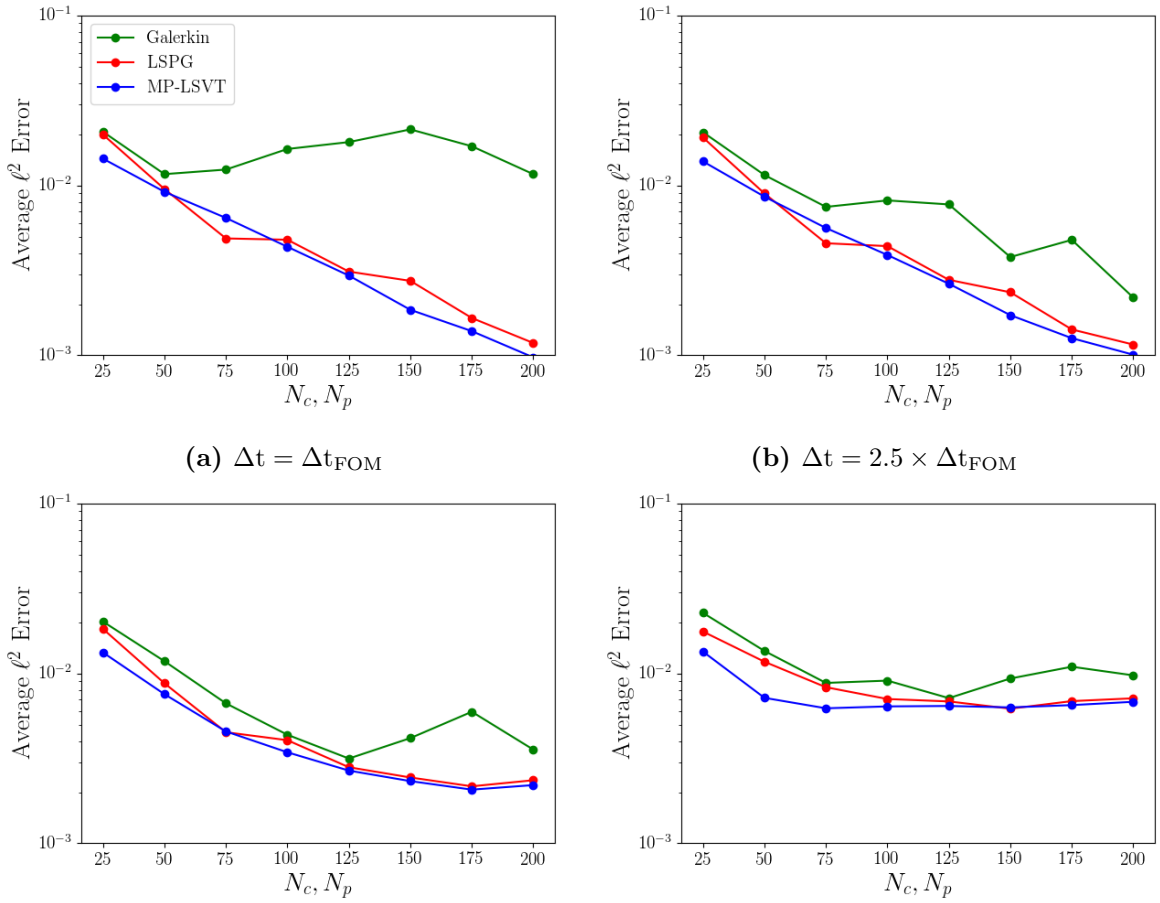



**Figure 6.7:** Cavity POD residual energy for conservative and primitive state datasets.




**(a)** Primitive variables **(b)** Conservative variables

**Figure 6.8:** Cavity time-average POD projection error.




**(a)** $\Delta t = \Delta t_{\text{FOM}}$ **(b)** $\Delta t = 2.5 \times \Delta t_{\text{FOM}}$

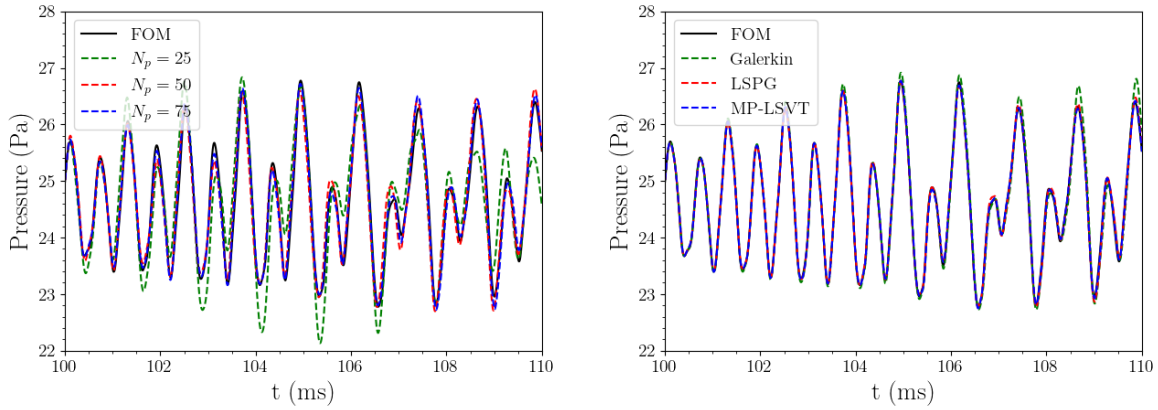
**(c)** $\Delta t = 5 \times \Delta t_{\text{FOM}}$ **(d)** $\Delta t = 10 \times \Delta t_{\text{FOM}}$

**Figure 6.9:** Cavity unsampled PROM time-average error, various $\Delta t$.

to perform well.

A trial basis dimension study and time step size study is conducted. The trial basis dimension $N_c/N_p$ is swept from 25 modes to 200 modes at 25-mode intervals. Four time step sizes are examined: $\Delta t \in \{1,\ 2.5,\ 5,\ 10\}$ $\mu$s, or 1, 2.5, 5, and 10 times that of the FOM simulation. As has been documented by previous work [69, 75, 125], increasing the PROM time step often achieves computational speedup with negligible increase in error relative to PROMs for which $\Delta t = \Delta t_{\text{FOM}}$, up to a point. Average error results for each evaluated time step are displayed in Fig. 6.9. Several indicative pressure probe measurements are displayed in Fig. 6.10.

Unremarkably, the LSPG and MP-LSVT PROMs outperform the Galerkin PROMs at all mode counts and time steps. Further, the LSPG and MP-LSVT PROMs generally exhibit non-increasing accuracy with trial basis enrichment, while error often *increases*




**(a)** MP-LSVT method, various $N_p$.

**(b)** $N_c, N_p = 150$, various methods.

**Figure 6.10:** Cavity unsampled PROM probe measurements, $\Delta t = 5 \times \Delta t_{FOM}$.

with trial basis enrichment for the Galerkin PROMs. For all time steps and trial basis dimensions evaluated, the LSPG and MP-LSVT PROMs perform similarly. This is not unexpected, as this is a non-reacting simulation which does not involve stiff source terms, extremely disparate spatio-temporal scales, or poor conditioning from which traditional LSPG PROMs might suffer. As mentioned previously, increasing the PROM time step may result in "free" computational cost savings. Indeed, increasing the PROM time step to $\Delta t = 2.5 \times \Delta t_{FOM}$ generates virtually no error increase for the LSPG and MP-LSVT PROMs, and even improves the performance of the Galerkin PROMs. However, at higher time step sizes, the accuracy of all PROMs for roughly $N_c, N_p > 50$ deteriorate drastically. In fact, for $\Delta t = 10 \times \Delta t_{FOM}$, accuracy improvement via trial basis enrichment saturates at $N_c, N_p = 50$, never dropping below 0.5%.

The pressure probe monitors in Fig. 6.10 display largely unsurprising results. As seen in Fig 6.10a, a very low trial basis dimension ($N_p = 25$), the solution quickly deviates and fails to reconstruct the FOM data faithfully. Large over- and under-shoot in the unsteady pressure signal are observed, and by the end of the simulation period the signal has devolved into small, unorganized fluctuations. Increasing the resolution to $N_p = 50$ improves the signal reconstruction, although small under- and over-shoot is observed by $t = 103$ ms. This small discrepancy is only marginally improved by increasing the trial basis dimension to $N_p = 75$, in agreement with the converging average error shown in Fig. 6.9c.

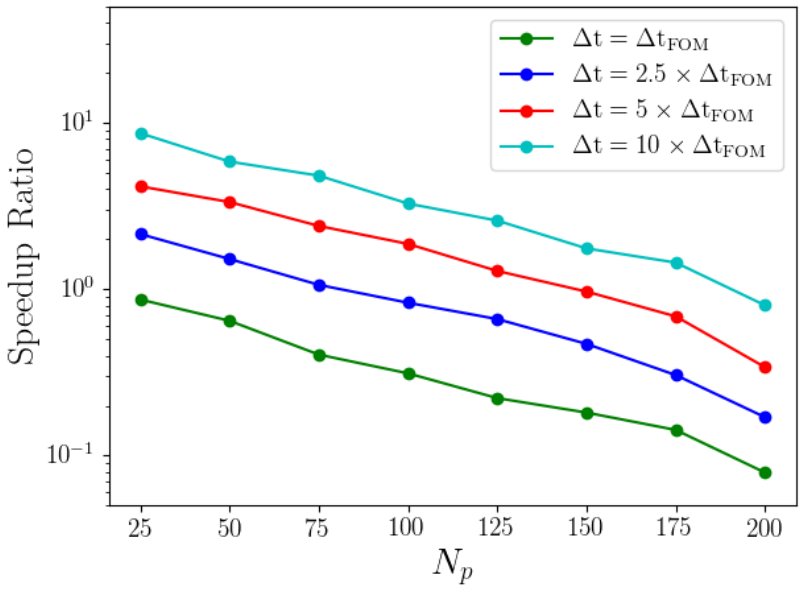
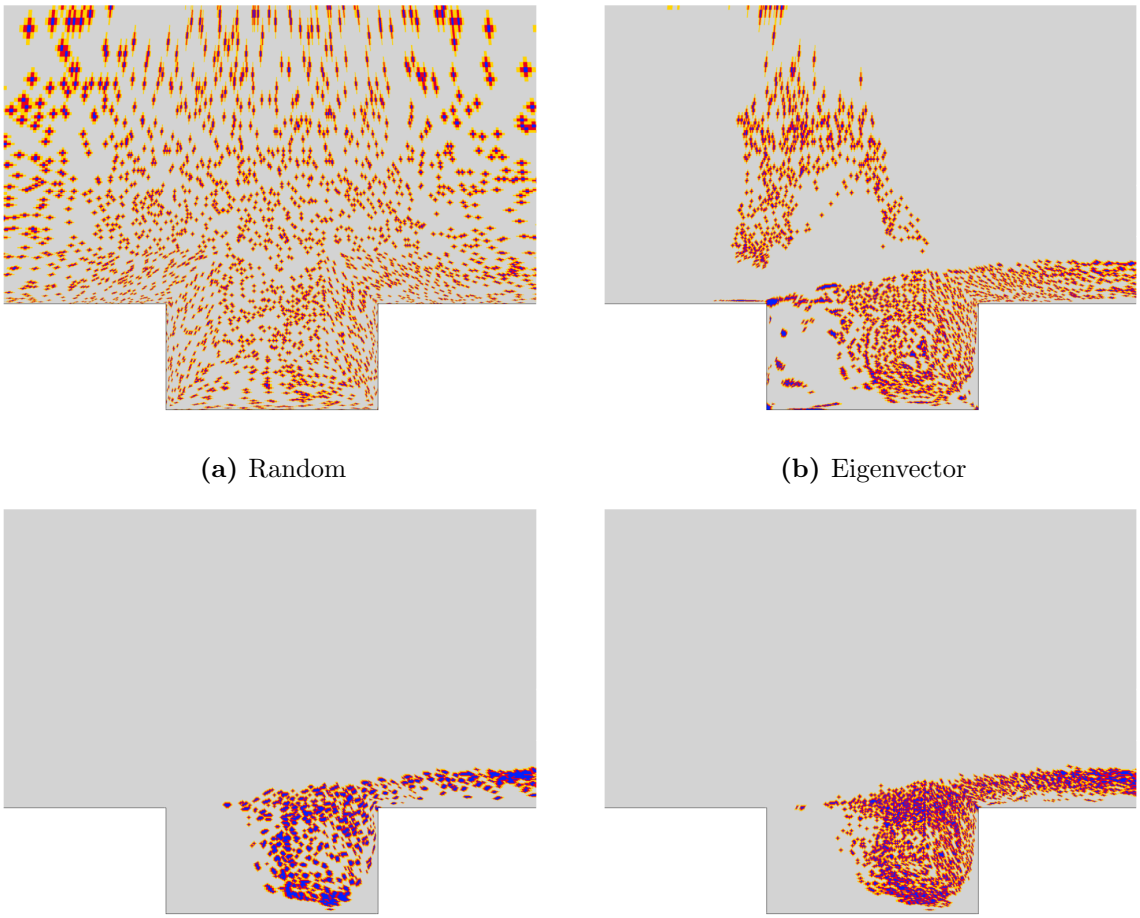



**Figure 6.11:** MP-LSVT unsampled PROM computational cost, relative to FOM cost, various $\Delta t$.

The computational cost of the unsampled MP-LSVT PROMs is now examined. Results are nearly identical for equivalent Galerkin and LSPG PROMS. As discussed previously, projection-based ROMs *alone* should not produce any significant computational cost savings, and in fact should increase cost due to lifting the state to the full-dimension and projecting the non-liner function onto the test space. Figure 6.11 quantifies just how much this change affects the run-time of the unsampled PROM. Note that using the same time step size as that of the FOM, the unsampled PROM is always more expensive than the FOM (below 1 on the $y$-axis) no matter the trial basis dimension. This highlights the necessity of hyper-reduction: projection-based order reduction of the governing system alone, even for extremely low trial/test basis dimensions, does not improve PROM computational cost. For $\Delta t = 2.5 \times \Delta t_{FOM}$, the unsampled PROM only achieves speedup for $N_p < 100$. This trend continues to improve the speedup for $\Delta t = 5 \times \Delta t_{FOM}$, $10 \times \Delta t_{FOM}$, though prior results showed that PROM accuracy quickly deteriorates at high time steps. For $N_p = 100$ and $\Delta t = 10 \times \Delta t_{FOM}$, the unsampled PROM is only capable of achieving three times speedup, which is hardly impressive given that these results only reconstruct the training period. To realize significant cost savings, hyper-reduction must be applied.



**(a)** Random **(b)** Eigenvector

**(c)** GNAT V1 **(d)** GNAT V2

**Figure 6.12:** Cavity sample mesh examples, $N_r = 250$, $N_s = 2.5\% \times N$.

### 6.1.3 Hyper-reduced PROMs

For the sake of brevity, only MP-LSVT HPROMs are examined. As will be seen in Section 6.2, Galerkin and LSPG PROMs appear unsuitable for practical reacting flows, and will not be discussed thereafter. All results presented in this section utilize a trial basis dimension of $N_p = 150$. All gappy POD regressor bases are constructed from POD modes of the conservative field dataset, i.e., $\boldsymbol{\Psi} = \mathbf{U}_c \in \mathbb{R}^{N \times N_r}$, as detailed in Section 4.4.2

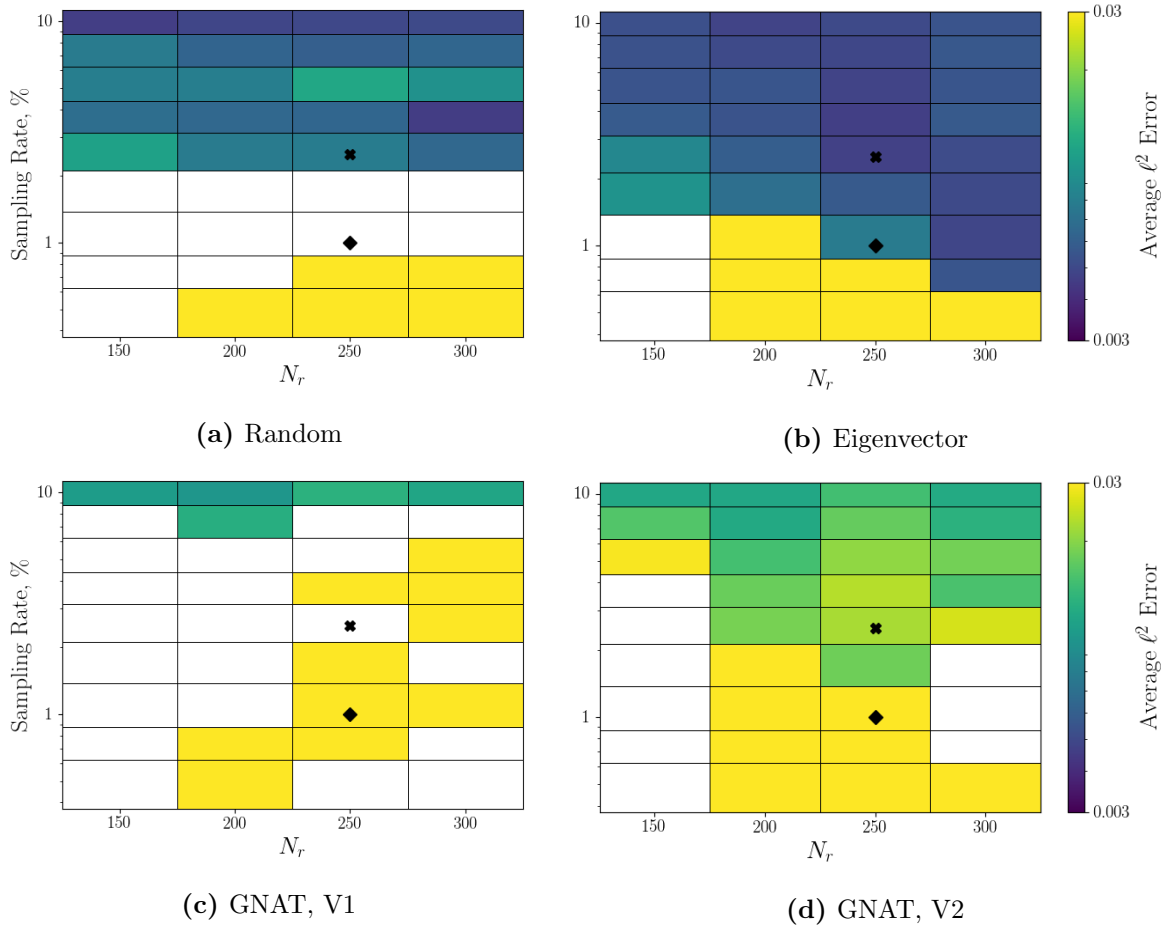
The accuracy and computational performance of the HPROMs is examined for sam-

| Sampling Rate (%) | 0.5 | 0.75 | 1 | 1.75 | 2.5 | 3.75 | 5.0 | 7.5 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| Cores | 2 | 2 | 2 | 2 | 3 | 5 | 6 | 9 | 13 |
| Cells/core (approx.) | 312 | 469 | 625 | 1,093 | 1,042 | 938 | 1,042 | 1,042 | 962 |

**Table 6.2:** Partitioning for cavity HPROM sample meshes.

pling rates of $N_s \in \{0.5,\ 0.75,\ 1.0,\ 1.75,\ 2.5,\ 3.75,\ 5.0,\ 7.5,\ 10\}\% \times N$, and gappy POD regressor dimensions of $N_r \in \{150,\ 200,\ 250,\ 300\}$. Views of indicative sample meshes constructed using the four sampling algorithms detailed in Section 4.5.2 are shown in Fig. 6.12. As in Section 4.6, blue cells indicate directly-sampled cells, red cells indicate flux cells, and yellow cells indicate gradient/vertex cells. The three greedy sampling algorithms generate sample meshes which are quite similar in some ways: the vast majority of sampled cells are located in the shear layer originating from the cavity leading edge and meandering downstream, as well as in the strong recirculation region in the downstream half of the cavity. The most glaring difference appears to be that the eigenvector-based algorithm also selects a number of points in the free-stream flow. Note that this region, although dominated by the free-stream conditions, experiences strong pressure oscillations emitted from the cavity trailing edge. Interestingly, the GNAT V1 algorithm selects sample cells in extremely tight clusters, while GNAT V2 generates a relatively more diffuse sample mesh, and the eigenvector-based algorithm's sample mesh is even more diffuse. Further, the eigenvector-based sampling selects a significant number of points on the leading edge of the cavity and in the upstream half of the cavity. It will be shown shortly how apparently minor discrepancies can have a drastic effect on HPROM performance. For all online HPROM results, each sample mesh is partitioned for parallel computations according to the sampling rate, as given in Table 6.2, regardless of the sampling algorithm utilized. The number of cores is altered for each sampling rate to ensure approximately equal load balancing, amounting to roughly 1,000 cells per core. Note that this is impossible for $N_s \leq 1\% \times N$, and two cores are used for such cases.

The accuracy of the online HPROMs on sample meshes generated by each sampling algorithm stand in stark contrast. Time-average $\ell^2$ error contour plots for $\Delta t = 5 \times \Delta t_{\text{FOM}}$ are given in Fig. 6.13,comparing accuracy at each combination of sampling rate $N_s$ and gappy POD regressor basis dimension $N_r$. White squares indicate simulations which exploded. Across all sampling algorithms, note the unsurprising result that increasing $N_s$ and $N_r$ tends to improve HPROM performance. However, it is quite plain that eigenvector-based sampling consistently generates more stable and accurate HPROMs



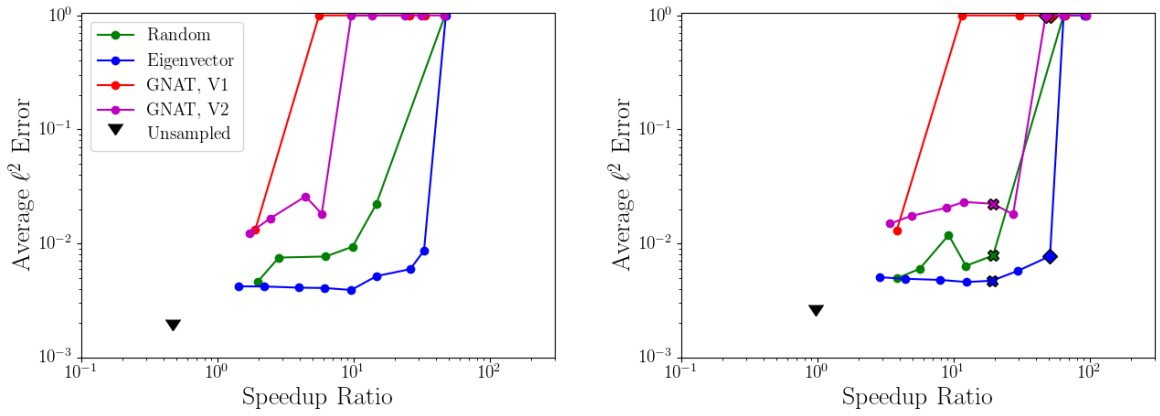

**(a)** Random

**(b)** Eigenvector

**(c)** GNAT, V1

**(d)** GNAT, V2

**Figure 6.13:** Cavity HPROM time-average error contours with respect to gappy POD regressor dimension and sampling rate, $\Delta t = 5 \times \Delta t_{\text{FOM}}$, various sampling algorithms.



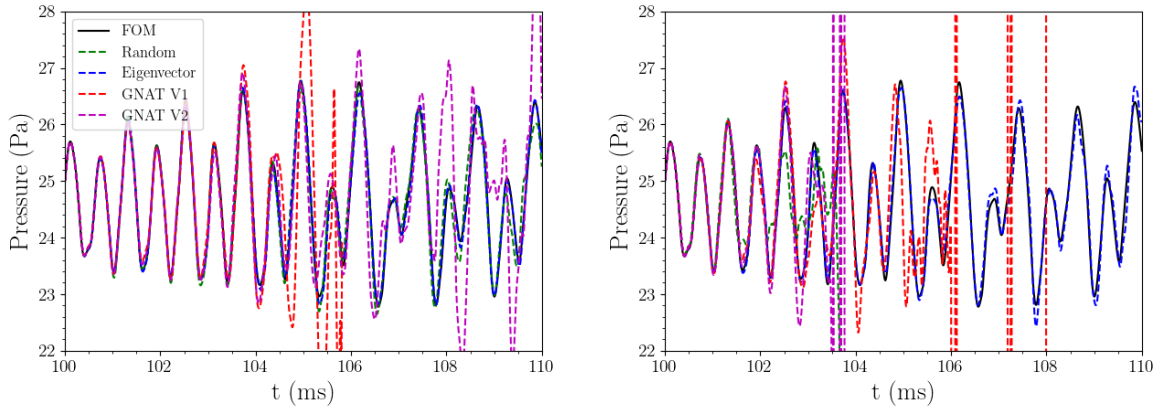

**(a)** $\Delta t = 2.5 \times \Delta t_{FOM}$ **(b)** $\Delta t = 5 \times \Delta t_{FOM}$

**Figure 6.14:** Cavity HPROM error vs. CPU-time speedup, $N_p = 150$, $N_r = 250$, various $\Delta t$.

than any other sampling algorithm, only failing at extremely low sampling rates. Interestingly, random sampling tends to outperform both GNAT sampling algorithms, though it only produces stable simulations for $N_s > 2.5\% \times N$. In fact, GNAT V1 produces consistently stable simulations only for $N_s = 10\% \times N$, and even those are highly inaccurate. The GNAT V2 algorithm is capable of producing stable simulations at lower sampling rates, but also incurs relatively high error in the solution.

Of course, computational cost savings is the prime objective of hyper-reduction, and an accurate HPROM does not guarantee a fast HPROM. As one might expect, increasing $N_s$ and $N_r$ increases HPROM computational cost, as will be explored more thoroughly in Section 6.2. Figure 6.14 displays the time-average error with respect to the speedup ratio at each sampling rates for all sampling algorithms. The eigenvector-based sampling enables stable, accurate HPROMs which are able to achieve over 200 times computational cost savings, while the equivalent unsampled PROMs are either equally or more expensive than the FOM. Similarly, random sampling is capable of producing accurate ($< 1\%$ error), though only at higher sampling rates and achieving relatively lower speedup ratios. The GNAT sampling algorithms generally fail to produce meaningful cost savings without incurring instability or high error.

To better visualize these accuracy discrepancies, Fig. 6.15 displays pressure probe monitors for each sampling algorithm for two different sampling rates. The corresponding average error for Fig. 6.15a are marked with an "X" in Figs. 6.13 and 6.14, and those




**(a)** $N_s = 2.5\% \times N$ **(b)** $N_s = 1\% \times N$

**Figure 6.15:** Cavity MP-LSVT HPROM probe measurements, $N_p = 150$, $N_r = 250$, $\Delta t = 5 \times \Delta t_{\text{FOM}}$.

for Fig. 6.15b are similarly marked by a diamond. In general, all sampling algorithms are able to produce stable and accurate PROMs for the simulation period $t \in [100,\ 102]$ ms. However, those HPROMs generated by the GNAT sampling algorithms quickly deviate from the FOM, exhibiting wild swings in the unsteady pressure signal. Although random sampling generates an unstable HPROM for $N_s = 1\% \times N$, it produces a fairly accurate simulation for $N_s = 2.5\% \times N$, only exhibiting minor under- and over-shoot towards the end of the simulation period. Eigenvector-based sampling produces accurate reconstruction of the unsteady pressure signal, particularly for $N_s = 2.5\% \times N$, though it does exhibit some discrepancies at later times for $N_s = 1\% \times N$.

Although the above results indicate that it is possible to achieve excellent computational cost savings with HPROMs, they also hint at the challenge of guaranteeing that they are accurate and robust. The drastic effects that the sample mesh and gappy POD regression have on HPROM performance invites more thorough exploration for larger, more challenging systems.

## 6.2 Continuously-variable Resonance Combustor

The first practical reacting flow case of this thesis is now detailed, namely a model of the continuously-variable resonance combustor (CVRC) with a truncated combustion

chamber. The CVRC experiment originated in work by Yu *et al.* [176, 177, 178] at Purdue University, whereby a single-element, gaseous propellant, coaxial dump injection rocket combustor was designed to allow for continuous actuation of the oxidizer injection post. This actuation revealed that certain oxidizer post lengths gave rise to high-frequency combustion instabilities. The CVRC has become a useful numerical benchmark case for simulating rocket combustion [179, 180] and investigating the mechanics which contribute to combustion instabilities in rocket engines [181, 182, 183].

In this work, the full injector and combustor geometry of the CVRC is not modeled, unlike in [181]. Instead, as will be detailed in the following section, the oxidizer injection choke manifold is neglected and the combustion chamber is truncated well downstream of the dump plane. This has the effect of decreasing computational costs, while still including much of the complex combusting flow phenomena such the mixing shear layer, recirculation of hot products, and large-scale transport and mixing in the combustion chamber. Acoustic forcing is not applied at the outlet to mimic the combustion instability, as this would necessitate complex boundary treatments in any PROMs, which are beyond the scope of this work. The reader is directed to work by Huang *et al.* [126] for a thorough investigation of artificial boundary forcing treatments for PROMs.

### 6.2.1 Full-order Model

The truncated CVRC case presented here generally follows that investigated by Harvazinski and Shimizu [183], with a truncated combustion chamber. The geometry is displayed in Fig. 6.16. The oxidizer post extends approximately 14 cm upstream of the dump plane ($x = 0$ m), with a diameter of 2.05 cm. The annular fuel injection port has an outer diameter of 23.09 cm and an inner diameter of 22.27 cm, extends 30.55 cm upstream of the dump plane, and enters the oxidizer stream 10.18 cm upstream of the dump plane. The combustion chamber has a diameter of 4.5 cm, and extends approximately 14 cm downstream of the dump plane. The oxidizer inlet injects 42.35% gaseous oxygen and 57.65% water vapor by mass at 1,030 K, with a specified mass flow rate of 0.32 kg/s. Gaseous methane is injected through the fuel duct at 300 K, with a mass flow rate of

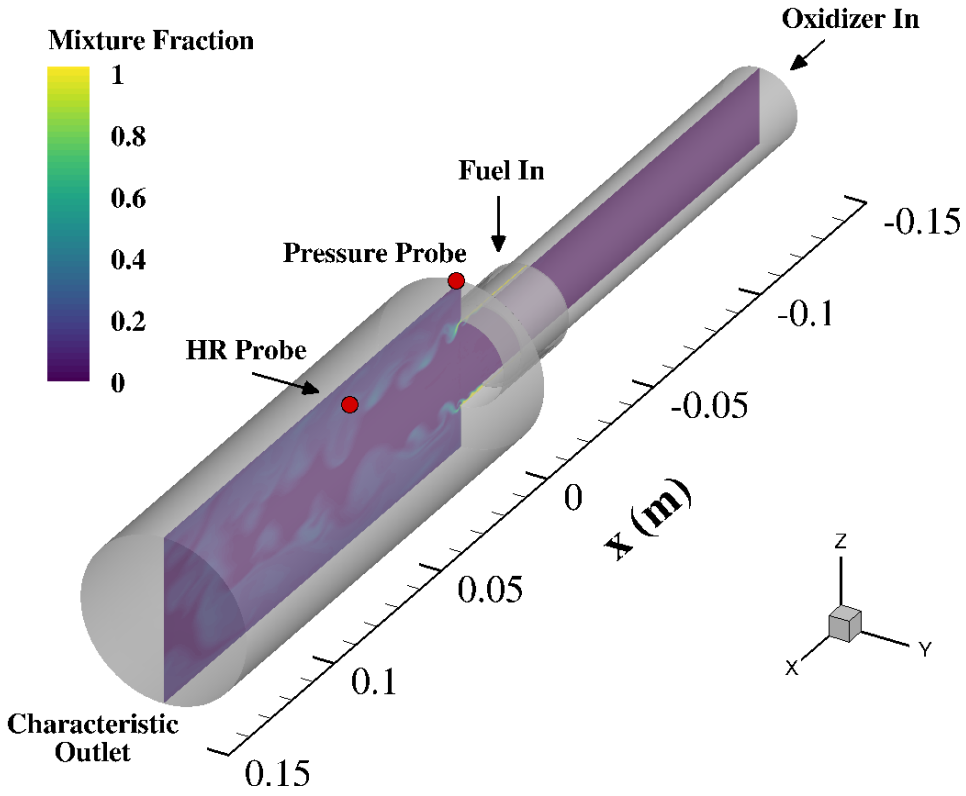



**Figure 6.16:** Truncated CVRC geometry with $x - z$ plane slice at t = 5.5 ms.

0.027 kg/s. The computational mesh is composed of 2,637,771 hexahedral cells, resulting in a total number of degrees of freedom $N$ = 18,464,397. No-slip, adiabatic conditions are enforced at the domain walls, and a subsonic characteristic boundary condition is applied at the outlet.

Combustion is modeled using the FPV approach [34] detailed in Section 2.2.2. As detailed previously, a library of steady diffusion flame solutions is pre-computed off-line, generating a lookup table mapping from the mixture fraction $Z$ and the progress variable $C$ to individual species mass fractions, i.e., $Y_i = Y_i(Z, C)$. For this case, the steady flame solutions are solved using the FlameMaster software [103] with the GRI-Mech 1.2 methane combustion mechanism [24]. The GRI-Mech 1.2 mechanism contains 32 chemical species and 177 reactions. All gases are treated as thermally perfect gases, and thermodynamic quantities (specific heats, enthalpy, and entropy) as well as transport properties (mass diffusivity, viscosity, and thermal conductivity) are computed from polynomial functions of temperature developed by McBride *et al.* [40] and described in Section 2.1.1.

The FOM simulation is computed with a physical time step of $\Delta t_{\text{FOM}} = 0.1$ $\mu$s. The

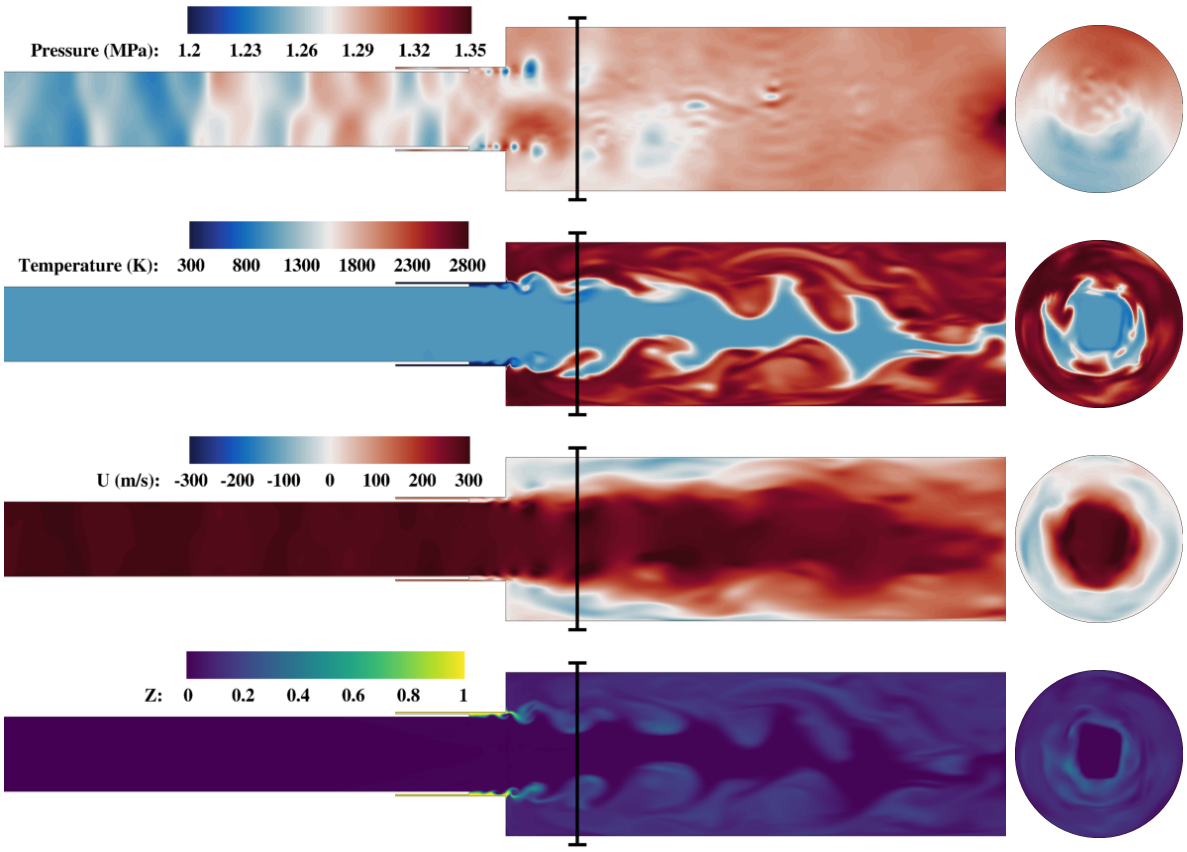



**Figure 6.17:** From top to bottom: pressure, temperature, axial velocity, and fuel mixture fraction slices at t = 5.5 ms.

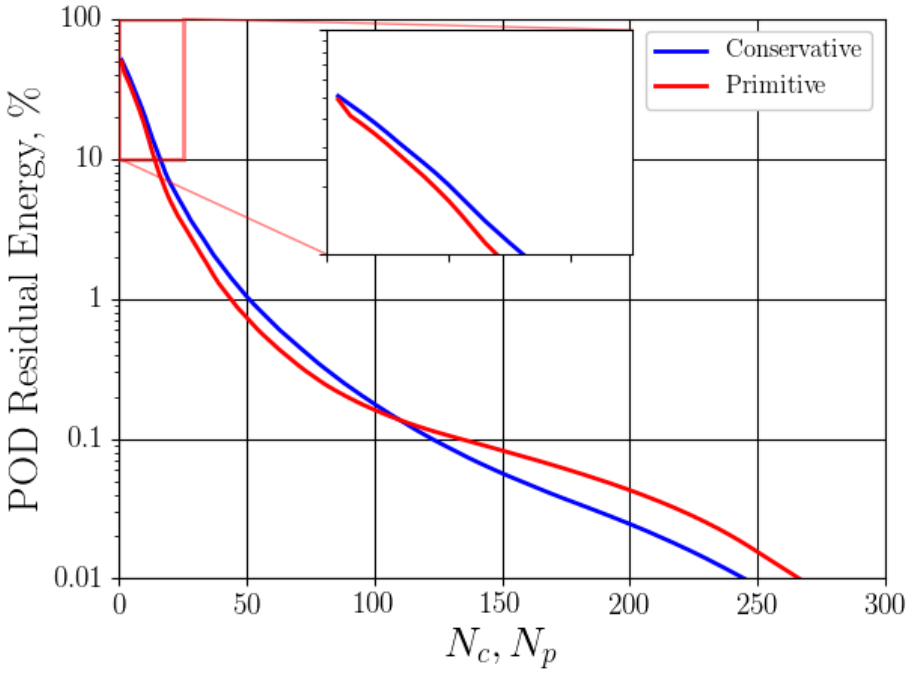



**Figure 6.18:** POD residual energy decay for truncated CVRC conservative and primitive state datasets.

fluid domain is initialized with a pressure of 1 MPa, and the combustion chamber is filled with hot products at 2,000 K to initiate combustion. Initial transients exit the domain before data collection begins at 5 ms of simulation time. Data snapshots are collected over a 0.5 ms window, sampled every five time steps, corresponding to roughly one flow-through period in the combustion chamber. This results in 1,001 snapshots (including the initial condition at t = 5 ms) of the conservative and primitive states. Representative snapshot slices of various flow fields are displayed in Fig. 6.17. Pressure point monitor measurements are collected from the dump plane corner at approximately $x = (0,\ 0,\ 2.25)$ cm, and unsteady heat release point monitor measurements are collected from the reacting mixing layer at approximately $x = (5,\ 0,\ 1.5)$ cm.

Figure 6.18 displays the POD residual energy for the model rocket combustor data. Achieving 1%, 0.1%, and 0.01% residual energy for the conservative dataset requires 51, 123, and 245 basis modes, respectively. For the primitive dataset, these levels require 44, 134, and 267 modes respectively. This decay is significantly slower than that observed for 2D open cavity flow (Fig. 6.7), and is indicative of the difficulty with which linear trial

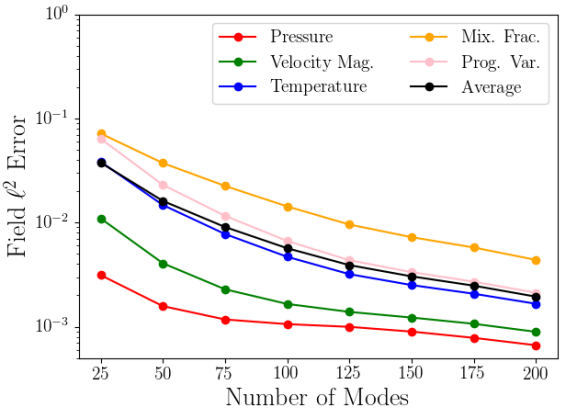



**Figure 6.19:** Primitive variables time-average projection error.

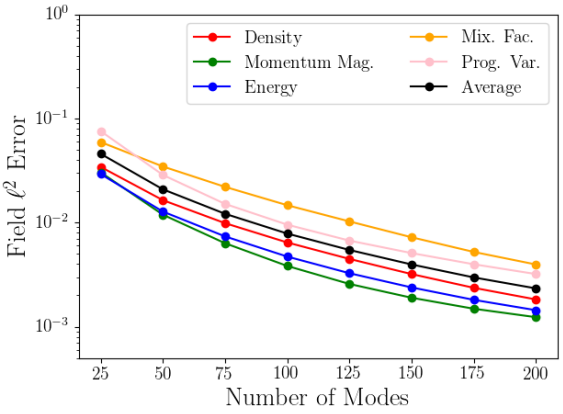



**Figure 6.20:** Conservative variables time-average projection error.

spaces capture the highly non-linear flow physics which characterize rocket combustion. This is further emphasized in the projection error plots shown in Figs. 6.19 and 6.20, where it is apparent that the fuel mixture fraction and progress variable fields induce much higher error levels at a given trial basis dimension than the primary flow fields.

As with the 2D cavity case, this full-order model should not be considered a well-resolved or accurate representation of the complex combustion phenomena which occur in real liquid-propellant rocket combustor. Although this model is derived from numerical experiments [181] which exhibited reasonable predictions of certain quantities of interest, the results presented here should be contextualized strictly as approximating the system dynamics modeled by the underlying numerical solver, which this thesis does not claim to perfectly represent reality.

### 6.2.2 Unsampled PROMs

The performance of the unsampled PROMs are again examined before proceeding to HPROMs. Discussion is restricted to MP-LSVT PROMs, and the exclusion of Galerkin and LSPG PROMs is explained at the end of this section. A trial basis dimension and time step study is again conducted. The trial basis dimension $N_p$ is again evaluated at 25-mode intervals from 25 modes to 200 modes. Four time step sizes are examined: $\Delta t \in \{0.1,\ 0.25,\ 0.5,\ 1\}$ $\mu$s, or 1, 2.5, 5, and 10 times that of the FOM simulation.

The time-average error results are displayed in Fig. 6.21. Note that, in general,

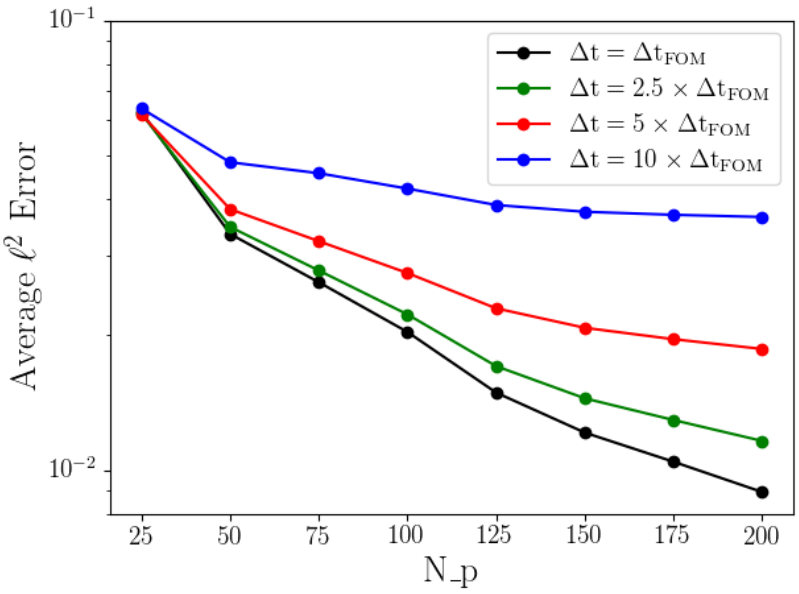
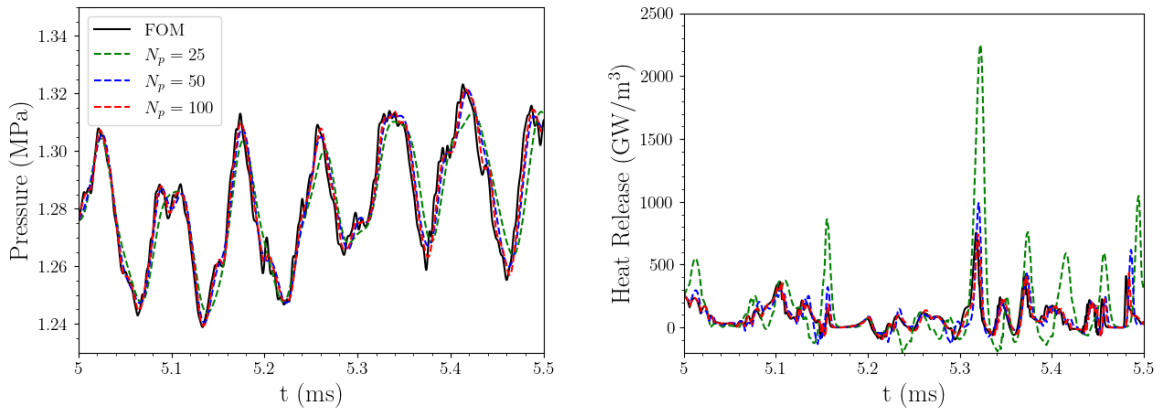



**Figure 6.21:** CVRC unsampled PROM time-average error, various $\Delta t$.

the error levels induced for this case are significantly higher than those observed in the cavity flow case. Whereas the unsampled cavity MP-LSVT PROMs easily achieved less than 1% relative $\ell^2$ error for all time steps for $N_p \geq 50$, this is only achieved here for $\Delta t = \Delta t_{FOM}$, $N_p = 200$. This is yet another indicator of the difficulty in accurately modeling reacting flows. However, observe the similar trend that moderate increases in the PROM time step result in negligible increase in PROM error, while larger time steps greatly diminish the PROM's accuracy and quickly saturates accuracy improvements with trial basis enrichment. For $\Delta t = 10 \times \Delta t_{FOM}$, unlike the cavity whose time-average error saturated at approximately 0.8%, error saturates at roughly 4% for this case.

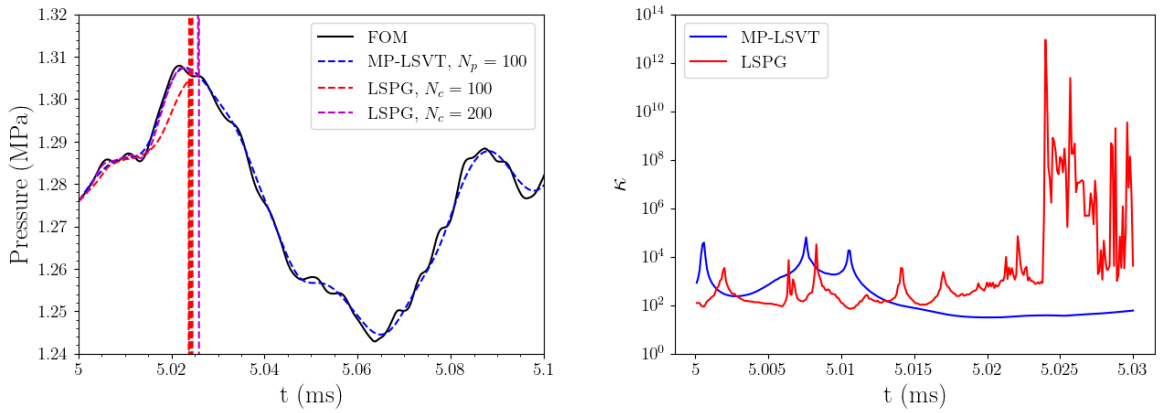
The effect of trial basis enrichment is visualized in Fig. 6.22. Interestingly, the pressure signal at the dump plane corner, shown in Fig. 6.22a, is relatively easy to capture, although significant smearing of small scale fluctuations can be observed for $N_p = 25$, and less so for $N_p = 50$. For a trial basis dimension of $N_p = 100$, even very small scale fluctuations are captured well. Comparisons for the reacting mixing layer heat release, however, are more telling. Figure 6.22b indicates that $N_p = 25$ results in dramatic over- and under-prediction of heat release across the entire simulation period. Increasing the trial basis resolution to $N_p = 50$ improves this error, but does not eliminate it. Again,




**(a)** Dump plane pressure.

**(b)** Shear layer heat release.

**Figure 6.22:** CVRC unsampled PROM probe measurements, $\Delta t = 5 \times \Delta t_{FOM}$, various $N_p$




**(a)** Probe measurements.

**(b)** Condition number, $N_c, N_p = 100$

**Figure 6.23:** CVRC unsampled MP-LSVT and LSPG PROM comparisons.

$N_p = 100$ results in excellent time-accurate reconstruction of the unsteady heat release in the reacting mixing layer.

The absence of any analysis for LSPG PROMs above and throughout the remained of this section is now explained. As documented by Huang *et al.* for a 2D single-element rocket combustor [75], Galerkin and LSPG PROMs exhibit increased stiffness in the resulting linear temporal evolution system, compared to that generated by the equivalent MP-LSVT PROM. The result is drastically degraded stability and accuracy, such that all investigated Galerkin PROMs were unstable and LSPG PROMs exhibited more than double the error measured for MP-LSVT PROMs with the same trial basis dimension. For the truncated CVRC case investigated here, *all* LSPG PROMs computed unstable solutions, even with significant basis enrichment. Figure 6.23a indicates the

| Sampling Rate (%) | 0.025 | 0.0375 | 0.05 | 0.075 | 0.1 | 0.175 |
|---|---|---|---|---|---|---|
| Cores | 2 | 2 | 2 | 2 | 3 | 5 |
| Cells/core (approx.) | 330 | 495 | 660 | 989 | 880 | 923 |
| Sampling Rate (%) | 0.25 | 0.375 | 0.5 | 0.75 | 1 | |
| Cores | 7 | 10 | 13 | 20 | 26 | |
| Cells/core (approx.) | 942 | 989 | 1,015 | 989 | 1,015 | |

**Table 6.3:** Partitioning for CVRC HPROM sample meshes.

effect of this instability, with a rapid explosion in pressure measurements at roughly t = 5.025 ms for $N_c \in \{100,\ 200\}$. The MP-LSVT PROM with $N_p = 100$ is both stable and noticeably more accurate than the equivalent LSPG PROM. Investigating further, Fig. 6.23b displays the evolution of the condition number of the first LSPG and MP-LSVT Newton iteration, i.e. $\kappa\left(\left[\mathbf{W}^k\right]^\top \mathbf{W}^k\right)$. This roughly measures the stiffness of the PROM linear solve, and reveals that while both methods experience similar ill conditioning at first, the conditioning of the MP-LSVT PROM gradually lessens while that of the LSPG PROM gradually increases and eventually explodes.

### 6.2.3 Mesh Sampling and Load Balancing

The pre-processing stage of developing hyper-reduced PROMs for the truncated CVRC is now discussed. Although a brief overview of hyper-reduction computational processes was given in Section 6.1, more attention is given here to some of the nuances related to load balancing and MPI communications for parallel computations.

As with the cavity flow case, the four sampling algorithms described in Section 4.5.2 are studied, and analyses are conducted for a range of sampling rates and gappy POD regressor basis dimensions. The sampling rates investigated and the parallel partitioning for each sample mesh (regardless of the sampling algorithm used) are summarized in Table 6.3, and are again adjusted to ensure that the load balance is approximately equal to 1,000 cells per core.

Visually inspecting several indicative samples meshes, with slices in Fig. 6.24 and iso-surfaces in Fig. 6.25, it is clear that the spatial distribution of the sample mesh points

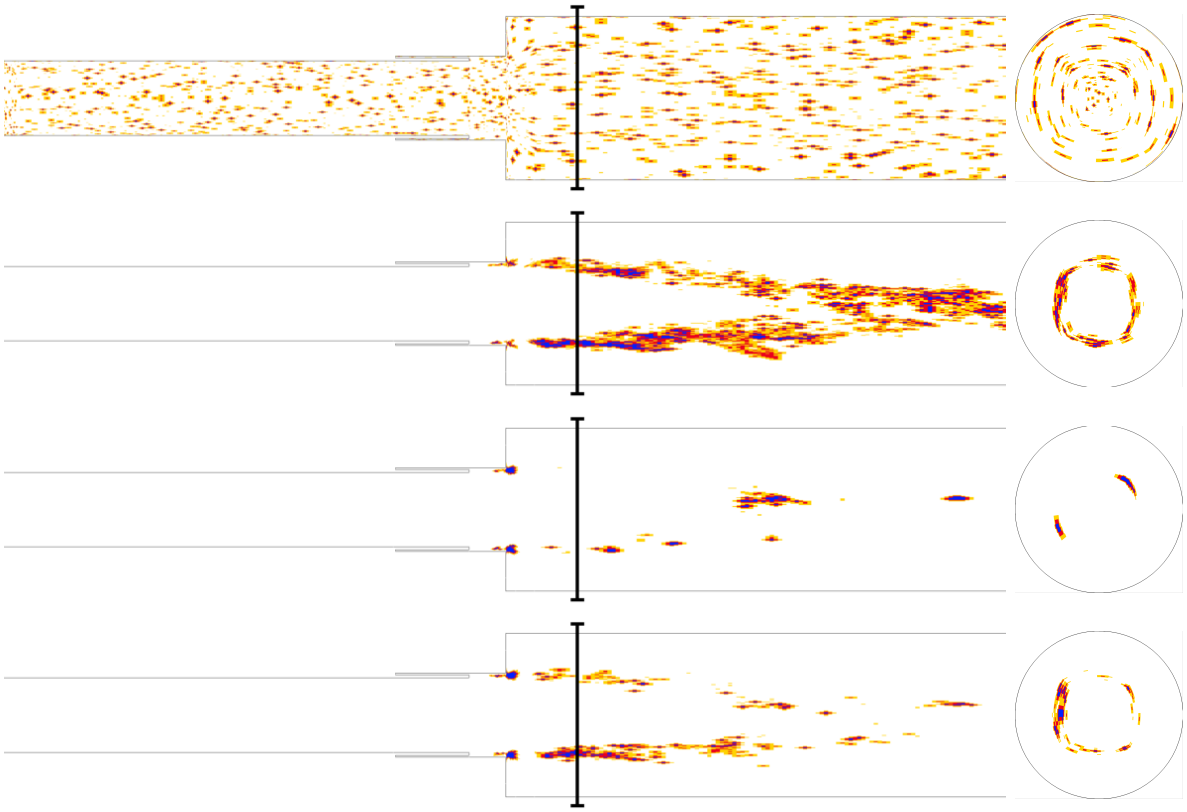
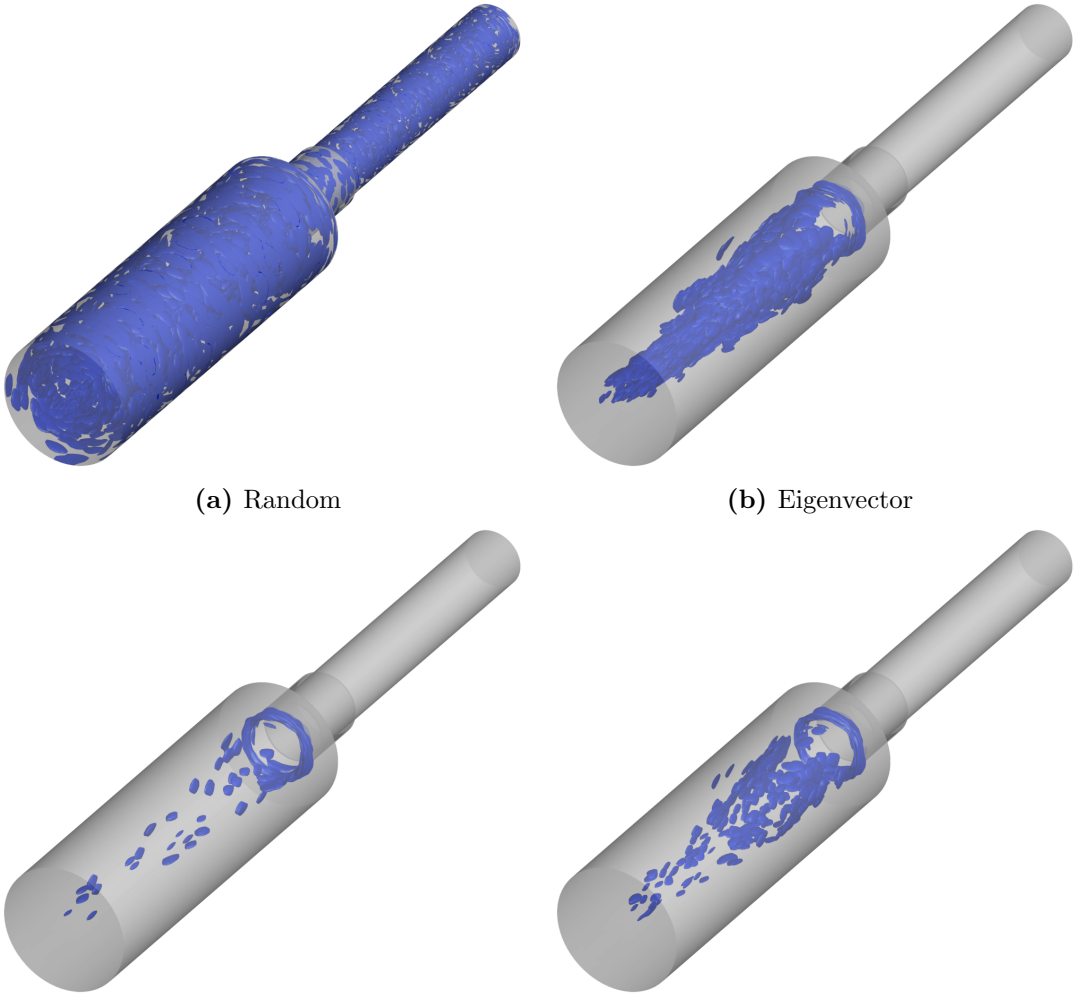


**Figure 6.24:** Example CVRC sample meshes for $N_s = 0.25\% \times N$, $N_r = 300$ for various sampling algorithms. From top to bottom: random, eigenvector-based, GNAT V1, and GNAT V2.

varies drastically between sampling strategies. Random sampling, as expected, selects points in a random pattern, including a significant number of points in the oxidizer duct. This might be considered computationally wasteful, as the physics in the oxidizer duct are quite simple, maintaining a roughly fixed temperature, velocity, and chemical composition, with only minor variations in pressure. Eigenvector-based sampling and both variants of GNAT sampling, on the other hand, display a vastly different result. In each case, sample mesh elements are clustered around the reacting shear layer downstream of the dump plane. This is particularly clear for eigenvector-based sampling, in which the sampling points form a conical shape aligned with the mixing layer of the annular fuel injection stream. This result is not entirely surprising, as the physics in the shear layer are by far the most complex in the system, featuring sharp gradients and strong non-linearities. Interestingly, both GNAT sampling methods generate extremely tight clusters in the vicinity of the fuel dump region, although the second variant selects some element in the downstream combustion chamber region. These findings are largely similar to those for the cavity flow case: among the greedy sampling algorithms, eigenvector sampling generates a larger, more diffuse sample mesh, while GNAT sampling clusters sampling points closely.

How these differences in sampling affect the load balancing of the mesh partitions is quantified in Fig. 6.26. Figure 6.26a displays the average number of cells per partition in the sample mesh as a percentage of the full mesh, when the sample mesh is split into ten partitions. Unsurprisingly, random sampling generates the largest meshes for all sampling rates, as the disperse sampling requires a larger number of auxiliary cells. Conversely, both GNAT sampling algorithms produce much smaller, compact sample meshes due to the tight clustering of sample points and commensurately fewer auxiliary cells they require. At low sampling rates, eigenvector-based sampling produces similar sample mesh sizes to random sampling, but approaches sizes similar to GNAT sampling at higher sampling rates.

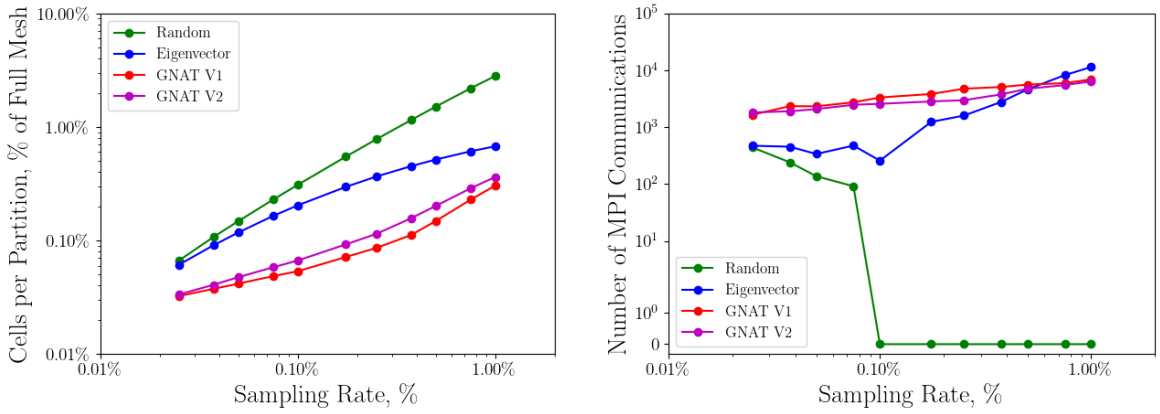
For much of the same reasons, the opposite is true for the volume of inter-core communications that are required for a given sample mesh. Figure 6.26b shows that in many



**(a)** Random **(b)** Eigenvector

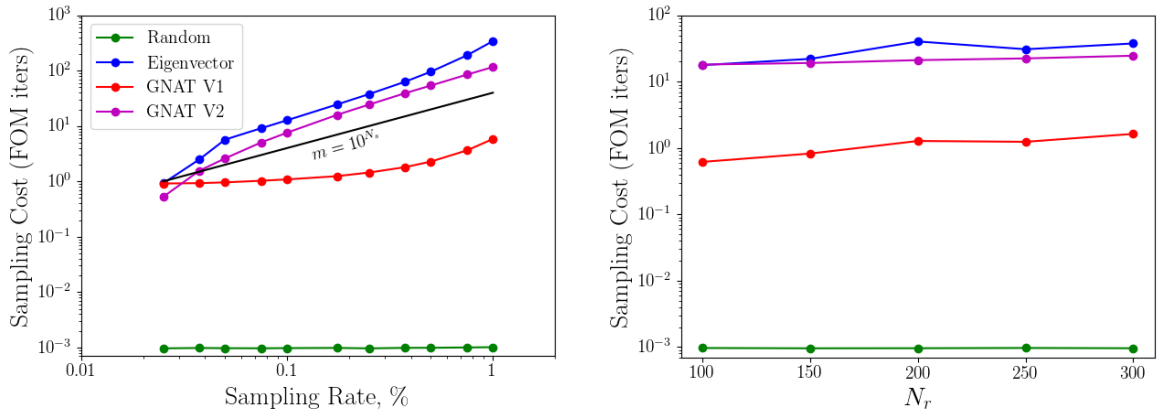
**(c)** GNAT V1 **(d)** GNAT V2

**Figure 6.25:** Directly-sampled cell isosurfaces, $N_s = 0.25\% \times N$, $N_r = 300$, various sampling algorithms.




**(a)** Average cells per partition, % of total mesh. **(b)** Total point-to-point MPI communications.

**Figure 6.26:** Sample mesh partition statistics, 10 partitions, various sampling rates.



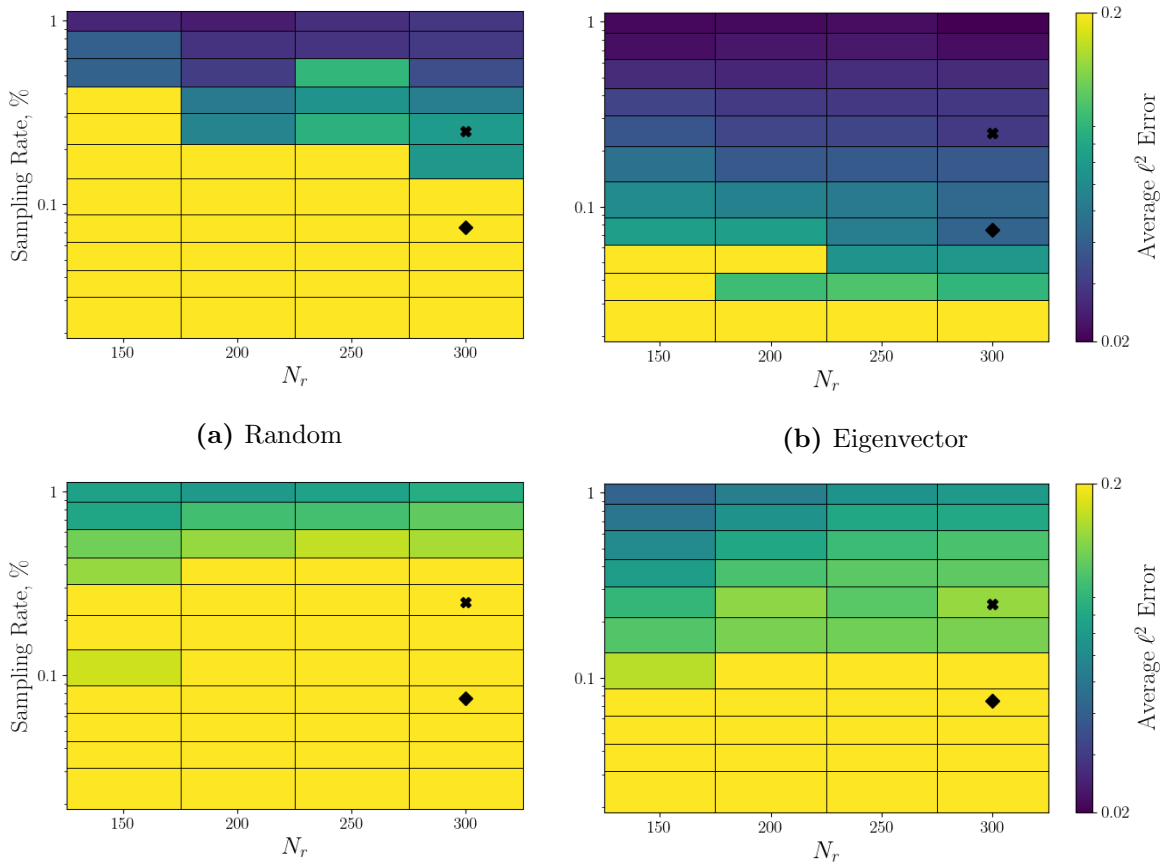

**(a)** W/r/t sampling rate, $N_r = 300$.

**(b)** W/r/t $N_r$, $N_s = 0.25\% \times N$.

**Figure 6.27:** Sample selection cost, relative to cost of a single FOM iteration.

cases, random sampling produces sample meshes which require no point-to-point MPI communications whatsoever. This is a function of the aforementioned disperse sampling which results in a large number of disjoint graphs which can be effectively separated by the partitioning software. This is not the case at extremely low sampling rates, which are dominated by mesh elements selected by the QDEIM procedure. Both GNAT sampling algorithms produce nearly identical volumes of communication at all sampling rates. This follows logically from the fact that tight sampling clusters require edge cuts to ensure effective load balancing. Eigenvector-based sampling induces a lower volume than GNAT sampling at lower sampling rates, but more at higher sampling rates.

Finally, Fig. 6.27 examines the offline cost of evaluating the various sampling algorithms. The cost is non-dimensionalized by the core-hours required to compute one FOM iteration. Of course, the cost of random sampling is nearly zero, scales negligibly with respect to the sampling rate 6.27a, and is not influenced by the number of modes retained in the regression basis $\mathbf{\Psi}$ (Fig. 6.27b). The greedy algorithms, on the other hand, incur noticeable computational cost. On the whole, the original GNAT sampling algorithm is significantly less expensive, obviously because it requires only $N_r$ evaluations of the greedy metric, while Peherstorfer's variant and eigenvector-based sampling require approximately $N_s/N_v$ evaluations. Eigenvector-based sampling is only slightly more costly than Peherstorfer's GNAT variant. All greedy methods appear to scale sublinearly with respect to $N_r$, while eigenvector-based sampling and Peherstorfer's GNAT variant seem




**(a)** Random

**(b)** Eigenvector

**(c)** GNAT, V1

**(d)** GNAT, V2

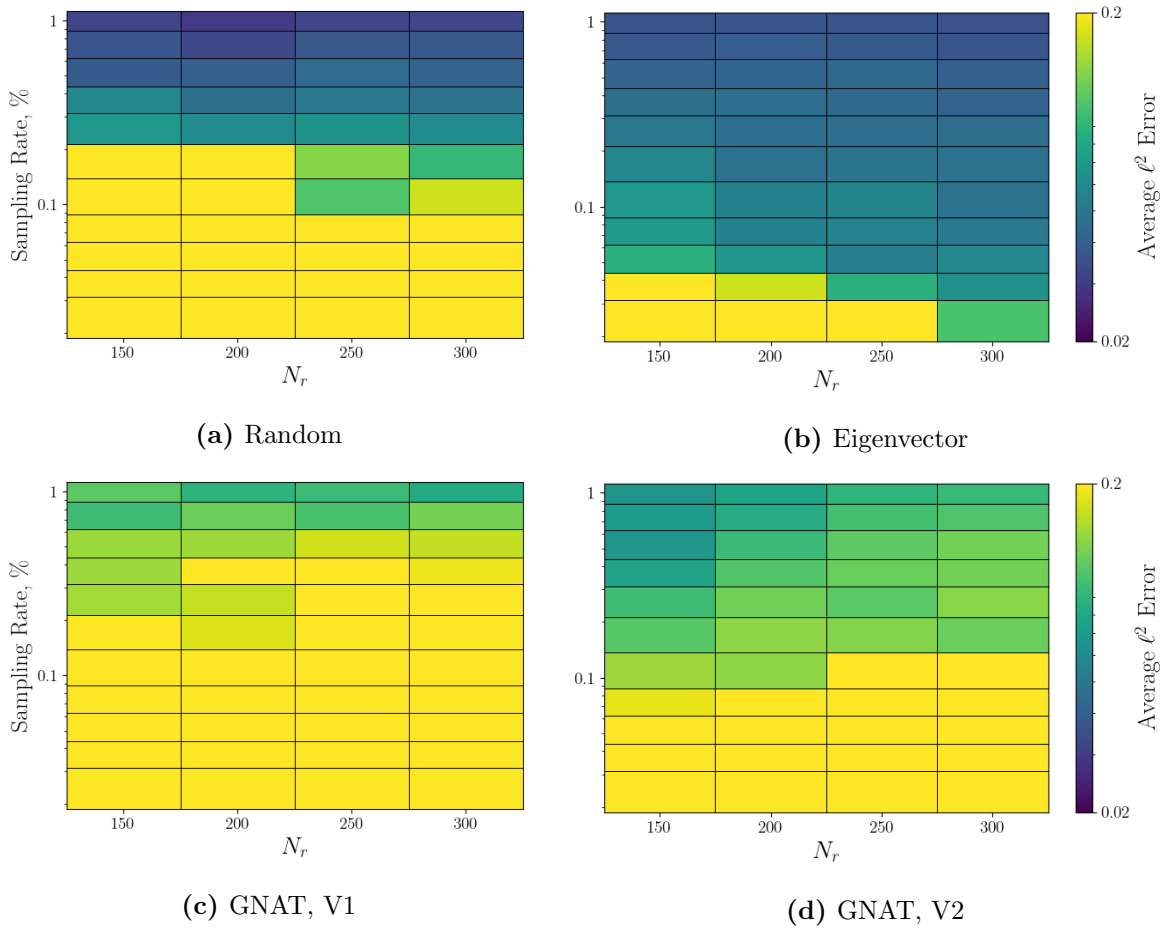
**Figure 6.28:** CVRC HPROM time-average error contours with respect to gappy POD regressor dimension and sampling rate, $\Delta t = 5 \times \Delta t_{\text{FOM}}$, various sampling algorithms.

to scale exponentially with respect to the sampling rate. Although these latter two methods cost over ten FOM iterations to compute for sampling rates above 0.1%, recall that this accounts to 0.2% of the total FOM cost (5,000 iterations).

### 6.2.4 Hyper-reduced PROMs

The same analysis conducted for the MP-LSVT PROMs of the 2D cavity case are repeated for the truncated CVRC. All HPROMs utilize a trial basis dimension of $N_p = 100$, as this was shown above to accurately reconstruct the pointwise unsteady heat release. Again, all gappy POD regressor bases are generated from the POD of the conservative field dataset. The sampling rates, number of partitions, and approximate number of cells per partition are given in Table 6.3.

Time-average $\ell^2$ error contour plots for $\Delta t \in \{5,\ 10\} \times \Delta t_{\text{FOM}}$, for various $N_r$ and $N_s$,




**(a)** Random

**(b)** Eigenvector

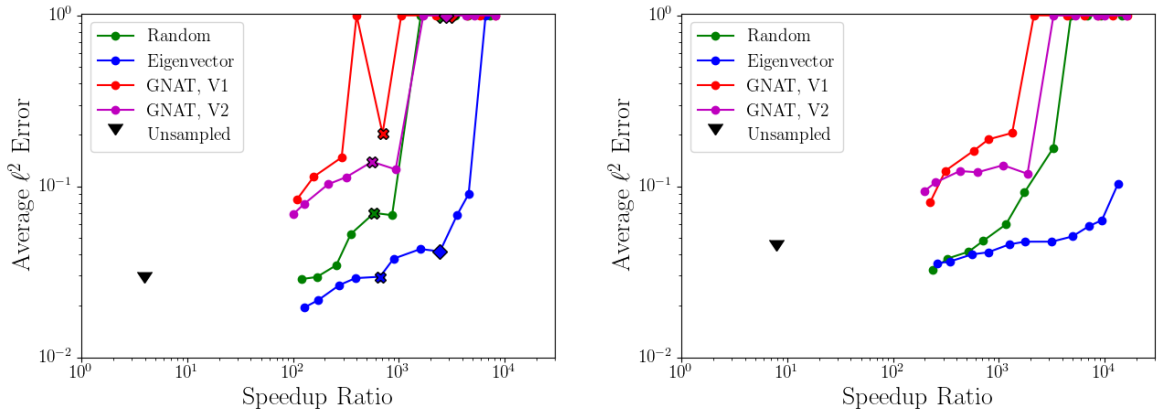
**(c)** GNAT, V1

**(d)** GNAT, V2

**Figure 6.29:** CVRC HPROM time-average error contours with respect to gappy POD regressor dimension and sampling rate, $\Delta t = 10 \times \Delta t_{\text{FOM}}$, various sampling algorithms.

are shown in Figs. 6.28 and 6.29. In this case, all HPROMs remain stable, though those simulations which accrued over 20% error (indicated by bright yellow) can be considered complete failures. For random and eigenvector-based sampling, the natural tendency for accuracy to improve with increased $N_s$ and $N_r$ is observed again. Interestingly, both variants of GNAT sampling seem to perform better at higher sampling rates for lower $N_r$, though these algorithms tend to perform very poorly across most configurations. Again, eigenvector-based sampling performs remarkably well, generating stable HPROMs with less than 5% time-average error for $N_r \geq 250$. Random sampling also appears capably of generating similar levels of accuracy, albeit at much higher sampling rates.

Figure 6.30 displays the cost-accuracy tradeoff incurred by varying the sampling rate, for fixed $N_r = 300$. As observed previously, decreasing the sampling rate tends to degrade simulation accuracy, but enables commensurate computational cost savings. Eigenvector-

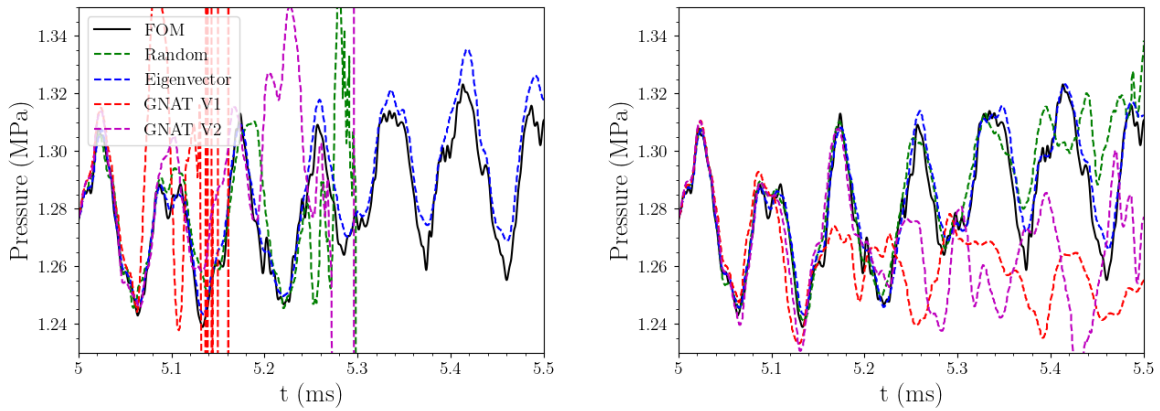



**(a)** $\Delta t = 5 \times \Delta t_{FOM}$ **(b)** $\Delta t = 10 \times \Delta t_{FOM}$

**Figure 6.30:** CVRC HPROM time-average error vs. CPU-time speedup, various $\Delta t$.

based sampling is able to generate HPROMs which achieve three to four orders of magnitude speedup over the FOM with minimal loss of accuracy with respect to the unsampled PROM. At extremely low sampling rates, though, the time-average error grows to over 8-10%. Random sampling is capable of achieving similar speedup results, though this tends to incur unacceptable degredation of accuracy, particularly at very low sampling rates. In general, both GNAT sampling variants are unable to produce accurate HPROMs for any sampling rate investigated, measuring over 10% error in almost all cases.

Figure 6.31 illustrates how the sampling algorithms and sampling rates affect the long-term reconstruction of the unsteady pressure signal at the dump plane corner (marked in Fig. 6.16). The average error for Fig. 6.31a and 6.31b are marked in Fig. 6.30a by an "X" and diamond, respectively. Almost all of the displayed simulations remain reasonably accurate up to $t = 5.1$ ms. As indicated by the above discussion, however, both GNAT variants appear incapable of generating accurate reconstructions beyond this point, either exploding or significantly deviating from the FOM pressure signal. For $N_s = 0.25 \times N$, random sampling is able to maintain an accurate reconstruction for t $< 5.35$ ms, but deteriorates near the end of the simulation. This is not the case for $N_s = 0.075 \times N$, where random sampling explodes shortly after t $= 5.25$ ms. On the other hand, eigenvector-based sampling is able to faithfully reconstruct the pressure signal in both instances, though for $N_s = 0.075 \times N$ the signal begins to deteriorate noticeably near the end of the simulation period.




**(a)** $N_s = 0.075\% \times N$ **(b)** $N_s = 0.25\% \times N$

**Figure 6.31:** CVRC MP-LSVT HPROM probe measurements, $N_p = 100$, $N_r = 300$, $\Delta t = 5 \times \Delta t_{\text{FOM}}$.

These observations are emphasized further by field plots drawn from the end of the simulation period, t = 5.5 ms, for $N_s = 0.25\%$. Figure 6.32 shows pressure slices, where it is abundantly clear that the HPROMs produced by random sampling and the GNAT variants have devolved into high-amplitude oscillations. Although the GNAT variants appear to avoid the extremely high-frequency oscillations observed in the random sampling cases, the point is ultimately irrelevant due to overall deviation from the solution. Eigenvector-based sampling computes a remarkably similar pressure field, though some of the very high-frequency content is absent in the combustion chamber (though this may be due entirely to an under-resolved trial space).

Similar results are observed in the temperature field, shown in Fig. 6.33. The result produced by eigenvector-based sampling captures the mixing and advection of the reactant shear layer, though there is again noticeable smearing and spurious oscillations near the flame front. Much like with the pressure field, random sampling produces an HPROM which devolves into high-frequency oscillations in the mixing layer, while both GNAT variants produce large oscillations throughout the combustion chamber.

### 6.2.5 Effects of Sampling Criteria

Three approaches for evaluating the greedy sampling algorithm metric and selecting those degrees of freedom to be appended to the sample set were described in Section 4.5.1: ag-

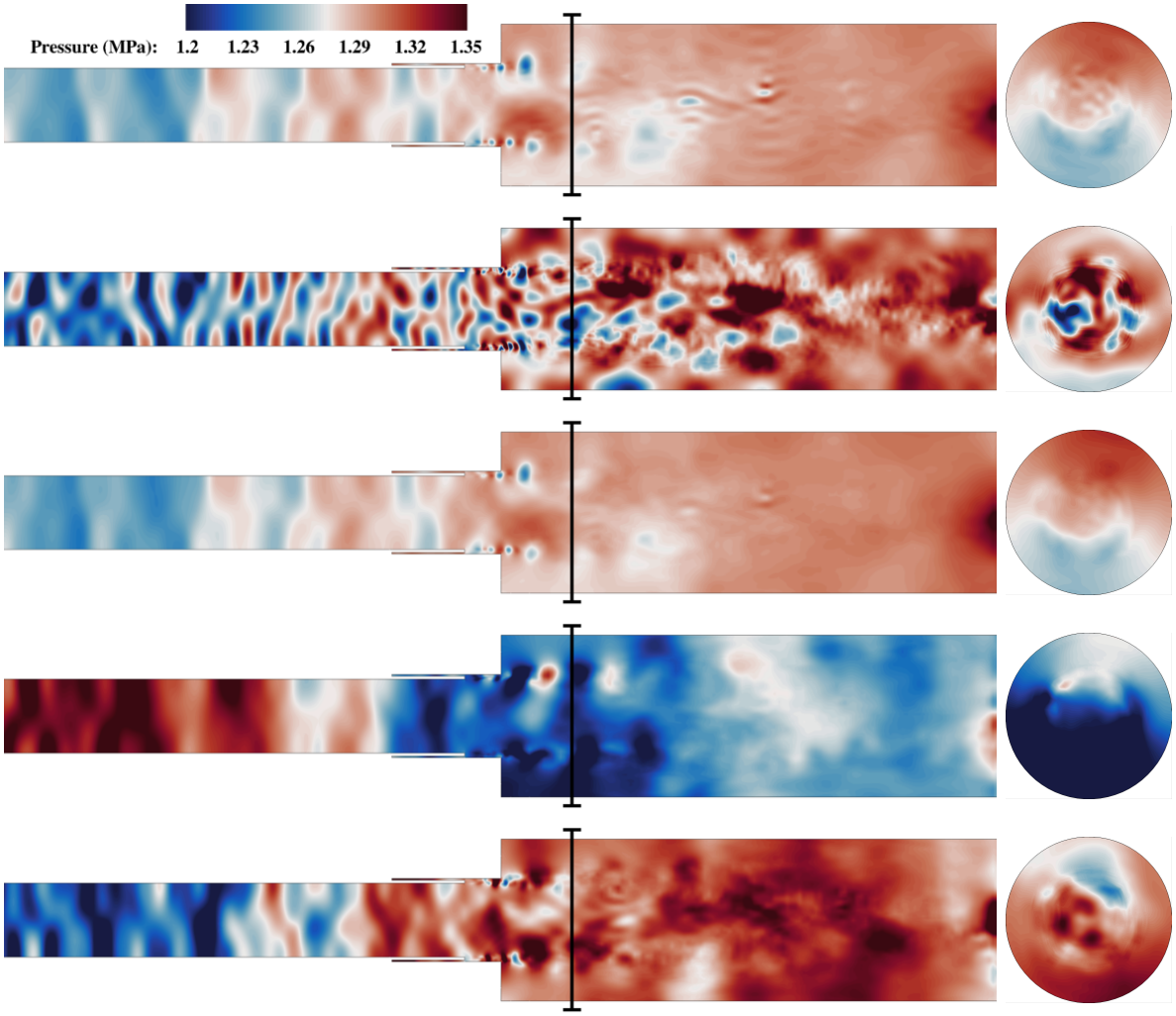



**Figure 6.32:** CVRC HPROM pressure slices, t = 5.5 ms, $N_s = 0.25\%$, $N_r = 300$, $\Delta t = 5 \times \Delta t_{\text{FOM}}$. From top to bottom: FOM, random, eigenvector, GNAT V1, and GNAT V2 sampling.

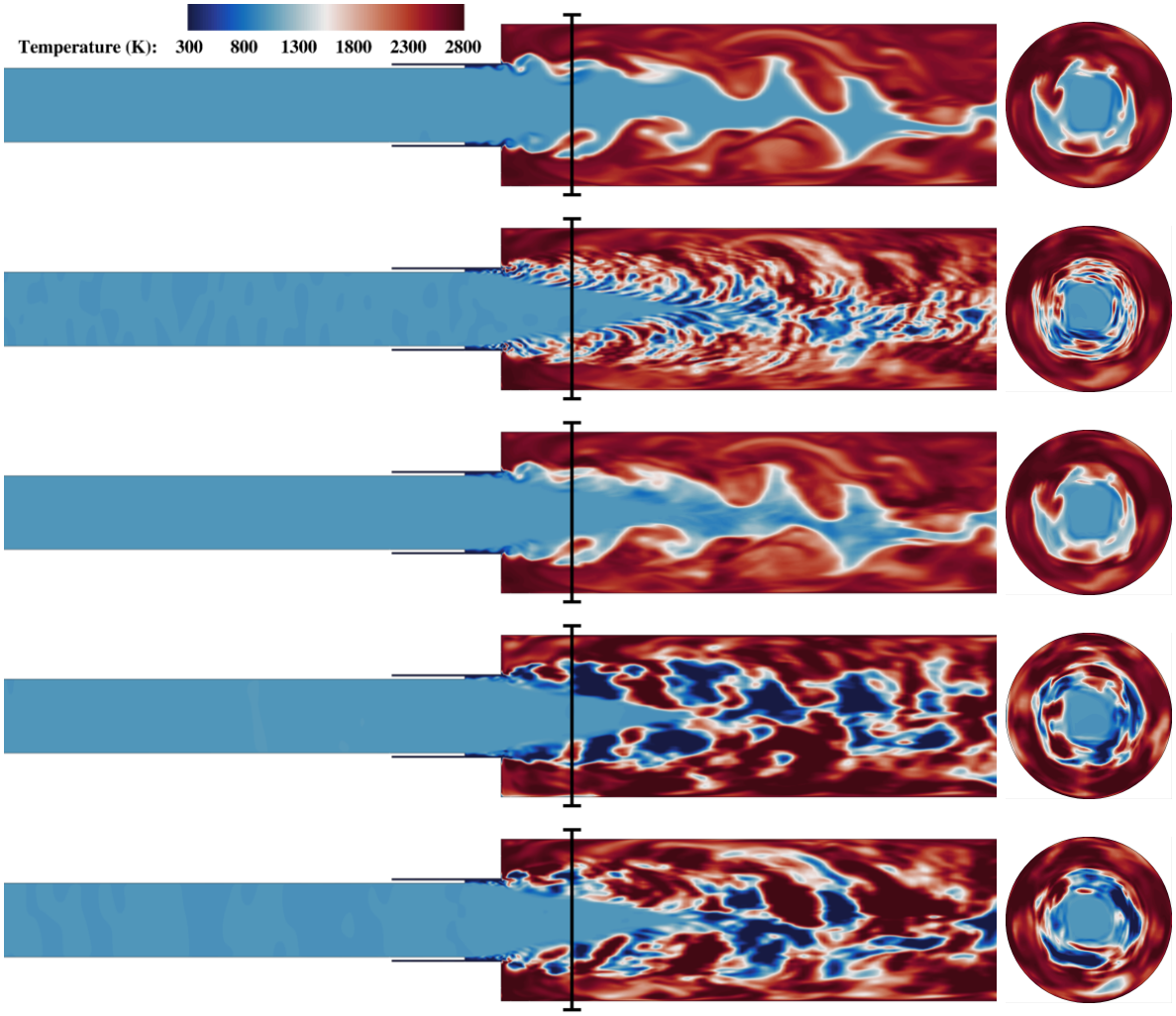



**Figure 6.33:** CVRC HPROM temperature slices, t = 5.5 ms, $N_s = 0.25\%$, $N_r = 300$, $\Delta t = 5 \times \Delta t_{\text{FOM}}$. From top to bottom: FOM, random, eigenvector, GNAT V1, and GNAT V2 sampling.

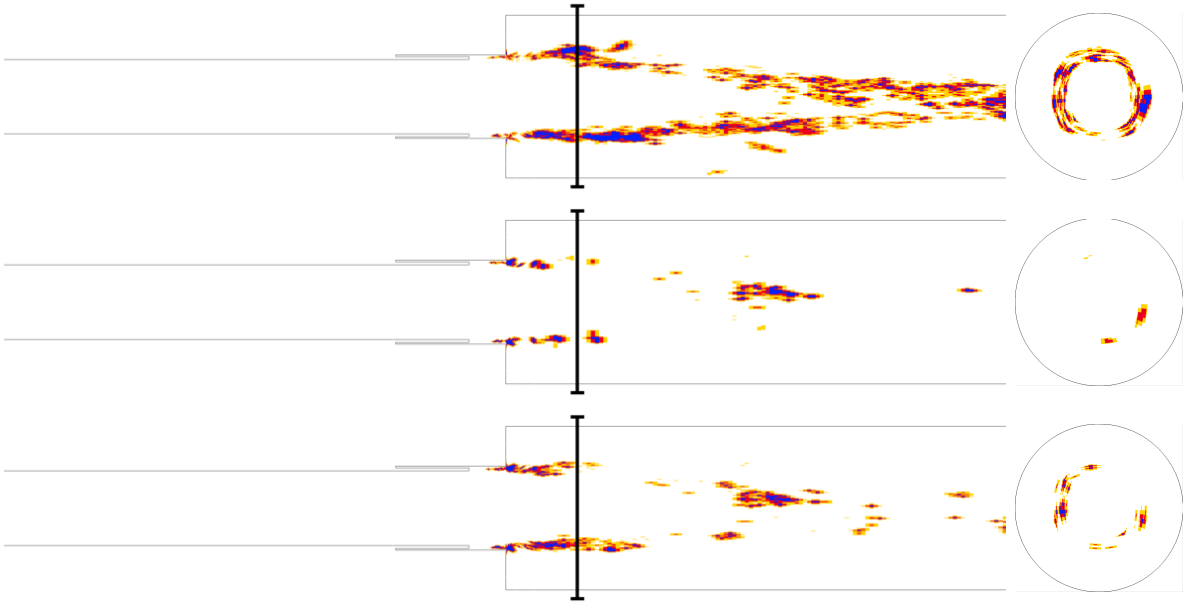
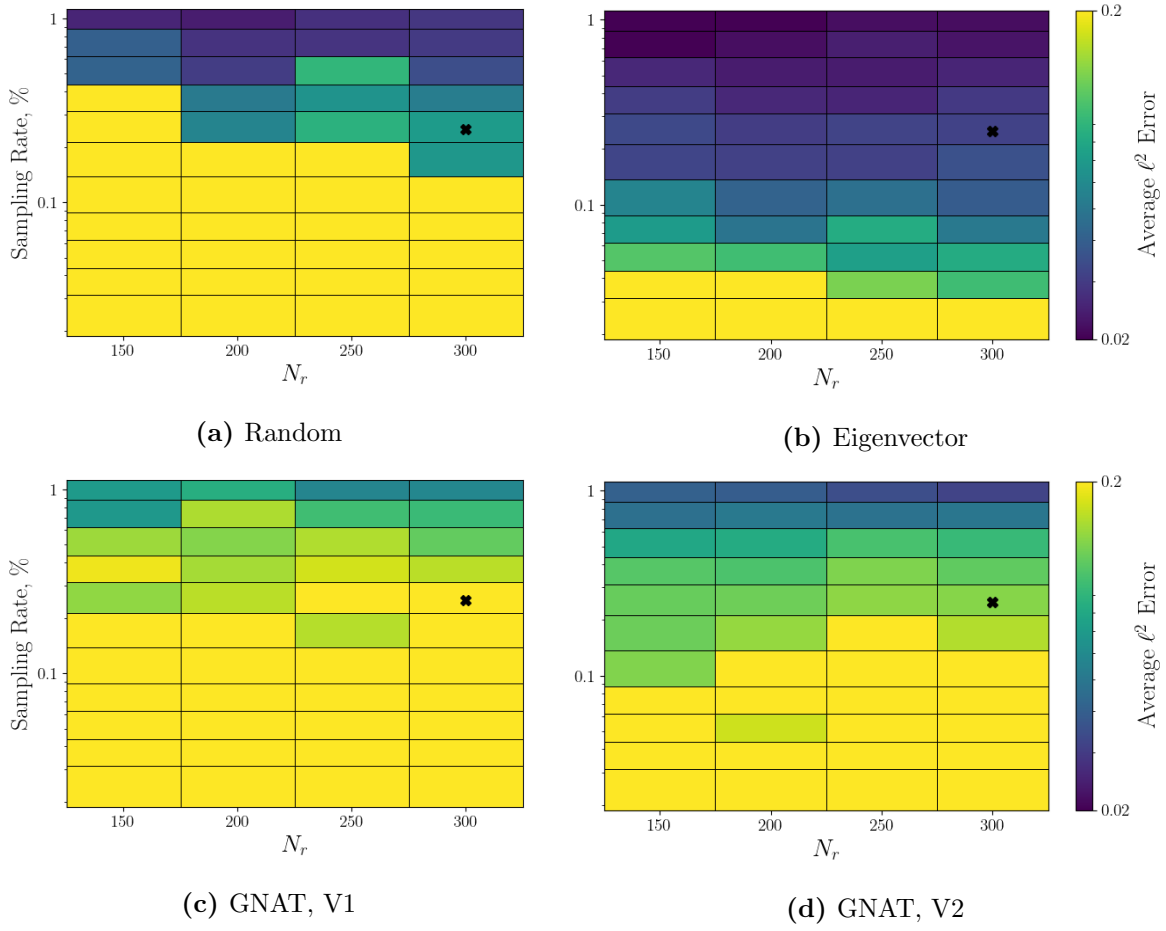


**Figure 6.34:** Example sample meshes for $N_s = 0.25\% \times N$, $N_r = 300$ for various sampling algorithms, using the post-sampling approach. From top to bottom: random, eigenvector-based, GNAT V1, and GNAT V2.

nostic, post-sampling, and comprehensive sampling. The comprehensive method has been claimed to provide a more complete description of the sampling metric for coupled fluid flow systems, and is used for all hyper-reduction results presented in this thesis. In this section, the validity of this claim is investigated. The performance of HPROMs generated by the post-sampling and comprehensive techniques are thus compared. Random sampling via post-sampling is omitted, as it is not a greedy sampling algorithm. The same sample mesh sizes as those listed in Table 6.3 are again used for all post-sampling meshes.

Figure 6.34 provide indicative sample mesh slices for post-sampling meshes, reflecting the same mesh size and regression basis dimension $N_r$ as those displayed in Fig. 6.24. The two approaches generate similar meshes, and close inspection is required to notice differences. One might notice slightly tighter clustering of sampled points for eigenvector-based sampling, while the GNAT sample meshes are slightly more diffuse.

The resulting simulation accuracy with respect to the sample rate $N_s$ and the regression basis dimension $N_r$ also remains largely the same, as seen in Fig. 6.35. Eigenvector-based sampling performs quite well at low sampling rates, random sampling performs well at high sampling rates, and both GNAT algorithms tend to perform poorly at any




**(a)** Random

**(b)** Eigenvector

**(c)** GNAT, V1

**(d)** GNAT, V2

**Figure 6.35:** CVRC HPROM time-average error contours for sample meshes constructed via the post-sampling approach, with respect to gappy POD regressor dimension and sampling rate, $\Delta t = 5 \times \Delta t_{FOM}$, various sampling algorithms.

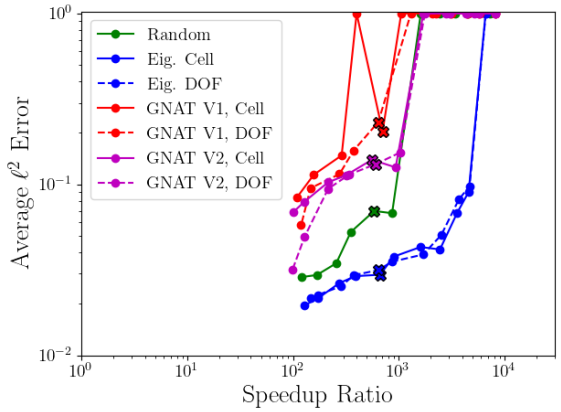



**Figure 6.36:** CVRC HPROM time-average error vs. CPU-time speedup, $\Delta t = 5 \times \Delta t_{FOM}$.

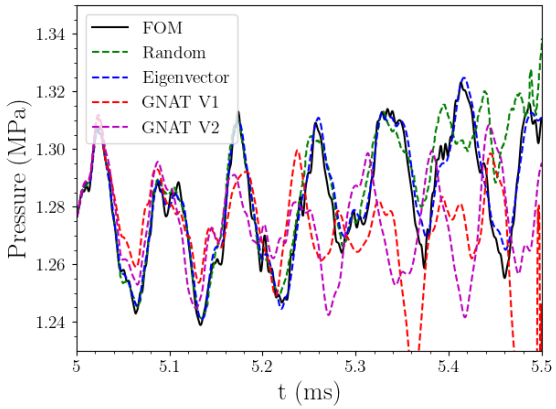



**Figure 6.37:** CVRC HPROM probe sampled via post-sampling, $N_p = 100$, $N_s = 0.25\% \times N$, $N_r = 300$, $\Delta t = 5 \times \Delta t_{FOM}$.

sampling rate. The general trend of improved accuracy with increased $N_r$ is observed again, most noticeably with eigenvector-based sampling.

Examining the cost-accuracy tradeoff in Fig. 6.36 emphasizes that the post-sampling approach has little, if any, influence on the accuracy and computational cost savings of the HPROMs. Except at very high sampling rates, where post-sampling appears to *improve* the accuracy of the GNAT algorithms, the lines representing post-sampling results lie nearly on top of those representing comprehensive sampling. Pressure probe results in Fig. 6.37 display nearly identical performance to those shown in Fig. 6.31b.

The results shown above indicate that the choice of sampling criterion plays a minimal role in enabling fast and accurate HPROMs, relative to the extreme sensitivity to the sampling algorithm. This logically follows from the idea that certain variables make outsized contributions to the greedy sampling metric, e.g. the temperature vs. pressure field, and so both approaches result in similar sample meshes despite non-dimensionalization of the data prior to generation of the regression basis $\mathbf{\Psi}$. In either case, those mesh elements which incur the greatest error are still selected.

### 6.2.6 Effects of Dual Basis Gappy POD

All results shown thus far have used a single gappy POD regression basis, $\mathbf{\Psi} \in \mathbb{R}^{N \times N_r}$. Following the justification outlined in Section 4.4.2, the regression basis has been chosen

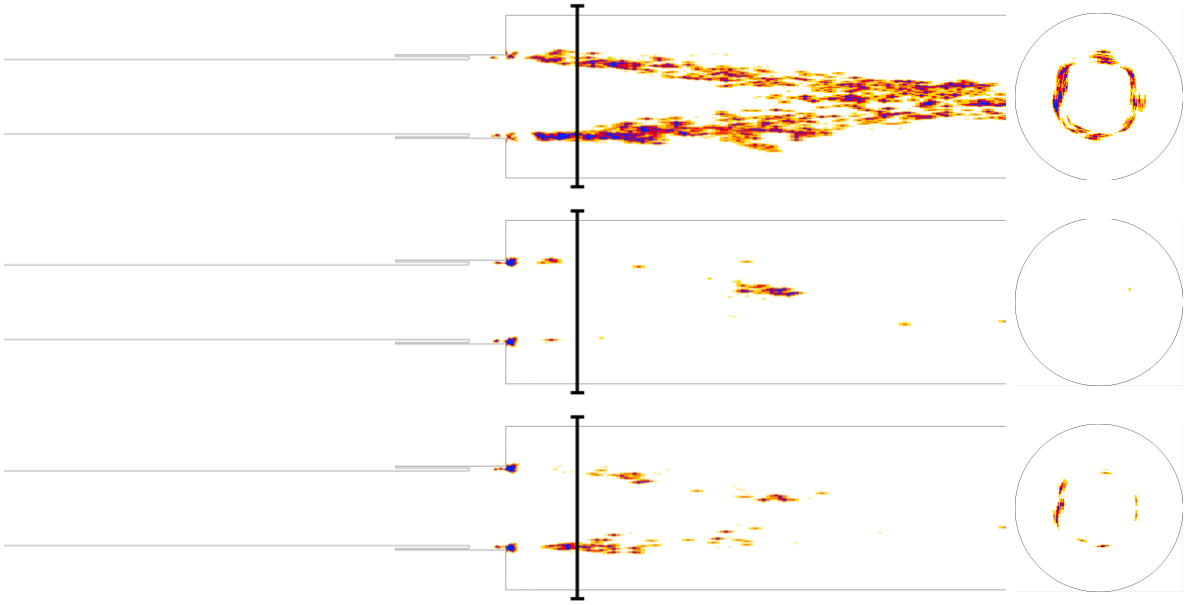


**Figure 6.38:** Example sample meshes for $N_s = 0.25\% \times N$, $N_r = 300$ for various sampling algorithms, using the post-sampling approach. From top to bottom: random, eigenvector-based, GNAT V1, and GNAT V2.

such that $\boldsymbol{\Psi} = \widetilde{\mathcal{U}}_c$; that is, the regression basis is computed from snapshots of the conservative fields. This approach has worked well, but alternative approaches have suggested representing components of the fully-discrete residual with separate gappy POD approximations (as in brief notes in [157] and [70]). In Section 4.4.4, a dual-basis formulation was proposed by which the fully-discrete residual is separated into the time integrator $\dot{\mathbf{q}}_c$ and all spatial discretization terms $\mathbf{f}(\mathbf{q}_c, \mathrm{t})$, with corresponding gappy POD regression bases $\boldsymbol{\Psi}_c \in \mathbb{R}^{N \times N_c}$ and $\boldsymbol{\Psi}_f \in \mathbb{R}^{N \times N_f}$ respectively. For the sake of completeness, HPROMs constructed via this approach are investigated below. Only one set of basis sizes is investigated: $N_r = N_c = 300$ and $N_f = 800$. Unlike in the above Section 6.2.5

The sample meshes generated by the dual-basis sampling procedure are virtually identical to those generated by the single-basis approach, as seen in Fig. 6.38. Close inspection reveals minor differences, but follow the same trends as those seen in Fig. 6.24. Despite the marked similarities of the sample meshes, the performance of the online HPROMs are strikingly different, as shown in Fig. 6.39. For a given sample rate $N_s$, the dual-basis formulation halves the computational cost savings and nearly doubles the error (or even becomes unstable). Although the relative discrepancies between the sampling algorithms persist (eigenvector-based sampling performing the best, followed by random

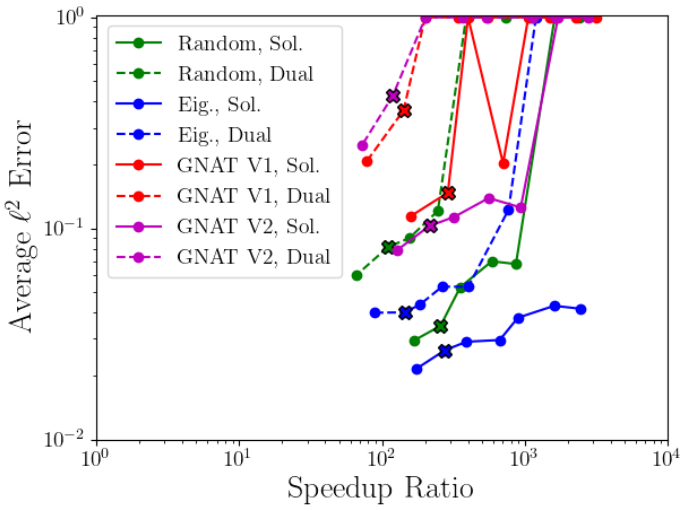



**Figure 6.39:** CVRC HPROM time-average error vs. CPU-time speedup, $\Delta t = 5 \times \Delta t_{\text{FOM}}$.

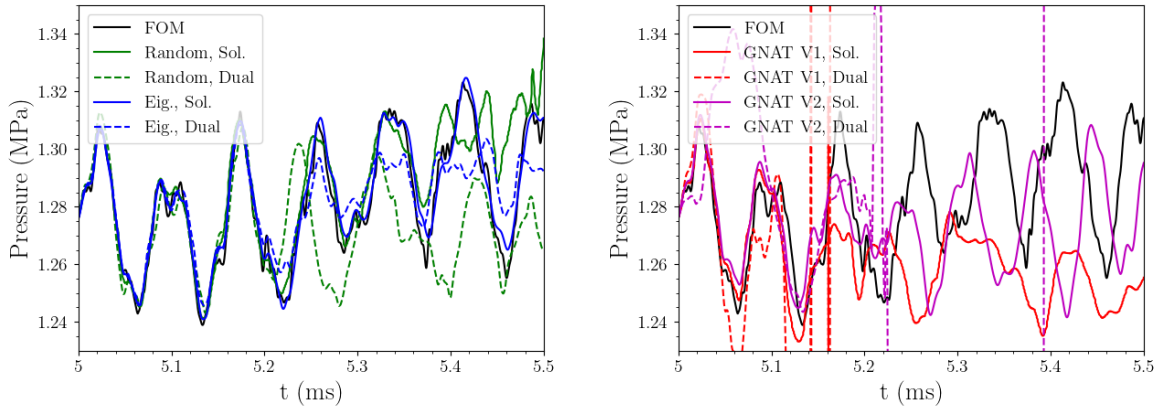



**Figure 6.40:** CVRC HPROM probe sampled via post-sampling, $N_p = 100$, $N_s = 0.25\% \times N$, $N_r = 300$, $\Delta t = 5 \times \Delta t_{\text{FOM}}$.

sampling, and GNAT tending to fail outright).

Pressure monitor results in Fig. 6.40 only emphasize the comparatively poor performance of the dual-basis formulation. The accuracy of the random and eigenvector-based HPROMs deteriorates much more rapidly (around t = 5.2 ms), and even the GNAT HPROMs are wholly unstable (opposed to greatly inaccurate). These results indicate that the dual-basis formulation does not benefit the performance of the HPROMs despite a supposed improvement in the accuracy of the constituent gappy POD regressions. This is, in some sense, a comfort, inasmuch as the simplest solution (a single gappy POD basis) appears to be the best. Of course, increased granularity of the fully-discrete residual (e.g. generating separate approximations for the inviscid and viscous fluxes) might improve on

this, but this introduces an exceptional level of complexity in the Gauss–Newton normal equations and was not deemed a worthwhile investigation.

## 6.3 Purdue Nine-element Combustor

In this section, PROM performance in modeling a multi-element rocket combustor experiencing high-frequency combustion instability is examined. This model is based on experiments conducted with the nine-element Transverse Instability Combustor Purdue University by Orth *et al.* [8]. In the following section, only a simplified description of the experiment as it pertains to the numerical model is described; the reader is encouraged to find greater detail on the experiment's design in [8]. As with the CVRC, this test article was designed in tandem with CFD researchers in an effort to establish a rocket combustor benchmark for the development of advanced reacting flow LES solvers. In this case, the number of parallel propellant injectors is increased to nine, introducing additional complexity due to interactions between injectors in the presence of strong transverse combustion instabilities. Indeed, work by Harvazinski *et al.* [184] was able to reasonably replicate the experimental self-excited combustion instability via LES with a reduced finite-rate chemistry mechanism. However, as will be demonstrated shortly, the computational cost of such simulations is exceptionally high and precludes any usefulness in the engineering design process. As such, this case provides an excellent target for reduced-order modeling.

### 6.3.1 Full-order Model

The computational model presented here derives from the work by Harvazinski *et al.* [184], using the same computational mesh, but marginally different boundary conditions and a different reaction mechanism. The spatial domain does not include the oxidizer preburner, oxidizer manifold, or fuel manifold elements of the experimental test article. A cutaway of the spatial domain is shown in Fig. 6.41, and an isometric view of the combustor is shown in Fig. 6.42. The combustor is composed of a linear array of nine coaxial propellant

injection elements, each spaced centerline-to-centerline by 26.67 mm apart. The oxidizer posts extend 96.01 mm upstream of the dump plane, and has a diameter of 11.25 mm upstream of the fuel injection port. Each oxidizer post supplies 96.5% gaseous oxygen and 3.5% water vapor (by weight) at 636 K. The entire oxidizer injection assembly supplies oxidizer at a rate of 0.7575 kg/s. Fuel is supplied by radial injection approximately 42 mm upstream of the dump plane, after which flow is turned 90° to exit the annular fuel port and mix with the oxidizer approximately 11.34 mm upstream of the dump plane. The inner diameter of each fuel annulus measures 13 mm, and the outer diameter (also the diameter of the propellant injection port into the combustion chamber) measures 15.75 mm. Each fuel injector supplies 100% gaseous methane at 287.6 K at a rate of 0.02369 kg/s. The main combustion chamber (upstream of the nozzle) measures 118.5 mm long ($x$-direction), 240 mm wide ($y$-direction), and 30.5 mm deep ($z$-direction). The nozzle is 81.5 mm long, and terminates in an exit port 142.1 mm wide and 10.8 mm deep.

Mass flow rate boundary conditions are applied to the oxidizer and fuel inlets, though a single rate is enforced for the entire oxidizer inlet area, while individual rates are enforced for each fuel inlet. Adiabatic, no-slip wall conditions are enforced at all walls. A fixed pressure outlet is enforced at the nozzle exit, specifying an exit pressure of 101,325 Pa to model venting to atmosphere.

Reactions are modeled by a 12-species (plus inert nitrogen), 38-reaction finite-rate mechanism referred to as FFCMy-12, which is developed by Xu and Wang [42, 185] as a reduced mechanism of the larger FFCM-1 mechanism [186] tailored for combustion of methane and oxygen in rocket combustors. As mentioned in Section 2.2.1, this mechanism includes third-body low-pressure corrections of the Hinshelwood [100] and Troe [101] forms. This mechanism has been utilized previously in the simulation of traditional liquid rocket engines [130, 187] as well as rotating detonation engines [188, 189]. The fluid is treated as a thermally-perfect gas, and thermodynamic and transport properties are computed by the empirical fit models described in Section 2.1.1. Subgrid dissipation is modeled via the Nicoud $\sigma$-model described in Section 2.1.2.

The computational mesh is composed of 14,387,292 hexahedral cells, resulting in a

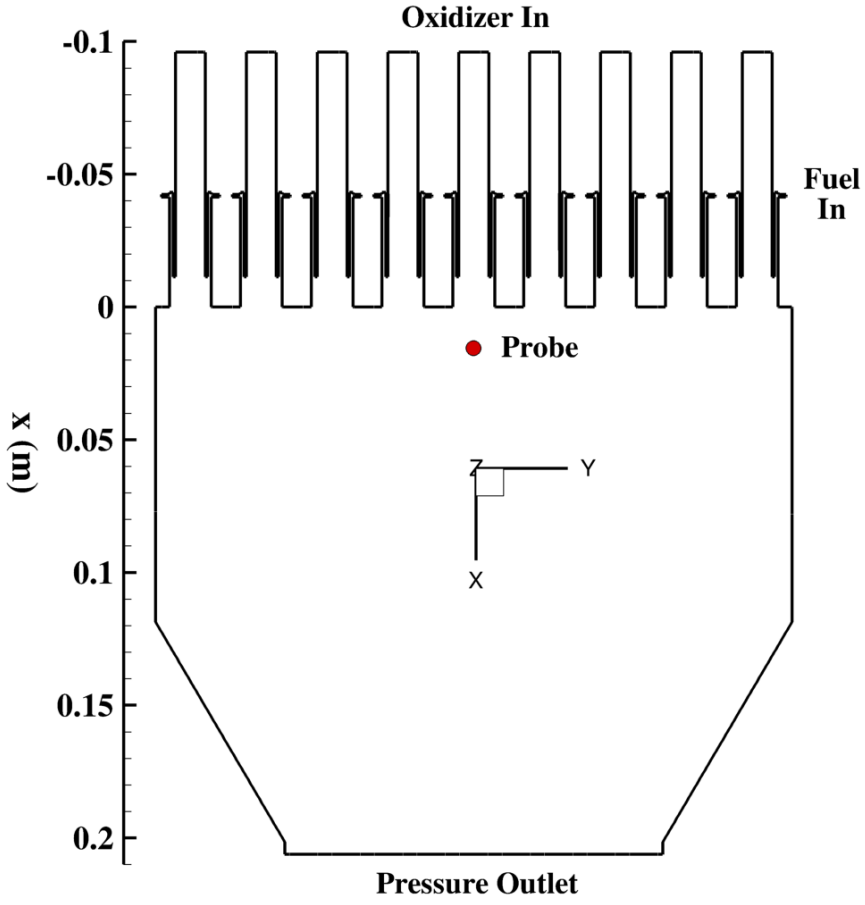



**Figure 6.41:** Nine-element combustor geometry, $x - y$ cutaway.

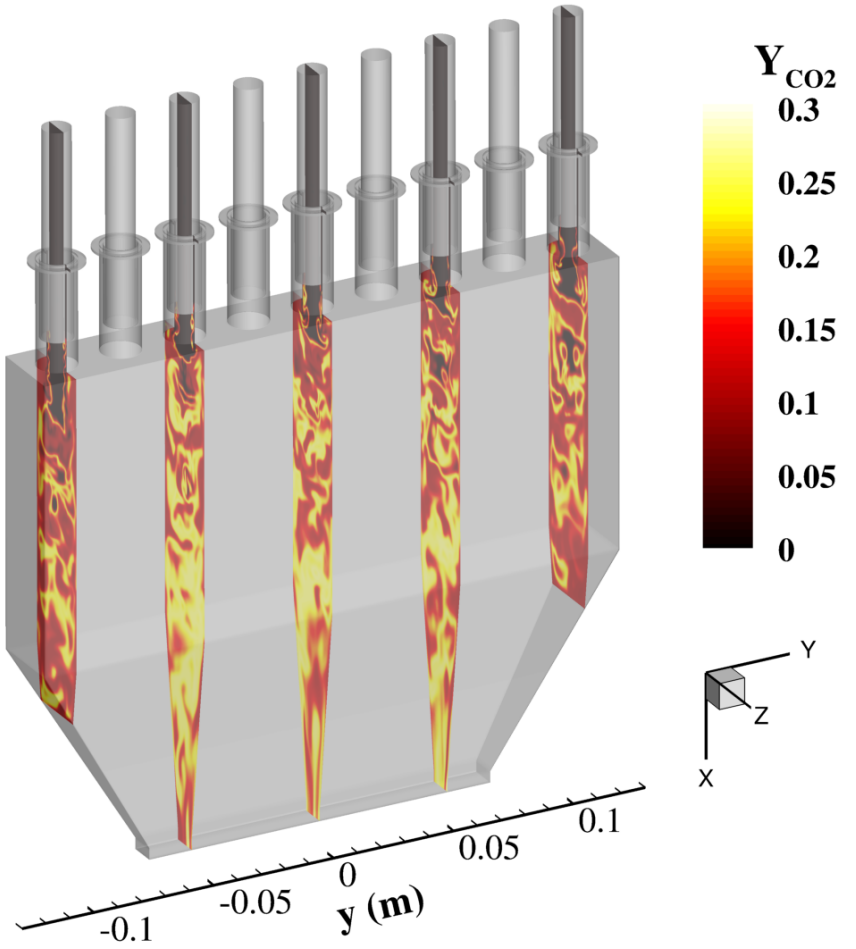



**Figure 6.42:** Nine-element combustor geometry, isometric view, with $Y_{CO_2}$ $y - z$ slices at t = 21.6 ms.

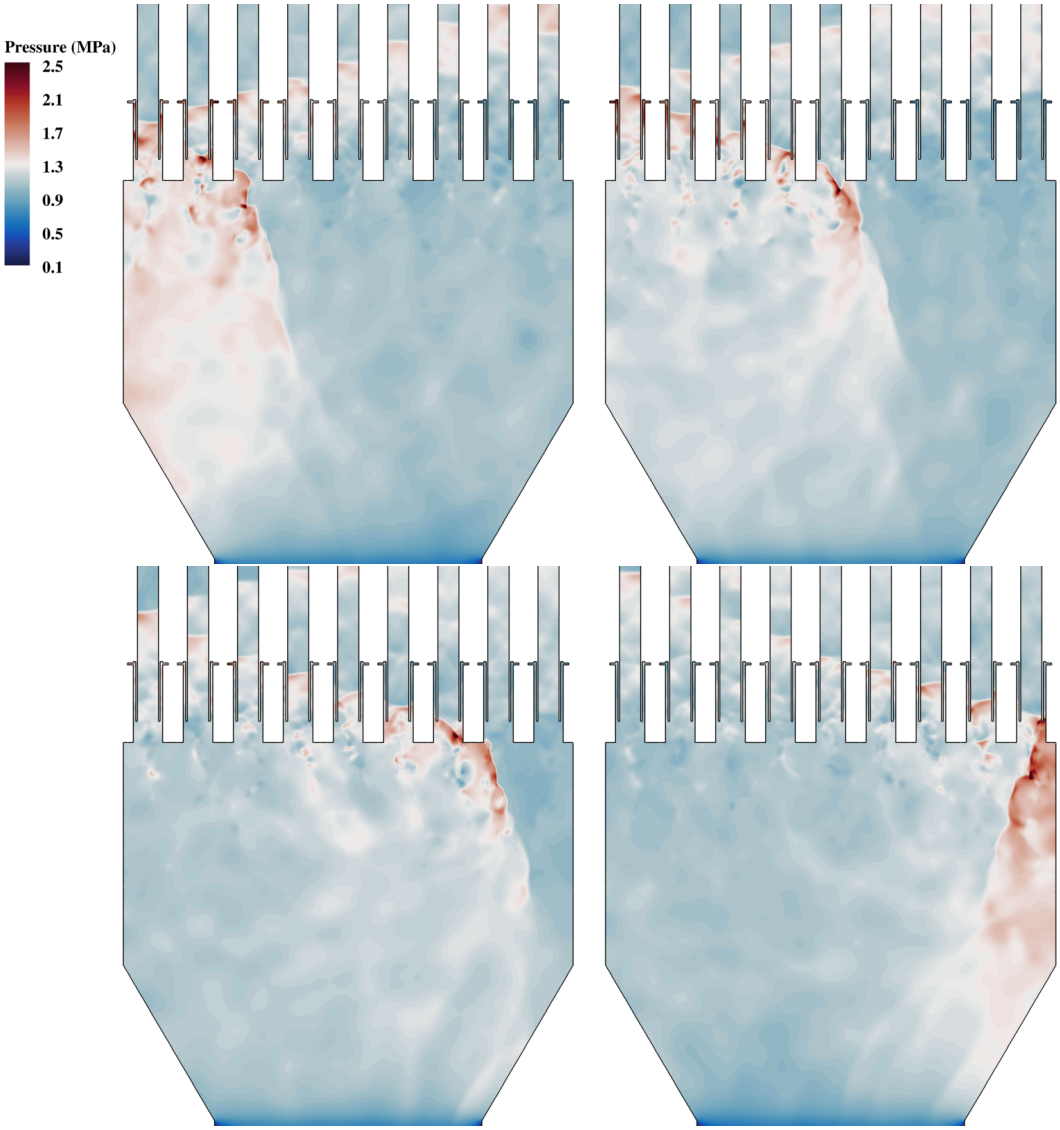



**Figure 6.43:** FOM pressure slices for $x-y$ plane slice at t = 21.65 (top left), 21.7 (top right), 21.75 (bottom left) and 21.8 (bottom right) ms.

total full-order dimension of $N = 244,583,964$. To the best of the author's knowledge, this represents the largest case examined for PROMs of unsteady fluid flows, and also accounts for a significant increase in modeling complexity relative to comparable studies of aerodynamics problems.

The FOM is initialized as follows. The entire domain is initialized with zero velocity and 1.138 MPa pressure. The oxidizer posts and propellant injection ports (excluding the fuel annuli) are initialized with 96.5% oxygen and 3.5% water vapor at 636 K. The fuel annuli are initialized with 100% methane at 287.6 K. The combustion chamber is initialized with hot products (44% water vapor and 56% carbon dioxide) at 2,000 K. The

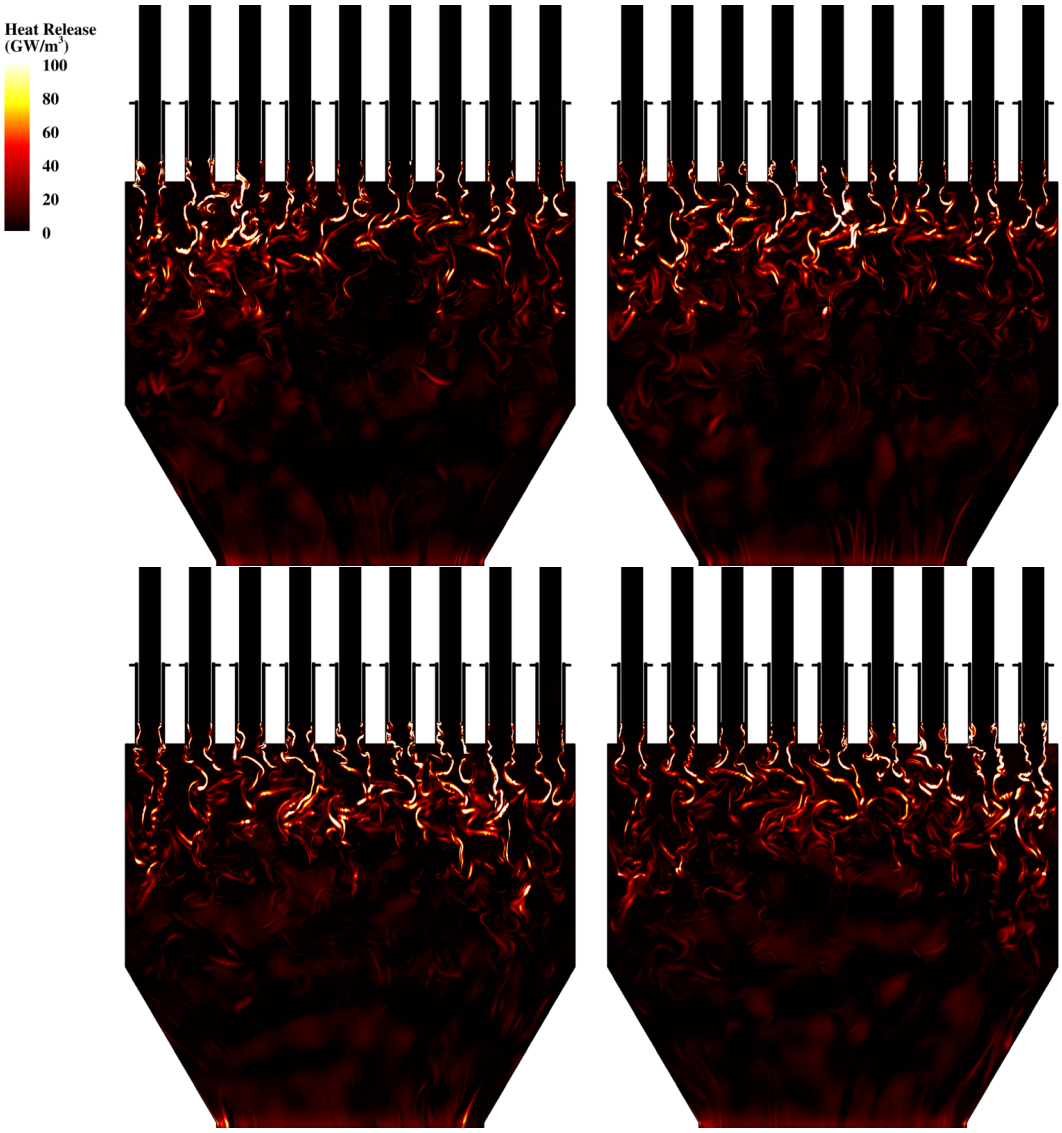



**Figure 6.44:** FOM heat release slices for $x - y$ plane slice at t = 21.65 (top left), 21.7 (top right), 21.75 (bottom left) and 21.8 (bottom right) ms.

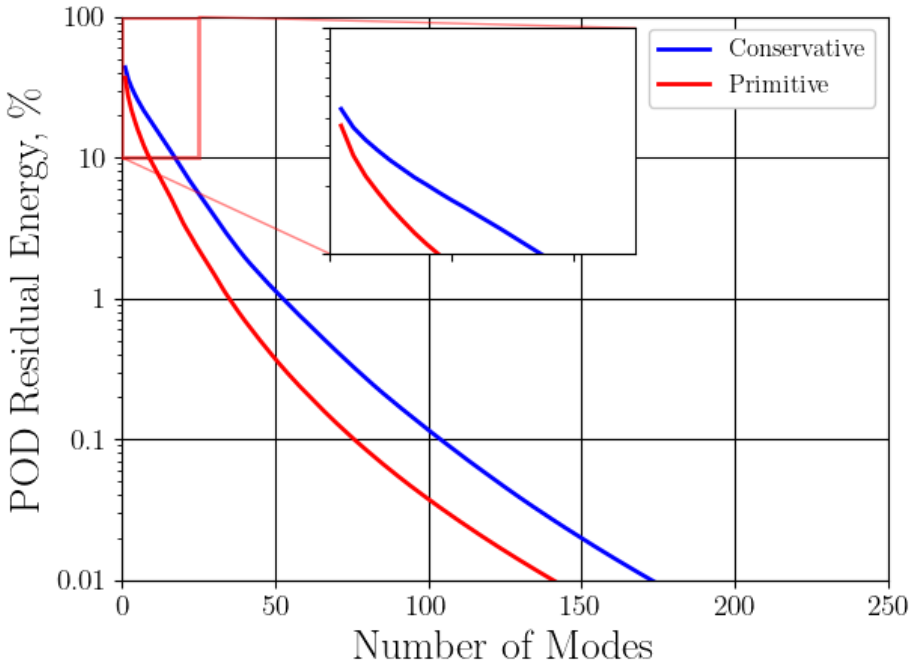



**Figure 6.45:** POD residual energy decay for nine-element combustor conservative and primitive state datasets.

physical time step size is $\Delta t_{FOM} = 0.1\mu s$, and the FOM is first run for 216,000 steps (21.6 ms) to allow the transverse instability to initiate. The instability is visualized in Fig. 6.43, where a high-pressure wave traverses the width of the combustion chamber (reflecting from the walls) over the course of approximately 0.195 ms. The pressure wave is accompanied by heightened local heat release, which can be seen in Fig. 6.44.

Starting at t = 21.6 ms, snapshots of the primitive and conservative states are

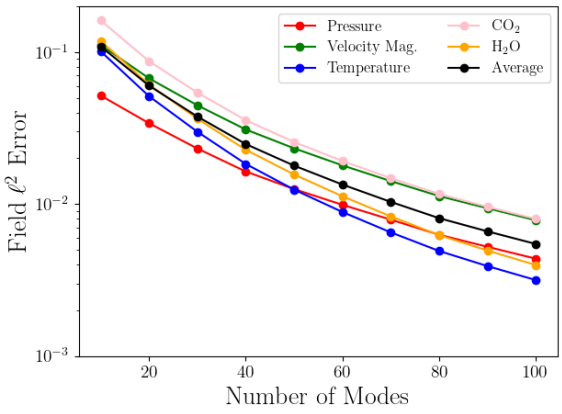



**Figure 6.46:** Primitive variables time-average projection error.

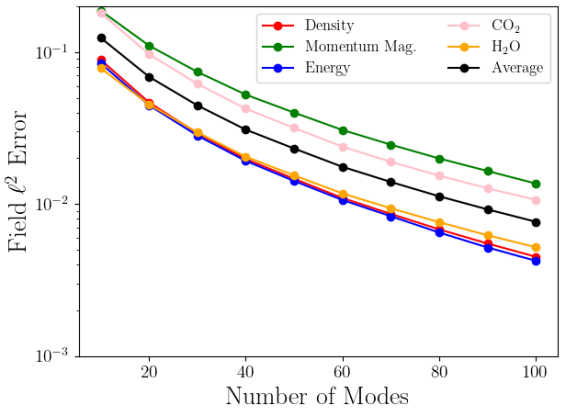



**Figure 6.47:** Conservative variables time-average projection error.

collected at every time step until t = 21.8, resulting in 2,001 snapshots (including $\mathbf{q}(\mathrm{t} = 21.5\mathrm{ms})$). This window captures one complete traversal of the high-pressure wave along the width of the combustor. The POD residual energy decay is shown in Fig. 6.45. Achieving 1%, 0.1%, and 0.01% of the conservative state POD residual energy requires 53, 104, and 173 modes respectively. For the primitive state, this requires 35, 76, and 141 modes respectively. As with the truncated CVRC, this case exhibits an extremely slow POD residual energy decay characteristic of such a highly-nonlinear convection-dominated flow. The projection error of the primitive and conservative state datasets are shown in Figs. 6.46 and 6.47 respectively. Interestingly, again the transverse and depth-wise velocity represent the largest sources of projection error. However, three of the most relevant transported scalars, carbon monoxide, carbon dioxide, and water vapor appear to induce less error than the fuel mixture fraction and progress variable did in the CVRC case.

### 6.3.2 Unsampled PROMs

Unsampled PROM accuracy is now investigated. Again, only MP-LSVT PROMs are examined, as Galerkin and LSPG PROMs were categorically unstable. This result is hardly surprising due to the extreme stiffness arising from the more detailed reaction mechanism. Results are evaluated for three time step values, $\Delta\mathrm{t} \in \{2.5,\ 5,\ 10\} \times \Delta\mathrm{t}_{\mathrm{FOM}}$, and primitive variable trial basis dimensions $N_p \in \{25,\ 50,\ 75,\ 100\}$.

For results presented in this and the following section, average error is not computed across all primitive field variables, as this quantity appears exceptionally high even for large $N_p$. This is entirely due to deviations in chemical intermediates, such as monatomic species (H, O) and radicals ($CH_3$, OH, $HO_2$), which are extremely sensitive to small changes in reactant and other intermediate species fields. Although one may argue that the accuracy of reconstructing the reactant and final product fields is of primary concern, incorrect estimates of reaction intermediates undoubtedly degrades overall performance (particularly in predictions of unsteady heat release rates). Ignoring intermediate species, averages are computed over the primary flow field variables (pressure, velocity, and tem-

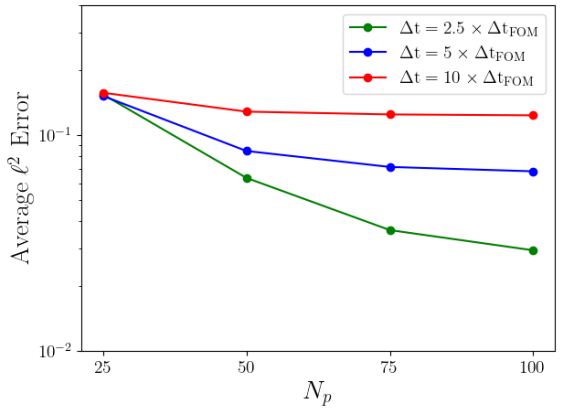



**Figure 6.48:** Nine-element combustor unsampled PROM time-average error, various $\Delta t$.

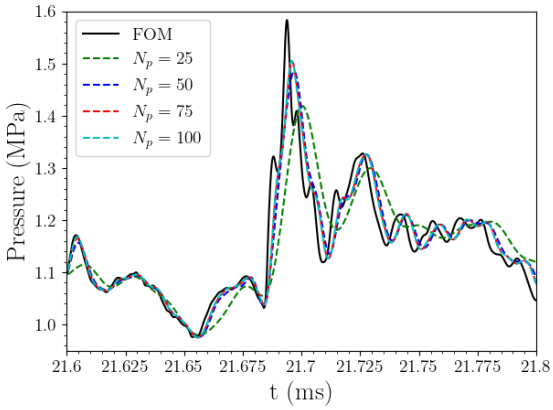
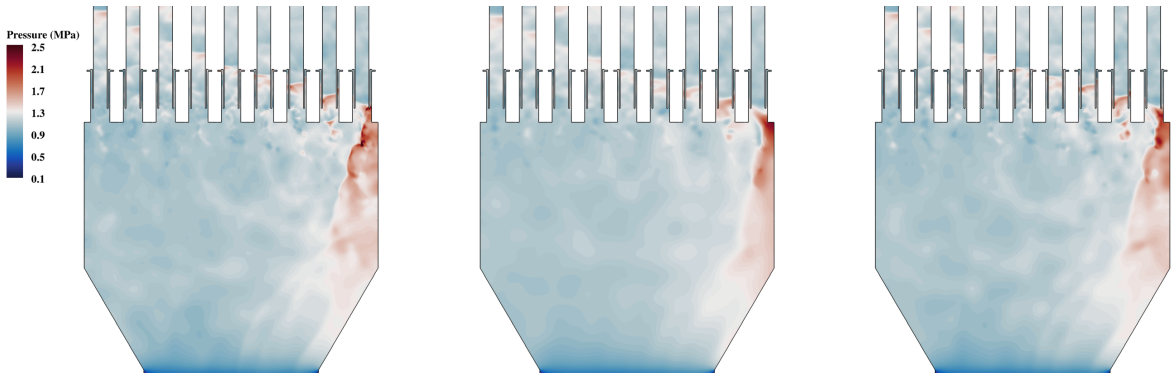



**Figure 6.49:** Nine-element combustor unsampled PROM pressure probe measurements, $\Delta t = 5 \times \Delta t_{FOM}$, various $N_p$.

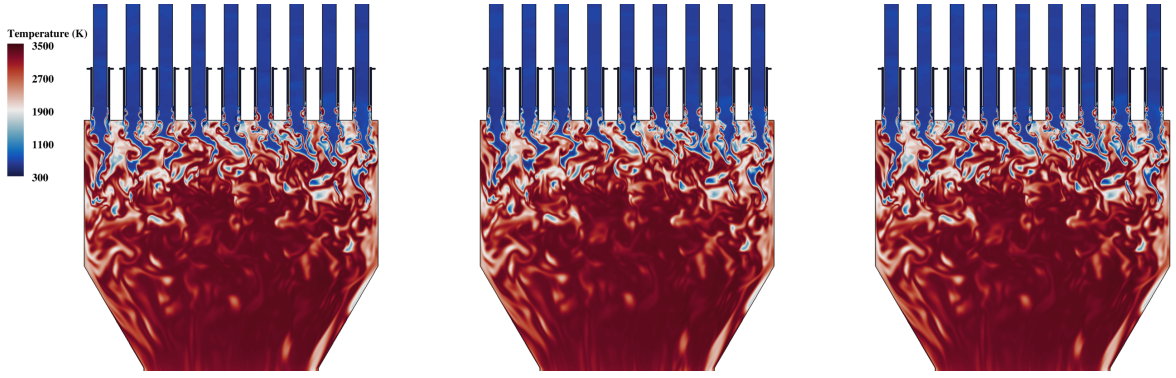
perature) and primary chemical species ($CH_4$, $O_2$, $H_2O$, CO, $CO_2$). Figure 6.48 displays time-average error results for the three time steps and four trial basis dimensions examined. Similar to observations for the CVRC, increasing the time step beyond $5 \times \Delta t_{FOM}$ results in significant solution degradation which does not improve much by enriching the trial space. In all cases, the error appears to level off after $N_p = 75$; this dimension will be used for all hyper-reduction results later in this chapter.

Examining several unsteady flow fields in Fig. 6.50 provides visual insight into the unsteady evolution of the tightly-coupled acoustic, thermodynamic, and chemical interactions modeled by the PROM. Snapshots are taken from the end of the simulation period. Following similar trends observed for the CVRC, Fig. 6.50a shows that enriching the trial basis tends to improve the reconstruction of sharp gradients, exemplified here by the high-pressure combustion instability. Such a trend is less immediately apparent for the temperature fields in Fig. 6.50b and the carbon monoxide fields in Fig. 6.50c. Even at low $N_p$, the PROM is capable of reconstructing these unsteady states quite well. This can be reasonably attributed to the fact that transport of these fields is characterized a much slower time scale than that of the transverse pressure wave. As bulk advection has less of an effect on the temperature and transported scalar fields during the time window examined, one expects that the linear trial basis is better suited for modeling those fields.

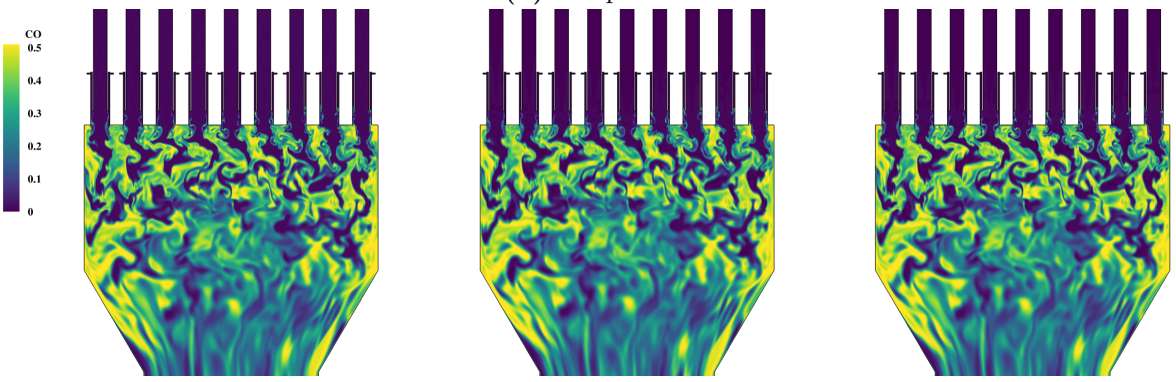
The point-wise accuracy of reconstructing the high-pressure combustion instability




**(a)** Static pressure




**(b)** Temperature




**(c)** CO mass fraction

**Figure 6.50:** Nine-element combustor unsampled MP-LSVT PROM slices, t = 21.8ms, $\Delta t = 5 \times \Delta t_{FOM}$. From left to right: FOM, $N_p = 25$, $N_p = 75$.

| Sampling Rate (%) | 0.025 | 0.05 | 0.1 | 0.25 | 0.5 | 1.0 |
|---|---|---|---|---|---|---|
| Cores | 44 | 88 | 132 | 308 | 572 | 1100 |
| Cells/core (approx.) | 82 | 82 | 109 | 117 | 126 | 131 |

**Table 6.4:** Partitioning for nine-element combustor HPROM sample meshes.

is examined in the probe monitor results shown in Fig. 6.49. As mentioned above, low trial basis dimensions ($N_p = 25$) tend to smear the pressure wave and generally fail to reconstruct the instantaneous probe measurements. Increasing the trial basis dimension to $N_p = 50$ improves on this measure, but enriching the trial basis further brings diminishing returns. Even increasing the basis dimension to $N_p = 100$ fails to capture much of the higher-frequency content or reproduce the maximum amplitude of the pressure signal. This implies a fundamental limitation of the linear trial basis, which struggles to recreate the complex interaction between the high-pressure transverse wave and the longitudinal reacting shear layers. However, given the extreme physical complexity and numerical stiffness of this problem, even these results are a testament to the robustness of the MP-LSVT method.

### 6.3.3 Hyper-reduced PROMs

Again, analyses for hyper-reduced MP-LSVT PROMs are repeated for the nine-element combustor. All results presented here use a physical time step of $\Delta t = 5 \times \Delta t_{\mathrm{FOM}}$, as well as a trial basis dimension of $N_p = 75$. The gappy POD regressors are constructed from the conservative field data. The investigation for this case is slightly more limited in scope than those presented for the cavity flow and truncated CVRC cases due to the significantly higher computational cost. As such, results exploring the effects of the sampling rate and gappy POD regressor dimension are coarser. Further, as will be shown shortly, only random sampling is capable of generating stable HPROM simulations; the partitioning data provided in Table 6.4 reflects this fact, as random sampling requires an exceptionally large number of auxiliary cells. The seemingly low cells/core counts is simply an indicator of the need for more cores to complete calculations on these large meshes in a timely fashion. Example isosurfaces of the sample mesh for random and

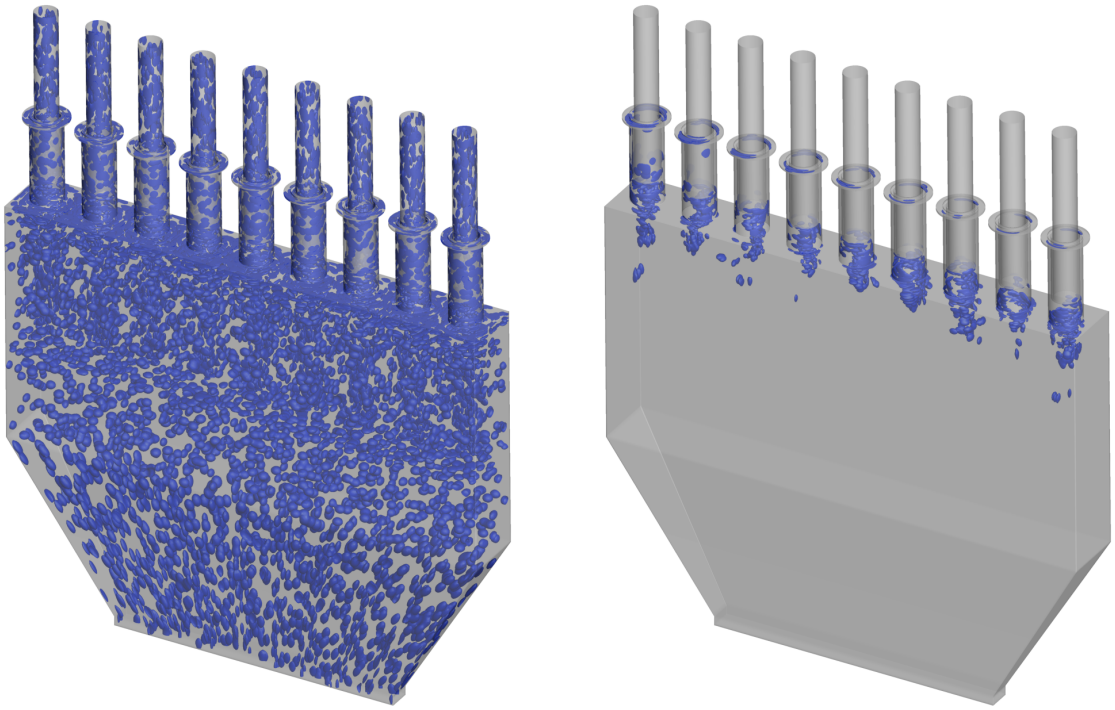


**Figure 6.51:** Example nine-element combustor sample meshes for $N_s = 0.1\% \times N$, $N_r = 100$, with random sampling (left) and eigenvector-based sampling (right).

eigenvector-based sampling are presented in Fig. 6.51 illustrate this fact well, showing that the greedy method generates an extremely compact sample mesh centered on the reacting shear layer of each injector, while the randomly-sampled mesh is distributed throughout the domain.

The poor performance of the eigenvector-based sampling algorithm is made immediately apparent in the pressure probe measurements displayed in Fig. 6.52. Even for $N_s = 0.1\% \times N$ (for which the algorithm performed quite well for the cavity and CVRC cases), the solution employing sample meshes computed via the eigenvector-based sampling almost immediately diverges and quickly becomes unstable. The same behavior is observed for sampling rates up to $N_s = 0.25\% \times N$, after which point the offline computational cost of computing the sample mesh becomes exorbitantly large (>20,000 core-hours). Comparatively, the random sampling algorithm requires negligible offline computational cost and is capable of producing a reasonable reconstruction of the unsteady pressure signal. Drawing attention again to the sample mesh visualization in Fig. 6.51, note that the eigenvector-based algorithm selects cells in tight clusters either at the fuel injection ports or in the mixing shear layer of each injector. Few cells are

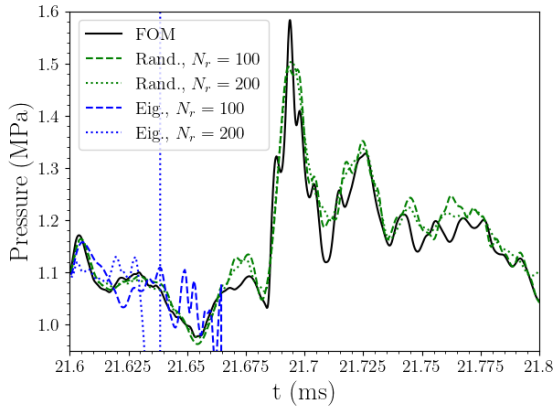



**Figure 6.52:** Nine-element combustor HP-PROM pressure probe measurements, $N_s = 0.1\% \times N$, various sampling algorithms and $N_r$.

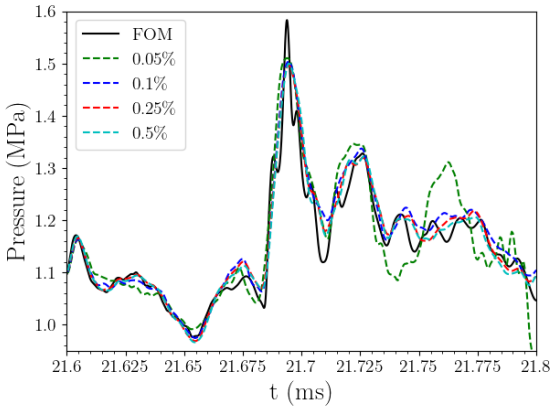
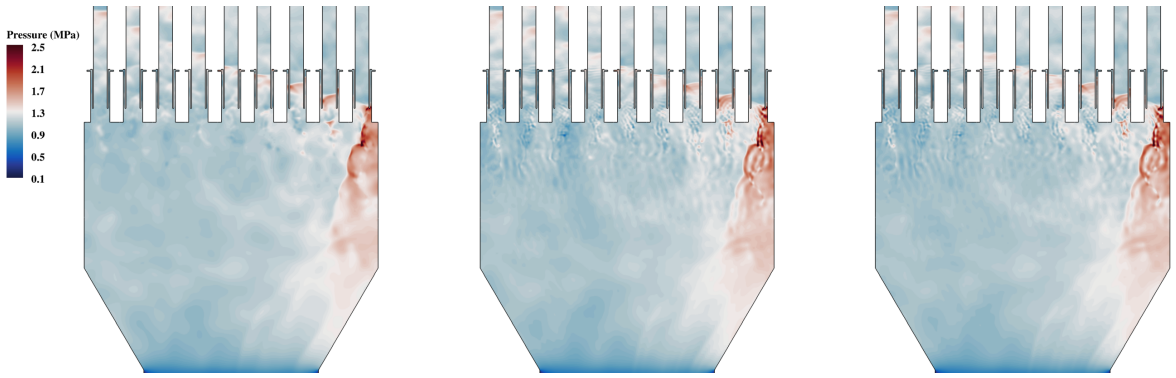



**Figure 6.53:** Nine-element combustor HP-PROM pressure probe measurements, random sampling, $N_r = 200$, various $N_s$.

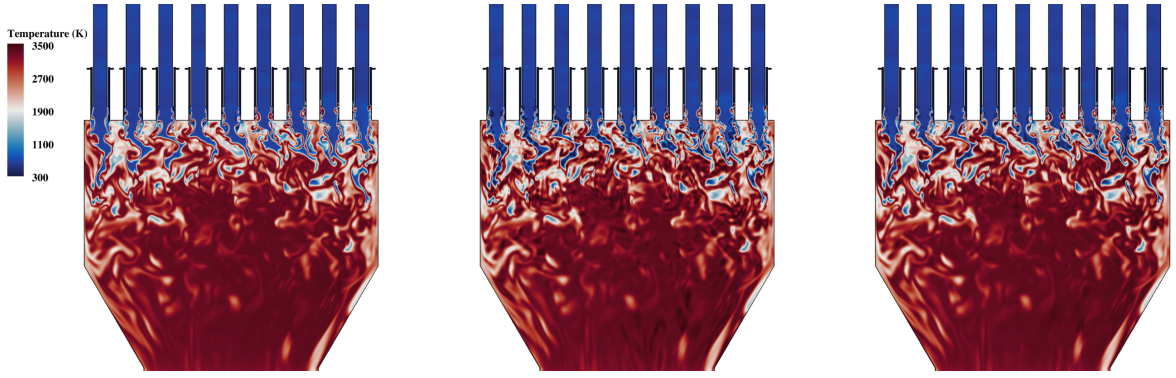
placed in the combustion chamber, where the high-pressure combustion instability occurs and is sustained by the high heat release of the ongoing reaction. In this context, it is not surprising that this greedy algorithm is incapable of modeling the traversal of the pressure wave across the width of the combustion chamber. Although not shown here, both variants of the GNAT sampling algorithm perform equally poorly (unsurprisingly, as those algorithms tend to cluster samples even more closely).

On the one hand, this result is somewhat discouraging after the excellent performance of the eigenvector-based sampling algorithm for the cavity and CVRC cases. However, it also sheds light on the need for improved hyper-reduction methods for problems which exhibit spatially-distributed convective phenomena (the transverse pressure wave), which may be tightly coupled to spatially-localized physics (propellant injection). As will be displayed in the following chapter, online trial basis and sample mesh adaptation proposes solutions to such problems, allowing the sample mesh to evolve with time and more accurately capture bulk advection effects. Such methods are not applied to this case, unfortunately, as the implementation of the underlying adaptation algorithms is not yet communication-scalable, and cannot solve adaptive ROM systems of such high dimensionality.

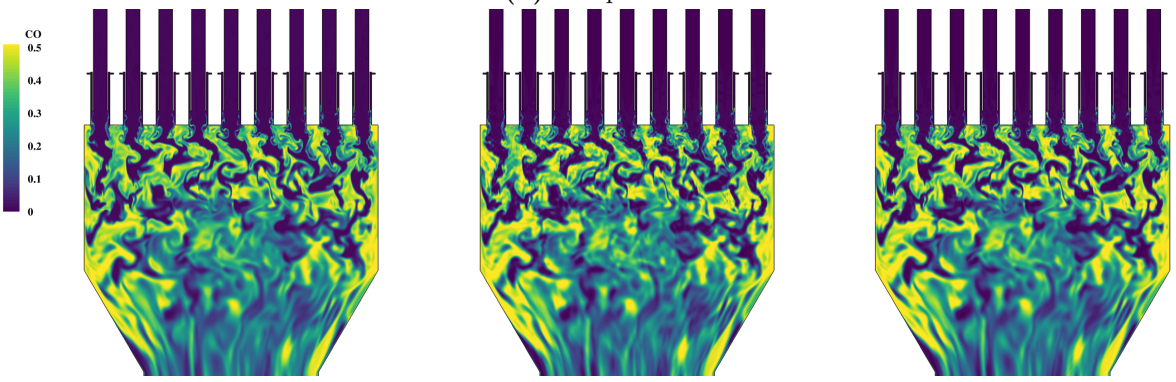
Despite the inadequate performance of the greedy sampling algorithms, the random sampling method is still capable of generating good results, even for a system of this




**(a)** Static pressure




**(b)** Temperature




**(c)** CO mass fraction

**Figure 6.54:** Nine-element HPROM slices, t = 21.8ms, $N_r = 200$, $\Delta t = 5 \times \Delta t_{FOM}$. From left to right: FOM, $N_s = 0.1\% \times N$, $N_s = 0.5\% \times N$.

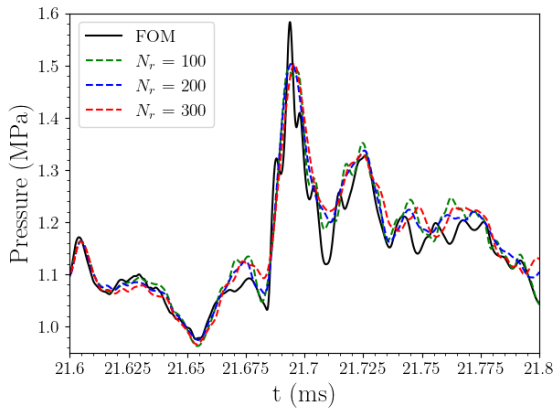



**Figure 6.55:** Nine-element combustor HPPROM pressure probe measurements, random sampling, $N_s = 0.1\% \times N$ various $N_r$.

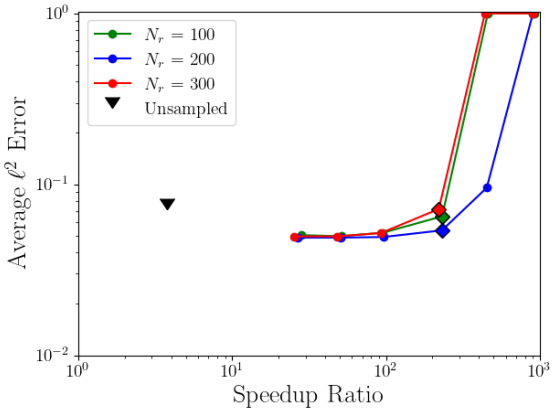



**Figure 6.56:** Nine-element combustor HPROM time-average error vs. CPU-time speedup, various $N_r$.

complexity and size. Figure 6.54 displays several sample slices to this effect, showing HPROM results at the end of the simulation time window for $N_s \in \{0.1,\ 0.5\}\% \times N$. Although the pressure field (Fig. 6.54a) exhibits significant spurious oscillations near the injector face, the temperature (Fig. 6.54b) and CO mass fraction (Fig. 6.50c) fields reconstruct those of the FOM with remarkable accuracy. The effect of the sampling rate is displayed via pressure monitor measurements in Fig. 6.53: below $N_s = 0.1\% \times N_s$, the long-term accuracy of the solution begins to degrade noticeably, while increasing the sampling rate results in marginal improvement. Although the HPROMs appear incapable of capturing the full amplitude of the pressure wave, this may be attributed to the coarseness of the trial basis, as it is observed that the full amplitude is not captured by the unsampled ROM even with $N_p = 100$.

The effect of the gappy POD regressor basis is noted by probe measurements in Fig. 6.55. To the human eye, there is little significant difference in the reconstruction of the pressure signal, implying a less significant effect of $N_r$ for this system than was observed for the cavity and CVRC cases. Aggregate error measurements, displayed with respect to computational cost speedup in Fig. 6.56, reveal similarly small discrepancies with regard to $N_r$. Unfortunately, as observed with the cavity and CVRC case, the random sampling algorithm is unable to generate the enormous cost savings and robust solutions that the eigenvector-based sampling was capable of for those cases. As the

greedy algorithms appeared generally unstable, the random algorithm is able to produce, at best, roughly two orders of magnitude computational cost savings while remaining stable and accurate. Although not as impressive as the three to four orders of magnitude speedup observed for the CVRC case, this still represents a major accomplishment in the application of these methods to a problem of such high dimensionality and extreme physical complexity. Further, it serves as a testament to the ability of the MP-LSVT framework to generate accurate and robust solutions in combination with good computational cost savings when paired with a careful hyper-reduction approach.

## 6.4 Conclusions

In this chapter, scalable hyper-reduced PROMs were tested for several multi-scale and multi-physics systems of increasing dimension and complexity. Beginning with a non-reacting transonic open cavity flow case, residual-based PROM methods including LSPG and MP-LSVT were shown to be superior to the standard Galerkin projection method. The MP-LSVT method further exhibited excellent accuracy in modeling extremely stiff rocket combustor systems, including single-element and linear nine-element combustors, for which the LSPG method was incapable of producing stable solutions. The sensitivity of HPROM load balancing, accuracy, and computational cost savings to a variety of model parameters were rigorously investigating, resulting in the following insights:

1. Randomized sampling, although simple and inexpensive to implement, generally resulted in greater memory consumption and more floating-point calculations than GNAT or eigenvector-based sampling. This is due to a wider spatial distribution of sampling points, which demands a commensurately greater number of auxiliary cells for computing fluxes, gradients, and vertex state reconstructions.

2. On the other hand, randomized sampling produced computational meshes which require comparatively few (often zero) MPI communications. This is in contrast to GNAT sampling and eigenvector-based sampling, which often required a significant number of MPI communications. Eigenvector-based sampling generally required

fewer MPI communications than GNAT sampling.

3. Gappy POD HPROMs were able to produce robust, accurate reconstructions of the trial basis training window, although the accuracy deteriorated at extremely low sampling rates. The hyper-reduced PROMs, along with careful load balancing, were able to easily achieve over three orders of magnitude speedup in core time over the FOM in some instances.

4. The original GNAT sampling procedure proposed tends to generate small sample meshes, but produces very inaccurate HPROMs compared to those generated by the fine-grained variant. While the original algorithm is significantly less expensive to evauate in the offline stage, the deterioration in accuracy and stability is extreme.

5. Interestingly, random sampling tends to outperform both GNAT greedy sampling algorithms, and consumes a fraction of the offline computational cost. At very low sampling rates however, random sampling also tends to produce unstable HPROMs.

6. On the whole, the eigenvector-based sampling strategy appeared to perform the best across most metrics of interest. Among the investigated strategies, it required the lowest memory consumption and fewest floating-point calculations, outperformed GNAT sampling in the number of required MPI communications, and generally produced the lowest average solution error for a given sampling rate across a range of time step sizes. The last point was especially true at extremely low sampling rates. Here, HPROMs using eigenvector-based sampling produced comparable error to that of the unsampled PROM, whereas HPROMs using random and GNAT sampling were highly erroneous or even unstable.

7. The above point is not true for the nine-element combustor, for which greedy sampling algorithms (including eigenvector-based sampling) failed to generate stable HPROMs. Random sampling was capable of generating modest speedups while retaining simulation accuracy. Future work must address similar systems which are characterized by advection occurring at vastly different time scales.

8. Increased time step sizes were able to push speedup further, though there is (unsurprisingly) a significant error increase for unsampled PROMs at much larger time steps. Interestingly, the accuracy of hyper-reduced PROMs remained fairly consistent at larger time steps.

The work presented in this chapter represents the first successful experiments of hyper-reduced projection-based ROMs of this size and physical complexity to date, to the best of the author's knowledge. These findings establish useful heuristics for future practitioners to aid the development of accurate and scalable HPROMs for systems exhibiting similarly challenging phenomena including strong gradients, propagating waves, exceptional stiffness, and extreme scale disparity.

# Chapter 7

# Adaptive HPROMs for Rocket Combustors

For the results shown in Chapter 6, all simulations have simply *reconstructed* the training data. That is, the geometry, initial conditions, boundary conditions, and simulation duration of each PROM are identical to those of the FOM from which the training data are extracted. Although these simulations were successful in this effort, the ultimate goal of any data-driven method is *generalizability*. To be truly useful for engineering applications, such methods must be able to make fast and accurate predictions for unseen configurations which are not accounted for in the training data.

## 7.1 Failure of Static Trial Bases

The PROMs investigated in Chapter 6 are, unfortunately, not generalizable. To demonstrate this for the truncated CVRC case examined in Section 6.2, the unsampled MP-LSVT PROM is allowed to run for a longer duration than the original FOM training data, to t = 5.6 ms. The resulting dump plane corner pressure probe is shown in Fig. 7.1. Even though the PROM is capable of simulating the training period with exceptionally high accuracy, the solution rapidly deviates shortly after the end of the training period at t = 5.5 ms. The dominant cause of this failure is the inability of the linear trial space to model realizations of the flow field which were not included in the training data. In a sense, this is an attempt at *extrapolation* in time (rather than a sort of *interpolation* within the training set). Although disappointing, this result is not at all surprising, as data-driven methods often struggle to model unseen data, particularly for highly non-linear systems.

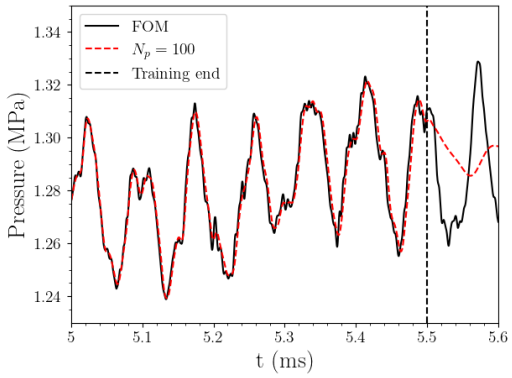



**Figure 7.1:** CVRC pressure probe, unsampled MP-LSVT, $N_p = 100$, $\Delta t = 5 \times \Delta t_{FOM}$.

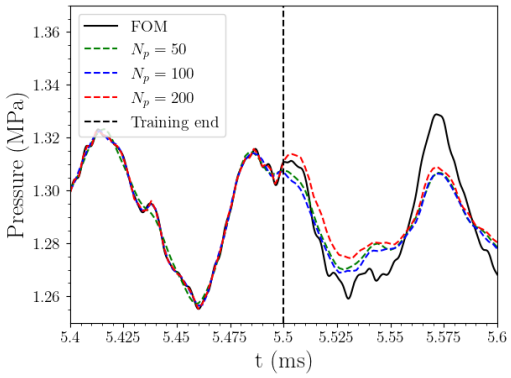



**Figure 7.2:** CVRC pressure probe, projected solution, various $N_p$.

The inability of the trial basis to represent unseen data is made readily apparent by observing the average projection error over time in Fig. 7.3. After t = 5.5 ms, the projection error drastically increases, and increasing the dimension of the trial basis is unable to improve this whatsoever. This can be visually observed in pressure probe measurements of the projected solution in Fig. 7.2, where large discrepancies can be observed beyond t = 5.5 ms.

Comparisons of the FOM and projected pressure and temperature fields at t = 5.6 ms can be seen in Figs. 7.4 and 7.5. In both instances, the solution appears smeared and more axisymmetric, lacking most of the transient features of the flow field such as the highly distorted flame front in the combustion chamber. As the online PROM cannot be expected to model the unsteady solution more accurately than the projected solution, the previous failure of the PROM is to be expected.

Given the poor performance of the unsampled PROM, it can be safely assumed that any HPROM will perform as poorly or worse. Ultimately, the failure lies purely with the fact that an accurate solution does not exist in the trial space. Alternatively, several methods have been proposed which *adapt* a linear trial basis and hyper-reduction sampling configuration during the unsteady solution of the linear subspace PROM. Examples of approaches which propose time-variant trial bases include dictionary-based methods (sometimes referred to as *local* bases) [190, 191, 192], space-time trial bases [193, 194], or basis transport maps [195]. Although these methods have displayed exceptional ac-

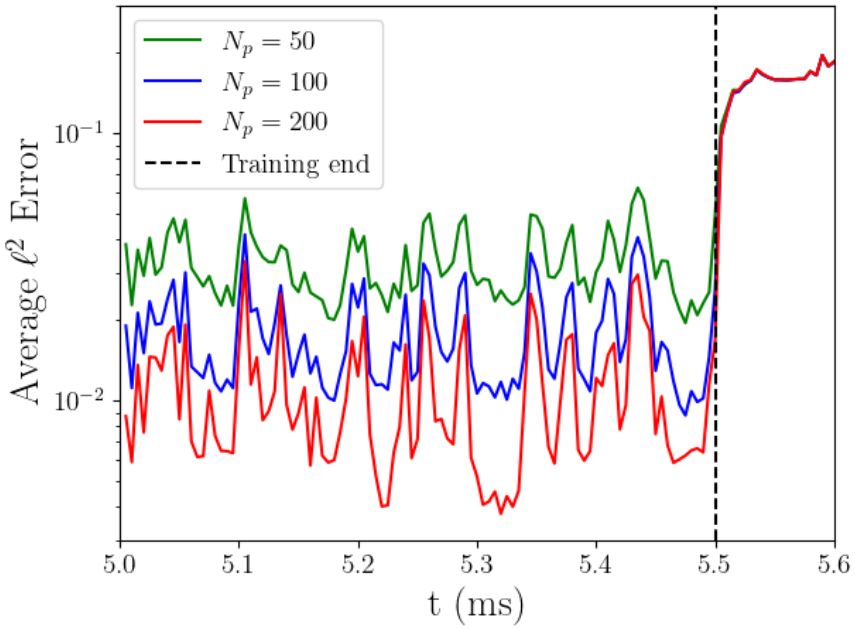



**Figure 7.3:** CVRC average projection error over time, various $N_p$.

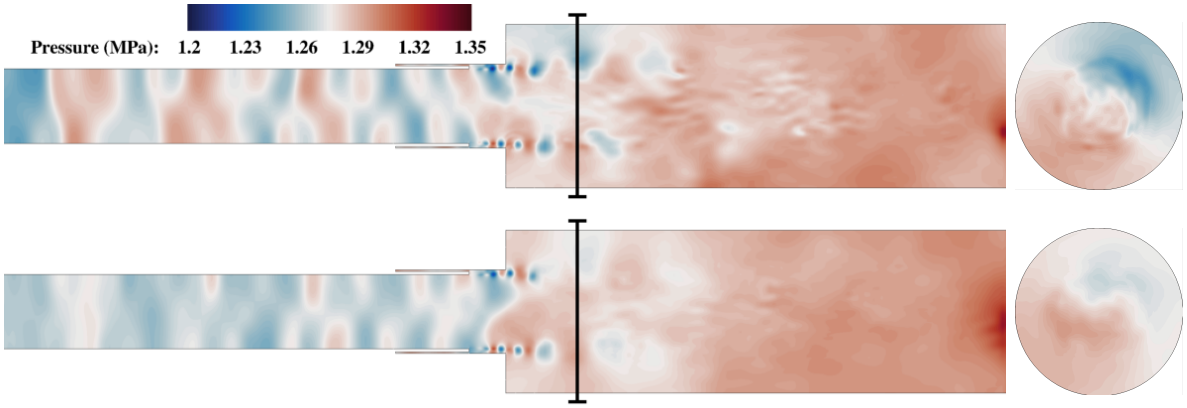



**Figure 7.4:** CVRC pressure field beyond training bounds, t = 5.6 ms, FOM at top and projected solution for $N_p = 100$ below.

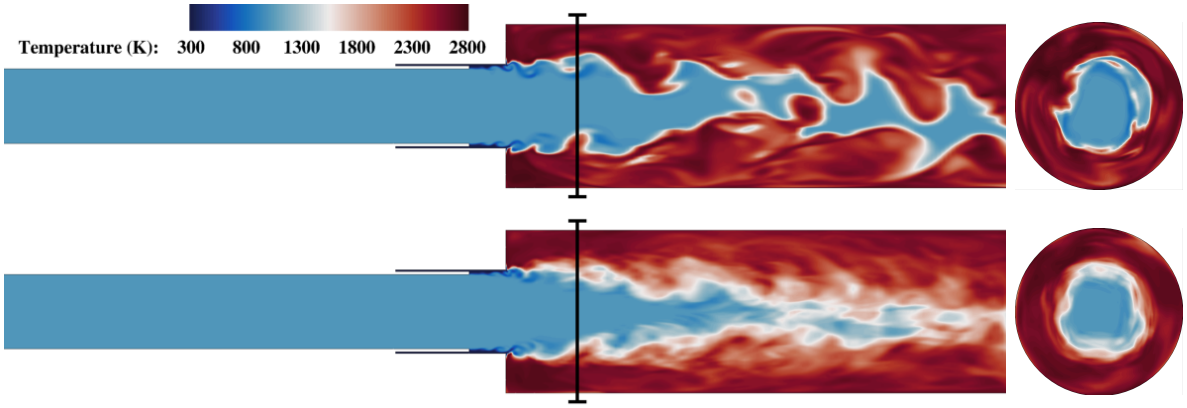



**Figure 7.5:** CVRC temperture field beyond training bounds, t = 5.6 ms, FOM at top and projected solution for $N_p = 100$ below.

curacy improvements over static basis methods for convection-dominated problems, they still suffer from the fact that the constituent bases (e.g., those that form the dictionary) are constructed from the training data and may not accurately represent the unsteady solution when it significantly diverges from states observed in the full-order datasets. A notable exception is the work by Etter and Carlberg [196], which updates the trial basis during online computations by vector space sieving.

The following sections explore adaptive PROMs by borrowing elements from the AADEIM framework by Peherstorfer [197, 153], which leverages limited, periodic queries of the full-order model to incorporate the online state evolution in adapting the sampling configuration. This is investigated in combination with the recent one-step adaptation framework by Huang and Duraisamy [126] as a method for adapting the trial basis.

## 7.2 AADEIM

The AADEIM framework, originally posed by Peherstorfer and Willcox [197] and later refined by Peherstorfer [153] is outlined here. This section borrows heavily from their notation for the sake of consistency. Only the sample mesh adaptation of AADEIM is utilized, though the trial basis adaptation method is detailed here for the sake of completeness.

The AADEIM procedure is predicated on the presentation of the fully discrete FOM O$\Delta$E as given in Eq. 4.23, repeated here as

$$\mathbf{q}_c^{n-1} - \mathbf{f}_\mathbf{r}\left(\mathbf{q}_c^n,\ \mathrm{t}^n\right) = \mathbf{0}. \tag{7.1}$$

Again, this is an extremely counterintuitive way to write the evolution equations, but has the benefit of framing the residual in such a way as to motivate equating the trial basis $\mathbf{U}_c$ and the regression basis $\boldsymbol{\Psi}$, as outlined in Section 4.4.2. From this frame of reference, AADEIM proposes adaptation of the trial basis $\mathbf{U}_c$ as additive low-rank updates of the form

$$\mathbf{U}_c^{n+1} = \mathbf{U}_c^n + \boldsymbol{\alpha}^n \left[\boldsymbol{\beta}^n\right]^\top. \tag{7.2}$$

The matrices $\boldsymbol{\alpha} \in \mathbb{R}^{N \times z}$ and $\boldsymbol{\beta} \in \mathbb{R}^{N_c \times z}$ compute the low-rank update, where $z$ indicates the rank of the update. The original AADEIM procedure draws a distinction between sampling points (and the corresponding sampling operator $\mathbf{S} \in \mathbb{R}^{N_s \times N}$) and interpolation points (and its own operator $\widehat{\mathbf{S}} \in \mathbb{R}^{N_r \times N}$), where the latter is linked to the original DEIM formulation. It is not necessary that the interpolation points be members of the sample set.

Given these sampling and interpolation operations, the trial basis update is formulated as the solution to the least-squares minimization of the error,

$$\boldsymbol{\alpha}^n \left[\boldsymbol{\beta}^n\right]^\top = \underset{\boldsymbol{\alpha} \in \mathbb{R}^{N \times z},\, \boldsymbol{\beta} \in \mathbb{R}^{N_c \times z}}{argmin} \left\| \mathbf{S}^n \left[ \left[ \mathbf{U}_c^n + \boldsymbol{\alpha} \boldsymbol{\beta}^\top \right] \left[ \widehat{\mathbf{S}}^n \mathbf{U}_c^n \right]^{-1} \widehat{\mathbf{S}}^n - \mathbf{I} \right] \mathbf{F}_r^n \right\|_F^2, \tag{7.3}$$

where the snapshot matrix $\mathbf{F}_r^n \in \mathbb{R}^{N \times w}$ is defined as

$$\mathbf{F}_r^n := \left[ \mathbf{f_r}\left(\widetilde{\mathbf{q}}_c^{n-w}\right), \ \ldots \ , \ \mathbf{f_r}\left(\widetilde{\mathbf{q}}_c^{n-1}\right) \right] \tag{7.4}$$

$$:= \left[ \widetilde{\mathbf{q}}_c^{n-w+1}, \ \ldots \ , \ \widetilde{\mathbf{q}}_c^{n} \right] \tag{7.5}$$

The window size $w \in \mathbb{N}_0$ represents the number of prior PROM solutions for which the regression error in Eq. 7.3 is considered. That is, during the online solution of the PROM, the state $\widetilde{\mathbf{q}}_c^n$ is stored in $\mathbf{F}_r^n$, and the solution of least-squares minimization of the regression error of this solution updates the the trial basis $\mathbf{U}_c$ during the PROM runtime. Note that in the work by Peherstorfer [153], those degrees of freedom which are *not* sampled by $\widehat{\mathbf{S}}$ are simply reconstructed by $\mathbf{S}\widetilde{\mathbf{q}}_c = \mathbf{S}\mathbf{U}_c^n \left[\widehat{\mathbf{S}}\mathbf{U}_c^n\right]^{-1} \widehat{\mathbf{S}} \left[\overline{\mathbf{q}}_c + \mathbf{H}_c \mathbf{U}_c \widehat{\mathbf{q}}_c^n\right]$.

The adaptation of the sample mesh is significantly simpler than that of the trial basis. The regression error is computed at every degree of freedom according to the formulation

$$\mathbf{R}^n = \left(\mathbf{I} - \boldsymbol{\Psi}^n \left[\mathbf{S}^n \boldsymbol{\Psi}^n\right]^+ \mathbf{S}^n \mathbf{F}_r^n\right). \tag{7.6}$$

Those $N_s$ degrees of freedom which incur the highest regression error are selected to construct the new sampling mesh defined by $\mathbf{S}^{n+1}$. Note that this step constitutes a *loss of independence* from the full-order dimension $N$, as the error must be computed

at every degree of freedom. The full-order model operator must be evaluated at an infrequent interval $N_u$ to calculate this. The effect of this query of the full-order system will be shown to be a major handicap of the method in later results.

## 7.3 One-step Basis Adaptation

The trial space adaptation proposed by Huang *et al.* [126] provides a simpler alternative to that provided by AADEIM. This method begins by suggesting a basis increment $\mathbf{U}'_c \in \mathbb{R}^{N \times N_c}$ which seeks to construct an improved trial space with which to represent the state vector computed from the FOM solution $\mathbf{q}_c$, given a fixed latent state $\widehat{\mathbf{q}}_c$. This can be formalized at the $n$th time step as

$$\mathbf{q}_c^n = \overline{\mathbf{q}}_c + \mathbf{H}_c \left[\mathbf{U}_c^n + [\mathbf{U}'_c]^n\right] \widehat{\mathbf{q}}_c^n, \tag{7.7}$$

where the FOM solution $\mathbf{q}_c^n$ is computed in tandem with the low-dimensional PROM solution $\widehat{\mathbf{q}}_c^n$. Rearranging terms arrives at

$$[\mathbf{U}'_c]^n \widehat{\mathbf{q}}_c^n = \mathbf{H}_c^{-1} \left[\mathbf{q}_c^n - \overline{\mathbf{q}}_c\right] - \mathbf{U}_c^n \widehat{\mathbf{q}}_c^n. \tag{7.8}$$

Obviously, this system of equations is underdetermined, attempting to solve for $N \times N_c$ variables given $N$ equations. As such, there is no unique solution for $[\mathbf{U}'_c]^n$. One solution suggested in [126] is given as

$$[\mathbf{U}'_c]^n = \frac{\left[\mathbf{H}_c^{-1} \left[\mathbf{q}_c^n - \overline{\mathbf{q}}_c\right] - \mathbf{U}_c^n \widehat{\mathbf{q}}_c^n\right] \left[\widehat{\mathbf{q}}_c^n\right]^\top}{\|\widehat{\mathbf{q}}_c^n\|_2^2} \tag{7.9}$$

Note that this approach is not strictly feasible given that the solution of the FOM equations $\mathbf{q}_c^n$ must be computed in order to evaluate the trial basis update, thus defeating the purpose of the PROM. However, the usefulness of this approach becomes apparent when hyper-reduction is applied.

Two basis update steps are delineated: that occurring after iterations which only compute a HPROM solution, and that occurring after an iteration during which the

sample mesh is updated as in the previous section. In the former case, the basis is only adapted at sampled points by the form

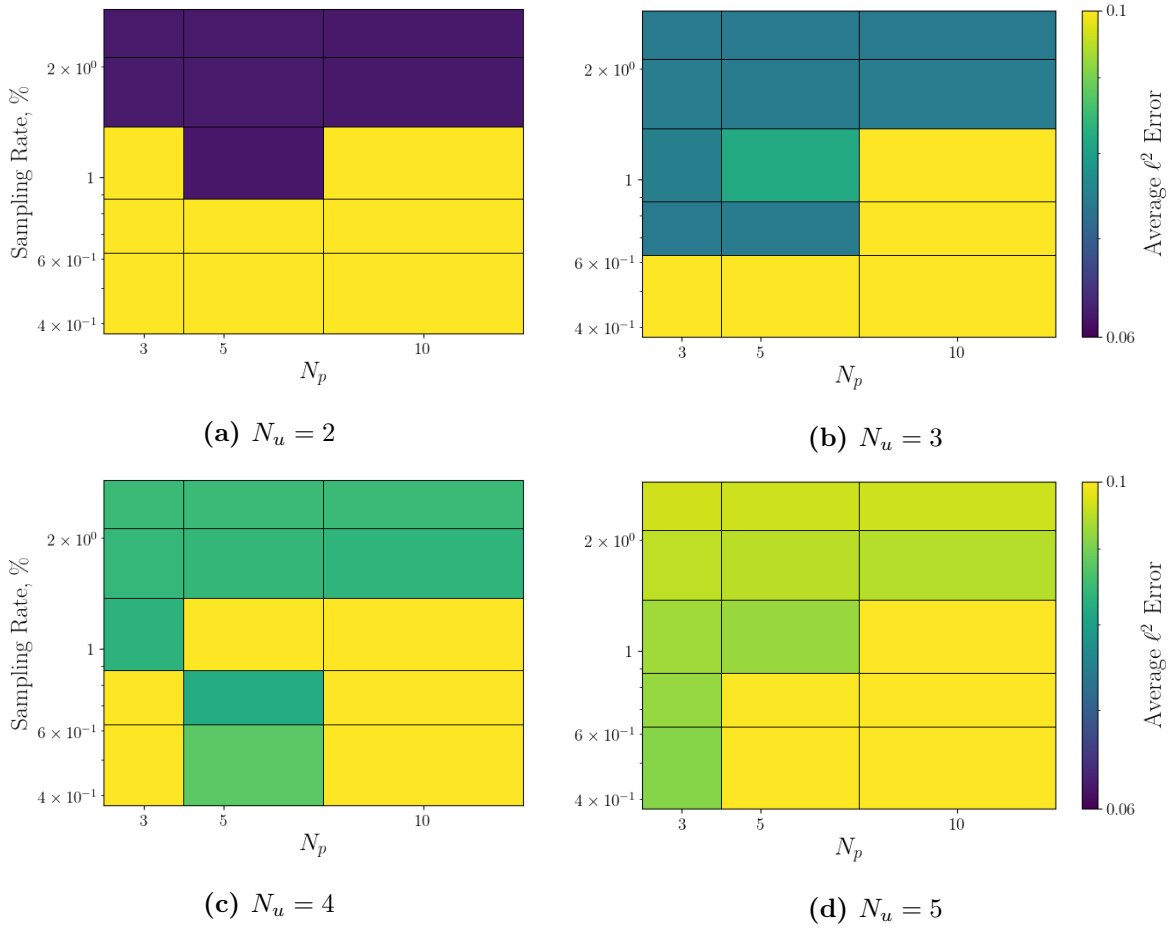
$$\mathbf{S}^n[\mathbf{U}_c']^{n+1} = \mathbf{S}^n[\mathbf{U}_c']^n + \frac{\mathbf{S}^n\left[\mathbf{H}_c^{-1}\left[\mathbf{q}_c^n - \overline{\mathbf{q}}_c\right] - \mathbf{U}_c^n\widehat{\mathbf{q}}_c^n\right]\left[\widehat{\mathbf{q}}_c^n\right]^\top}{\|\widehat{\mathbf{q}}_c^n\|_2^2} \tag{7.10}$$

## 7.4 Online Adaptation for the CVRC

Following the above process, future-state prediction is attempted for the truncated CVRC case introduced in Section 6.2.

The same simulation duration, t $\in$ [5.0, 5.5] ms, is investigated, but the training window is altered to t $\in$ [5.0, 5.005] ms, with snapshots collected every time step at $\Delta$t $= 1 \times 10^{-7}$ for a total of 50 snapshots. The total prediction period is thus 100 times larger than the training period. The physical time step of all adaptive HPROMs is the same as that of the FOM. Initial sample meshes are all computed using the random sampling algorithm. Adaptation begins at the tenth time step, after which the trial basis is updated at every iteration. For a given adaptation step, the first half of the subiterations are dedicated to the HPROM solve, while the latter half are used for the FOM solution. Adaptive HPROMs are computed for all permutations of trial basis dimension $N_p \in \{2,\ 5,\ 10\}$, sampling rate $N_s \in \{0.5,\ 0.75,\ 1.0,\ 1.75,\ 2.5\}\% \times N$, and update interval $N_u \in \{2,\ 3,\ 4,\ 5\}$.

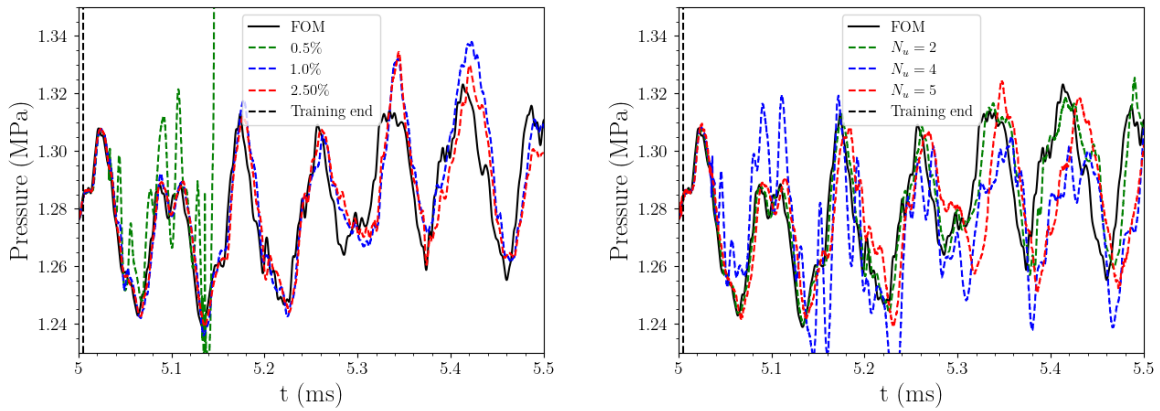
Error contours with respect to the trial basis dimension and sampling rate, for a given update frequency, are given in Fig. 7.6. Those cases marked in bright yellow are effectively failed simulations, incurring well in excess of 10% error. Overall, the observed error for error stable solutions is significantly higher than that observed for the static HPROMs in Section 6.2, ranging roughly within 6-10%. Further, note that the tested sampling rates are much higher than those studied for the static HPROMs, and tend to require 1% sampling to achieve stability. However, recall that the equivalent static HPROM, trained from only the first 50 $\mu$s of the FOM simulation, would invariable be unstable. Ultimately, the simulations behave similarly with respect to the sampling rate,




**(a)** $N_u = 2$ **(b)** $N_u = 3$

**(c)** $N_u = 4$ **(d)** $N_u = 5$

**Figure 7.6:** CVRC adaptive HPROM time-average error contours with respect to trial basis dimension and sampling rate, various $N_u$.




**(a)** $N_u = 3$, various $N_s$.

**(b)** $N_s = 1.0\%$, various $N_u$.

**Figure 7.7:** CVRC adaptive HPROM pressure probes, $N_p = 5$.

generally improving with increased sampling. Also to be expected, accuracy tends to degrade with increased update frequency, with $N_u = 5$ generating approximately $\geq 10\%$ error in all cases. Surprisingly, stability tends to decrease with respect to the trial basis dimension. This may be a function of the non-uniqueness of the one-step adaptation process, whereby a larger trial space induces a sub-optimal update. As a sanity check, the degenerative solution for $N_p = 1$ and $\mathbf{U}_p^n = \widetilde{\mathbf{q}}_p^n$ does not produce a stable solution.

Pressure probe monitors for several adaptive HPROM configurations are shown in Fig. 7.7. The end of the training period is marked by a vertical dashed line, at the far left of the images. Figure 7.7a visualizes the effect of the sampling rate on predictive performance, with the obvious conclusion that increasing the sampling rate tends to improve the pressure signal accuracy. While the case for which $N_s = 0.5\% \times N$ quickly becomes unstable, higher sampling rates generate near-perfect predictions up to t = 5.3 ms. After this point, however, significant over- and undershoots of the signal are observed, though the frequency of the dominant pressure oscillation is maintained. Figure 7.7b compares various update intervals, where previous observations are confirmed. Near-perfect prediction is achieved for $N_u = 2$, while $N_u = 5$ causes large discrepancies beginning at t = 5.2 ms and even appears to slightly decrease the frequency of the dominant acoustic mode.

Field comparisons at t = 5.5 for various update intervals are shown in Figs. 7.8 and 7.9. Although relative accuracy is not immediately obvious, in the pressure fields shown in Fig. 7.8 it is apparent that the highly-inaccurate case for $N_u = 4$ gives rise

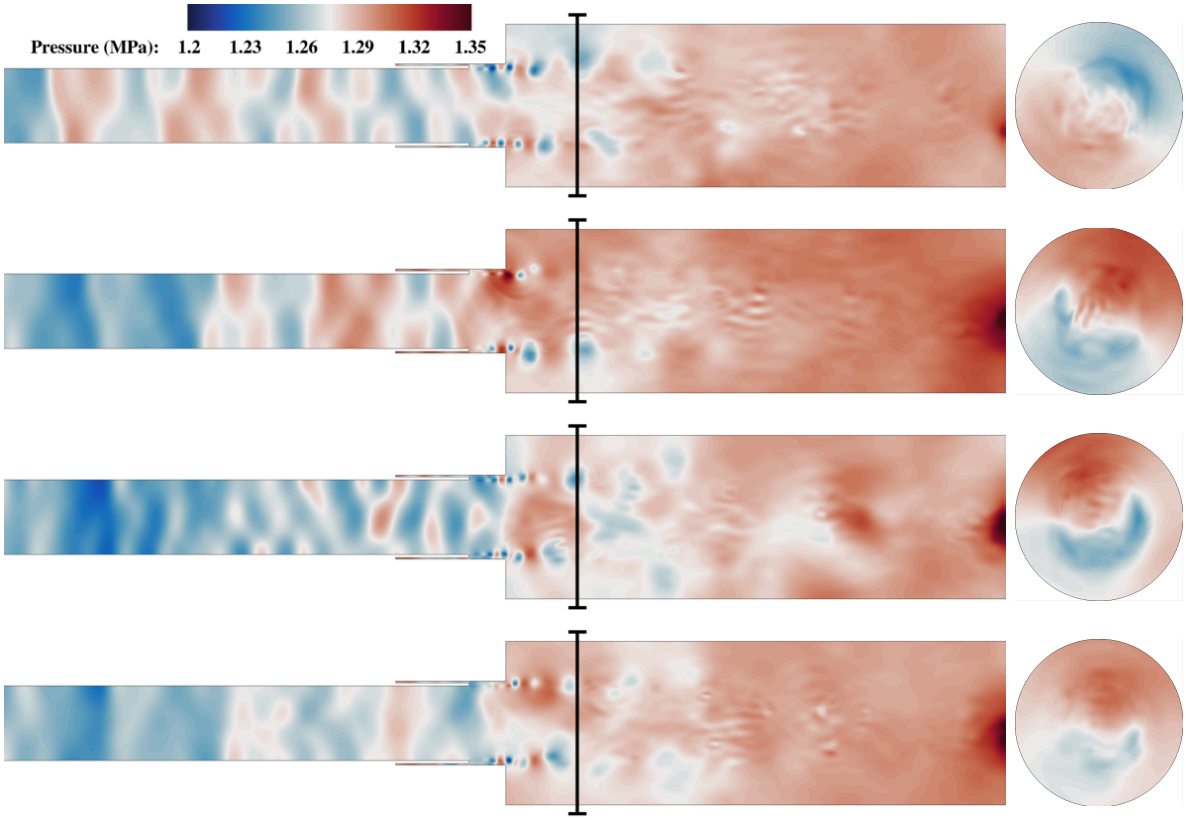



**Figure 7.8:** CVRC pressure field t = 5.5 ms, $N_p = 5$, $N_s = 1\% \times N$. From top to bottom: FOM, $N_u = 2$, $N_u = 4$, $N_u = 5$.

to spurious high-frequency oscillations along the center axis of the combustion chamber, just downstream of the dump plane. The evolution of the temperature field in Fig. 7.9 is similarly opaque, though all appear to generate the characteristic shear layer and vigorous mixing near the exit.

The process of sample mesh adaptation is briefly touched upon. Recall that the error metric used to select a new sample set is computed from the regression error of the full field $\widetilde{\mathbf{q}}_p$. This metric is computed in one shot, and does not involve any greedy iteration process like those outlined in Section 4.5.2. As such, the sample mesh slices shown in Fig. 7.10 are largely unsurpising, generating extremely tight clusters of samples similar to those generated by GNAT sampling as seen in Section 6.2. Again, samples tend to be selected in the reacting shear layer, which is characterized by highly unsteady mixing and sharp gradients at the flame front. Isosurfaces of the directly-sampled mesh elements in Fig. 7.11 emphasize this, generating a nearly conical sample mesh similar to those observed for eigenvector-based sampling in Section 6.2.

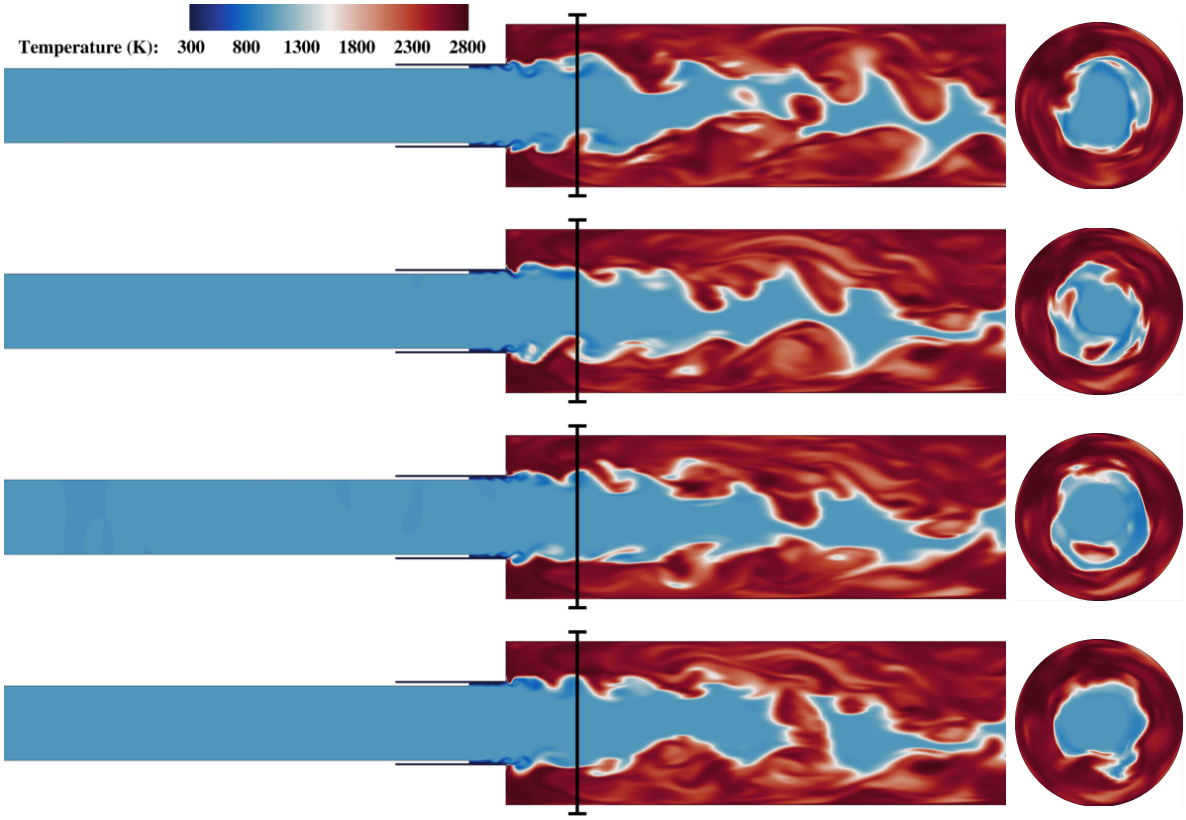



**Figure 7.9:** CVRC temperature field t = 5.5 ms, $N_p = 5$, $N_s = 1\% \times N$. From top to bottom: FOM, $N_u = 2$, $N_u = 4$, $N_u = 5$.

Although the predictive performance of these adaptive HPROMs shown above are quite remarkable, they are not without major drawbacks. Primary among those for the adaptive approach used here is the need to compute the full-order solution at the interval specified by $N_u$. For the cases explored here, for which half of the subiterations for an adaptation step are dedicated to the FOM solve, the absolute maximum computational speedup that can be achieved is thus equal to $2 \times N_u$. This is confirmed in Fig. 7.12, which displays the tradeoff in error with respect to computational speedup for all HPROMs computed. Each line represents one value of $N_u$, and each point represents a different sampling rate. Although these results indicate that the solver is able to nearly achieve the upper bound of computational speedup, the cost savings are paltry compared to those observed for the static HPROMs in Chapter 6. Further, the sampling rate appears to have little effect on the overall accuracy or computational speedup beyond a certain point, as any speedup is inevitably dwarfed by the cost of the sampling update step.

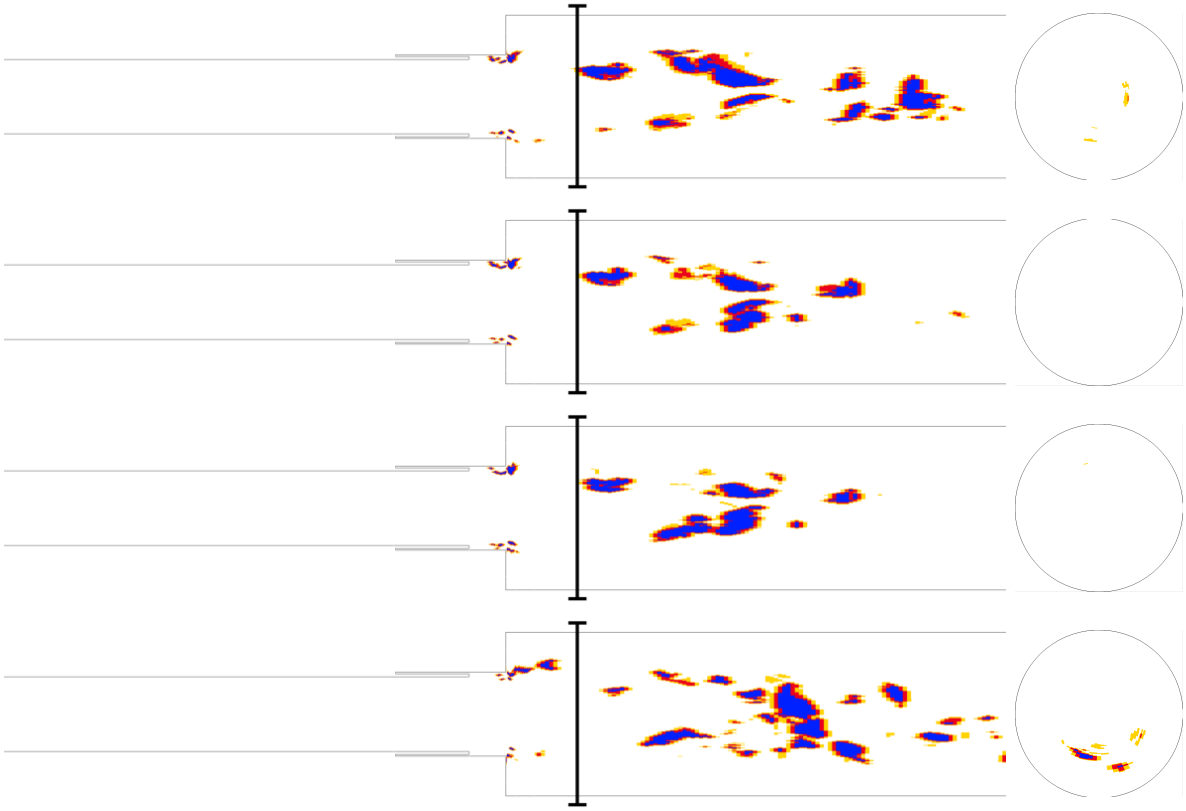


**Figure 7.10:** CVRC sample mesh, $N_p = 5$, $N_s = 1\% \times N$. From top to bottom, t = 5.125 ms, 5.25 ms, 5.375 ms, 5.5 ms.

## 7.5 Conclusions

The above results represent an exploratory effort in the application of online trial basis and sample mesh adaptation in pursuit of truly predictive HPROMs. Although the methodology and initial implementation was not conducted by the author, personal contributions centered on optimization of the implementation to enable the feasible application of these methods of problems of unprecedented scale. Although the methodology is fundamentally limited in its ability to enable significant computational cost savings, these results display that the method is capable of generating (quantitatively) accurate simulations of high-dimensional, highly non-linear combusting flows beyond the limits of the training dataset with measurable speedup.

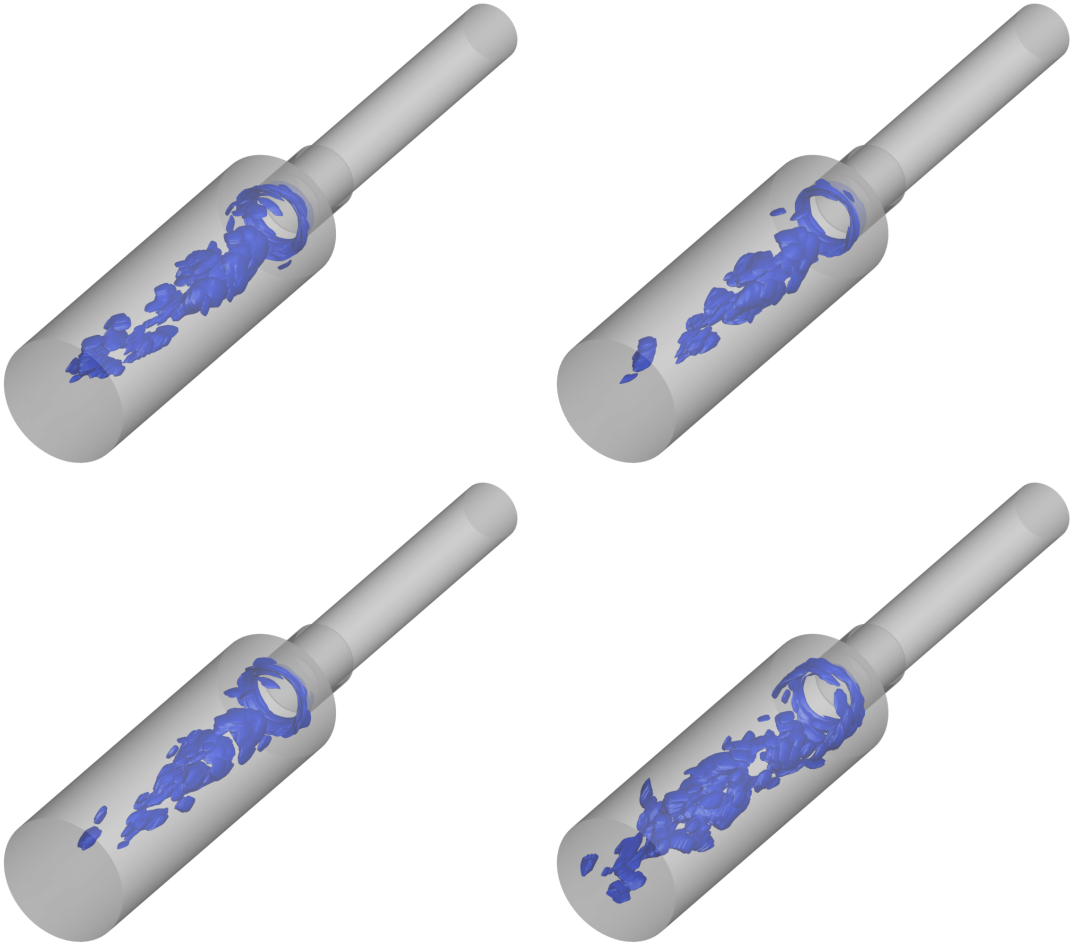


**Figure 7.11:** Directly-sampled cell isosurfaces, $N_p = 5$, $N_s = 1\% \times N$. Top left is t = 5.125 ms, top right is 5.25 ms, bottom left is 5.375 ms, bottom right is 5.5 ms.

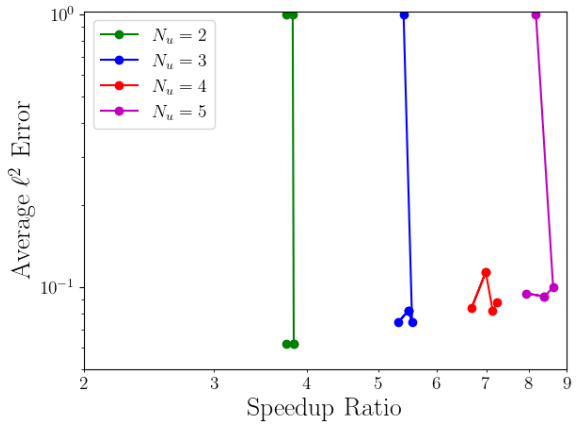



**Figure 7.12:** CVRC adaptive HPROM time-average error vs. computational speedup, $N_p = 5$, various $N_u$.

# Chapter 8

# Best Practices for PROMs of Reacting Flows

In the author's personal experience, there is often a certain opacity in the projection-based ROM literature on the subject of data preparation and robustness controls. In many cases this is simply due to the fact that many PROM studies deal with systems which are governed by a single state variable (Burgers' equation, Kuramoto–Sivashinsky equation), are governed by state variables that are naturally of similar magnitudes (shallow water equations, Euler equations for Sod shock tube), or are commonly non-dimensionalized in open-source or commercial solvers (incompressible Navier–Stokes). In such situations, the dimensionality of the state variables is either irrelevant or has little effect on the accuracy of state representations and the solution of the PROM equations. Sometimes, however, small but crucial details of data preparation or the PROM solution are left out as they are considered self-evident to the authors (as has been discovered anecdotally by this author and colleagues). To date, the work by Parish and Rizzi [120] provides the most explicit description and comprehensive study of ensuring a dimensionally-consistent POD formulation and PROM solution by careful choice of inner products. Although the focus of their work is the compressible Euler equations, they provides broadly-meaningful insights for dimensional dynamical systems.

All but one numerical experiment in this thesis deals with high-pressure, compressible reacting flows, which are characterized by both vastly disparate magnitudes in the system state and an inability to readily non-dimensionalize the governing equations. As such, careful data preparation and unsteady solution robustness controls are extremely important in enabling the stable and accurate solution of projection-based reduced-order

models. Further, this thesis aims to be as transparent about every element of the PROM construction process to help enable successful PROMs for future researchers working with similarly-complex systems. Some of the key elements in this process are detailed the numerical experiments which follow.

All results shown in this chapter are unsampled MP-LSVT PROMs of the truncated CVRC case investigated in Section 6.2. A time step of $\Delta \mathrm{t} = 5 \times \Delta \mathrm{t}_{\mathrm{FOM}}$ is used in all cases.

## 8.1 Centering and Scaling

As previously defined, the general (linear or non-linear) low-dimensional state representation for the conserved and target states are defined, respectively, as

$$\widetilde{\mathbf{q}}_c := \overline{\mathbf{q}}_c + \mathbf{H}_c \boldsymbol{\varphi}_{g,c}\left(\widehat{\mathbf{q}}_c\right), \tag{8.1}$$

$$\widetilde{\mathbf{q}}_p := \overline{\mathbf{q}}_p + \mathbf{H}_p \boldsymbol{\varphi}_{g,p}\left(\widehat{\mathbf{q}}_p\right), \tag{8.2}$$

where, again, the functions $\boldsymbol{\varphi}_{g,c}\left(\widehat{\mathbf{q}}_c\right) = \mathbf{U}_c \widehat{\mathbf{q}}_c$ and $\boldsymbol{\varphi}_{g,p}\left(\widehat{\mathbf{q}}_p\right) = \mathbf{U}_p \widehat{\mathbf{q}}_p$ for a linear trial space. This decomposition by centering ($\overline{\mathbf{q}}_c$, $\overline{\mathbf{q}}_p$) and the scaling ($\mathbf{H}_c$, $\mathbf{H}_p$) operation is often referred to as *feature scaling* in the machine learning community, whereby the training datasets are constructed (as noted in Eqs 3.6 and 3.42) by

$$\mathbf{Q}'_c = \left[\mathbf{H}_c^{-1}\left[\mathbf{q}_c\left(\mathrm{t}^0\right) - \overline{\mathbf{q}}_c\right], \ \ldots \ , \ \mathbf{H}_c^{-1}\left[\mathbf{q}_c\left(\mathrm{T}\right) - \overline{\mathbf{q}}_c\right]\right], \tag{8.3}$$

$$\mathbf{Q}'_p = \left[\mathbf{H}_p^{-1}\left[\mathbf{q}_p\left(\mathrm{t}^0\right) - \overline{\mathbf{q}}_c\right], \ \ldots \ , \ \mathbf{H}_p^{-1}\left[\mathbf{q}_p\left(\mathrm{T}\right) - \overline{\mathbf{q}}_p\right]\right]. \tag{8.4}$$

The choice of $\overline{\mathbf{q}}_c/\mathbf{H}_c$ and $\overline{\mathbf{q}}_p/\mathbf{H}_p$ has a measurable influence on the accuracy of the above approximations, particularly for variables of extremely disparate magnitudes. Before demonstrating this fact, several popular methods of centering and scaling are outlined and compared.

**Centering**

With the exception of centering for min-max scaling, all centering methods described here are considered to be spatially-variant, i.e. $\overline{\mathbf{q}}_c := \overline{\mathbf{q}}_c(\mathbf{x})$, $\overline{\mathbf{q}}_p := \overline{\mathbf{q}}_p(\mathbf{x})$. Each is described in turn.

1. *Initial condition*: Centering about the initial condition, i.e. $\overline{\mathbf{q}}_c = \mathbf{q}_c(\mathrm{t}^0)$ or $\overline{\mathbf{q}}_p = \mathbf{q}_p(\mathrm{t}^0)$, results in approximation of the unsteady state as perturbations about the initial condition. In the case of a linear trial space, this guarantees exact satisfaction of the initial conditions, as the projection of the zero vector (the initial condition subtracted by itself) is identically zero. In the case of autoencoder non-linear manifold methods, however, this is merely satisfied approximately, and near-satisfaction is encouraged by including the zero vector in the training set and initializing the non-linear manifold PROM from the encoding of the zero vector, as in [116].

2. *Mean*: The mean field centering computes the centering vector as the arithmetic mean of the data snapshots,

$$\overline{\mathbf{q}}_c = \frac{1}{N_\mathrm{T}} \sum_{n=1}^{N_\mathrm{T}} \mathbf{q}_c^n \tag{8.5}$$

$$\overline{\mathbf{q}}_p = \frac{1}{N_\mathrm{T}} \sum_{n=1}^{N_\mathrm{T}} \mathbf{q}_p^n \tag{8.6}$$

   This concept is fairly common in the turbulence modeling community, which often seeks to accurately describe unsteady perturbations about the time-averaged field for statistically-stationary flows. In the same sense, centering data snapshots about the mean field ensures that the low-dimensional representation accurately captures these small-scale fluctuations which would otherwise be dwarfed by the mean field.

**Scaling**

For all methods described below, the scaling matrices $\mathbf{H}_c$, $\mathbf{H}_p$ are composed of constant scalars which are specific to a given state variable (e.g., density, velocity) but are not

specific to spatial location. This can be written as as

$$\mathbf{H}_c := diag\left(\mathbf{h}_{c,1}^{\top},\ \ldots,\ \mathbf{h}_{c,N_v}^{\top}\right), \quad \mathbf{h}_{c,v}\left(\mathbf{x}_i\right) = h_{c,v}\ \forall\ i \in \{1,\ \ldots,\ N_e\} \tag{8.7}$$

$$\mathbf{H}_p := diag\left(\mathbf{h}_{p,1}^{\top},\ \ldots,\ \mathbf{h}_{p,N_v}^{\top}\right), \quad \mathbf{h}_{p,v}\left(\mathbf{x}_i\right) = h_{p,v}\ \forall\ i \in \{1,\ \ldots,\ N_e\} \tag{8.8}$$

where $\mathbf{h}_{c,v},\ \mathbf{h}_{p,v} \in \mathbb{R}$ is the constant conservative/target scaling value for the $v$ state variable.

1. *$\ell^2$-norm*: Scaling by the $\ell^2$-norm method is motivated by that proposed by Lumley and Poje [198], and is computed as

$$h_{c,v} = \frac{1}{N_{\mathrm{T}} N_e} \sum_{n=1}^{N_{\mathrm{T}}} \left\| \mathbf{q}_{c,v}^{n} - \overline{\mathbf{q}}_{c,v} \right\|^2 \tag{8.9}$$

$$h_{p,v} = \frac{1}{N_{\mathrm{T}} N_e} \sum_{n=1}^{N_{\mathrm{T}}} \left\| \mathbf{q}_{p,v}^{n} - \overline{\mathbf{q}}_{p,v} \right\|^2 \tag{8.10}$$

   This has the effect of ensuring that all state variable vectors have a length (in the Euclidean norm) close to unity. This is closely related to the POD, which computes the distance between the data and their projection in the $\ell^2$ norm.

2. *Min-max*: Min-max feature scaling, composed of associated centering and scaling operations, has the effect of ensuring that all values in the modified dataset fall in the range $[0.0,\ 1.0]$. For the $v$th state variable (e.g. density, velocity), the centering vector is computed as

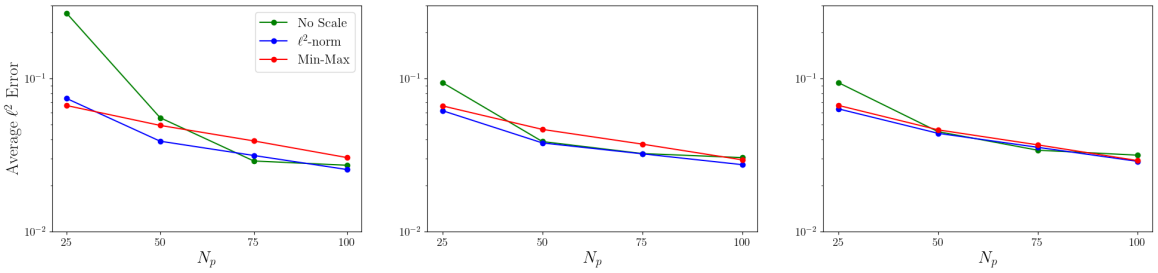
$$\overline{q}_{c,v} = min\left(\mathbf{Q}_{c,v}\right), \quad \overline{\mathbf{q}}_{c,v}\left(\mathbf{x}_i\right) = \overline{q}_{c,v}\ \forall\ i \in \{1,\ \ldots,\ N_e\} \tag{8.11}$$

$$\overline{q}_{p,v} = min\left(\mathbf{Q}_{p,v}\right), \quad \overline{\mathbf{q}}_{p,v}\left(\mathbf{x}_i\right) = \overline{q}_{p,v}\ \forall\ i \in \{1,\ \ldots,\ N_e\} \tag{8.12}$$

   where the data snapshot matrix for the $v$th state variable is given by

$$\mathbf{Q}_{c,v} := \left[\mathbf{q}_{c,v}\left(\mathrm{t}^0\right),\ \ldots,\ \mathbf{q}_{c,v}\left(\mathrm{T}\right)\right] \tag{8.13}$$

$$\mathbf{Q}_{p,v} := \left[\mathbf{q}_{p,v}\left(\mathrm{t}^0\right),\ \ldots,\ \mathbf{q}_{p,v}\left(\mathrm{T}\right)\right] \tag{8.14}$$




**(a)** No centering. **(b)** Initial condition. **(c)** Time average.

**Figure 8.1:** CVRC unsampled MP-LSVT PROM time-average error, $N_p = 25$, various trial space centerings.

The scaling values are then computed as

$$h_{c,v} = max\left(\mathbf{Q}_{c,v}\right) - min\left(\mathbf{Q}_{c,v}\right) \tag{8.15}$$

$$h_{p,v} = max\left(\mathbf{Q}_{p,v}\right) - min\left(\mathbf{Q}_{p,v}\right) \tag{8.16}$$

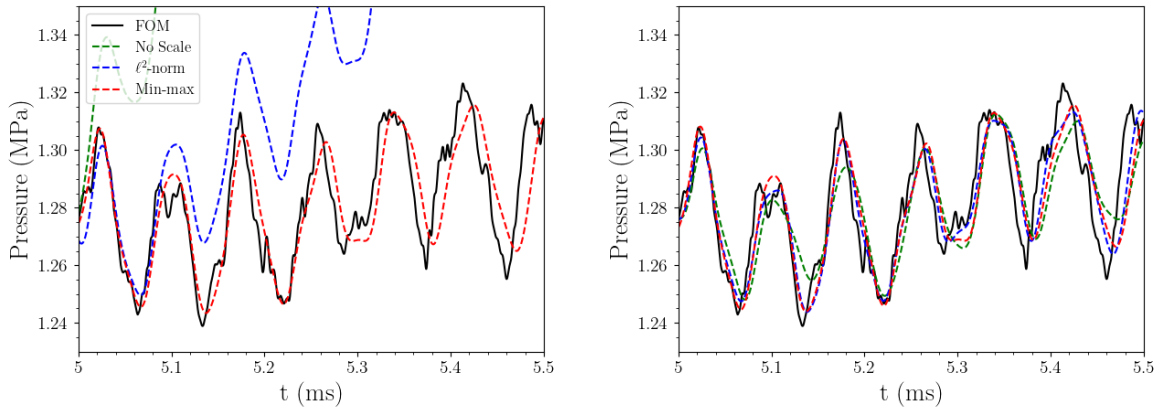
where the data matrices are given as in Eqs. 8.13 and 8.14. Note that this min-max scaling can be applied on top of an alternative centering method, e.g. centering by the initial condition and then min-max scaling. Such an operation for the conservative state would take the form

$$\mathbf{q}'_{c,v} = \frac{\mathbf{q}_{c,v} - \overline{\mathbf{q}}_c - min\left(\mathbf{Q}_{c,v}\right)}{max\left(\mathbf{Q}_{c,v}\right) - min\left(\mathbf{Q}_{c,v}\right)} \tag{8.17}$$

As such, min-max scaling will be referred to exclusively as a scaling operation which can be combined with a given centering method, or applied without alternative centering.

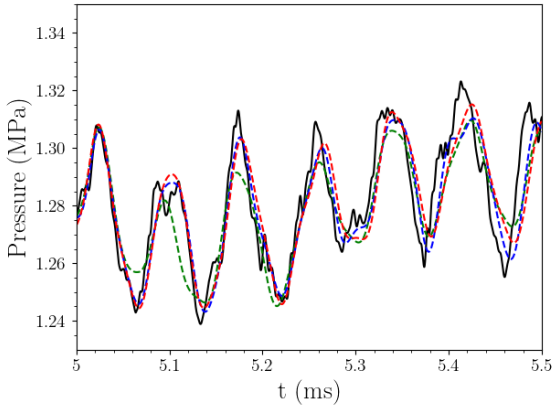
Although min-max scaling is fairly common within the machine learning community, and is a simple method of ensuring that the data are roughly the same order of magnitude, this scaling has the effect of emphasizing outliers which define the maximum and minimum bounds of the state variables.

Time-average error results for the MP-LSVT PROMs of the truncated CVRC, for $N_p = \{25,\ 50,\ 75,\ 100\}$, are shown in Fig. 8.1, along with indicative pressure probe measurements in Fig. 8.2. Both instances in which centering is applied (either initial condition




**(a)** No centering.

**(b)** Initial condition.




**(c)** Time average.

**Figure 8.2:** CVRC unsampled MP-LSVT PROM pressure probes, $N_p = 25$, various trial space centerings.

or time-average field) perform similarly, and scaling the state appears to improve the solution over cases without scaling. If no centering, however, the PROM accuracy is rather poor even in cases where scaling is also applied.

## 8.2 Residual Weighting

As discussed in the derivations of PROMs in Sections 3.2 and 3.3, practical engineering systems are often described by state variables of vastly different magnitudes. In the case of fluid flows in rocket combustors, density (and transported scalars) are usually $\mathcal{O}\left(1\text{-}10\right)$ kg/m$^3$, velocity (and momentum) are usually $\mathcal{O}\left(100\right)$ m/s (kg/m$^2$-s), temperatures are often $\mathcal{O}\left(1,000\right)$ K, and pressures are in excess of $\mathcal{O}\left(1\times10^6\right)$ Pa. As a result, it is often important to non-dimensionalize or precondition these systems to ensure proper linear solver convergence, as the dimensional system may be poorly conditioned.

A similar notion is relevant to LSPG and MP-LSVT PROMs, whose un-normalized formulation is given respectively by

$$\widehat{\mathbf{q}}_c^n = \underset{\mathbf{y}\in\mathbb{R}^{N_c}}{argmin} \ \left\|\mathbf{r}\left(\mathbf{y}\right)\right\|_2, \tag{8.18}$$

$$\widehat{\mathbf{q}}_p^n = \underset{\mathbf{y}\in\mathbb{R}^{N_p}}{argmin} \ \left\|\mathbf{r}\left(\mathbf{y}\right)\right\|_2, \tag{8.19}$$

where the formulation of the fully-discrete residual $\mathbf{r}\left(\cdot\right)$ with respect to the primitive and conservative latent variables is used interchangeably, but represents the same quantity. The solution of this non-linear least-squares problem by quasi-Newton methods may suffer from poor convergence, as high-magnitude elements of the residual may contribute disproportionately to the $\ell^2$-norm.

In light of this, as given in the formulations in Eqs. 3.28 and 3.48, a diagonal residual weighting matrix $\mathbf{H}_r \in \mathbb{R}^{N\times N}$ is introduced which modifies the PROM formulation to

weighted non-linear least-squares problems of the form

$$\widehat{\mathbf{q}}_c^n = \underset{\mathbf{y}\in\mathbb{R}^{N_c}}{argmin} \ \left\|\mathbf{H}_r^{-1}\mathbf{r}\left(\mathbf{y}\right)\right\|_2, \tag{8.20}$$

$$\widehat{\mathbf{q}}_p^n = \underset{\mathbf{y}\in\mathbb{R}^{N_p}}{argmin} \ \left\|\mathbf{H}_r^{-1}\mathbf{r}\left(\mathbf{y}\right)\right\|_2. \tag{8.21}$$

The operation $\mathbf{H}_r^{-1}$ thus seeks to normalize the residual terms such that they contribute approximately equally to the non-linear least-squares problem and improve iterative convergence. For all results in this thesis, $\mathbf{H}_r$ takes a similar form to that of the scaling matrices introduced in Section 8.1. That is, it is constructed from scalars which are constant for a single governing equation (e.g. mass conservation, total energy conservation), and do not vary for degrees of freedom associated with different mesh elements. This is written as

$$\mathbf{H}_r := diag\left(\mathbf{h}_{r,1}^\top,\ \ldots,\ \mathbf{h}_{r,N_v}^\top\right), \quad \mathbf{h}_{r,v}\left(\mathbf{x}_i\right) = h_{r,v} \ \forall\ i \in \{1,\ \ldots,\ N_e\} \tag{8.22}$$

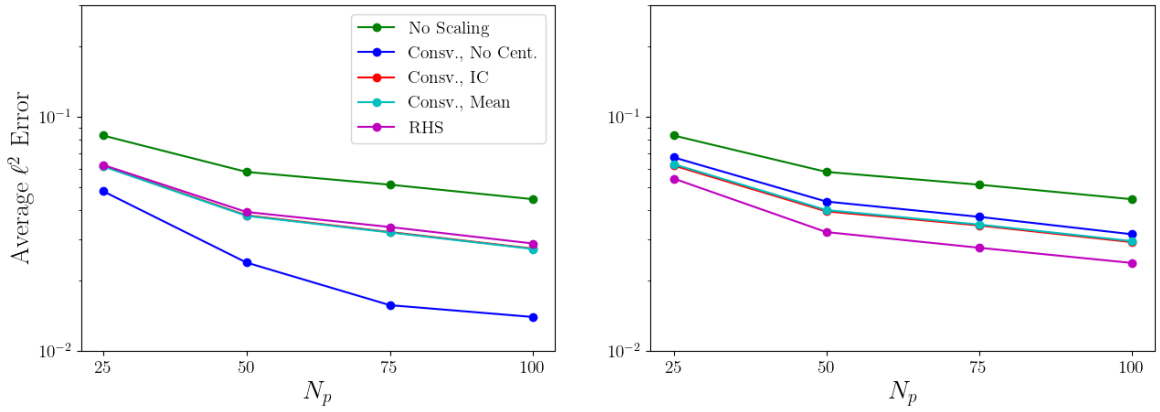
In this thesis, calculation of the residual weighting factors is predicated on the formulation of the PROM residual as

$$\mathbf{r}\left(\widehat{\mathbf{q}}^n\right) := \dot{\mathbf{q}}_c^n - \mathbf{f}\left(\mathbf{g}\left(\widehat{\mathbf{q}}^n\right),\ \mathrm{t}\right) = \mathbf{0}. \tag{8.23}$$

That is, the fully-discrete system is seperable into the time-integrator of the conservative state and a non-linear "right-hand side" (RHS) function $\mathbf{f}\left(\cdot,\ \mathrm{t}\right)$. Further, for the specific case of linear multi-step time integration schemes, for which the discretization of the time derivative takes the general form

$$\dot{\mathbf{q}}_c^n := \mathbf{q}_c^n + \sum_{i=1}^{s} a_i \mathbf{q}_c^{n-i} \tag{8.24}$$

As mentioned in Section 4.4, a consistent linear multi-step scheme satisfies $\sum_{i=1}^{s} a_i = -1$. Substituting the linear approximation of the conservative state, the discrete time




**(a)** $\ell^2$-norm. **(b)** Min-max.

**Figure 8.3:** CVRC unsampled MP-LSVT PROM time-average error, $N_p = 25$, various residual scalings.

derivative can be written as

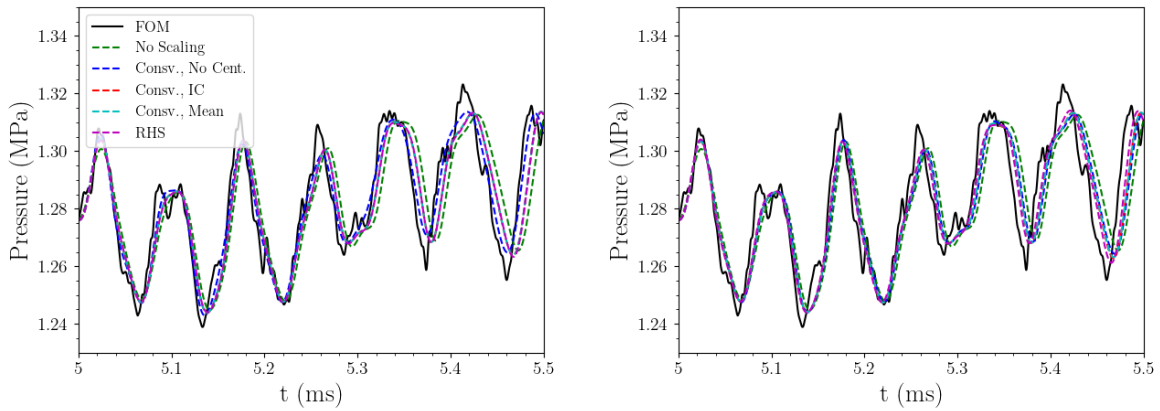
$$\dot{\mathbf{q}}_c^n = \mathbf{H}_c \mathbf{U}_c \left( \widehat{\mathbf{q}}_c^n + \sum_{i=1}^{s} a_i \widehat{\mathbf{q}}_c^{n-i} \right) \tag{8.25}$$

No similar formulation can be ascribed to the RHS function.

The above discussion motivates two primary approaches for computing the constant scaling factors $h_{r,v}$.

1. *Conservative state*: This approach computes a residual scaling identical to that of the conservative state scaling matrix, $\mathbf{H}_c$. As outlined in Section 8.1, this is associated with a particular centering and scaling method. Centering about the conservative initial condition and time-average field are investigated here, along with $\ell^2$-norm and min-max scaling.

2. *Right-hand side*: This approach computes the residual scaling from unsteady snapshots of the non-linear function $\mathbf{f}(\cdot,\ \mathrm{t})$. As mentioned above, it does not make sense to center these snapshots, as the linear multi-step schemes used in this thesis cannot be decomposed such that a centering vector cancels out. As such, the following results compute RHS scaling by $\ell^2$-norm and min-max scaling, without centering. The latter option neglects subtracting the field variable minimums.




**(a)** $\ell^2$-norm. **(b)** Min-max.

**Figure 8.4:** CVRC unsampled MP-LSVT PROM pressure probes, $N_p = 25$, various residual scalings.

MP-LSVT PROMs are again computed for the truncated CVRC. The primitive state is centered about the initial condition and scaled by the $\ell^2$-norm in all cases. Time-average error and probe measurements are again displayed in Fig. 8.3 and 8.4, respectively. Here, the effect of residual weighting is slightly more nuanced than that of solution centering and scaling. On the whole, all cases are reasonably accurate. However, certain choices of residual scaling have the effect of causing/preventing an apparent decrease in the frequency of the dominant pressure mode, making it appear as if the solution is lagging behind that of the FOM. In particular, it appears that computing the residual scaling via the un-centered conservative variables alleviates this problem to the greatest extent, and generally results in the lowest time-average error.

## 8.3 Limiters

As discussed at length in work by Huang *et al.* [73] and Blonigan *et al.* [199], PROMs for systems described by flow fields characterized by sharp gradients (such as shocks or flames) often generate non-physical solutions, such as negative densities, temperatures, or pressures. These inevitably cause the solver to fail in attempts to, say, compute the square root of a negative value. The cause of these issues in linear subspace PROMs is often due to "ringing," a common issue in linear representations of non-linear functions. Closely

related to Gibbs phenomena for Fourier series representations, ringing appears as large over- and undershoots in approximating sharp gradients, along with local oscillations in the vicinity. Severe ringing can result in non-physical solutions due to the aforementioned over- and undershoots. This is generally caused by under-resolution of the trial space, and usually improves with increasing $N_c/N_p$.

A variety of tactics for combatting high-frequency oscillatory behavior in PROMs has been proposed, including adaptations of classical artifical viscosity [200, 201] and filtering [202, 203] approaches to PROMs. Perhaps the simplest approach to eliminating non-physical solutions are limiting, or clipping, functions, which bound the solution from above, below, or both. This can be formalized by the limiting function

$$q = max\left(min\left(q,\ q_{max}\right),\ q_{min}\right) \tag{8.26}$$

The work by Blonigan *et al.* [199] on a hypersonic re-entry vehicle uses this approach to bound the density and temperature from below, ensuring that they never fall below zero and ensures a physical solution. The work by Huang *et al.* [73] on a model rocket combustor goes further, bounding temperature from below using a value slighly smaller than the injection temperature of the cold fuel, and bounding from above using a value slightly larger than the adiabatic flame temperature of the reactants injected. This latter approach limits the solution more aggressive, but is ultimately informed by the physical constraints of the modeled system. Additionally, this approach is shown to not only improve stability, but also improve the long-term accuracy of the online PROM. Later work by Huang and coworkers [74, 75] also explores similar heuristic limiters for species mass fraction fields; these are not discussed further in this thesis.

As an aside, the work by Blonigan *et al.* [199] classifies this limiting approach as a type of non-linear trial manifold, inasmuch the clipping function is non-linear and all realizable solutions are computed via this non-linear function. While this distinction is accurate, this thesis does not use such a label, as clipping functions are already commonly used in simulations of reacting flows to ensure that all species mass fractions sum to unity. Referring to this approach as a non-linear manifold PROM might unduly confuse

the reader with respect to the neural network non-linear trial manifolds discussed in Sec. 3.1.2.

## 8.4 Variable Transformations

Throughout the results presented in the above chapters, it is clear that the MP-LSVT method is capable of generating stable and accurate PROMs for reacting flow systems. Although the MP-LSVT is a theoretically simple modification of the well-established LSPG projection, the effect of this change on PROM performance is profound. Although the two methods performed equivalently for the non-reacting 2D transonic cavity flow in Section 6.1, MP-LSVT was shown in Section 6.2 to preserve PROM stability for the 3D CVRC while LSPG quickly became unstable. This has similarly been observed in the original development of the MP-LSVT method by Huang *et al.* [75].

The core concept of the variable transformation has other, more subtle advantages. For one, choosing an alternative set of target variables allows a PROM practitioner to more confidently estimate *a priori* the accuracy of approximating flow fields which are of more practical interest than the conservative variables. Calculations of POD bases or autoencoders can be appropriately weighted and trained to approximate certain fields with higher fidelity. For example, if the unsteady temperature field is of particular importance to an application, the training loss in approximating the temperature field can be scaled to ensure higher accuracy. This is not nearly as simple when training using the conservative variables, which may be involved in calculating various practical quantities of interest to varying degrees.

The use of an alternate set of target variables also enables vastly simpler implementations of limiters such as those discussed in Section 8.3. When the conservative state is modeled, the field to be limited must first be calculated from the conservative variables, and then the conservative fields recalculated to reflect the change in the limited field. For example, to limit species mass fractions, the mass fractions would first be computed from the density-weighted mass fractions, clip the mass fraction fields, then recalculate

the density-weighted mass fraction fields. This process incurs additional computational cost and modeling complexity which can be circumvented by directly modeling the field to be limited as part of the MP-LSVT target state.

For the above reasons, and due to its observed accuracy and robustness, the model preserving variable transformation can be considered a best practice in highly stiff problems such as reacting flows. Whether similar improvements can be realized for other dynamical systems remains to be seen. Further, a significant amount of work remains to determine whether there are additional target flow variable sets (e.g. entropy variables) which can result in even better PROMs than those investigated using the primitive variables.

## 8.5 Sample Selection Computations

As discussed in Section 4.5.2 and empirically tested in Section 6.2 as well as prior work by Wentland *et al.* [125], constructing a sampled mesh for HPROMs which results in a stable and accurate solution at low sampling rates can be extremely challenging. Traditional methods, such as GNAT sampling originally proposed by Carlberg *et al.* [68], as well as simple random sampling, appear to perform suboptimally for HPROMs of complex reacting flow systems. Eigenvector-based sampling, originally developed by Zimmermann and Willox [147] and simplified by Peherstorfer*et al* [76], has proven remarkably effective in enabling accurate HPROMs with very large computational cost savings. Based on findings here and in prior works [76, 125], eigenvector-based sampling seems a natural choice for ensuring HPROM effectiveness.

However, as demonstrated by offline cost measurements in Fig. 6.27, eigenvector-based sampling can account for non-negligible portion of the full computational budget even when utilizing scalable, high-performance linear algebra tools such as PLATFORM [132]. Distributed-memory software is generally a necessity when operating on datasets in excess of $\mathcal{O}(10)$ GB, but such methods require special considerations. In particular, greedy methods implemented in distributed-memory applications are hampered by repeated sorting of non-contiguous memory and blocking, global reductions which are required to de-

termine the minimum/maximum instance of the greedy sampling metric. Attempting to mitigate this communication overhead by limiting the total number of process invariably reduces the parallelism of evaluating the greedy metric. Hence, a delicate balancing act is required to enable efficient offline calculations.

One possible approach for mitigating the overall cost of the offline sampling calculations is suggested by the difference between the two GNAT sampling variants investigated in this thesis. In the original algorithm, $\texttt{ceil}(N_s/N_r)$ are selected at every greedy iteration, while the alternative algorithm selected $N_v$ at every greedy iteration. The latter is far more granular and approaches a more optimal solution, but incurs a much higher computational cost due to requiring potentially many more evaluations of the greedy metric. However, these approaches imply a spectrum, by which the number of samples to be selected is specified by the user at runtime. Smaller values should results in improved modeling accuracy, while higher values reduce offline costs. A compromise between these two effects may drastically reduce the computational burden of greedy sampling algorithms while ensuring stable and accurate HPROMs. Further, it may prove to be a useful approach in improving online sample mesh adaptation, where single-shot sampling may require exceptionally large sample meshes to ensure accuracy, as demonstrated in Chapter 7.

# Chapter 9

# Conclusion

This thesis investigated projection-based reduced-order models for multi-scale transport-dominated systems, with a particular emphasis on reacting flows and rocket combustor applications. Attention was given to the application of these ROMs for problems of much larger dimensions and complexity than has been attempted in the literature. The main insights drawn from these experiments are highlighted below, including particularly successful methods and practices, as well as challenges which frustrated attempts at producing stable, accurate, and efficient simulations. A few additional questions which might serve as interesting topics for future research are also presented.

## 9.1 Summary and Insights

Chapters 2–4 provided a detailed background on modern projection-based ROMs in the context of numerical models of compressible reacting flows. Classical projection methods, namely Galerkin and LSPG projection, were contrasted against the recent MP-LSVT method, which was later shown in this thesis to be superior to classical methods in modeling reacting flows. Important nuances in the development of scalable hyper-reduced PROMs of non-linear PDEs were detailed, including their theoretical formulation, several sample mesh selection algorithms, gappy POD regressor calculation, and load balancing. Outlines of online basis and sample mesh adaptation approaches were also provided.

Chapter 5 investigated a 1D freely-propagating, acoustically-forced model premixed flame. These calculations were enabled by PERFORM, an open-source Python frame-

work for developing novel ROMs for reacting flows. While the 1D model flame is an extremely simple model of advection-dominated reacting flow, intrusive linear PROMs failed outright in both reconstructing training data and predicting unseen data. The ability of a linear trial space in approximating sharp gradients and traveling waves was shown to be very poor. Alternatively, modern neural network approaches were shown to be capable of approximating the solution very accurately. Deep autoencoder neural networks enabled efficient dimension reduction with extremely small latent space dimensions, inducing projection error which was orders of magnitude lower than that observed for comparable linear trial spaces. However, the integration of these autoencoders in non-linear manifold PROMs proved less encouraging, generating solution trajectories for unseen outlet pressure forcing frequencies which often failed to accurately predict the upstream acoustic content or flame speed. Further, these non-linear manifold PROMs were exorbitantly computationally-expensive, due largely to the evaluation of neural network Jacobians by automatic differentiation. To mitigate this online cost, non-intrusive ROMs were presented using recurrent neural networks, specifically LSTMs, to model the time evolution of the latent variables. This circumvented the need to repeatedly evaluate the governing equations or compute Jacobians and projections, drastically cutting down online costs. Further, the non-intrusive ROMs generated excellent online predictions for training and unseen data alike. The apparent robustness and predictive accuracy of these non-intrusive ROMs are appealing, though the computational cost of training these neural network ROMs was shown to be exceptionally high.

Chapter 6 rigorously investigated the construction of accurate and scalable HPROMs for highly non-linear fluid flow systems, applying the techniques outlined in Chapter 4 to a 2D transonic flow over an open cavity, a 3D single-element model rocket combustor, and a multi-element laboratory rocket combustor. To the best of the author's knowledge, this last experiment represents the largest and one of the most physically-complex PROMs in the literature to date. The effects of the sampling algorithm, sample mesh size, and gappy POD regressor dimension were examined. In particular, these results revealed that traditional sampling algorithms, such as random sampling or GNAT sampling (an

extension of DEIM greedy sampling), were generally unable to generate significant computational speedups without inducing unacceptable loss of stability and accuracy. The eigenvector-based sampling proposed by Peherstorfer [76], on the other hand, was capable to constructing a sample mesh which enabled over three orders of magnitude cost savings while retaining excellent model accuracy. However, the greedy sampling algorithms were shown to incur significant offline computational cost, albeit only for relatively large sample meshes.

Chapter 7 investigated limits of PROMs in making future state predictions for the two 3D rocket combustor cases. In order to construct truly generalizable models, basis and sample mesh adaptation methods were applied to these problems. Successful future-state predictions were achieved, retaining stability and accuracy for time windows two orders of magnitude larger than the training datasets. The results presented were seen to offer an order of magnitude in cost savings, compared to many orders of magnitude savings achieved with static basis ROMs. This reveals a strong need for advanced online greedy sampling and load balancing techniques to minimize sample mesh sizes and MPI communication overhead.

Chapter 8 stepped back to take a big-picture examination of some of the underlying data preparation and online robustness control methods which enabled the successful results presented in previous chapters. All results presented were computed for the truncated single-element rocket combustor presented in Chapter 6. Several data centering and scaling methods were compared in generating accurate trial bases and subsequent online PROMs, showing that centering training datasets enhances accuracy significantly. Residual weighting, loosely associated with the more familiar concepts of non-dimensionalization and preconditioning, was also shown to be crucial in generating robust PROMs. In particular, computing residual scaling from uncentered snapshots of the conservative variables proved effective in enhancing long-term PROM accuracy. Finally, the importance of physics-informed temperature limiters, variable transformations, and efficient sample mesh computations was discussed, with recommendations for specific applications to reacting flow simulations.

The sum of these results imply that projection-based reduced-order models are very nearly capable of producing efficient and generalizable data-driven models for large-scale multi-physics systems. The MP-LSVT method, improved sampling algorithms, and online adaptation approaches have been shown here to be effective solutions to many of the challenges previously faced by the PROM community. However, this work has also exposed several key issues which remain to be solved, particularly in efficient online adaptation methods. Further, novel neural network ROM approaches do not appear to be a cost-effective solution to this model generalization problem, as they are shown to be onerously expensive to train even for simple one-dimensional reacting flow problems. On the whole, however, these results are greatly encouraging and suggest many interesting paths for future research, as detailed below.

## 9.2 Future Work

Developing data-driven models for reacting flows is a massive undertaking, and this thesis has only investigated a small portion of possible research paths. Based on my experience, I suggest a few key areas that might build off this work, restricting discussion to projection-based reduced-order models. Roughly in order of descending difficulty (in my opinion), these are:

1. PROMs with advanced chemistry and multi-phase models: The chemistry models utilized in this thesis are fairly rudimentary compared to those used by contemporary researchers in turbulent combustion. Further, the assumption of gaseous propellants ignores important spray physics in liquid-propellant rocket engines. Determining the sensitivity of PROMs in predicting complex combustion physics, such as radiation, soot generation, wall heat transfer, and spray atomization is a logical step in advancing PROMs to practical engineering systems including gas turbines, automotive engines, and jet engines.

2. PROMs for cylindrical multi-element rocket engines: A cylindrical single-element combustor and a linear multi-element combustor were investigated in this thesis.

In both, dominant system acoustics act largely in one direction. Cylindrical multi-element combustors, which are standard for industrial rocket engines, often experience complex rotational acoustic modes. The ability of PROMs to accurately model interactions between propellant injectors arranged in such configurations would help establish their utility for practical applications, rather than for the laboratory-scale experiments presented here.

3. Online adaptive sampling algorithms: An extremely naive sampling metric for online sample mesh adaptation was used for the adaptive HPROMs presented in Chapter 7, simply measuring the discrepancy the predicted FOM and PROM state at a given iteration. Although this approach was capable of making accurate future-state predictions, it required very large sample meshes to achieve a stable solution, and hence produced minimal computational cost savings. As seen is Chapter 6, alternative greedy algorithms are capable of producing accurate reconstructions on small sample meshes, but are far too computationally-expensive to compute during online calculations. Developing efficient sampling adaptation metrics will be crucial for enabling extremely fast adaptive HPROMs.

4. Comparison of linear adaptive PROMs and non-linear manifold ROMs: The key difficulty in modeling the one-dimensional model premixed flame in Chapter 5 with a linear representation was the poor approximation of sharp gradients under strong advection. Although non-linear manifold ROMs drastically improved this representation, they are extremely expensive to train. As linear adaptive PROMs incur negligible training cost, but have been shown to accurately predict advection-dominated flows [78], a direct comparison against non-linear manifold ROMs might reveal that an adaptive linear representation is both accurate and cost-effective.

5. Cost accounting for large-scale ML ROMs: A very superficial cost accounting for a few non-linear manifold ROMs, including offline training and online evaluation time, was presented for a simple one-dimensional model premixed flame in Chapter 5. Even for such a low-dimensional system, the offline cost is orders of magnitude

greater than that for linear PROMs. This would only be exacerbated for larger, more practical engineering systems, where the high training cost of deep neural networks in increased exponentially due to the curse of dimensionality. An honest evaluation of this computational burden might inform future developments in data-driven ROMs.

# Bibliography


[1] SpaceX. *Falcon User's Guide*. https://www.spacex.com/media/falcon_users_guide_042020.pdf. Apr. 2020.

[2] *GEnx Engine*. https://www.geaerospace.com/propulsion/commercial/genx. Accessed: 2022-10-07.

[3] George P. Sutton. "History of liquid propellant rocket engines in the United States". In: *Journal of Propulsion and Power* 19.6 (2003), pp. 978–1007. DOI: 10.2514/2.6942.

[4] US Defense Logistics Agency. *Aerospace Energy Standard Prices for DOD Customers Effective 1 Oct 2017*. https://www.dla.mil/Portals/104/Documents/Energy/Standard%20Prices/Aerospace%20Prices/E_2017Oct1AerospaceStandardPrices_170913.pdf?ver=2017-09-13-145335-477. Sept. 2017.

[5] NASA. *Liquid Propellant Rocket Combustion Instability*. Ed. by David T. Harrje. 1972.

[6] Otto K Goetz and Jan C. Monk. "Combustion Device Failures During Space Shuttle Main Engine Development". In: *5th International Symposium on Liquid Space Propulsion*. 2003.

[7] *Saturation Properties for Hydrogen - Pressure Increments*. https://webbook.nist.gov/cgi/fluid.cgi?Action=Load&ID=C1333740&Type=SatT&Digits=5&PLow=.5&PHigh=1.5&PInc=.1&RefState=DEF&TUnit=K&PUnit=atm&DUnit=kg/m3&HUnit=kJ/mol&WUnit=m/s&VisUnit=uPa*s&STUnit=N/m. Accessed: 2022-10-07.

[8] Michael R. Orth, Christopher Vodney, Thomas Liu, William Z. Hallum, Timothee Pourpoint, and William E. Anderson. "Measurement of linear growth of self-excited instabilities in an idealized rocket combustor". In: *AIAA Aerospace Sciences Meeting, 2018*. 2018. DOI: 10.2514/6.2018-1185.

[9] Anthony Young. *The Saturn V F-1 Engine*. Ed. by David M. Harland. Praxis, 2008. ISBN: 9780387096292. DOI: 10.1007/978-0-387-09630-8.

[10] Joseph C. Oefelein and Yang Vigor. "Comprehensive review of liquid-propellant combustion instabilities in F-1 engines". In: *Journal of Propulsion and Power* 9.5 (1993), pp. 657–677. DOI: 10.2514/3.23674.

[11] Dan Zhao and X. Y. Li. "A review of acoustic dampers applied to combustion chambers in aerospace industry". In: *Progress in Aerospace Sciences* 74 (2015), pp. 114–130. DOI: 10.1016/j.paerosci.2014.12.003.

[12] Vigor Yang and William Anderson, eds. *Liquid Rocket Engine Combustion Instability*. American Institute of Aeronautic and Astronautics, 1995. ISBN: 1563471833. DOI: 10.2514/4.866371.

[13] F. E.C. Culick. "Some recent results for nonlinear acoustics in combustion chambers". In: *AIAA Journal* 32.1 (1994), pp. 146–169. DOI: 10.2514/3.11962.

[14] N. Noiray, D. Durox, T. Schuller, and S. Candel. "A unified framework for nonlinear combustion instability analysis based on the flame describing function". In: *Journal of Fluid Mechanics* 615 (2008), pp. 139–167. DOI: 10.1017/S0022112008003613.

[15] Tory Bruno (@torybruno). *All prior attempts at a space launch class methane engine failed due to intractable combustion instability or other issues.* Jan. 25, 2021. eprint: Twitter. URL: https://twitter.com/torybruno/status/1353785512234987523.

[16] Scott M. Murman, Michael J. Aftosmis, and Stuart E. Rogers. "Characterization of Space Shuttle Ascent Debris Aerodynamics Using CFD Methods". In: *43rd AIAA Aerospace Sciences Meeting and Exhibit*. 2005. DOI: 10.2514/6.2005-1223.

[17] Toshio Kobayashi and Kozo Kitoh. "A Review of CFD Methods and Their Application to Automobile Aerodynamics". In: *SAE Technical Papers* (1992). DOI: 10.4271/920338.

[18] S. R. Shah, S. V. Jain, R. N. Patel, and V. J. Lakhera. "CFD for centrifugal pumps: A review of the state-of-the-art". In: *Procedia Engineering* 51 (2013), pp. 715–720. DOI: 10.1016/j.proeng.2013.01.102.

[19] Jonathon Sumner, Christophe Sibuet Watters, and Christian Masson. "CFD in Wind Energy: The Virtual, Multiscale Wind Tunnel". In: *Energies* 3 (2010), pp. 989–1013. DOI: 10.3390/en3050989.

[20] Giancarlo Alfonsi. "Reynolds-Averaged Navier-Stokes Equations for Turbulence Modeling". In: *Applied Mechanics Reviews* 62 (2009). DOI: 10.1115/1.3124648.

[21] P. J. Mason. "Large-eddy simulation: A critical review of the technique". In: *Quarterly Journal of the Royal Meteorological Society* 120 (1994), pp. 1–26. DOI: 10.1002/qj.49712051503.

[22] Pietro Catalano and Marcello Amato. "An evaluation of RANS turbulence modelling for aerodynamic applications". In: *Aerospace Science and Technology* 7 (2003), pp. 493–509. DOI: 10.1016/S1270-9638(03)00061-0.

[23] Franck Nicoud, Hubert Baya Toda, Olivier Cabrit, Sanjeeb Bose, and Jungil Lee. "Using singular values to build a subgrid-scale model for large eddy simulations". In: *Physics of Fluids* 23 (2011). DOI: 10.1063/1.3623274.

[24] Gregory P. Smith, David M. Golden, Michael Frenklach, Nigel W. Moriarty, Boris Eiteneer, Mikhail Goldenberg, C. Thomas Bowman, Ronald K. Hanson, Soonho Song, William C. Gardiner, Jr., Vitali V. Lissianski, and Zhiwei Qin. *GRI-Mech 3.0*. http://www.me.berkeley.edu/gri_mech/.

[25] C. J. Sung, C. K. Law, and J. Y. Chen. "An augmented reduced mechanism for methane oxidation with comprehensive global parametric validation". In: *27th International Symposium on Combustion*. 1998, pp. 295–304. DOI: 10.1016/S0082-0784(98)80416-5.

[26] Westbrook, Charles K. and Dryer, Frederick L. "Chemical kinetic modeling of hydrocarbon combustion". In: *Progress in Energy and Combustion Science* 10 (1984), pp. 1–57. DOI: 10.1016/0360-1285(84)90118-7.

[27] Tianfeng F. Lu and Chung K. Law. "Strategies for mechanism reduction for large hydrocarbons: n-heptane". In: *Combustion and Flame* 154 (2008), pp. 153–163. DOI: 10.1016/j.combustflame.2007.11.013.

[28] Marc T. Henry de Frahan, Jon S. Rood, Marc S. Day, Hariswaran Sitaraman, Shashank Yellapantula, Bruce A. Perry, Ray W. Grout, Ann Almgren, Weiqun Zhang, John B. Bell, and Jacqueline H. Chen. "PeleC: An adaptive mesh refinement solver for compressible reacting flows". In: *International Journal of High Performance Computing Applications* (2022). DOI: 10.1177/10943420221121151.

[29] Suo Yang. "Effects of Detailed Finite Rate Chemistry in Turbulent Combustion". PhD thesis. Georgia Institute of Technology, 2017.

[30] A. R. Masri. "Turbulent Combustion of Sprays: From Dilute to Dense". In: *Combustion Science and Technology* 188.10 (2016), pp. 1619–1639. DOI: 10.1080/00102202.2016.1198788.

[31] Michael F. Modest and Daniel C. Haworth. *Radiative Heat Transfer in Turbulent Combustion Systems*. Springer, 2016. ISBN: 9783319272917. DOI: 10.1007/978-3-319-27291-7.

[32] Hamid Omidvarborna, Ashok Kumar, and Dong Shik Kim. "Recent studies on soot modeling for diesel combustion". In: *Renewable and Sustainable Energy Reviews* 48 (2015), pp. 635–647. DOI: 10.1016/j.rser.2015.04.019.

[33] Norbert Peters. *Turbulent Combustion*. Cambridge University, 2000. DOI: 10.1017/CBO9780511612701.

[34] Charles D. Pierce. "Progress-variable Approach for Large-eddy Simulation of Turbulent Combustion". PhD thesis. Stanford University, 2001.

[35] O. Colin, F. Ducros, D. Veynante, and T. Poinsot. "A thickened flame model for large eddy simulations of turbulent premixed combustion". In: *Physics of Fluids* 12.7 (2000), pp. 1843–1863. DOI: 10.1063/1.870436.

[36] S. B. Pope. "PDF methods for turbulent reactive flows". In: *Progress in Energy and Combustion Science* 11 (1985), pp. 119–192. DOI: 10.1016/0360-1285(85)90002-4.

[37] W. P. Jones and S. Navarro-Martinez. "Large eddy simulation of autoignition with a subgrid probability density function method". In: *Combustion and Flame* 150 (2007), pp. 170–187. DOI: 10.1016/j.combustflame.2007.04.003.

[38] Edward Knudsen, Shashank, and Heinz Pitsch. "Modeling partially premixed combustion behavior in multiphase LES". In: *Combustion and Flame* 162 (2015), pp. 159–180. DOI: 10.1016/j.combustflame.2014.07.013.

[39] John D. Kelleher and Brendan Tierney. *Data Science*. MIT Press, 2018. ISBN: 9780262347020. DOI: 10.7551/mitpress/11140.001.0001.

[40] Bonnie J. McBride, Sanford Gordon, and Martin A. Reno. *Coefficients for Calculating Thermodynamic and Transport Properties of Individual Species*. Tech. rep. 1993.

[41] Tianfeng Lu and Chung K. Law. "A directed relation graph method for mechanism reduction". In: *Proceedings of the Combustion Institute* 30 (2005), pp. 1333–1341. DOI: 10.1016/j.proci.2004.08.145.

[42] Hai Wang, Rui Xu, Kun Wang, Craig T Bowman, Ronald K Hanson, David F Davidson, Kenneth Brezinsky, and Fokion N Egolfopoulos. "A physics-based approach to modeling real-fuel combustion chemistry – I . Evidence from experiments , and thermodynamic , chemical kinetic and statistical considerations". In: *Combustion and Flame* 193 (2018), pp. 502–519. DOI: 10.1016/j.combustflame.2018.03.019.

[43] T Schuller, S Ducruix, D Durox, and S Candel. "Modeling Tools for the Prediction of Premixed Flame Transfer Functions". In: *Proceedings of the Combustion Institute*. Vol. 29. 2002, pp. 107–113. DOI: 10.1016/S1540-7489(02)80018-9.

[44] Gowtham Manikanta Reddy Tamanampudi, Swanand Sardeshmukh, William Anderson, and Cheng Huang. "Combustion instability modeling using multi-mode flame transfer functions and a nonlinear Euler solver". In: *International Journal of Spray and Combustion Dynamics* 12 (2020). DOI: 10.1177/1756827720950320.

[45] Patrick Jenny, Dirk Roekaerts, and Nijso Beishuizen. "Modeling of turbulent dilute spray combustion". In: *Progress in Energy and Combustion Science* 38 (2012), pp. 846–887. DOI: 10.1016/j.pecs.2012.07.001.

[46] Gal Berkooz, Philip Holmes, and John L Lumley. "The Proper Orthogonal Decomposition in the Analysis of Turbulent Flows". In: *Annual Review of Fluid Mechanics* 25 (1993), pp. 539–575. DOI: 10.1146/annurev.fl.25.010193.002543.

[47] Serkan Gugercin and Athanasios C Antoulas. "A Survey of Model Reduction by Balanced Truncation and Some New Results". In: *International Journal of Control* 77.8 (2004), pp. 748–766. DOI: 10.1080/00207170410001713448.

[48] Federico Negri Alfio Quarteroni Andrea Manzoni. *Reduced Basis Methods for Partial Differential Equations*. Springer, 2016. ISBN: 9783319154312. DOI: 10.1007/978-3-319-15431-2.

[49] Mark A Kramer. "Nonlinear Principal Component Analysis Using Autoassociative Neural Networks". In: *AIChE Journal* 37.2 (1991), pp. 233–243. DOI: 10.1002/aic.690370209.

[50] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. "Nonlinear Component Analysis as a Kernel Eigenvalue Problem". In: *Neural Computation* 10.5 (1998), pp. 1299–1319. DOI: 10.1162/089976698300017467.

[51] Christopher M Bishop, Markus Svensen, and Christopher K.I. Williams. "GTM: A Principled Alternative to the Self-Organizing Map". In: *Advances in Neural*

*Information Processing Systems*. 1997, pp. 354–360. DOI: 10.1007/3-540-61510-5_31.

[52] Marko Budisic, Ryan Mohr, and Igor Mezic. "Applied Koopmanism". In: *Chaos* 22 (2012). DOI: 10.1063/1.4772195.

[53] Peter J. Schmid. "Dynamic mode decomposition of numerical and experimental data". In: *Journal of Fluid Mechanics* 656 (2010), pp. 5–28. DOI: 10.1017/S0022112010001217.

[54] Benjamin Peherstorfer and Karen Willcox. "Data-driven operator inference for nonintrusive projection-based model reduction". In: *Computer Methods in Applied Mechanics and Engineering* 306 (2016), pp. 196–215. DOI: 10.1016/j.cma.2016.03.025.

[55] Francisco J. Gonzalez and Maciej Balajewicz. "Deep convolutional recurrent autoencoders for learning low-dimensional feature dynamics of fluid systems". In: *pre-print* (2018). arXiv: 1808.01346 [math.DS].

[56] Jiayang Xu and Karthik Duraisamy. "Multi-level convolutional autoencoder networks for parametric prediction of spatio-temporal dynamics". In: *Computer Methods in Applied Mechanics and Engineering* 372 (2020). DOI: 10.1016/j.cma.2020.113379.

[57] César Quilodrán-Casas, Rossella Arcucci, Laetitia Mottet, Yike Guo, and Christopher Pain. "Adversarial autoencoders and adversarial LSTM for improved forecasts of urban air pollution simulations". In: *ICLR SimDL Workshop*. 2021. DOI: 10.48550/arXiv.2104.06297.

[58] Nadine Aubry, Philip Holmes, John L. Lumley, and Emily Stone. "The dynamics of coherent structures in the wall region of a turbulent boundary layer". In: *Journal of Fluid Mechanics* 192 (1988), pp. 115–173. DOI: 10.1017/S0022112088001818.

[59] 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". In: *Physics of Fluids A* 3 (1991), pp. 2337–2354. DOI: 10.1063/1.857881.

[60] Anil E. Deane and Catherine Mavriplis. "Low-dimensional description of the dynamics in separated flow past thick airfoils". In: *AIAA Journal* 32.6 (1994), pp. 1222–1227. DOI: 10.2514/3.12123.

[61] W. Cazemier, R. W.C.P. Verstappen, and A. E.P. Veldman. "Proper orthogonal decomposition and low-dimensional models for driven cavity flows". In: *Physics of Fluids* 10.7 (1998), pp. 1685–1699. DOI: 10.1063/1.869686.

[62] T. Bui-Thanh, K. Willcox, O. Ghattas, and B. van Bloemen Waanders. "Goal-oriented, model-constrained optimization for reduction of large-scale systems". In: *Journal of Computational Physics* 224 (2007), pp. 880–896. DOI: 10.1016/j.jcp.2006.10.026.

[63] D. Rempfer. "On low-dimensional Galerkin models for fluid flow". In: *Theoretical and Computational Fluid Dynamics* 14 (2000), pp. 75–88. DOI: 10.1007/s001620050131.

[64] Clarence W Rowley, Tim Colonius, and Richard M Murray. "Model reduction for compressible flows using POD and Galerkin projection". In: *Physica D* 189 (2004), pp. 115–129. DOI: 10.1016/j.physd.2003.03.001.

[65] Irina Kalashnikova and Mathew F. Barone. "Stable and Efficient Galerkin Reduced Order Models for Non-Linear Fluid Flow". In: *6th AIAA Theoretical Fluid Mechanics Conference* (2011). DOI: 10.2514/6.2011-3110.

[66] Sebastian Grimberg, Charbel Farhat, and Noah Youkilis. "On the stability of projection-based model order reduction for convection-dominated laminar and turbulent flows". In: *Journal of Computational Physics* 419 (2020). DOI: 10.1016/j.jcp.2020.109681.

[67] Kevin Carlberg, Charbel Bou-Mosleh, and Charbel Farhat. "Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive

tensor approximations". In: *International Journal for Numerical Methods in Engineering* 86 (2011), pp. 155–181. DOI: 10.1002/nme.

[68] Kevin Carlberg, Charbel Farhat, Julien Cortial, and David Amsallem. "The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows". In: *Journal of Computational Physics* 242 (2013), pp. 623–647. DOI: 10.1016/j.jcp.2013.02.028.

[69] Kevin Carlberg, Matthew Barone, and Harbir Antil. "Galerkin v. least-squares Petrov–Galerkin projection in nonlinear model reduction". In: *Journal of Computational Physics* 330 (2017), pp. 693–734. DOI: 10.1016/j.jcp.2016.10.033.

[70] Sebastian Grimberg, Charbel Farhat, Radek Tezaur, and Charbel Bou-Mosleh. "Mesh sampling and weighting for the hyperreduction of nonlinear Petrov-Galerkin reduced-order models with local reduced-order bases". In: *International Journal for Numerical Methods in Engineering* 122 (2021), pp. 1846–1874. DOI: 10.1002/nme.6603.

[71] Cheng Huang, Jiayang Xu, Karthik Duraisamy, and Charles L. Merkle. "Exploration of reduced order models for rocket combustion applications". In: *AIAA Aerospace Sciences Meeting* (2018). DOI: 10.2514/6.2018-1183.

[72] Cheng Huang, Karthik Duraisamy, and Charles L. Merkle. "Challenges in reduced order modeling of reacting flow". In: *AIAA Propulsion and Energy Forum* (2018). DOI: 10.2514/6.2018-4675.

[73] Cheng Huang, Karthik Duraisamy, and Charles L. Merkle. "Investigations and improvement of robustness of reduced-order models of reacting flow". In: *AIAA Scitech Forum* (2019). DOI: 10.2514/6.2019-2012.

[74] Cheng Huang, Karthik Duraisamy, and Charles Merkle. "Data-Informed Species Limiters for Local Robustness Control of Reduced-Order Models of Reacting Flow". In: *AIAA Scitech Forum* (2020). DOI: 10.2514/6.2020-2141.

[75] Cheng Huang, Christopher R. Wentland, Karthik Duraisamy, and Charles Merkle. "Model reduction for multi-scale transport problems using model-form preserving least-squares projections with variable transformation". In: *Journal of Computational Physics* 448 (2022). DOI: 10.1016/j.jcp.2021.110742.

[76] Benjamin Peherstorfer, Serkan Gugercin, and Zlatko Drmač. "Stability of discrete empirical interpolation and gappy proper orthogonal decomposition with randomized and deterministic sampling points". In: *SIAM Journal on Scientific Computing* 42.5 (2020), pp. 2837–2864. DOI: 10.1137/19M1307391.

[77] Benjamin Peherstorfer. "Breaking the Kolmogorov Barrier with Nonlinear Model Reduction". In: *Notices of the American Mathematical Society* (2022). DOI: 10.1090/noti2475.

[78] Wayne Isaac Tan Uy, Christopher R Wentland, Cheng Huang, and Benjamin Peherstorfer. "Reduced models with nonlinear approximations of latent dynamics for model premixed flame problems". In: *pre-print* (2022). arXiv: 2209.06957v1 [math.NA].

[79] Nicholas Arnold-Medabalimi, Christopher R. Wentland, Cheng Huang, and Karthik Duraisamy. "Parallel Linear Algebra Tool FOr Reduced Modeling (PLATFORM)". In: *SoftwareX* 21 (2023). DOI: 10.1016/j.softx.2023.101313.

[80] Ding Li, Sankaran Venkateswaran, Keramat Fakhari, and Charles L. Merkle. "Convergence Assessment of General Fluid Equations on Unstructured Hybrid Grids". In: *15th AIAA Computational Fluid Dynamics Conference*. 2001. DOI: 10.2514/6.2001-2557.

[81] Ding Li, Guoping Xia, Venkateswaran Sankaran, and Charles L Merkle. "Computational Framework for Complex Fluid Physics Applications". In: *Third International Conference on Computational Fluid Dynamics*. 2006.

[82] G. Xia, M. Harvazinski, W. Anderson, and C. L. Merkle. "Investigation of Modeling and Physical Parameters on Instability Prediction in a Model Rocket Combustor". In: *AIAA Joint Propulsion Conference* (2011). DOI: 10.2514/6.2011-6030.

[83] Cheng Huang, Rohan Gejji, William Anderson, Changjin Yoon, and Venkateswaran Sankaran. "Combustion Dynamics Behavior in a Single-Element Lean Direct Injection (LDI) Gas Turbine Combustor". In: *AIAA Joint Propulsion Conference*. 2014. DOI: 10.2514/6.2014-3433.

[84] Adam L. Comer, Cheng Huang, Brent A. Rankin, Matthew E. Harvazinski, and Venkateswaran Sankaran. "Modeling and simulation of bluff body stabilized turbulent premixed flames". In: *AIAA Scitech Forum* (2016). DOI: 10.2514/6.2016-1936.

[85] C. R. Wilke. "A Viscosity Equation for Gas Mixtures". In: *The Journal of Chemical Physics* 18.4 (1950), pp. 517–519. DOI: 10.1063/1.1747673.

[86] S. Mathur, P. K. Tondon, and S. C. Saxena. "Thermal conductivity of binary, ternary and quaternary mixtures of rare gases". In: *Molecular Physics* 12.6 (1967), pp. 569–579. DOI: 10.1080/00268976700100731.

[87] J.H. Burgoyne and F. Weinberg. "A method of analysis of a plane combustion wave". In: *Symposium (International) on Combustion* 4 (1952), pp. 294–302.

[88] William Sutherland. "The viscosity of gases and molecular force". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 36.223 (1893), pp. 507–531. DOI: 10.1080/14786449308620508.

[89] A Svehla. *Transport Chemical Coefficients Equilibrium for the NASA Program Lewis*. Tech. rep. 1995.

[90] Charles F. Curtiss and Joseph O. Hirschfelder. "Transport Properties of Multicomponent Gas Mixtures". In: *The Journal of Chemical Physics* 17.6 (1949), pp. 550–555. DOI: 10.1063/1.1747319.

[91] Bruce E. Poling, John M. Prausnitz, and John P. O'Connell. *The Properties of Gases and Liquid*. McGraw-Hill, 2001. ISBN: 0070116822.

[92] Philip D. Neufeld, A. R. Janzen, and R. A. Aziz. "Empirical equations to calculate 16 of the transport collision integrals Ω(l,8)* for the lennard-jones (12-6) potential". In: *The Journal of Chemical Physics* 57.3 (1972), pp. 1100–1102. DOI: 10.1063/1.1678363.

[93] Rober J. Kee, Graham Dixon-Lewis, Jurgen Warnatz, Michael E. Coltrin, James A. Miller, and Harry K. Moffat. *A FORTRAN Computer Code Package for the Evaluation of Gas-phase, Multicomponent Transport Properties*. Tech. rep. 1998.

[94] Stephen B. Pope. *Turbulent Flows*. Cambridge University Press, 2000. DOI: 10.1017/CBO9780511840531.

[95] Jay P. Boris. "On Large Eddy Simulation Using Subgrid Turbulence Models, Comment 1". In: *Whither Turbulence? Turbulence at the Crossroads*. 1989, pp. 344–353. DOI: 10.1007/3-540-52535-1_53.

[96] D. H. Porter, A. Pouquet, and P. R. Woodward. "Kolmogorov-like spectra in decaying three-dimensional supersonic flows". In: *Physics of Fluids* 6 (1994), pp. 2133–2142. DOI: 10.1063/1.868217.

[97] F. F. Grinstein and C. Fureby. "LES studies of the flow in a swirl gas combustor". In: *Proceedings of the Combustion Institute* 30 (2005), pp. 1791–1798. DOI: 10.1016/j.proci.2004.08.082.

[98] Rickard E. Bensow and Göran Bark. "Implicit LES Predictions of the Cavitating Flow on a Propeller". In: *Journal of Fluids Engineering* 132 (2010). DOI: 10.1115/1.4001342.

[99] Fernando F. Grinstein, Len G. Margolin, and William J. Rider, eds. *Implicit Large Eddy Simulation: Computing Turbulent Fluid Dynamics*. Cambridge University Press, 2007. DOI: 10.1017/CBO9780511618604.

[100] C.N. Hinshelwood. "On the Theory of Unimolecular Reactions". In: *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences* 113.763 (1926), pp. 230–233.

[101] R.G. Gilbert, K. Luther, and J. Troe. "Theory of Thermal Unimolecular Reactions in the Fall-off Range. II. Weak Collision Rate Constants". In: *Berichte der Bunsengesellschaft für physikalische Chemie* 87 (1983), pp. 169–177. DOI: 10.1002/bbpc.19830870218.

[102] N Peters. "Laminar Diffusion Flamelet Models in Non-premixed Turbulent Combustion". In: *Progress in Energy and Combustion Science* 10 (1984), pp. 319–339. DOI: 10.1016/0360-1285(84)90114-X.

[103] Heinz Pitsch. *FlameMaster: a C++ Computer Program for 0D Combustion and 1D Laminar Flame Calculations*. URL: https://www.itv.rwth-aachen.de/en/downloads/flamemaster/.

[104] P. L. Roe. "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes". In: *Journal of Computational Physics* 43 (1981), pp. 357–372. DOI: 10.1016/0021-9991(81)90128-5.

[105] Curtis R Mitchell. "Improved Reconstruction Schemes for the Navier-Stokes Equations on Unstructured Meshes". In: *AIAA Aerospace Sciences Meeting*. 1994. DOI: 10.2514/6.1994-642.

[106] Russ D. Rausch, John T. Batinat, and Henry T.Y. Yang. "Spatial Adaptation Procedures on Unstructured Meshes for Accurate Unsteady Aerodynamic Flow Computation". In: *AIAA Structures, Structural Dynamics, and Materials*. 1991. DOI: 10.2514/6.1991-1106.

[107] Timothy J. Barth and Dennis C. Jespersen. "The Design and Application of Upwind Schemes on Unstructured Meshes". In: *AIAA Aerospace Sciences Meeting*. 1989. DOI: 10.2514/6.1989-366.

[108] Shishir A. Pandya, Sankaran Venkateswaran, and Thomas H. Pulliam. "Implementation of preconditioned dual-time procedures in OVERFLOW". In: *AIAA Aerospace Sciences Meeting*. 2003. DOI: 10.2514/6.2003-72.

[109] R. W. MacCormack. "Current Status of Numerical Solutions of the Navier-Stokes Equations". In: *AIAA Aerospace Sciences Meeting*. 1985. DOI: 10.2514/6.1985-32.

[110] Peter Benner, Serkan Gugercin, and Karen Willcox. "A survey of projection-based model reduction methods for parametric dynamical systems". In: *SIAM Review* 57.4 (2015), pp. 483–531. DOI: 10.1137/130932715.

[111] Benjamin Stamm Jan S. Hesthaven Gianluigi Rozza. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. Springer, 2016. ISBN: 9783319224701. DOI: 10.1007/978-3-319-22470-1.

[112] Lawrence Sirovich. "Turbulence and the Dynamics of Coherent Structures Part I: Coherent Structures". In: *Quarterly of Applied Mathematics* 45.3 (1987), pp. 561–571. DOI: 10.1090/qam/910462.

[113] N. Halko, P. G. Martinsson, and J. A. Tropp. "Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions". In: *SIAM Review* 53.2 (2011), pp. 217–288. DOI: 10.1137/090771806.

[114] Shane A. McQuarrie, Cheng Huang, and Karen E. Willcox. "Data-driven reduced-order models via regularised Operator Inference for a single-injector combustion process". In: *Journal of the Royal Society of New Zealand* (2021). DOI: 10.1080/03036758.2020.1863237.

[115] Allan Pinkus. *n-Widths in Approximation Theory*. Springer Berlin, Heidelberg, 1985. ISBN: 9783642698941. DOI: 10.1007/978-3-642-69894-1.

[116] Kookjin Lee and Kevin T. Carlberg. "Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders". In: *Journal of Computational Physics* 404 (2020), p. 108973. DOI: 10.1016/j.jcp.2019.108973.

[117] Sebastian Ruder. "An overview of gradient descent optimization algorithms". In: *pre-print* (2016). arXiv: 1609.04747 [cs.LG].

[118] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. "Language Models are Few-Shot Learners". In: *Advances in Neural Information Processing Systems*. 2020.

[119] William Fedus, Barret Zoph, and Noam Shazeer. "Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity". In: *Journal of Machine Learning Research* 23 (2022).

[120] Eric J Parish and Francesco Rizzi. "On the impact of dimensionally-consistent and physics-based inner products for POD-Galerkin and least-squares model reduction of compressible flows". In: *pre-print* (2022). arXiv: 2203.16492 [math.NA].

[121] Payton Lindsay, Jeffrey Fike, Irina Tezaur, and Kevin Carlberg. "Preconditioned Least-Squares Petrov-Galerkin Reduced Order Models". In: *pre-print* (2022). arXiv: 2203.12180 [math.NA].

[122] Sebastian Grimberg and Charbel Farhat. "Hyperreduction of CFD models of turbulent flows using a machine learning approach". In: *AIAA Scitech Forum*. 2020. DOI: 10.2514/6.2020-0363.

[123] Kevin Carlberg, Youngsoo Choi, and Syuzanna Sargsyan. "Conservative model reduction for finite-volume models". In: *Journal of Computational Physics* 371 (2018), pp. 280–314. DOI: 10.1016/j.jcp.2018.05.019.

[124] Kookjin Lee and Kevin T. Carlberg. "Deep Conservation: A Latent-Dynamics Model for Exact Satisfaction of Physical Conservation Laws". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. 2021. DOI: 10.1609/aaai.v35i1.16102.

[125] Christopher R Wentland, Cheng Huang, and Karthikeyan Duraisamy. "Investigation of Sampling Strategies for Reduced-Order Models of Rocket Combustors". In: *AIAA Scitech Forum*. 2021. DOI: 10.2514/6.2021-1371.

[126] Cheng Huang, Karthik Duraisamy, and Charles Merkle. "Component-Based Reduced Order Modeling of Large-Scale Complex Systems". In: *Frontiers in Physics* 10 (2022). DOI: 10.3389/fphy.2022.900064.

[127] J. Choi, J. Dongarra, R. Pozo, and D. Walker. "ScaLAPACK: a scalable linear algebra library for distributed memory concurrent computers". In: *The Fourth Symposium on the Frontiers of Massively Parallel Computation*. 1992. DOI: 10.1109/FMPC.1992.234898.

[128] Mark Gates, Jakub Kurzak, Ali Charara, Asim YarKhan, and Jack Dongarra. "SLATE: Design of a Modern Distributed and Accelerated Linear Algebra Library". In: *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. 2019. DOI: 10.1145/3295500.3356223.

[129] Nicholas Arnold-Medabalimi, Cheng Huang, and Karthik Duraisamy. "Data-Driven Modal Decomposition Techniques for High-Dimensional Flow Fields". In: *Data Analysis for Direct Numerical Simulations of Turbulent Combustion: From Equation-Based Analysis to Machine Learning*. Ed. by Heinz Pitsch and Antonio Attili. Springer International Publishing, 2020, pp. 135–155. ISBN: 978-3-030-44718-2. DOI: 10.1007/978-3-030-44718-2_7.

[130] Matthew E. Harvazinski, Tristan Fuller, William E. Anderson, Nicholas Arnold-Medabalimi, and Cheng Huang. "Large Eddy Simulations of a Liquid Rocket Injector under Multiple Operating Conditions". In: *AIAA Scitech Forum*. 2020. DOI: 10.2514/6.2020-0421.

[131] Shaowu Pan, Nicholas Arnold-Medabalimi, and Karthik Duraisamy. "Sparsity-promoting algorithms for the discovery of informative Koopman-invariant subspaces". In: *Journal of Fluid Mechanics* 917 (2021). DOI: 10.1017/jfm.2021.271.

[132] Nicholas Arnold-Medabalimi, Cheng Huang, and Karthik Duraisamy. "Large-eddy simulation and challenges for projection-based reduced-order modeling of a gas turbine model combustor". In: *International Journal of Spray and Combustion Dynamics* 14.1-2 (2022), pp. 153–175. DOI: 10.1177/17568277221100650.

[133] S. Chaturantabut and D.C. Sorensen. "Nonlinear model reduction via discrete empirical interpolation". In: *SIAM Journal on Scientific Computing* 32 (5 2010), pp. 2737–2764. DOI: 10.1137/09076649.

[134] R Everson and L Sirovich. "Karhunen–Loeve procedure for gappy data". In: *Journal of the Optical Society of America A* 12.8 (1995), pp. 1657–1664. DOI: 10.1364/JOSAA.12.001657.

[135] Steven S. An, Theodore Kim, and Doug L. James. "Optimizing cubature for efficient integration of subspace deformations". In: *ACM Transactions on Graphics* 27.5 (2008). DOI: 10.1145/1409060.1409118.

[136] J. A. Hernández, M. A. Caicedo, and A. Ferrer. "Dimensional hyper-reduction of nonlinear finite element models via empirical cubature". In: *Computer Methods in Applied Mechanics and Engineering* 313 (2017), pp. 687–722. DOI: 10.1016/j.cma.2016.10.022.

[137] Charbel Farhat, Philip Avery, Todd Chapman, and Julien Cortial. "Dimensional reduction of nonlinear finite element dynamic models with finite rotations and energy-based mesh sampling and weighting for computational efficiency". In: *International Journal for Numerical Methods in Engineering* 98 (2014), pp. 625–662. DOI: 10.1002/nme.4668.

[138] Konstantinos Vlachas, David Najera-Flores, Carianne Martinez, Adam R Brink, and Eleni Chatzi. "A Physics-Based Reduced Order Model with Machine Learning-Boosted Hyper-Reduction". In: *Topics in Modal Analysis & Parameter Identification, Volume 8*. Ed. by Brandon J Dilworth, Timothy Marinone, and Michael Mains. Springer, 2023, pp. 131–139. ISBN: 978-3-031-05445-7.

[139] Youngkyu Kim, Youngsoo Choi, David Widemann, and Tarek Zohdi. "A fast and accurate physics-informed neural network reduced order model with shallow masked autoencoder". In: *Journal of Computational Physics* 451 (2022). DOI: 10.1016/j.jcp.2021.110841.

[140] Patricia Astrid. "Fast Reduced Order Modeling Technique for Large Scale LTV Systems". In: *Proceedings of the American Control Conference*. 2004, pp. 762–767. DOI: 10.1109/ACC.2004.182340.

[141] Patricia Astrid, Siep Weiland, Karen Willcox, and Ton Backx. "Missing point estimation in models described by proper orthogonal decomposition". In: *IEEE Transactions on Automatic Control* 53.10 (2008), pp. 2237–2251. DOI: 10.1109/TAC.2008.2006102.

[142] Robert Bos, Xavier Bombois, and Paul Van Den Hof. "Accelerating large-scale non-linear models for monitoring and control using spatial and temporal correlations". In: *Proceedings of the American Control Conference*. 2004, pp. 3705–3710. DOI: 10.1109/ACC.2004.182863.

[143] Patrick Allen LeGresley. "Application of Proper Orthogonal Decomposition (POD) to Design Decomposition Methods". PhD thesis. Stanford University, 2005.

[144] Maxime Barrault, Yvon Maday, Ngoc Cuong Nguyen, and Anthony T. Patera. "An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations". In: *Comptes Rendus Mathematique* 339.9 (2004), pp. 667–672. DOI: 10.1016/j.crma.2004.08.006.

[145] Karen Willcox. "Unsteady flow sensing and estimation via the gappy proper orthogonal decomposition". In: *Computers & Fluids* 35.2 (2006). DOI: 10.1016/j.compfluid.2004.11.006.

[146] Krithika Manohar, Bingni W. Brunton, J. Nathan Kutz, and Steven L. Brunton. "Data-Driven Sparse Sensor Placement for Reconstruction: Demonstrating the Benefits of Exploiting Known Patterns". In: *IEEE Control Systems Magazine* 38.3 (2018), pp. 63–86. DOI: 10.1109/MCS.2018.2810460.

[147] Ralf Zimmermann and Karen Willcox. "An accelerated greedy missing point estimation procedure". In: *SIAM Journal on Scientific Computing* 38.5 (2016), pp. 2827–2850. DOI: 10.1137/15M1042899.

[148] Saifon Chaturantabut and Danny C. Sorensen. "Application of POD and DEIM on dimension reduction of non-linear miscible viscous fingering in porous media". In: *Mathematical and Computer Modelling of Dynamical Systems* 17.4 (2011), pp. 337–353. DOI: 10.1080/13873954.2011.547660.

[149] R. Ştefânescu and I. M. Navon. "POD/DEIM nonlinear model order reduction of an ADI implicit shallow water equations model". In: *Journal of Computational Physics* 237 (2013), pp. 95–114. DOI: 10.1016/j.jcp.2012.11.035.

[150] D. Wirtz, D. C. Sorensen, and B. Haasdonk. "A posteriori error estimation for DEIM reduced nonlinear dynamical systems". In: *SIAM Journal on Scientific Computing* 36.2 (2014), A311–A338. DOI: 10.1137/120899042.

[151] David Amsallem, Matthew Zahr, Youngsoo Choi, and Charbel Farhat. "Design optimization using hyper-reduced-order models". In: *Structural and Multidisciplinary Optimization* 51 (2015), pp. 919–940. DOI: 10.1007/s00158-014-1183-y.

[152] Alessandro Alla and J. Nathan Kutz. "Nonlinear model reduction via dynamic mode decomposition". In: *SIAM Journal on Scientific Computing* 39.5 (2017), B778–B796. DOI: 10.1137/16M1059308.

[153] Benjamin Peherstorfer. "Model reduction for transport-dominated problems via online adaptive bases and adaptive sampling". In: *SIAM Journal on Scientific Computing* 42.5 (2020), A2803–A2836. DOI: 10.1137/19M1257275.

[154] Feng Bai and Yi Wang. "A reduced order modeling method based on GNAT-embedded hybrid snapshot simulation". In: *Mathematics and Computers in Simulation* 199 (2022), pp. 100–132. DOI: 10.1016/j.matcom.2022.03.006.

[155] Jessica T. Lauzon, Siu Wun Cheung, Yeonjong Shin, Youngsoo Choi, Dylan Matthew Copeland, and Kevin Huynh. "S-OPT: A Points Selection Algorithm for Hyper-

Reduction in Reduced Order Models". In: *pre-print* (2022). arXiv: 2203.16494 [math.NA].

[156] Radek Tezaur, Faisal As'ad, and Charbel Farhat. "Robust and globally efficient reduction of parametric, highly nonlinear computational models and real time online performance". In: *Computer Methods in Applied Mechanics and Engineering* 399 (2022). DOI: 10.1016/j.cma.2022.115392.

[157] Irina Tezaur, Jeffrey Fike, Kevin Carlberg, Matthew Barone, Danielle Maddix, and Erin Mussoni. *Advanced Fluid Reduced Order Models for Compressible Flow.* Tech. rep. Sandia National Laboratories, 2017.

[158] Yuxiang Beckett Zhou. "Model Reduction for Nonlinear Dynamical Systems with Parametric Uncertainties". MA thesis. Massachusetts Institute of Technology, 2012.

[159] Zlatko Drmač and Serkan Gugercin. "A new selection operator for the discrete empirical interpolation method-improved a priori error bound and extensions". In: *SIAM Journal on Scientific Computing* 38.2 (2016), A631–A648. DOI: 10.1137/15M1019271.

[160] George Karypis and Vipin Kumar. "A fast and high quality multilevel scheme for partitioning irregular graphs". In: *SIAM Journal on Scientific Computing* 20 (1 1999), pp. 359–392. DOI: 10.1137/S1064827595287997.

[161] Christopher R. Wentland and Karthik Duraisamy. "PERFORM: A Python package for developing reduced-order models for reacting fluid flows". In: *Journal of Open Source Software* 7.79 (2022), p. 3428. DOI: 10.21105/joss.03428.

[162] Elnaz Rezaian, Cheng Huang, and Karthik Duraisamy. "Non-intrusive balancing transformation of highly stiff systems with lightly damped impulse response". In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 380 (2022). DOI: 10.1098/rsta.2021.0202.

[163] Christopher R. Wentland, Cheng Huang, and Karthikeyan Duraisamy. "Closure of Reacting Flow Reduced-Order Models via the Adjoint Petrov-Galerkin Method". In: *AIAA Aviation Forum.* 2019. DOI: 10.2514/6.2019-3531.

[164] Elizabeth Qian, Boris Kramer, Benjamin Peherstorfer, and Karen Willcox. "Lift & Learn: Physics-informed machine learning for large-scale nonlinear dynamical systems". In: *Physica D: Nonlinear Phenomena* 406 (2020). DOI: 10.1016/j.physd.2020.132401.

[165] Romit Maulik, Bethany Lusch, and Prasanna Balaprakash. "Non-autoregressive time-series methods for stable parametric reduced-order models". In: *Physics of Fluids* 32 (2020). DOI: 10.1063/5.0019884.

[166] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. "A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures". In: *Neural Computation* 31 (2019), pp. 1235–1270. DOI: 10.1162/neco_a_01199.

[167] Anatol Roshko. *Some Measurements of Flow in a Rectangular Cutout*. Tech. rep. 1952.

[168] K Krishnamurty. *Acoustic Radiation from Two-dimensional Rectangular Cutouts in Aerodynamic Surfaces*. Tech. rep. 1955.

[169] J.E. Rossiter. *Wind-Tunnel Experiments on the Flow over Rectangular Cavities at Subsonic and Transonic Speeds*. Tech. rep. 1964.

[170] Tim Colonius, Amit J Basu, and Clarence W Rowley. "Computation of sound generation and flow/acoustic instabilities in the flow past an open cavity". In: *ASME/JSME Joint Fluids Engineering Conference*. 1999.

[171] Clarence W. Rowley, Tim Colonius, and Amit J. Basu. "On self-sustained oscillations in two-dimensional compressible flow over rectangular cavities". In: *Journal of Fluid Mechanics* 455 (2002), pp. 315–346. ISSN: 00221120. DOI: 10.1017/S0022112001007534.

[172] Lionel Larchevêque, Pierre Sagaut, and Odile Labbé. "Large-eddy simulation of a subsonic cavity flow including asymmetric three-dimensional effects". In: *Journal of Fluid Mechanics* 577 (2007), pp. 105–126. DOI: 10.1017/S0022112006004502.

[173] Justin L. Wagner, Katya M. Casper, Steven J. Beresh, Patrick S. Hunter, Russell W. Spillers, John F. Henfling, and Randall L. Mayes. "Fluid-structure interactions in compressible cavity flows". In: *Physics of Fluids* 27.6 (2015). DOI: 10.1063/1.4922021.

[174] Katya M. Casper, Justin L. Wagner, Steven J. Beresh, Russell W. Spillers, John F. Henfling, and Lawrence J. Dechant. "Spatial distribution of pressure resonance in compressible cavity flow". In: *Journal of Fluid Mechanics* 848 (2018), pp. 660–675. DOI: 10.1017/jfm.2018.346.

[175] Irina Tezaur, Jeffrey Fike, Kevin Carlberg, Maciej Balajewicz, Matthew Barone, and Erin Mussoni. *Model Reduction for Compressible Cavity Simulations Towards Uncertainty Quantification of Structural Loading*. Tech. rep. Sandia National Laboratories, 2016.

[176] Yen C. Yu, Stefan M. Koeglmeier, James C. Sisco, and William E. Anderson. "Combustion Instability of Gaseous Fuels in a Continuously Variable Resonance Chamber (CVRC)". In: *AIAA Joint Propulsion Conference*. 2008. DOI: 10.2514/6.2008-4657.

[177] Yen C. Yu, Loral O'Hara, James C. Sisco, and William E. Anderson. "Experimental Study of High-Frequency Combustion Instability in a Continuously Variable Resonance Combustor (CVRC)". In: *AIAA Aerospace Sciences Meeting*. 2009. DOI: 10.2514/6.2009-234.

[178] Yen C Yu, James C Sisco, Stanford Rosen, Ajay Madhav, and William E Anderson. "Spontaneous Longitudinal Combustion Instability in a Continuously Variable Resonance Combustor". In: *Journal of Propulsion and Power* Vol 28.5 (2012), pp. 876–887.

[179] Romain Garby, Laurent Selle, and Thierry Poinsot. "Large-Eddy Simulation of combustion instabilities in a variable-length combustor". In: *Comptes Rendus Mécanique* 341 (2013), pp. 220–229. DOI: 10.1016/j.crme.2012.10.020.

[180] Tuan M. Nguyen, Pavel P. Popov, and William A. Sirignano. "Longitudinal Combustion Instability in a Rocket Engine with a Single Coaxial Injector". In: *Journal of Propulsion and Power* 34.2 (2018), pp. 354–373. DOI: 10.2514/1.b36516.

[181] Matthew E Harvazinski, Cheng Huang, Venkateswaran Sankaran, Thomas W Feldman, William E Anderson, Charles L Merkle, and Douglas G Talley. "Coupling between hydrodynamics, acoustics, and heat release in a self-excited unstable combustor". In: *Physics of Fluids* 27 (2015). DOI: 10.1063/1.4916673.

[182] Matthew E. Harvazinski, Douglas G. Talley, and Venkateswaran Sankaran. "Influence of boundary condition treatment on longitudinal-mode combustion instability predictions". In: *Journal of Propulsion and Power* 32.2 (2016), pp. 529–532. DOI: 10.2514/1.B35972.

[183] Matthew E Harvazinski and Taro Shimizu. "Computational Investigation on the Effect of the Oxidizer Inlet Temperature on Combustion Instability". In: *AIAA Propulsion and Energy Forum*. 2019. DOI: 10.2514/6.2019-4109.

[184] Matthew E. Harvazinski, Rohan M. Gejji, Douglas G. Talley, Michael R. Orth, William E. Anderson, and Timothee L. Pourpoint. "Modeling of transverse combustion instability". In: *AIAA Scitech Forum*. 2019. DOI: 10.2514/6.2019-1732.

[185] Rui Xu, Kun Wang, Sayak Banerjee, Jiankun Shao, Tom Parise, Yangye Zhu, Shengkai Wang, Ashkan Movaghar, Dong Joon, Runhua Zhao, Xu Han, Yang Gao, Tianfeng Lu, Kenneth Brezinsky, Fokion N Egolfopoulos, David F Davidson, Ronald K Hanson, Craig T Bowman, and Hai Wang. "A physics-based approach to modeling real-fuel combustion chemistry – II . Reaction kinetic models of jet and rocket fuels". In: *Combustion and Flame* 193 (2018), pp. 520–537. DOI: 10.1016/j.combustflame.2018.03.021.

[186] Y. Tao G.P. Smith and H. Wang. *Foundational Fuel Chemistry Model Version 1.0 (FFCM-1)*. 2016. URL: https://web.stanford.edu/group/haiwanglab/FFCM1/pages/FFCM1.html.

[187] Matthew E. Harvazinski. "Bulk mode investigations of a liquid rocket injector". In: *AIAA Scitech Forum*. 2021. DOI: 10.2514/6.2021-1248.

[188] Supraj Prakash, Venkat Raman, Christopher F. Lietz, William A. Hargus, and Stephen A. Schumaker. "Numerical simulation of a methane-oxygen rotating detonation rocket engine". In: *Proceedings of the Combustion Institute* 38 (2021), pp. 3777–3786. DOI: 10.1016/j.proci.2020.06.288.

[189] Armani Batista, Mathias C. Ross, Christopher Lietz, and William A. Hargus. "Descending modal transition dynamics in a large eddy simulation of a rotating detonation rocket engine". In: *Energies* 14 (2021). DOI: 10.3390/en14123387.

[190] David Amsallem, Matthew J. Zahr, and Charbel Farhat. "Nonlinear model order reduction based on local reduced-order bases". In: *International Journal for Numerical Methods in Engineering* 92 (2012), pp. 891–916. DOI: 10.1002/nme.4371.

[191] Benjamin Peherstorfer, Daniel Butnaru, Karen Willcox, and Bungartz Hans-Joachim. "Localized Discrete Empirical Interpolation Method". In: *SIAM Journal on Scientific Computing* 36.1 (2014), A168–A192. DOI: 10.1137/130924408.

[192] Rémi Abgrall, David Amsallem, and Roxana Crisovan. "Robust model reduction by L1 -norm minimization and approximation via dictionaries: application to nonlinear hyperbolic problems". In: *Advanced Modeling and Simulation in Engineering Sciences* 3.1 (2016). DOI: 10.1186/s40323-015-0055-3.

[193] Youngsoo Choi and Kevin Carlberg. "Space-time least-squares Petrov-Galerkin projection for nonlinear model reduction". In: *SIAM Journal on Scientific Computing* 41.1 (2019), A26–A58. DOI: 10.1137/17M1120531.

[194] Chi Hoang, Kenny Chowdhary, Kookjin Lee, and Jaideep Ray. "Projection-based model reduction of dynamical systems using space-time subspace and machine learning". In: *Computer Methods in Applied Mechanics and Engineering* 389 (2022). DOI: 10.1016/j.cma.2021.114341.

[195] Angelo Iollo and Damiano Lombardi. "Advection modes by optimal mass transfer". In: *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* 89 (2014). DOI: 10.1103/PhysRevE.89.022923.

[196] Philip A. Etter and Kevin T. Carlberg. "Online adaptive basis refinement and compression for reduced-order models via vector-space sieving". In: *Computer Methods in Applied Mechanics and Engineering* 364 (2019). DOI: 10.1016/j.cma.2020.112931.

[197] Benjamin Peherstorfer and Karen Willcox. "Online adaptive model reduction for nonlinear systems via low-rank updates". In: *SIAM Journal on Scientific Computing* 37.4 (2015), A2123–A2150. DOI: 10.1137/140989169.

[198] John L. Lumley and Andrew Poje. "Low-dimensional models for flows with density fluctuations". In: *Physics of Fluids* 9.7 (1997), pp. 2023–2031. DOI: 10.1063/1.869321.

[199] Patrick J. Blonigan, Kevin Carlberg, Francesco Rizzi, Micah Howard, and Jeffrey A. Fike. "Model reduction for hypersonic aerodynamics via conservative LSPG projection and hyper-reduction". In: *AIAA Scitech Forum*. 2020. DOI: 10.2514/6.2020-0104.

[200] S. Sirisup and G. E. Karniadakis. "A spectral viscosity method for correcting the long-term behavior of POD models". In: *Journal of Computational Physics* 194 (2004), pp. 92–116. DOI: 10.1016/j.jcp.2003.08.021.

[201] Omer San and Traian Iliescu. "Proper orthogonal decomposition closure models for fluid flows: Burgers equation". In: (2013). arXiv: 1308.3276.

[202] Selin Ardag, Stefan Siegel, Jurgen Seidel, Kelly Cohen, and Thomas McLaughlin. "Filtered POD-based low-dimensional modeling of the 3D turbulent flow behind a circular cylinder". In: *International Journal for Numerical Methods in Fluids* 66 (2011), pp. 1–16. DOI: 10.1002/fld.2238.

[203] D. Wells, Z. Wang, X. Xie, and T. Iliescu. "An evolve-then-filter regularized reduced order model for convection-dominated flows". In: *International Journal for Numerical Methods in Fluids* 84 (2017), pp. 598–615. DOI: 10.1002/fld.4363.