OPERATOR THEORETIC MODEL PREDICTIVE CONTROL OF MOVING BOUNDARY DYNAMICAL SYSTEMS: APPLICATION TO HYDRAULIC FRACTURING

A Dissertation

by

ABHINAV NARASINGAM

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Joseph Sang-Il Kwon
Committee Members,	Costas Kravaris
	M. M. Faruque Hasan
	Yalchin Efendiev
Head of Department,	Arul Jayaraman

August 2020

Major Subject: Chemical Engineering

ABSTRACT

Many chemical processes are characterized by nonlinear models with moving boundaries that describe their dynamics. Such a nonlinear trajectory can explore many regions of the state space and the spatial domains of interest as well as the dominant spatial patterns of the system change with time. Therefore, traditional model approximation methods may fail to accurately capture the dynamics. This underlines the need to develop strategies for obtaining computationally efficient yet highly accurate models that aid in the prediction and control of these systems.

In such systems, care should be taken to capture the local behavior of every portion of the solution trajectory in the state space. To achieve this, the first step of this research work computes (temporally) local reduced order models by first performing time-domain partitioning using a novel clustering strategy based on Mixed Integer Nonlinear Programming. The reduced bases are then derived within each cluster using two model reduction methods, Proper Orthogonal Decomposition (POD) and Dynamic Mode Decomposition (DMD). The proposed framework is tested on a nonlinear hydraulic fracturing process, and a Model Predictive Controller (MPC) is formulated using the (temporally) local reduced order models to design an optimum pumping schedule that ensures uniform proppant concentration for maximum oil and gas extraction.

The idea of model reduction is then extended to the parameter space in order to estimate unknown rock properties that directly affect the hydraulic fracturing process. Owing to the complex dynamics exhibited, this results in the formulation of a large-scale inverse problem that is ill-posed. To deal with this unidentifiability issue, the number of unknown model parameters are reduced while preserving the spatial features in the geological properties via POD. The statistical information is then updated using the available process measurements via a Monte Carlo data assimilation technique, the Ensemble Kalman Filter.

Although the local model reduction framework is superior in performance, a single model that is accurate within a larger domain in the state space would be beneficial in some cases. Therefore, the next part explores an alternative approach that uses Koopman operator theory to develop linear predictors to approximate nonlinear systems. Before employing it to process systems, we address several key challenges associated with it. In particular, we solve a sparse regression problem over a large set of candidate functional forms to effectively identify the required nonlinear transformation (observable functions). The developed models are shown to reveal important physical phenomena such as proppant transport and fracture propagation inside a fracture. It also highlights how a priori knowledge can be incorporated easily into the algorithm and results in accurate models that are used for controller synthesis.

Finally, Koopman theory is integrated with Lyapunov-based MPC (LMPC) for stabilizing nonlinear systems. LMPC possesses all the advantages of a standard MPC with added explicit Lyapunov constraints and is particularly suited as it explicitly characterizes a set of initial conditions starting from where the closed–loop stability can be guaranteed. By leveraging the linear structure of the Koopman models, control Lyapunov functions (CLFs) within LMPC are embedded in the nonlinear transformation to yield a standard convex (quadratic) optimization problem. This method is then improved by formulating the controller in Koopman eigenfunction coordinates, which leads to a bilinear model for control affine systems. Provided there exists a continuously differentiable inverse mapping between the original state space and (lifted) function space, the designed controller is capable of translating the feedback stabilizability of the Koopman bilinear system to the original nonlinear system. Due to the bilinear structure of the Koopman model, seeking a CLF is no longer a bottleneck for LMPC. Benchmark numerical examples demonstrate the utility of the proposed feedback control design.

ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Joseph Sang-Il Kwon who guided me in every phase of my doctoral studies. He provided me tremendous support and encouragement both in professional and personal life. I would like to thank my committee Dr. Kravaris, Dr. Hasan and Dr. Efendiev for guiding me through my research and for their invaluable feedback which helped me push further and widen my perspective. I would also like to specially thank Dr. Eduardo Gildin, who was kind enough to substitute for one of the committee members during my preliminary examination and his student Anqi Bao who collaborated with me on a project.

I would like to thank Dr. Panagiotis Christofides, Professor at UCLA, and his student Dr. Zhe Wu for their valuable insights and helping me understand the nuances of Lyapunov-based model predictive control. I would also like to thank all my colleagues from Dr. Kwon's lab, Harwinder Singh Sidhu, Dr. Prashanth Siddhamshetty, Dongheon Lee, Hyun-Kyu Choi, Niranjan Arvind Sitapure, Mohammed Saad Faizan Bangi, and Bhavana Bhadriraju Venkata Naga Sai for being a part of my journey. I would like to offer special thanks to Dr. Sang-Hwan Son for his advice and support on the theoretical aspects of stable and robust control. I thank the Texas A&M High Performance Research Computing Center for making available state-of-the-art super computing facilities for my research. I would also like to thank all the staff in the chemical engineering department for helping promptly whenever required and providing a hassle free work environment.

Finally, I am grateful to my family and friends for supporting me throughout my life. Words cannot possibly describe the sacrifices they have made for me or the support they have given.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a dissertation committee consisting of Dr. Joseph Kwon (chair), Dr. Costas Kravaris and Dr. Faruque Hasan of the Artie McFerrin Department of Chemical Engineering, and Dr. Yalchin Efendiev of the Department of Mathematics.

The high-fidelity hydraulic fracturing model used for generating data required for analyzing the validity of the developed frameworks and algorithms was provided by Dr. Prashanth Siddhamshetty from Dr. Joseph Kwon's lab; the student is either explicitly mentioned in the thesis and/or in papers that were published or that will be published in the future.

All other work conducted for the dissertation was completed by the student independently.

Funding Sources

This work was made possible in part by the startup fund awarded to Dr. Joseph Kwon by the Artie McFerrin Department of Chemical Engineering at Texas A&M University. This work was also made possible by financial support from the National Science Foundation under the grant CBET-1804407 and U.S. Department of Energy under grant numbers DE-EE0007888-10-8, DE-EE0007888-9.3.

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

The theme that appears throughout this research work is to employ *approximate (reduced)* models for the purpose of *real-time control* of complex nonlinear dynamical systems, especially those characterized by moving boundaries. This work tackles the challenge of model approximation (reduction) of nonlinear systems mainly from two different algorithmic standpoints: local (cluster) based model reduction and nonlinear transformation based approximations. In both these approaches, we employ operator theoretic perspective of dynamical systems as a foundation to develop the approximate models. For the first approach, the state-space is first divided into clusters and reduced-order models are developed locally within each cluster to capture the essential features of every portion of the system trajectory. In the second approach, the dynamics are first "lifted" onto a higher-dimensional space using nonlinear transformation and linear predictors are constructed to approximate these dynamics in a principled manner.

1.1 Motivation

Many chemical processes are characterized by complex nonlinear models that can describe their dynamics to near-perfect accuracy. Undoubtedly, the computational burden associated with solving these high-fidelity models can be exorbitant, particularly when performing fully-resolved simulations at very fine time scales and/or spatial mesh sizes. These costs can be further magnified and impose challenges in the context of dynamic optimization, parameter estimation and controller synthesis. For example, flow control is largely dependent upon solving the Navier-Stokes equations which in terms of numerical computations can require large grid sizes, typically $O(n = 10^6)$. The memory required to store such matrices, $O(n^2)$, is itself around 7 TB and the cost scales as $O(n^3)$ for computations such as matrix inversion or solution of Lyapunov or Riccati equations or inequalities, which is prohibitive even with today's advanced hardware. Fortunately, in several problems, a much simpler, latent structure can be identified by using physical or mathematical arguments that could make the aforementioned tasks computationally tractable. In recent times, the plummeting cost of sensors, data storage and computational resources have resulted in the accessibility of abundant time-series data for various processes. In combination with mathematical models, this data can lead us to identify the above mentioned reduced, predictive descriptions of the original complex systems. These reduced models preserve a few salient features of the underlying phenomena at a more practical scale. From a computational point of view, quite frequently, the time evolution of all the observables is irrelevant for physical and practical purposes, so the analysis is instead focused on a reduced set of variables. This reduced set of variables parameterizes a low-dimensional system characterizing the long-term dynamics of the process for which the control tools are applicable.

1.2 Operator-theoretic model approximation

The classical dynamical systems theory focuses on Poincaré's geometric approach, i.e., state-space description through which the evolution of the dynamics can be understood. Such an approach has met with success in a variety of settings and, at this point, one hardly needs to justify the use of geometric theory when working on a particular problem. However, in a mathematical and engineering sense, a phenomenon of interest can be represented in different ways. For example, instead of working in the state-space one can shift the focus to the space of functions (of the state) called *observables* and watch how these functions evolve in time. Furthermore, it would be highly desirable if one could find such a set of functions whose evolution is governed by a linear operator. In a general dynamical system, one measures the system output which can be interpreted as some function of the states. This means that the spectral properties of the linear operator (i.e., eigenvalues and eigenfunctions) governing the evolution of observables encode global information that allows future state prediction and scalable reconstruction of the underlying dynamics simply from measurement data [1]. Expressing the system in a function (output) space makes sense especially in the era of big data.

The operator theoretic formalism for nonlinear systems goes back to seminal works by Koopman and Von Neumann [2, 3] where it was shown that Hamiltonian dynamics can be analyzed using an infinite-dimensional linear operator on the space of all possible observable functions that can be measured from the underlying state. The Koopman operator is a linear operator that describes the temporal evolution of scalar observables (which are essentially functions of system states) which are driven by the underlying nonlinear dynamics. Specifically, the Koopman operator, when acted upon an observable, governs its evolution along the original system trajectory. Hence, the operator-theoretic description provides global insight into the system dynamics. This makes the Koopman operator a natural choice for data-driven analysis of dynamical systems and is appropriate for controller design.

Another feature of Koopman operator approach that makes it extremely appealing for controller design is that it is a linear operator, although infinite dimensional, even when associated with nonlinear dynamics. Thus it extends the spectral analysis concepts of linear systems to the dynamics of observables in nonlinear systems. Specifically, the eigenvalues, eigenfunctions and invariant subspaces of the Koopman operator encode global information and provide valuable insights that allows future state prediction and scalable reconstruction of the underlying dynamics [4]. It has been shown in the literature that Koopman eigenfunctions are strongly connected to the geometric properties of the system [5, 6], and are related to global linearization of the system [7]. Recently, the connection between (existence of) specific Koopman eigenfunctions and the global stability analysis of has been explored [8]. Koopman eigenfunctions and the corresponding eigenvalues also facilitate the estimation of limit cycles as well as their basins of attraction [9]. Furthermore, the full state of the system can be projected onto the eigenfunctions of the Koopman operator using a linear combination of (Koopman) modes characterized by a fixed frequency and rate of decay. Therefore, dominant patterns of the underlying nonlinear system can be captured using these modes as useful coherent structures [1].

Since its revival in [1] , Koopman operator theory has been the focus of many research efforts as a popular tool for data-driven analysis and control of nonlinear dynamical systems [10, 11, 12, 13]. Recently, Koopman operator has been used to study nonlinear stability and a sufficient condition for global stability has been established with respect to the existence of stable eigenfunctions [8]. With respect to control, Koopman operator theory has been successfully used for feedback control of fluid flows, tracking reference output signals in Burgers equation and incompressible flow past a cylinder [12]. In [10] it has been shown that Koopman based linear predictors show superior performance over local linearization methods both in prediction and control of the Korteweg-de Vries PDE. It has also been extended to input-output systems for observer synthesis [14, 15] and constrained state estimation [16].

1.3 Operator theory and model reduction

A commonly used idea in applied mathematics and engineering for problems that deal with infinite features is that of the expansion of a possibly complicated function of space and time (for systems described by partial differential equations (PDEs)) into an infinite sum of simpler components. The most common examples are of course the Taylor and Fourier expansions (or decompositions). For these decompositions, a fixed set of functions of space is chosen, and the evolving function or field is projected onto those to obtain time-dependent coefficients. We could call these decompositions with a predetermined basis. With time, many model reduction techniques have been proposed and implemented in the context of several applications. The exhaustive list of available methods can be broadly classified into three categories: a) projection/modal decomposition methods [17, 18, 19, 20], b) input-output methods [21, 22, 23, 24, 25], and c) machine learning methods [26, 27]. Among them, the class of projection-based techniques seek to exploit the fact that the solutions of large-scale complex systems can be projected (at some angle) onto a low-dimensional manifold to yield reduced-order models. Generally, they take a set of data and from it compute a set of modes, or characteristic features to form a basis that spans a subspace onto which the original data is projected. Intuitively, operator theory which facilitates spectral analysis via the linear operator can therefore be a natural choice for model reduction of high-dimensional systems. The meaning of the basis chosen, modes and the angle of projection will depend on the particular type of decomposition used. Three of the most widely used decomposition methods are introduced below and they form the foundation of the developed model reduction framework.

1.3.1 Proper Orthogonal Decomposition

Today, perhaps the most popular projection method for model reduction is based on Proper Orthogonal Decomposition (POD) which basically maximizes the energy (L_2- norm) of an orthogonal projection of a given data set. POD has come to be known by many different names, depending on the field of application, like principal component analysis (PCA) [28], the Karhunen– Loève transform [29], and empirical orthogonal function (EOF) analysis [30]. All of these reduce to computing the Singular Value Decomposition (SVD) of a data set. It was first introduced to fluid systems by Lorenz [30] and later by Lumley [31] for identifying coherent structures in turbulent flows [17]. The method essentially captures energetically dominant features of fluid flows and has been widely used to achieve better understanding of complex flow physics. The potential to extract empirical information from experimental or large-scale simulation data is one of the reasons for the wide popularity of POD-based techniques. It utilizes detailed numerical simulations of high-fidelity systems (in practice, we can use process/experimental data) to compute empirical eigenfunctions that accurately describe the solution subspace. In 1987, Sirovich introduced the method of snapshots assuming that the eigenfunctions can be expressed as a linear combination of the snapshots [32, 33, 34]. Under this assumption, the eigenfunctions can be algebraically computed by solving an eigenvalue problem associated with a covariance matrix constructed from the inner product of each snapshot with the rest of the ensemble. These empirical eigenfunctions are then used in a projection method such as the Galerkin's framework to derive low-dimensional systems of the original nonlinear dynamical systems [35, 36, 37]. The POD-based MOR technique has found successful implementation in several applications [38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51].

The main drawback of POD is that the reduced models obtained by subsequent Galerkin projection of the governing equations do not always represent the *dynamics* accurately. This is because although the POD modes are energetically optimal; sometimes the evolution of dynamics are affected by low energy modes [52]. For instance, when analyzing time-series data, if the data are reordered the POD modes still remain unchanged. In other words, the time evolution or dynamics of the system are not encoded into the POD modes. In its basic form, POD is a statistical tool used to capture most important features for reconstructing a given data set. Furthermore, POD does not account for the external inputs or actuation which is very important when dealing with forced dynamics.

1.3.2 Dynamic Mode Decomposition

Unlike POD, Dynamic Mode Decomposition (DMD) has been developed specifically for analyzing the dynamics of nonlinearly evolving dynamics. It was first introduced by Schmid [18] as a generalization of linear stability analysis and then independently by Rowley [19] as a particular numerical algorithm to compute dynamically relevant modes for globally linearizing nonlinear dynamics based on linear operator theory. Computationally, DMD assumes a linear model that best represents the underlying dynamics, even if those dynamics stem from a nonlinear process. Although it might seem equivocal describing a nonlinear system by superposition of modes whose dynamics are governed by the corresponding eigenvalues, Schmid argues that it is valid because DMD analyzes "linear tangent approximation to the underlying flow" [53]. In fact, Rowley explores DMD as a numerical approximation to Koopman spectral analysis providing theoretical justification for characterizing nonlinear systems [19]. The connection between DMD and operator theory is further elaborated below.

After gaining quick popularity, DMD has found successful implementation in many fluid mechanics applications to analyze both numerical [54, 55, 56, 57, 58] as well as experimental flow field data and help characterizing relevant physical mechanisms [59, 60, 61, 62, 63, 64]. Several efforts have been made to explore the connections of DMD with other methods, such as Eigensystem Realization Algorithm (ERA) [65], Fourier Analysis [66], POD [59] and Koopman analysis [67, 68]. Several variants of the DMD algorithm have also been proposed, including optimized DMD [66], optimal mode decomposition [69, 70], sparsity promoting DMD [71] and extended DMD [72]. For more details on DMD literature, the review article by Mezić [1] is recommended.

1.3.2.1 Dynamic Mode Decomposition with control

Building on its success, the authors in [73] extended the concepts of DMD and introduced the method of Dynamic Mode Decomposition with control (DMDc) to utilize both measurements of the system and applied external inputs in extracting the underlying dynamics. Additionally, DMDc also provides a description of how the control inputs affect the system, and with this understanding of the input-to-output behavior, a reduced-order model can be generated and used in the design of feedback control systems to regulate the original high-dimensional systems. DMDc inherits a number of advantages of DMD in that it is a completely data-driven framework and can be applied to nonlinear systems. Furthermore, there are a number of connections between DMDc and other popular system identification methods such as Numerical Algorithms for Subspace State Space System Identification (N4SID) [22], Multivariable Output Error State Space (MOESP) [23], Observer Kalman Filter Identification (OKID) [24], and Canonical Variate Analysis (CVA) [74]. Algorithmically, these methods involve regression, model reduction, and parameter estimation steps similar to DMDc. However, differences do exist in terms of the similarity transformation required for projection and the use of an orthogonal complement of control inputs to generate the approximate solution [75]. Therefore, DMDc can be used in diverse engineering applications, one of which is presented in this work, where the study of dynamics while simultaneously considering the applied control input to the complex systems is important.

1.3.3 Extended Dynamic Mode Decomposition

As mentioned earlier, the Koopman operator theory is feasible only if one can numerically approximate the infinite-dimensional operator. Of all the methods available to do so, one that is of particular interest to us is the Extended DMD (EDMD). EDMD has been developed in [72] as a generalization of DMD which seeks to enrich the observable space by using nonlinear functions. The EDMD algorithm approximates the leading eigenfunctions and eigenvalues of the Koopman operator from time series data and a dictionary of observables that spans a subspace of the scalar observable functions. Specifically, EDMD identifies the “slow” subspace of the Koopman operator which approximates the long term dynamics of observables by neglecting the fast transients. Once identified, they enable the reconstruction of system states as a linear combination of the Koopman eigenfunctions. For a comprehensive treatment of the EDMD method, we refer the readers to [72]. An alternative way, one that is more relevant to model identification and thus we follow in this work, is to envision EDMD as a state-space transformation, so that the dynamics appear to be linear, to accurately compute a finite-dimensional approximation of the controlled Koopman operator. Essentially, the EDMD procedure boils down to a nonlinear transformation of the data (lifting to the space of observables) followed by solving a regression problem to construct the required linear predictors.

1.3.3.1 Connections with DMD

DMD has many conceptually equivalent but mathematically different definitions. In fact, one specific definition taken from [65] proves that EDMD is equivalent to DMD for a very specific and restrictive choice of the dictionary of observables. If the dictionary of observable functions considered in the EDMD algorithm are restricted to linear monomials of the states (i.e., the states alone) then the eigenfunctions computed by DMD and EDMD would be identical. Therefore, conceptually, DMD can be thought of as producing an approximation of Koopman eigenfunctions if the dictionary is constructed using only linear monomials. In general, this approximation is accurate only in some small neighborhood and therefore DMD produces only a local linearization. This reiterates the need to improve accuracy of DMD based model reduction either by computing local bases as mentioned above or using nonlinear basis functions in the dictionary as in the case of EDMD.

1.4 Moving boundary systems

Nonlinear dynamics can be associated with various complexities like chaos, singularities, etc. In this regard, nonlinear transport processes commonly encountered in chemical and petroleum systems can often be associated with moving boundaries. More specifically, the spatial domain of interest as well as the dominant spatial patterns of the system change with time. Although they found successful implementation in a broad range of applications, the above mentioned methods fail to capture the fully-resolved dynamics of such complex systems with a desired accuracy even with an unaffordable large number of basis functions. The performance of the global eigenfunctions becomes even worse when the local dynamic behavior changes significantly owing to nonlinear process parameters that change with space and time. Therefore, there is a need to tailor the existing model approximation methods to successfully handle complex systems associated with moving boundaries.

2. APPLICATION: HYDRAULIC FRACTURING

2.1 Introduction

The application of interest throughout this research work is hydraulic fracturing. Over the past two decades, the natural gas from shale plays has been one of the fastest growing total primary energy contributors in the United States; it has risen from less than 1% of domestic gas production in 2000 to over 20% by 2010 [76]. Although shale gas has been produced for years from shales with natural fractures, the recent shale boom has been primarily due to the use of two technologies: horizontal drilling and hydraulic fracturing. Typically, shale formations are characterized by low matrix permeabilities ($\leq 0.01 - 0.0001$ mD) and hence preclude the use of conventional drilling techniques. Therefore, the recovery of shale oil and gas in commercial quantities is economically viable by creating extensive artificial fractures around wellbore [77]. Specifically, in the fracturing process, a mixture of water, sand and chemicals is injected into the horizontal borehole of the well at very high pressure to fracture the shale rocks and release the gas. The created fractures facilitate the extraction of oil and gas by providing high conductivity pathways from the formation to the wellbore, while the proppant (sand whose permeability is very high compared to the surrounding rock) trapped inside the fracture walls at the end of the process increases the formation permeability within the stimulated reservoir volume. Thus, the final fracture conductivity is paramount in maximizing the productivity of the stimulated well, which is the ultimate goal of a fracturing process in practice.

2.2 Hydraulic fracturing as a control problem

The final fracture conductivity chiefly depends on two factors: propped fracture geometry and proppant distribution throughout the fracture at the end of pumping. In order to achieve a high fracture conductivity, it is essential to create fractures with the desired geometry as well as achieve uniform proppant concentration throughout the fracture at the end of pumping operation. Traditionally, during the planning stage, an optimization-based methodology called Unified Fracture

Design (UFD) is used for the design of optimal hydraulic fracturing treatments [78, 79]. Specifically, based on the total quantity and properties of the proppant to be injected, reservoir properties and drainage area, UFD determines the optimal fracture dimensions such as the fracture half-length and fracture width that maximizes the well productivity index. Once a desired geometry is decided, during the implementation stage, the fracture geometry is adjusted by varying the flow rate of the injected proppant. Simultaneously, the spatial distribution of the proppant concentration across the fracture is regulated by varying the inlet proppant concentration (i.e., pumping schedule). To produce a fracture with desired features determined by the aforementioned optimization-based approach and to have a desired distribution of proppant across the fracture at the end of pumping, it is important to develop a technique to generate an optimal pumping schedule. Therefore, the design of this optimal pumping schedule can be naturally cast as a control problem. However, the problem with controlling these petroleum systems is that the mathematical models used to describe them are very complex due to highly-coupled nonlinear system dynamics and parametric uncertainty. Solving these forward problems at very fine scales is computationally very expensive and becomes prohibitive in the case of (optimal) controller design where such forward problems need to be solved iteratively.

Owing to this, initial efforts viewed the fracturing process as an open-loop problem where the pumping schedule followed a pre-determined power-law which assumed a constant leak-off and does not take into account practical constraints [80]. To deal with its limitations, researchers also leveraged exhaustive forward simulations which were used to adjust the pumping schedule iteratively until the desired fracture geometry and fracture conductivity were achieved. This approach outperforms Nolte's pumping schedule at the expense of computational resources [81, 82]. In recent years, several works successfully considered the closed-loop operation of the hydraulic fracturing process by using model-based predictive controllers that take advantage of feedback measurements to compensate for minor model uncertainty and disturbances [83, 84, 85]. However, the performance of these real-time control schemes highly depends on the availability of computationally efficient approximate models capable of capturing the nonlinear process behavior.

2.3 Dynamic model of hydraulic fracturing

Typically, a hydraulic fracturing process consists of two sub-processes: fracture propagation and proppant transport. The corresponding dynamic model is obtained using lubrication theory (fluid momentum) and elasticity equation (rock deformation). We consider one of the classic 2D hydraulic fracturing models, known as the Perkins-Kern-Nordgren (PKN) model [86, 87]. This model is usually applied to the cases of long fracture length (hundreds of meters in length, shown as the x-axis in Figure 2.1), constant height (y-axis) and a small width (z-axis). The rock bed in which the fractures propagate is assumed to be a homogeneous elastic formation characterized by a constant Young's modulus E and Poisson's ratio ν. Since the fracture length is considered much larger compared to other dimensions, the net pressure is independent of the y-axis making the cross-section elliptical, the most defining feature of the PKN model.

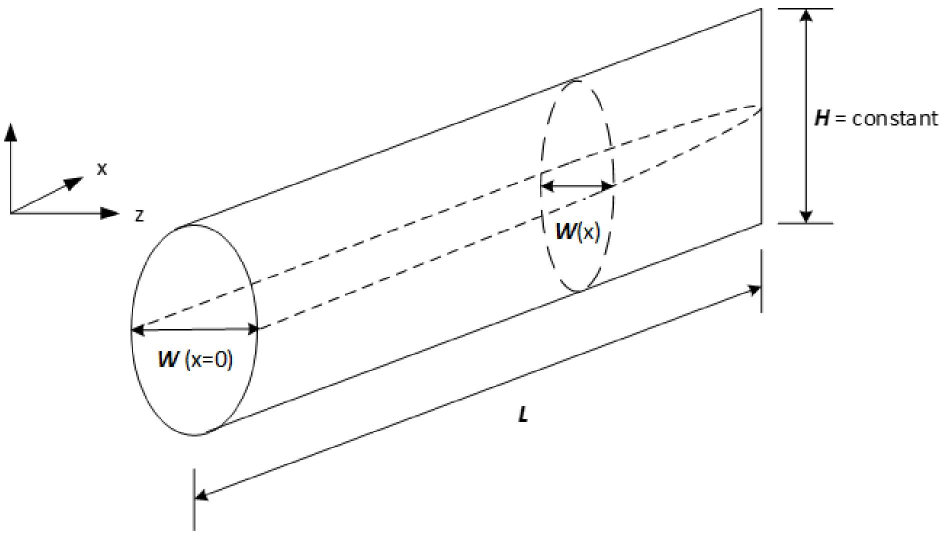


Figure 2.1: The PKN fracture model considered in this work. Reprinted with permission form [88].

2.3.1 Fracture propagation

A brief description of the equations governing the fracture propagation is presented below. The fluid flow rate inside the fracture is determined by the following equation for flow of a Newtonian fluid in an elliptical section using the lubrication theory.

$$\frac{dP}{dx} = -\frac{64\mu q_x}{\pi H W^3} \tag{2.1}$$

where P is the net pressure, μ is the fracturing fluid viscosity, q_x is the local flow rate in the horizontal direction x, H is the constant fracture height and W is the width of the fracture.

The fracture width (the minor axis of the ellipse) caused by the pressure due to the fluid loading is calculated from the solution of the plane strain condition (elasticity equation) as follows

$$W = \frac{2PH(1-\nu^2)}{E} \tag{2.2}$$

By taking into account the fracture volume changes and the fluid leak-off into the surrounding reservoir, the continuity equation gives the (incompressible) fluid volume conservation as

$$\frac{\partial A}{\partial t} + \frac{\partial q_x}{\partial x} + HU = 0 \tag{2.3}$$

where $A = \pi WH/4$ is the cross-sectional area of the elliptic fracture, and $U(t)$ is the fluid leak-off rate per unit height during the fracture propagation which is given by the Carter's equation [77]

$$U = \frac{2C_{\text{leak}}}{\sqrt{t-\tau(z)}} \tag{2.4}$$

where C_{leak} is the overall leak-off coefficient, t is the elapsed time since fracturing was initiated, and $\tau(z)$ is the time at which a specific fracture location gets exposed for the first time.

At the wellbore, the flow rate q_z is specified, and at the fracture tip $L(t)$, the fracture is always closed (i.e., the width of the fracture is zero). These lead to the following two boundary conditions:

$$q_z(0,t) = Q_0; \qquad W(L(t),t) = 0, \tag{2.5}$$

where Q_0 is the fluid injection rate at the wellbore. Initially, the fracture is closed leading to the following initial condition:

$$W(z,0) = 0 \tag{2.6}$$

2.3.2 Proppant transport

To model proppant transport, it is assumed that the injected proppant travels at the carrier fluid's velocity along the horizontal direction governed by advection while simultaneously settling at the fracture bottom forming a proppant bank induced by gravity. The following set of equations is used to describe the proppant transport phenomenon.

The advection of the suspended proppant can be expressed as:

$$\begin{aligned} &\frac{\partial(WC)}{\partial t} + \frac{\partial}{\partial x}(WCV_p) = 0 \\ &C(0,t) = C_0(t) \quad and \quad C(x,0) = 0 \end{aligned} \tag{2.7}$$

where $C(x,t)$ is the volumetric proppant concentration inside the fracture, $C_0(t)$ is the injected proppant concentration at the wellbore. The interaction between the individual proppant particles is assumed to be negligible because of the low proppant concentration. The drag and gravitational forces acting on proppant particles are considered for proppant settling. The relationship between the velocity of an individual proppant particle, V_p, the velocity of the fluid, V, and gravitational settling velocity, V_s, is given by [89]:

$$V_p = V - (1-C)V_s \tag{2.8}$$

The gravity-induced proppant settling velocity V_s can be computed as [90]

$$V_s = \frac{(1-C)^2(\rho_{sd}-\rho_f)gd^2}{10^{1.82C}18\mu} \tag{2.9}$$

where ρ_{sd} is the proppant particle density, ρ_f is the pure fluid density, g is the gravitational acceleration constant, d is the proppant diameter, and μ is the fracture fluid viscosity whose relationship with concentration can be modeled through the following empirical expression [91]:

$$\mu(C) = \mu_0\left(1-\frac{C}{C_{max}}\right)^{-\alpha} \tag{2.10}$$

where μ_0 is the pure fluid viscosity, α is an exponent in the range of 1.2 to 1.8, and C_{max} is the maximum theoretical concentration determined by $C_{max} = (1-\phi)\rho_{sd}$ where ϕ is the proppant bank porosity. The evolution of proppant bank height, δ, by the settling flux is described by [92],

$$\frac{d(\delta W)}{dt} = \frac{CV_sW}{(1-\phi)} \tag{2.11}$$

where there is initially no proppant bank, so the initial condition is that $\delta(z,0) = 0$. Please note that due to dilute suspension, as long as the operation is carried out for a short period of time the proppant bank height will remain much smaller than the fracture height $(\delta \ll H)$.

Please note that in the above equations we have not considered the effect of fracture wall roughness on the fluid transport. There have been several studies examining these effects of fracture aperture and fracture roughness on the mechanical (elastic) and hydraulic (fluid transport) response of a fractured rock. For example, the effect of roughness on mechanical response includes the stress dependence, which has a significant impact on seismic monitoring [93, 94], that is shown to be useful in potentially estimating the fracture size/length [95, 96]. Additionally, as the contact area between the fracture surfaces increases with the normal stress, or as the fracture surfaces become rough, the fluid flow pattern also changes affecting the hydraulic response of the fractured rock. Under the rough wall conditions, the fluid will not flow through the entire fracture surface but rather through a few preferred paths (channels). Several researchers have come up with models [97, 98, 99, 100, 101, 102] for studying the characteristics of fluid flow within rough fracture walls. All these models propose modifications to the lubrication (Boussinesq) equation which relates the fracture aperture and the pressure gradient via a cubic law. Ideally, under the assumption of smooth fracture walls, the fracture aperture in the equation is considered constant but in the presence of roughness, the proposed models treat the fracture aperture as a function of fracture roughness using empirical correlations. However, since all the modifications proposed will take effect only in modeling the relation between the fracture aperture and the fracture roughness, it does not significantly affect the data assimilation process adopted later in this manuscript. The procedure would remain the same with some changes to the above dynamic model of the hydraulic fracturing process.

An important characteristic of the hydraulic fracturing process is that as the fracture propagates in the lateral direction, the system boundary changes making the spatial domain of interest time-dependent. An efficient coupling of multiple nonlinear equations that describe the important physical phenomena in hydraulic fracturing systems is essential to update important variables at each time step. Due to the moving boundary nature of the problem, the number of equations to be solved grows as a fracture treatment continues, significantly increasing the computational requirements. Therefore, the aim of this work is to explore the use of Koopman operator theory to determine a simplified representation of the nonlinear dynamics such that the use of established control design methodologies becomes readily applicable.

2.4 Full-order simulation

The data for fracture propagation dynamics required for constructing the approximate models in the remainder of this dissertation are obtained by solving the high-fidelity model described by Eqs. (2.1)-(2.11). To deal with the non linearity and moving boundary nature, a novel in-house numerical scheme based on a periodic re-meshing strategy was used (the readers can refer to [85] for more details).

One of the most commonly used methods for dealing with time-dependent spatial domains is to use coordinate transformation; however, it requires an explicit expression for the growth of the boundary $l(t)$ to normalize the spatial coordinate [103]. Such an expression is not available a priori for this particular application. In such cases, we can divide, the spatial (or temporal) coordinate into equal intervals and solve the system of algebraic equations to determine the corresponding grid size in the other coordinate so that the boundary always remains at a grid point [104, 105, 106]. Based on this, a novel numerical scheme has been developed for solving the above governing equations by effectively handling the issues with the time-dependent spatial domain and coupling of nonlinear equations. For the sake of completeness, the numerical solution procedure is briefly described below.

2.4.1 Meshing strategies

A one-dimensional grid system is generated to represent the created fracture geometry. To deal with a time-dependent spatial domain, there are two widely used meshing strategies: moving meshing and periodic re-meshing of a fixed domain. While the former method provides a less accurate solution (because of the limited number of meshes) with a reasonable computational requirement, the latter allows for an accurate solution at the expense of CPU time (the degree of re-meshing could lead to an interpolation error in the solution). In order to capture the detailed process dynamics of the system that has a boundary condition of spatial domain which is subject to change, a fixed mesh strategy is employed by additionally adapting the size of integration time step.

2.4.2 Numerical solution procedure

The steps of the numerical algorithm are shown below:

1. At time step t_k, the fracture length $L(t_{k+1})$ is obtained by elongating the fracture tip by Δz, $L(t_{k+1}) = L(t_k) + \Delta z$.

2. The coupled equations of Eqs. (2.1)–(2.11) are solved for the fracture width $W(z, t_{k+1})$, the net pressure $P(z, t_{k+1})$, the flow rate $q_z(z, t_{k+1})$, and the proppant concentration $C(z, t_{k+1})$ across the fracture via a finite element method.

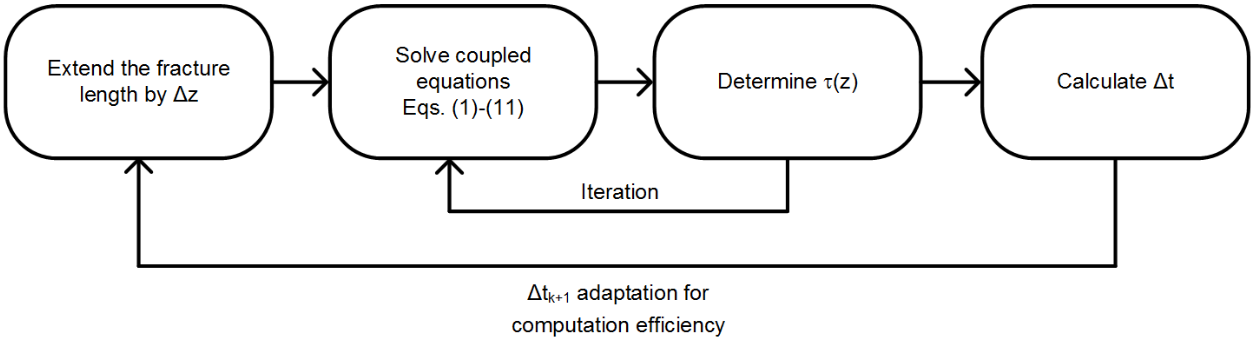


Figure 2.2: The steps of numerical simulation for fracture propagation. Modified with permission from [107].

3. Calculate $\tau(z_{k+1})$ in Eq. (2.4) iteratively by repeating Steps 2 and 3.

4. The time interval Δt_{k+1} is determined. For computational efficiency, the size of a numerical integration step can be adapted based on the Courant-Friedrichs-Lewy (CFL) number.

5. Set $k \to k + 1$ and go to Step 1.

For one-dimensional case, the CFL condition has the following form for explicit numerical schemes:

$$\frac{u\Delta t}{\Delta z} \leq 1 \tag{2.12}$$

where $u = |dW/dt|$ is the fracture width growth rate. This technique, improving the computational efficiency by increasing the CFL number, has been widely accepted [108]. In this work, Δz is fixed, and u increases with spatial domain (i.e., the fracture width changes more rapidly near the fracture tip compared to that near the wellbore), which provides room for improvement in the computational efficiency by increasing Δt near the wellbore.

2.5 High-fidelity simulation results

Using aforementioned techniques, an accurate full-order solution of the PDE system was obtained using the finite element method. The values of various process parameters used in our calculations were obtained from [83] and tabulated below. The fracture propagation was terminated at 135 m, and the spatial domain was discretized with each grid point having a size of 0.3 m, resulting in a total of $n = 451$ points. The total number of temporal snapshots obtained per trajectory are 19611. The spatio-temporal evolution of the fracture width for a sample injection flow rate of $Q_0 = 0.05 \text{ m}^3/\text{s}$ is shown in Figure 2.3. It can be observed that the growth rate of the fracture width is very high in the beginning but it slows down with time. At the same time, the fracture length grows steadily at a constant rate. The moving boundary nature of the problem is evident in the above figure with the zero values at a spatial point indicating that the fracture has not yet propagated to that location. Similarly, the numerical experiments were carried out for random input profiles to collect the concentration snapshots and construct the required data matrices for system identification. The full-order solution of the proppant concentration inside the fracture for a sample injected concentration ($u(t) = C_0(t)$) is shown in Figure 2.4. Please note that the results obtained from the numerical solutions have not been validated against any experimental data.

Parameter	Symbol	Value	Units
Leak-off coefficient	C_{leak}	6.3×10^{-5}	$m \cdot s^{-1/2}$
Maximum concentration	C_{max}	0.64	-
Young's modulus	E	5×10^{-9}	Pa
Proppant permeability	k_f	60000	mD
Formation permeability	k_r	1.5	mD
Vertical Fracture height	H	20	m
Proppant particle density	ρ_{sd}	2648	kg/m^3
Pure fluid density	ρ_f	1000	kg/m^3
Fracture fluid viscosity	μ	0.56	$Pa \cdot s$
Poisson ratio of formation	ν	0.2	-

Table 2.1: Model parameters used for the full-order simulation of the hydraulic fracturing dynamics. Adapted with permission from [107]

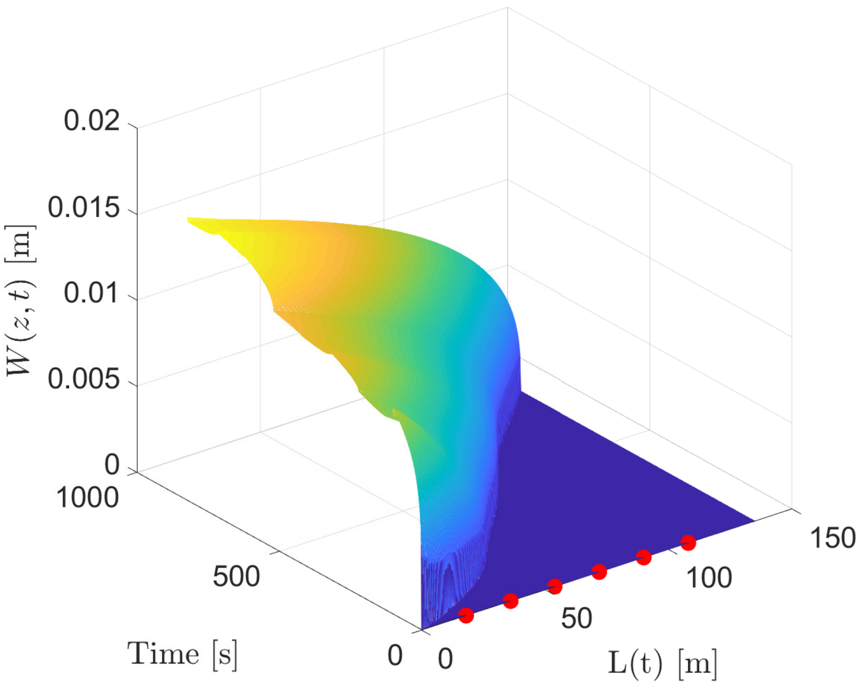


Figure 2.3: Spatio-temporal evolution of fracture width. The red markers indicate sensor placement for width measurements. Reprinted with permission from [109]

2.5.1 Measurement data

Because the hydraulic fracturing process takes place deep below the earth's surface, the availability of width measurements at all the spatial locations is not guaranteed. In fact, in a typical hydraulic fracturing process, real time measurements may include downhole pressure and microseismic data which will be processed to provide the fracture width at the wellbore and the fracture length, respectively [110, 111, 112]. Microseismic monitoring (MSM) is the detection of signals produced by small seismic events and is used for determining the location and geometry of fractures. When a crack occurs in a rock formation, energy is released, referred to as microseismicity, and some of the energy travels away from the source through the surrounding rock as seismic waves. These waves temporarily deform the material as they travel in the form of P-waves, or primary waves, which are relatively fast propagating waves and S-waves, or secondary waves, which are slower-propagating shear waves. MSM operations incorporate arrays of three-component geophones or accelerometers deployed in a nearby monitoring well that pick up these seismic events. The distance of the originating source of a microseismic event to each geophone is calculated based on the difference in arrival time between P- and S- waves and a previously calibrated seismic-wave velocity model. The detection results of the same event by three or more geophones allow determination of the location of the event source in three-dimensional (3D) space. Analysts interpret these MSM locations to show induced fracture length, height and azimuth. Downhole pressure data can be used to estimate the fracture width near the wellbore via pressure-width relationship equations.

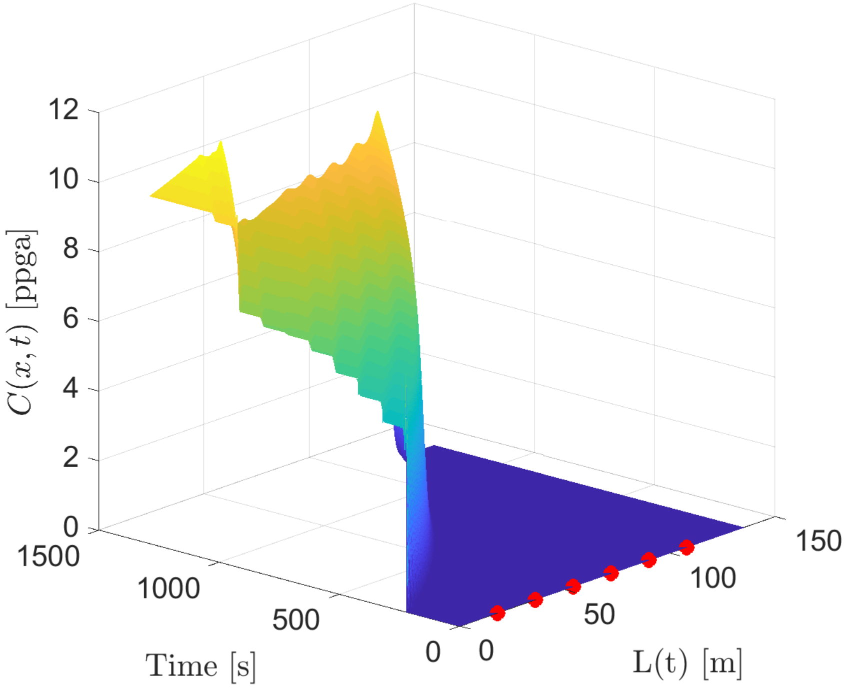


Figure 2.4: Spatio-temporal evolution of proppant concentration. The red markers indicate sensor placement for measurements. Reprinted with permission from [109].

Therefore, unless otherwise specified, in this work, only a partially observed system is considered; the average fracture width, $W_{avg}(t)$, fracture length, $L(t)$ and fracture width at six uniformly spaced locations, $[W_1(t), \cdots, W_6(t)]$, obtained from the numerical experiment were considered as the true process measurements, i.e., $x(t) = [W_{avg}(t), L(t), W_1(t), \cdots, W_6(t)]^T \in \mathbb{R}^8$.

2.6 Research objectives

The overall objective of this doctoral study is to develop accurate data-driven approximations for feedback control of complex nonlinear dynamical systems. From the application point of view, the developed methods are applied and their performance is validated on a hydraulic fracturing process with simultaneous fracture propagation and proppant transport. For this specific application, the control objective is to design a pumping schedule that regulates the spatial variation of proppant concentration and fracture geometry across the fracture at the end of pumping for shale reservoirs. The unifying theme throughout this research is to achieve this from the viewpoint of operator formalism of dynamical systems. The specific objectives of this research are:

1. Develop a novel framework for model order reduction by considering local reduced-order bases which are particularly tailored for systems characterized by moving boundaries or parameter variations where approximating the solution of interest in a fixed lower-dimensional subspace of global basis vectors may fail.

2. Extend the idea of model reduction to the parameter space to develop an integrated framework that can successfully handle ill-posed inverse problems such as those encountered in oil and gas simulations (history-matching problems).

3. Study the connection between operator theory and spectral decomposition-based methods to address the drawbacks of local model-reduction and develop a principled framework for generating highly accurate approximate models while still maintaining physical interpretability.

4. Leverage Koopman operator theory to design stable feedback controllers for nonlinear systems. Analyze and comment on the stability analysis of the original nonlinear system under the implementation of the designed controller in the Koopman space.

2.7 Dissertation layout

Chapter 3 outlines the proposed local model reduction framework on a complex nonlinear dynamical system characterized by moving boundaries. Specifically, we develop local reduced-order models using POD to generate the local basis functions and Galerkin projection is used to project onto the local subspaces to yield local models. Next, DMDc is used within the developed framework to address some of the limitations of POD. Specifically, DMDc results in local linear time invariant dynamical systems within each sub region leading to a completely equation-free architecture for local model reduction. We then use the obtained local linear models within a MPC formulation to design an optimal and practical pumping schedule to achieve a uniform proppant concentration across fracture at the end of pumping. Finally, the generated pumping schedule is applied to the high-fidelity hydraulic fracturing model, and the performance is compared with an existing system identification method.

In Chapter 4, the concept of model reduction is extended to the parameter estimation problem for controller design. Specifically, POD is used to perform parameterization to reduce the dimensionality of spatially varying Young's modulus profiles in a conventional rock formation and it is combined with a data assimilation technique, the ensemble Kalman filter (EnKF), to estimate true parameter values in the reduced low-dimensional subspace. Then, the local model reduction framework developed in Chapter 3 is used to design feedback controllers to achieve uniform proppant concentration in conventional oil reservoirs by explicitly taking into account the desired fracture geometry and the total amount of injected proppant.

Although the local model reduction framework shows improved accuracy, it is susceptible to generating unstable models due to discontinuity between various clusters. In some cases, having a single model which can approximate the system in the entire state-space would be beneficial. In Chapter 5, we explore the idea of nonlinear transformation for model approximation by shifting the focus from state-space to a space of functions (of the state) which forms the second part of this study; using Koopman operator theory to develop linear predictors for nonlinear dynamical systems. Koopman theory has been developed to analyze flow fields by the fluids community, however its use in the field of chemical engineering is very limited. Before its application to chemical process systems, there are some key challenges that need to be addressed. Particularly, coming up with a relevant choice for the required nonlinear transformation is not trivial and will be the focus of our research efforts. We outline a proposal for sparse identification of the basis functions that are used as transformed coordinates. We solve a sparse regression problem over a large set of candidate functional forms to determine the basis. The method balances model complexity and accuracy by selecting a sparse model that avoids over fitting to accurately represent the system dynamics when subjected to a different input profile. We implement the idea on the hydraulic fracturing application.

In Chapter 6, we aim to leverage the fact that Koopman theory will result in linear embedding even for nonlinear systems to develop a stabilizing control framework for nonlinear systems. Although Koopman MPC has been successfully applied for data-driven control of nonlinear flows, the tightness of the Koopman linear predictors is not established and the closed-loop stability is not guaranteed in the literature. Therefore, we propose to study the stability properties of the data-driven Koopman based controllers. Specifically, we look to integrate the Koopman linear predictors with Lyapunov-based model predictive controller (LMPC) to achieve feedback stabilization of the original nonlinear system.

In Chapter 7, we improve upon the idea presented in Chapter 6 to design data-driven stabilizing feedback controllers which do not require explicit mathematical expression of the original nonlinear dynamics, which is becoming increasingly common for complex processes. Moreover, the controllers are completely formulated in the Koopman space with the help of Koopman eigenfunctions and we show that under some conditions, the stability of the closed-loop system in the Koopman space can be easily translated to the original system under the implementation of the developed predictive controllers.

3. A FRAMEWORK FOR LOCAL MODEL REDUCTION[*]

3.1 Introduction

As discussed earlier, a single global reduced-order model may not accurately reproduce the dynamics of a complex nonlinear system especially if the system features moving boundaries or varying parameters. In such cases, in order to capture the local dynamics of a complex nonlinear system more effectively, the basis used for projection must be locally tailored to capture the behavior of every portion of the solution trajectory. The concept of exploiting local bases to accurately describe complex nonlinear systems has been the subject of intensive research in the field of applied mathematics in recent years. A simple aspect of this concept was introduced in [113] where the time domain was partitioned into multiple subdomains and local basis functions were used to construct a reduced-order model. Local bases were also exploited in [114] in the context of aeroelastic applications according to a space domain partition. The concept of partitioning was extended to combine with a machine learning technique to reduce the computational requirement of an estimator design in [115].

In our work, we first partition the time domain into multiple clusters where the snapshots contained within each cluster exhibit a similar behavior with each other and are relatively dissimilar with the ones contained within the other clusters. This is done to ensure that every cluster approximately represents similar dynamics. For practical implementation of the data clustering we use a global optimum search (GOS) algorithm. The GOS is an intuitive and robust Mixed-Integer Nonlinear Programming (MINLP) based clustering algorithm developed in [116]. The GOS algorithm explicitly seeks to minimize the Eucledian distances between data points and the cluster centers. More importantly, the GOS clustering algorithm incorporates a convenient method to predict the

---

[*] Reprinted with permission from "Temporal clustering for order reduction of nonlinear parabolic PDE systems with time-dependent spatial domains: Application to a hydraulic fracturing process," by A. Narasingam, P. Siddhamshetty and J. S. Kwon, AIChE Journal, 2017, 63, 3818-3831. Copyright 2017 by John Wiley and Sons.

[*] Reprinted with permission from "Development of local dynamic mode decomposition with control: Application to model predictive control of hydraulic fracturing," by A. Narasingam and J. S. Kwon, Computers & Chemical Engineering, 2017, 106, 501-511. Copyright is retained by the authors for all articles published in Elsevier journals.

optimal number of clusters by not only minimizing the sum of the intra-cluster distances, but also maximizing the sum of the inter-cluster distances. Although it does not theoretically guarantee a global optimum, the GOS algorithm tends to give good results and performs favorably over other traditional heuristic algorithms including k-means [116].

3.2 Schematic of the proposed framework

The following points outline the proposed methodology which are discussed in detail below.

1. Initially, the nonlinear parabolic partial differential equation (PDE) system is solved using a high-order discretization scheme, or alternatively using data collected from experiments, to construct a representative ensemble of solutions.

2. The GOS algorithm is then applied on the solution snapshot set to obtain an optimal number of clusters and the corresponding cluster configuration. The GOS algorithm solves the optimization problem by formulating it as an MINLP problem.

3. Then, a projection-based model reduction technique is applied to the solution snapshots contained within each cluster to obtain a set of local basis functions that capture the dominant spatial characteristics of the solution profile.

4. The computed basis functions are used to derive low-order systems within each cluster. The derived reduced-order models accurately capture the dominant dynamics of the original nonlinear system and can be numerically integrated to approximate the full-order solution.

A schematic representation elaborating the different steps of the proposed local model order-reduction methodology is presented in Figure 3.1.

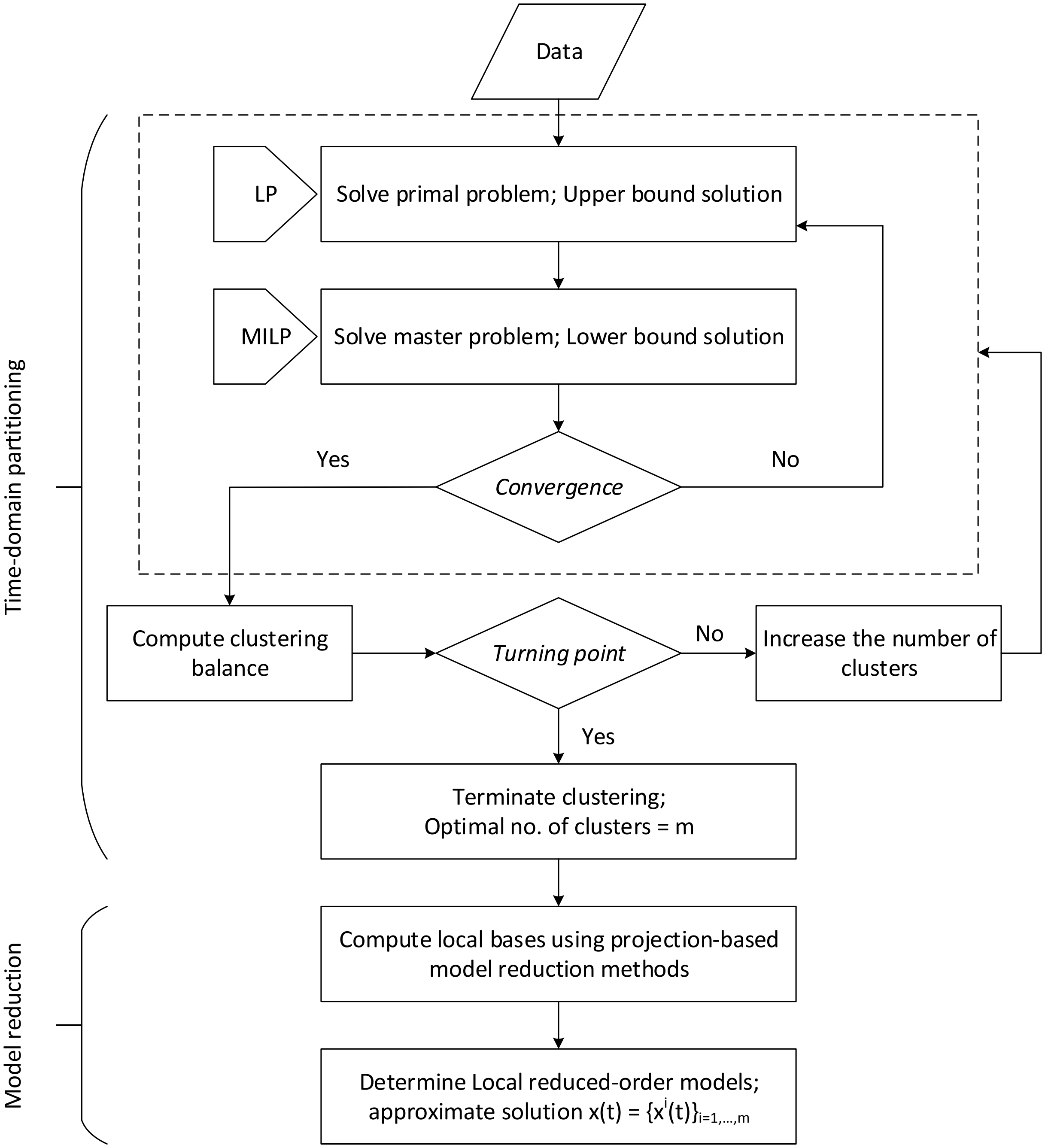


Figure 3.1: Flow diagram of local model reduction framework.

3.3 Mathematical preliminaries

3.3.1 Parabolic PDE systems with time-dependent domains

A general quasi-linear parabolic PDE system with time-dependent spatial domains can be represented with the following formulation [117]:

$$\frac{\partial \bar{\mathbf{x}}}{\partial t} = \mathcal{L}_1(\bar{\mathbf{x}}) + \mathcal{L}_2(\bar{\mathbf{x}}) + f(t, \bar{\mathbf{x}}) \tag{3.1}$$

subject to the moving boundary conditions,

$$\begin{aligned} C_1\bar{\mathbf{x}}(0,t) + D_1\frac{\partial \bar{\mathbf{x}}}{\partial z}(0,t) = \mathbf{R_1} \\ C_2\bar{\mathbf{x}}(l(t),t) + D_2\frac{\partial \bar{\mathbf{x}}}{\partial z}(l(t),t) = \mathbf{R_2} \end{aligned} \tag{3.2}$$

and the initial condition,

$$\bar{\mathbf{x}}(z,0) = \bar{\mathbf{x}}_\mathbf{0}(z) \tag{3.3}$$

where $\bar{\mathbf{x}}(z,t) = [x_1(z,t), \dots, x_n(z,t)]^T$ denotes the vector of state variables, $z \in [0, l(t)] \subset \mathbb{R}$ is the domain of spatial coordinate z, $t \in [0, \infty]$ is the time coordinate, $\mathcal{L}_1(\bar{\mathbf{x}})$, $\mathcal{L}_2(\bar{\mathbf{x}})$ are nonlinear differential operators which involve the first- and second-order spatial derivatives respectively, $f(t, z, \bar{\mathbf{x}})$ is a nonlinear vector function, C_1, D_1, C_2, D_2, are constant matrices, $\mathbf{R_1}$, $\mathbf{R_2}$ are column vectors and $\bar{\mathbf{x}}_\mathbf{0}(z)$ is the initial condition.

It is easy to see that the hydraulic fracturing application considered in this work falls under this category. However, the difficulty with the fracturing system shown in Eqs. (2.1)-(2.11) is that an explicit expression for the displacement of the boundary $z = l(t)$ is unknown.

3.3.2 Global Optimum Search

The proposed nonlinear MOR technique is based on the computation of temporally-local eigenfunctions. Within this context, the time domain of the solution is divided into subdomains (clusters) in such a way that the snapshots contained within each cluster exhibit a similar nonlinear behavior as compared to that of the others. Although there are a large number of clustering algorithms prevalent in the literature, the one selected here is the GOS clustering technique developed in [116] The GOS clustering algorithm is a robust yet intuitive clustering algorithm formulated as a Mixed-Integer Nonlinear Programming (MINLP) problem. Essentially, it tries to minimize the distances between the snapshots and the centers of their assigned clusters, and is formulated as:

$$\begin{aligned} & \underset{y_{jk}, c_{ki}}{Minimize} && \sum_{i=1}^{s}\sum_{j=1}^{n}\sum_{k=1}^{m} y_{jk}(x_{ij}-c_{ki})^2 \\ & \text{s.t} && \sum_{k=1}^{m} y_{jk} = 1, \quad \forall j = 1, \ldots, n \end{aligned} \tag{3.4}$$

where y_{jk} are binary variables which indicate whether a snapshot j falls within the cluster k and c_{ki} are continuous variables representing the cluster centers (i.e., center of mass obtained by averaging the snapshots contained within the cluster) and m is the total number of clusters. Furthermore, simplifying the objective function by expanding the quadratic equation and incorporating the necessary first order optimality condition, the optimization problem is formulated as:

$$\begin{aligned} & \underset{y_{jk}, c_{ki}}{Minimize} && \sum_{i=1}^{s}\sum_{j=1}^{n} x_{ij}^2 - \sum_{i=1}^{s}\sum_{j=1}^{n}\sum_{k=1}^{m}(x_{ij}y_{jk}c_{ki}) \\ & \text{s.t} && c_{ki}\sum_{j=1}^{n} y_{jk} - \sum_{j=1}^{n} x_{ij}y_{jk} = 0, \quad \forall i, k \\ & && \sum_{k=1}^{m} y_{jk} = 1, \quad \forall j = 1, ..., n \\ & && 1 \leq \sum_{j=1}^{n} y_{jk} \leq n - m + 1 \\ & && y_{jk} \in \{0, 1\}, \quad \forall j, k \\ & && c_{ki}^{L} \leq c_{ki} \leq c_{ki}^{U}, \quad \forall i, k \end{aligned} \tag{3.5}$$

In the above expression the bounds of c_{ki} are obtained by observing the range of snapshot data:

$$c_{ki}^{L} = min\{x_{ij}\}, \quad \forall i = 1, \ldots, s$$

$$c_{ki}^U = max\{x_{ij}\}, \quad \forall i = 1, \ldots, s$$

where c_{ki}^L is the lower and c_{ki}^U is the upper bound of c_{ki}, respectively. The necessary optimality condition is included in the first set of constraints, the second requires that each snapshot be placed in only one cluster, and the third specifies that there is at least one and no more than $(n - m + 1)$ snapshots in a cluster. The above optimization problems are MINLPs with bilinear terms and can be solved by using various available relaxation techniques which seek to convert them into Mixed-Integer Linear Programming (MILP) problems. But these approaches rely on introducing additional variables and constraints which makes the problem computationally expensive especially if the number of data points is large. Thus, the GOS algorithm was developed as an alternative based on a variation of the Generalized Benders Decomposition (GBD) [118]. For brevity, only an outline of the algorithm as applied to the context of clustering is presented here. For a more detailed description, we refer the readers to [116, 118, 119]. In brief, the GBD method decomposes the problem into a primal and the master problem. The primal problem is solved for continuous variables while fixing the integer (binary) variables and the objective function value provides an upper bound to the original problem. The master problem is derived from the nonlinear duality theory, making use of Lagrange multipliers and continuous variables obtained in the primal problem, and its solution provides the lower bound on the original problem. The two sequences of upper and lower bounds are iteratively updated until they converge to a finite value. The primal and the master problem formulations and the algorithms are presented below:

Step1 Primal problem

With fixed starting values for the binary variables, i.e, y^*_{jk} where it is the optimal value obtained by solving the master problem, the primal problem becomes a Linear Programming (LP) problem:

$$\begin{aligned}
&\underset{c_{ki}}{Minimize} && \sum_{i=1}^{s}\sum_{j=1}^{n} x_{ij}^2 - \sum_{i=1}^{s}\sum_{j=1}^{n}\sum_{k=1}^{m}(x_{ij}y^*_{jk}c_{ki}) \\
&\text{s.t} && c_{ki}\sum_{j=1}^{n} y^*_{jk} - \sum_{j=1}^{n} x_{ij}y^*_{jk} = 0, \quad \forall i,k \\
& && c^L_{ki} \le c_{ki} \le c^U_{ki}, \quad \forall i,k
\end{aligned} \tag{3.6}$$

In the primal we include only those constraints that involve c_{ki} and the rest are dropped out. The LP is solved to obtain c_{ki}, and the Lagrange multipliers λ_{ki} for each of the constraints. The objective function value is the upper bound of the solution to the original MINLP problem, Eq. (3.5).

Step 2 Master problem

The values obtained from the primal, c^*_{ki} and λ^*_{ki}, are inputted into the master problem which makes it a Mixed Integer Linear Programming (MILP) problem:

$$\begin{aligned}
&\underset{y_{jk},\mu_B}{Minimize} && \mu_B \\
&\text{s.t} && \mu_B \ge \sum_{i=1}^{s}\sum_{j=1}^{n} x_{ij}^2 - \sum_{i=1}^{s}\sum_{j=1}^{n}\sum_{k=1}^{m}(x_{ij}y_{jk}c^*_{ki}) \\
& && \quad + \sum_{i=1}^{s}\sum_{k=1}^{m} \lambda^*_{ki}\left(c^*_{ki}\sum_{j=1}^{n} y_{jk} - \sum_{j=1}^{n} x_{ij}y_{jk}\right) \\
& && \sum_{k=1}^{m} y_{jk} = 1, \quad \forall j = 1,\ldots,n \\
& && 1 \le \sum_{j=1}^{n} y_{jk} \le n - m + 1, \quad \forall k \\
& && y_{jk} = 0 - 1, \quad \forall j,k
\end{aligned} \tag{3.7}$$

The master problem solves for y_{jk} and μ_B and provides the lower bound to the solution of the original MINLP problem, Eq. (3.5).

Step 3 Termination

The solutions from the master problem, y^*_{jk}, are inputted in the primal problem and steps 1 and 2 are repeated until the solution converges (i.e a termination criterion is met). The termination criterion for GBD is based on the difference between the updated upper bound and the current lower bound. If this difference is less than or equal to a pre-specified tolerance $\epsilon \geq 0$ then we terminate.

Remark 1. *Note that in a typical GBD algorithm, there may be an infeasible primal problem, which should be handled by reformulation of the problem statement. In this work, we can always find a feasible assignment of snapshots to each cluster. Moreover, there is only one set of continuous variables (i.e., c_{ki}). Both these features render all primal problems feasible in our case.*

Remark 2. *The best feasible solution of the primal problem found up to the current iteration represents the upper bound. The solution of the master problem provides the lower bound and this lower bound increases monotonically because we keep adding constraints to it as the iterations increase, effectively demanding the solution of the master problem in the current iteration to be larger than or equal to the previous one. Hence, there will always be a point at which the upper and lower bounds converge.*

Remark 3. *Note that, in the primal problem the binary variables are fixed to different 0-1 combinations and solved. Hence, there is no reason for the obtained upper bounds to satisfy any monotonicity property (unlike the lower bounds from the master problem which increase with the iterations). Therefore, the upper bound is updated only when it is less than or equal to the previous step.*

Remark 4. *Note also that while the GOS algorithm tends to give good optimal solutions, it does not have a theoretical guarantee to return a global optimum. Providing a good initial point is necessary to ensure that we obtain a good (in a sense that it is better than other heuristic approaches)* *result. A good initial point will depend on the nature of the data being clustered. That being said, we suggest implementing a pre-clustering strategy which will incorporate any knowledge of the process data. For example, if certain snapshots are not to be arranged in the same cluster because of process constraints, the pre-clustering step will help in addressing this by placing these snapshots uniquely into different clusters by adding this constraint implicitly to the optimization problem.*

Step 4 Optimal number of clusters

For a cluster configuration to be optimal, not only the members within a cluster must be similar in behavior with each other (characterized by the minimum intra-cluster error sum) but at the same time they should be dissimilar with the members of other clusters (characterized by inter-cluster error sum which is to be maximized in our setting). Both these sums are combined to define a metric called clustering balance which is used to predict the optimal number of clusters. The clustering balance ϵ, initially proposed in [120], is the weighted sum of the intra-cluster error sum Λ and the inter-cluster error sum Γ. We note here that assigning different weight factors in the clustering balance term might result in different data clustering. Based on the problem specifications, one can assign different weights which will put more emphasis on either the similarity of snapshots within each cluster or the dissimilarity between different clusters. This choice of weights is completely at the discretion of the user and has to be made depending on the problem specifications. For a general case, it has been shown in [116] that the appropriate weight between Γ and Λ is 0.5.

$$\begin{aligned}
&\text{Global center,} && c_i^\circ = \frac{1}{n}\sum_{j=1}^{n} x_{ij}\\
&\text{Intra-cluster error sum,} && \Lambda = \sum_{i=1}^{s}\sum_{j=1}^{n}\sum_{k=1}^{m} y_{jk}\|x_{ij} - c_{ki}\|_2^2\\
&\text{Inter-cluster error sum,} && \Gamma = \sum_{i=1}^{s}\sum_{k=1}^{m} \|c_{ki} - c_i^\circ\|_2^2\\
&\text{Clustering balance,} && \epsilon = \frac{1}{2}(\Lambda + \Gamma)
\end{aligned} \tag{3.8}$$

Once steps 1 to 3 are completed, the algorithm then proceeds to calculate the inter-cluster error sum with the updated cluster centers and thereby the clustering balance. The iterative steps 1 through 4 are repeated by selecting a new initialization point and running the algorithm again. Owing to the decreasing nature of intra-cluster error sum and increasing nature of inter-cluster error sum with the number of clusters, a turning point (the minimum) in the clustering balance is attained and the algorithm terminates. We now have determined the optimal number of clusters and the corresponding cluster configuration.

3.4 Computational methods

Chapter 3.4 describes the practical implementation of the proposed local model reduction framework using two commonly used spectral decomposition methods: POD and DMD. Using each method, we study the performance of the local framework in accurately approximating the hydraulic fracturing dynamics.

3.4.1 Local Proper Orthogonal Decomposition

3.4.1.1 Proper Orthogonal Decomposition

Proper Orthogonal Decomposition, also known as Karhunen-Loève expansion, is a widely used MOR technique. It essentially provides a set of orthogonal bases that span the solution subspace of a dynamic system. The bases are then used in conjunction with a projection method to represent the high-dimensional model (or the given data) with a certain low-dimensional reduced-order system. POD uses a modal decomposition that is completely data-dependent without assuming any knowledge of the underlying process.

In Chapter 3.4.1.1, we review the mathematical formulation for proper orthogonal decomposition in the context of nonlinear parabolic PDE systems. Let x_k be the representative ensemble of solutions which is constructed by solving the PDE via a high-order discretization scheme or alternatively, using available process data obtained from experiments. The POD method aims at obtaining the most dominant spatial patterns of this ensemble. This is equivalent to finding the basis functions $\phi(z)$ that maximizes the ensemble average of the inner products between $x_k(z)$ and

$\phi(z)$ [103, 121]:

$$\begin{aligned} &Maximize \quad \langle(\phi(z), x_k(z))^2\rangle \\ &\text{s.t} \qquad\qquad ((\phi(z), \phi(z)) = 1 \end{aligned} \tag{3.9}$$

where $\langle\cdot\rangle$ denotes the averaging operation, $(\cdot,\cdot) = \int_0^{l(t)} (\cdot,\cdot)_{\mathbb{R}^n}\, dz$, and $(\cdot,\cdot)_{\mathbb{R}^n}$ denotes the standard inner product in $\mathbb{R}^n$. The constraint imposed in Eq. (3.9) makes the computation unique and can be taken into account by the use of a Lagrange multiplier, λ:

$$L[\phi] = \langle(\phi(z), x_k(z))^2\rangle - \lambda((\phi(z), \phi(z)) - 1) \tag{3.10}$$

The optimization problem described by Eq. (3.9) seeks to find a solution such that the average energy content is greater if the field x is projected along ϕ, than along any other basis function. The necessary condition for reaching the extremum is that the functional derivative is equal to zero. According to [103], this condition can be formulated as the following integral eigenvalue problem:

$$\begin{aligned} &R\phi = \lambda\phi \longrightarrow \int_0^{l(t)} K(z,z')\phi(z')dz' = \lambda\phi(z) \\ &K(z,z') = \langle(x_k(z), x_k(z'))\rangle = \frac{1}{N}\sum_{k=1}^{N} x_k(z)x_k(z') \\ &R := \int_0^{l(t)} K(z,z')dz' \end{aligned} \tag{3.11}$$

where $K(z,z')$ is the averaged auto correlation function, N is the total number of snapshots and R is a linear operator defined as above. The computation of the above integral eigenvalue problem is an expensive task. To overcome this problem, Sirovich introduced the method of snapshots [32]. The basic idea is to assume the eigenfunction can be expressed as a linear combination of the snapshots:

$$\phi(z) = \sum_{\kappa} c_\kappa x_\kappa(z) \tag{3.12}$$

Substituting the above expression for $\phi(z)$ in Eq. (3.11), we obtain the following eigenvalue problem:

$$\begin{aligned}
&\int_0^{l(t)} \frac{1}{N}\sum_{k=1}^{N} x_k(z)x_k(z')\Big(\sum_{\kappa=1}^{N} c_\kappa x_\kappa(z')\Big) = \lambda \sum_{\kappa=1}^{N} c_\kappa x_\kappa(z) \\
&B^{k\kappa} := \frac{1}{N}\int_0^1 x_k(z')x_\kappa(z')dz' \\
&\mathcal{B}\mathbf{c} = \lambda\mathbf{c}
\end{aligned} \tag{3.13}$$

where $\mathcal{B}$ is the $N \times N$ correlation matrix comprised of elements $B^{k\kappa}$ and $\mathbf{c} = [c_1, c_2, \ldots, c_N]$ are the set of eigenvectors corresponding to λ. These eigenvectors can be used to construct the eigenfunctions $\phi(z)$, also called the proper orthogonal modes, using Eq. (3.12). Next, we find a d-dimensional subspace $V_d \subset V$ by using the first d eigenvalues ($\lambda_1 \geq \cdots \lambda_d \geq \cdots \lambda_N$), and their corresponding eigenvectors, that describe the dominant spatial patterns of the system. The main result of the POD method is that the subspace V_d representing the data is given by $V_d = span\{\phi_1, \phi_2, \ldots, \phi_d\}$.

Remark 5. *The corresponding eigenvalues λ_k, $k = 1, \ldots, N$, also called the proper orthogonal values, are real and non-negative since $\mathcal{B}$ is a symmetric, positive semi-definite matrix.*

Remark 6. *Once the calculated eigenvalues are normalized, they represent the energy percentage which is the fraction of time spent by the solution along the spatial structure of the particular eigenfunction.*

Remark 7. *The obtained eigenfunctions satisfy orthonormality:*

$$\int_0^{l(t)} \phi_i(z)\phi_j(z)dz = \begin{cases} 0 & when \quad i \neq j \\ 1 & when \quad i = j \end{cases} \tag{3.14}$$

3.4.1.2 Galerkin's Method

In Chapter 3.4.1.2, we present the Galerkin's method to derive low-dimensional ODE systems that accurately reproduce the dynamics of the nonlinear parabolic PDE system defined by Eq. (3.1). In this respect, we first formulate the PDE system in a Hilbert space $\mathcal{H}(t)$, consisting of n-

dimensional vector functions defined on the spatial domain, that satisfies the boundary conditions of Eq. (3.2), by defining the state function x on $\mathcal{H}(t)$ as [103, 122]:

$$x(t) = \bar{\mathbf{x}}(z,t), \quad t > 0, \quad z \in [0, l(t)] \tag{3.15}$$

and operator $\mathcal{A}$ as:

$$\mathcal{A}x = \mathcal{L}_1(\bar{\mathbf{x}}) + \mathcal{L}_2(\bar{\mathbf{x}}) + \frac{\dot{l}}{l(t)} z \frac{\partial \bar{\mathbf{x}}}{\partial z} \tag{3.16}$$

We want to note that $\frac{\dot{l}}{l(t)} z \frac{\partial \bar{\mathbf{x}}}{\partial z}$ accounts for convective transport attributed to the time-dependent domain. In the case of parabolic PDE systems with fixed domains, the term does not exist because $\dot{l}(t) \equiv 0$. Then the system of Eq. (3.1) becomes,

$$\dot{x}(t) = \mathcal{A}x + \mathcal{F}(t, x(t)), \quad x(0) = x_0 \tag{3.17}$$

where $x_0 = \bar{x}_0(z)$ and $\mathcal{F}(t, x(t)) = f(t, \bar{\mathbf{x}}(z,t))$. Now, we can apply the standard Galerkin's method to the infinite-dimensional system of Eq. (3.17) to derive a finite-dimensional system. Note that the eigenspectrum (set of all eigenvalues) of the time-varying operator $\mathcal{A}$ can be partitioned into a dominant part consisting of d slow eigenvalues and a stable complement containing the remaining fast eigenvalues [122, 123]. Therefore, the Hilbert space $\mathcal{H}$ consists of two modal subspaces $\mathcal{H}_s$ and $\mathcal{H}_f$ defined as $\mathcal{H}_s = span\{\phi_1, \phi_2, \ldots, \phi_d\}$ and $\mathcal{H}_f = span\{\phi_{d+1}, \phi_{d+2}, \ldots\}$. The state x of the system of Eq. (3.17) can be decomposed as:

$$x = x_s + x_f = P_s x + P_f x \tag{3.18}$$

where P_s and P_f are two orthogonal projection operators that project the state x onto the subspaces $\mathcal{H}_s$ and $\mathcal{H}_f$.

We can rewrite the system of Eq. (3.17) using the above decomposition as:

$$\begin{aligned}
&\dot{x}_s(t) = \mathcal{A}_s x_s + \mathcal{F}_s(t, x_s, x_f) \\
&x_s(0) = P_s x(0) = P_s x_0 \\
&\dot{x}_f(t) = \mathcal{A}_f x_f + \mathcal{F}_f(t, x_s, x_f) \\
&x_f(0) = P_f x(0) = P_f x_0
\end{aligned} \tag{3.19}$$

where $\mathcal{A}_s = P_s\mathcal{A}$, $\mathcal{F}_s = P_s\mathcal{F}$, $\mathcal{A}_f = P_f\mathcal{A}$, $\mathcal{F}_f = P_f\mathcal{F}$. We can neglect the fast modes, $x_f = 0$, and specify the dynamical system that evolves on the finite dimensional subspace as [123]:

$$\dot{x}_s(t) = \mathcal{A}_s x_s + \mathcal{F}_s(t, x_s, 0), \quad x_s(0) = P_s x_0 \tag{3.20}$$

To apply this method computationally, we write the finite-dimensional system with respect to the bases (such a basis may be obtained from POD as described in Chapter 3.4.1.1), spanning the subspace onto which it is projected:

$$x_s(t) = \sum_{i=1}^{d} a_i(t)\phi_i \tag{3.21}$$

The Galerkin's method is applied to a system of PDEs in the form of Eq. (3.20) in the following way:

$$(\dot{x}_s, \phi_j) = (\mathcal{A}_s x_s, \phi_j) + (\mathcal{F}_s(t, x_s, 0), \phi_j) \tag{3.22}$$

Substituting Eq. (3.21) for x_s and using the orthonormal properties of the eigenfunctions yields:

$$\frac{da_j}{dt} = \sum_{i=1}^{d} a_i(\mathcal{A}_s\phi_i, \phi_j) + (\mathcal{F}_s(t, \sum_{i=1}^{d} a_i\phi_i, 0), \phi_j) \tag{3.23}$$

where $j = 1, \ldots, d$. This is the reduced-order model obtained by applying the POD and the Galerkin's methods to Eq. (3.1). The inner products in Eq. (3.23) are functionals of the known, time-independent POD modes $\phi(z)$, and are pre-computed before solving for the time-dependent coefficients a_j.

3.4.1.3 Numerical algorithm

With all the required tools now in hand, we are now ready to present the numerical algorithm for nonlinear model order reduction based on the computation of temporally-local eigenfunctions. The approach presented in this work consists of three stages. In the first stage, the snapshots are partitioned into clusters based on the local dynamics exhibited by the solution. In the second stage, the POD method is applied to each cluster to compute the temporally-local eigenfunctions that describe the dominant spatial patterns within each cluster. The final stage includes projecting the full-order solution onto the finite-dimensional solution space by the Galerkin's method using the orthogonal eigenfunctions obtained in the preceding step. The detailed algorithm is presented below in Algorithm 1.

3.4.1.4 Simulation results

As described in Chapter 2, the data required for implementing the above algorithm is obtained as a solution to the high-fidelity model. In this specific work, we have only considered the fracture propagation part described by Eqs. (2.1)-(2.6) which led to a total of 451 and 19611 nodes in the spatial and temporal coordinates, respectively. Figure 2.3 shows the evolution of the fracture width for $Q_0 = 0.05 \text{ m}^3/\text{s}$ starting from the initial condition. We now continue with the computation of the set of temporally-local eigenfunctions using the proposed methodology. A total of 981 snapshots, out of the 19611 generated from the full-order model, were taken at uniform time intervals and used for the computation of the low-dimensional model. Then, the GOS algorithm, described in Chapter 3.3.2, was employed to assign the snapshots into multiple clusters. The GOS algorithm was implemented using the modeling language GAMS (General Algebraic Modeling System) [124] and the algorithmic decomposition methodology APROS (Algorithms for PROcess Synthesis) [125]. GAMS was interfaced with the MILP solver, CPLEX, to determine the solutions to the primal and master problems. Please refer to Table 3.1 for a summary of the model statistics.

Remark 8. *The sensitivity of the proposed method to the number of snapshots is essentially the*

Algorithm 1 Nonlinear model order-reduction using temporally-local eigenfunctions

```
1: Obtain N snapshots, $\{x_1, ..., x_N\}$.
2: Apply the GOS algorithm to generate clusters and assign the snapshots to each cluster. For the cluster $k$, compute the cluster centers $\mathbf{c}_k$ and optimal number of clusters $m$ that minimizes the clustering balance $\epsilon$. The clustered snapshots $\mathcal{C}^k = \{x_1^k, \ldots, x_{n^k}^k\}$ for $k = 1, \ldots, m$ will satisfy:
       $$\bigcup_{k=1}^{m} \mathcal{C}^k = \{x_1, \ldots, x_N\} \qquad (3.24)$$
3: Compute the correlation matrix and solve the eigenvalue problem.
       $$B_{ij}^k = \frac{1}{n^k} \int_0^1 x_i^k(z') x_j^k(z') dz' \quad \forall i, j$$
       $$\mathcal{B}^k \mathbf{v}^k = \lambda^k \mathbf{v}^k \qquad (3.25)$$
   where $\sum_{k=1}^{m} n^k = N$.
4: Calculate the local eigenfunctions, $\phi^k$, by selecting a dimension $d^k$ for the cluster $k$.
       $$\phi_i^k = \sum_{j=1}^{n^k} \mathbf{v}_{i,j}^k x_j^k \quad i = 1, \ldots, d^k \qquad (3.26)$$
5: Apply Galerkin's method to derive $d^k$-dimensional ODE systems for the cluster k.
6: Solve the system of ODEs numerically to obtain the time-dependent coefficients $[a_1^k, \ldots, a_{d^k}^k]$
7: Compute the approximate solution:
       $$\bar{\mathbf{x}}^k(z, t) \approx \sum_{i=1}^{d^k} \mathbf{a}_i^k \phi_i^k \qquad (3.27)$$
```

same as the global POD method. One should ensure that the collected snapshots contain all the essential information regarding the spatial and dynamic behavior of the underlying system.

Figure 3.2 presents the intra-cluster and inter-cluster error sum profiles for different number of clusters. As can be seen from the graph, the intra-cluster error sum decreases with the total number of clusters which is expected from a divisive clustering algorithm. The intra-cluster error sum measures the extent of dissimilarity between snapshots within the same cluster, and hence should be minimized. On the other hand, the inter-cluster error sum measures the extent of dissimilarity between different clusters and hence is to be maximized. Using the two error sums, clustering balance was calculated for different cluster numbers. Figure 3.3 shows how the clustering balance

Clusters	iterations	equations	binary (y_{jk})	continuous (c_{ki})	CPU time (s)
2	1	1809	1962	824	5.35
4	3	2637	3924	1646	12.41
6	7	3461	5886	2468	46.05
8	6	4292	7848	3290	46.72
10	6	5116	9810	4112	57.24
12	10	5941	11772	4934	158.94
16	16	7604	15696	6578	504.31
20	20	9251	19620	8222	438.12

Table 3.1: Summary of model statistics for the clustering problem solved using the GOS algorithm. Reprinted with permission from [107]

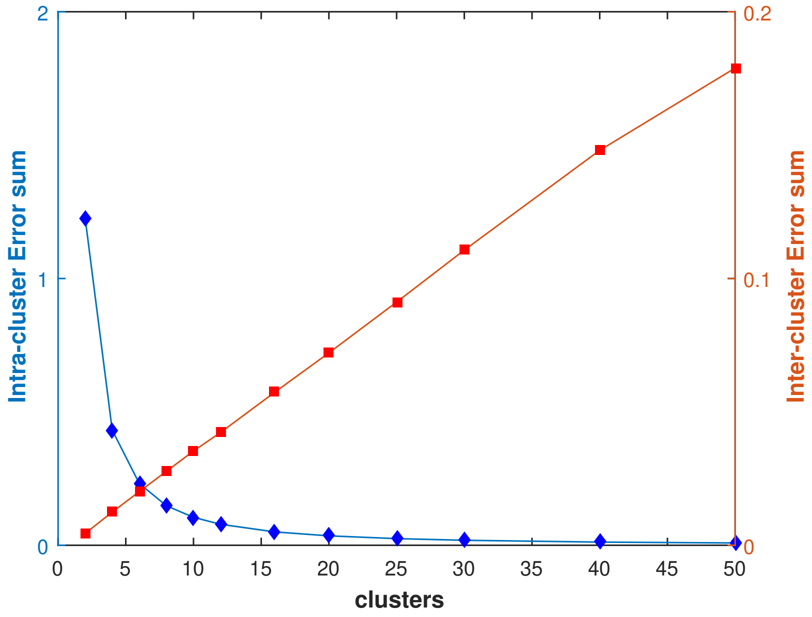


Figure 3.2: Plots of the Intra-cluster error sum (◆) and Inter-cluster error sum (■) from the clustering of 981 snapshots generated from the full-order solution data, each containing 411 spatial data points. Reprinted with permission from [107]

changes with the total cluster number. We can predict the optimal number of clusters by selecting the turning point in the curve. The GOS algorithm determined the optimal number of clusters as $m = 16$ and the corresponding cluster configuration, $\mathcal{C}^k$, has been extracted.

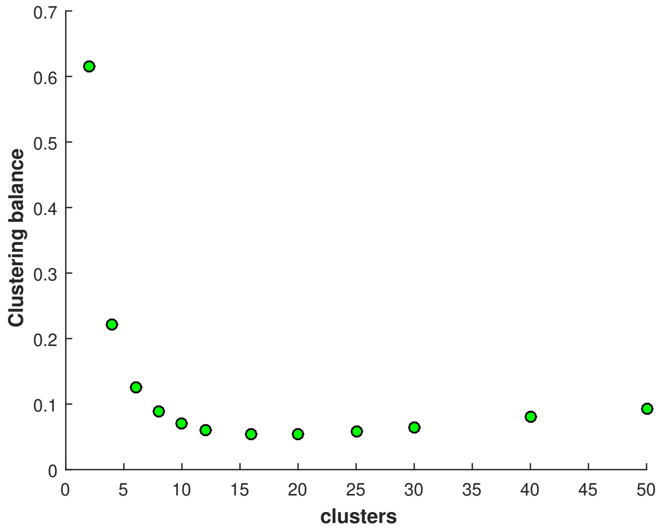


Figure 3.3: Clustering balance curve to predict the optimal number of clusters (m = 16), as shown by the turning point in the curve. Reprinted with permission from [107]

Once the optimal number of clusters have been determined, the POD method was applied to the snapshots in each cluster to compute eigenfunctions that span the dominant local spatial patterns embedded in each cluster. Then, the d-dimensional local POD basis is constructed by taking the first d largest eigenvalues obtained from solving the eigenvalue problem. Specifically, in our case of $m = 16$ clusters, the number of eigenfunctions (dimension for the low-order model) selected is $d^k = 2$, $\forall k = 1, \ldots, 16$. We present the two empirical eigenfunctions for each of the 16 clusters in Figures 3.4 and 3.5. Finally, the computed eigenfunctions are used in the Galerkin's framework to successfully derive the low-order ODE system, which is subsequently solved by discretizing the time coordinate to calculate the approximate solution to the full-order solution.

Remark 9. *We want to note the proposed approach does not provide a way to select the dimension r of the reduced-order model. One rule in selecting this dimension is the energy of the normalized eigenvalues. However, different clusters have different eigenfunctions (thereby different corresponding eigenvalues), and thus, they will result in different r values. Additionally, as pointed out earlier including those modes which correspond to low energies will introduce fluctuations in the*

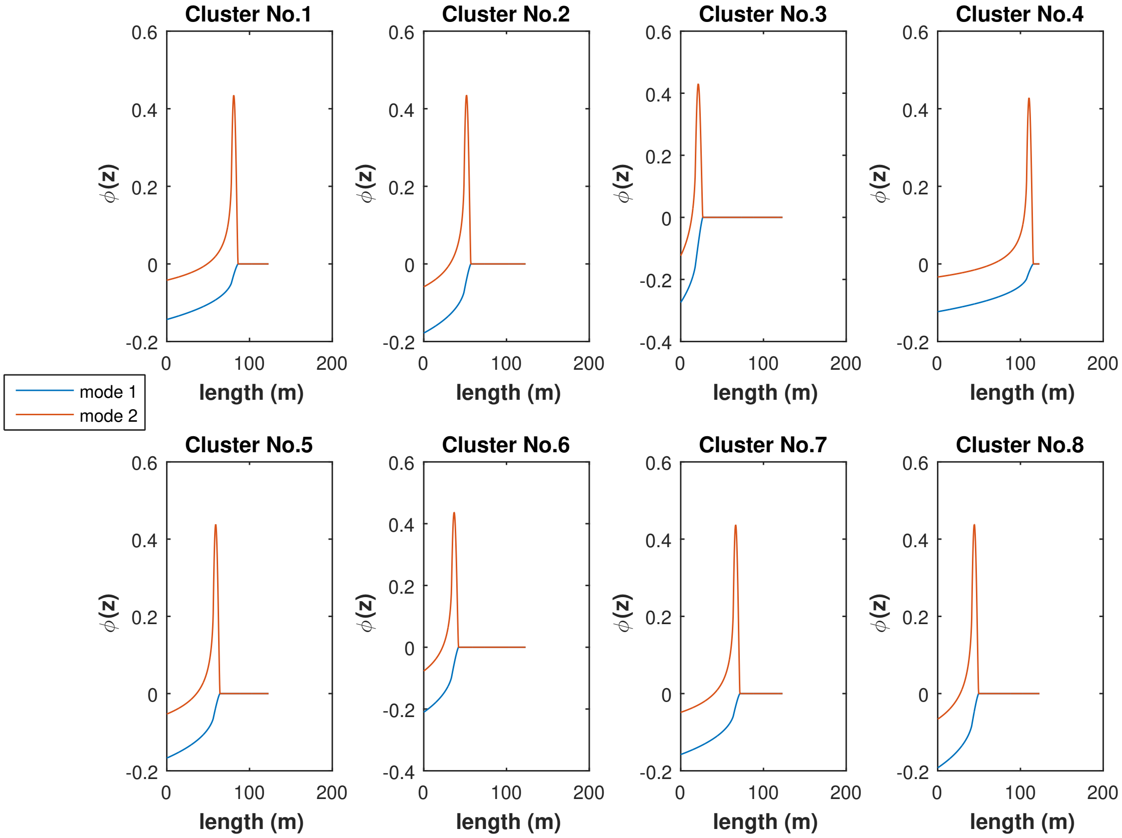


Figure 3.4: The two temporally-local eigenfunctions for clusters $1-8$. Reprinted with permission from [107]

system and will lead to numerical instability.

The standard inner product introduced earlier is defined for systems with time-invariant spatial domains. However, leveraging the fact that the fracture width is zero when the fracture does not propagate to a specified length, this problem can be viewed as a time-invariant case. This leads to a matrix for the spatiotemporal data that has nonzero values in the upper triangle and it has 0's at the lower triangle which corresponds to all the spatial locations where the fracture hasn't propagated yet at each time instant. Therefore, the definition of the standard inner product still holds. Additionally, the differential operator and thereby its eigenvalues can also be considered

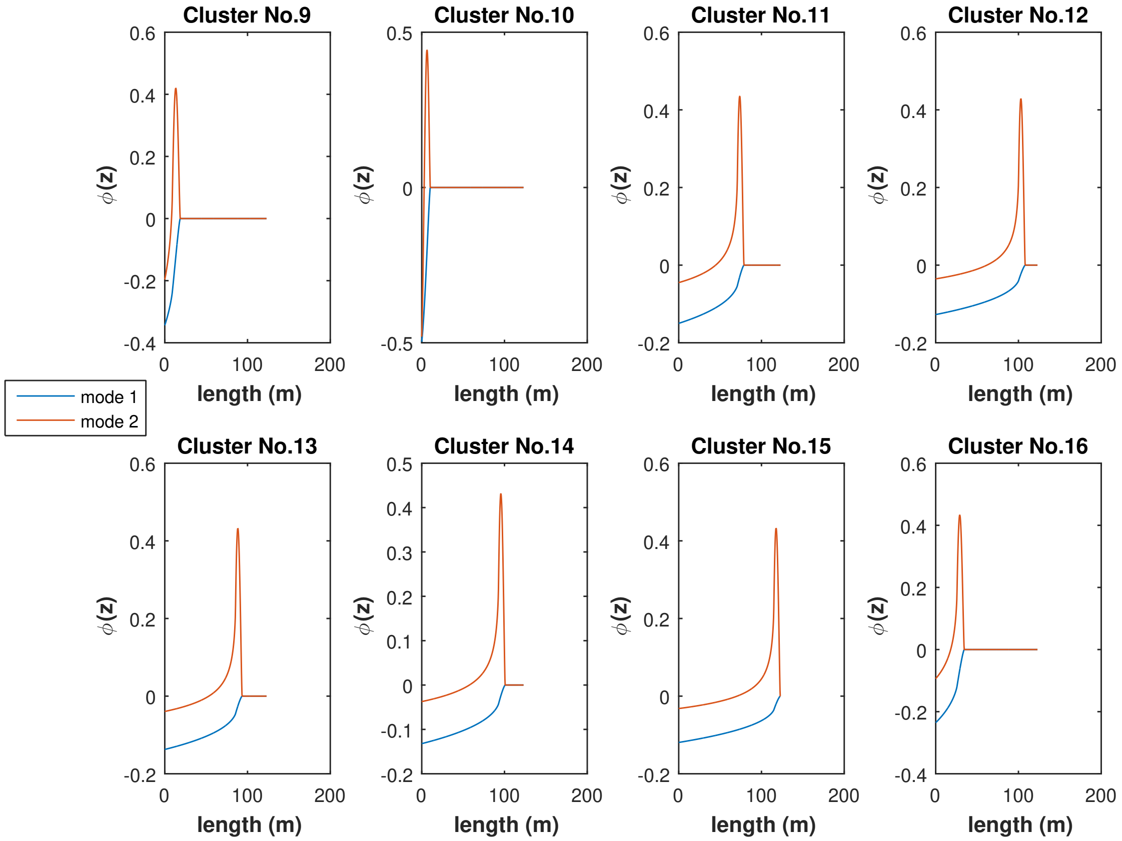


Figure 3.5: The two temporally-local eigenfunctions for clusters $8-16$. Reprinted with permission from [107]

time-invariant. This makes the computations for the KL decomposition and, subsequently, the Galerkin's projection valid for the system considered.

Figure 3.6 shows the complete width profile with respect to spatial and temporal coordinates obtained from the reduced order model (ROM) based on the temporally-local eigenfunctions. Furthermore, Figure 3.7 shows the comparison between the full-order solution and the reduced-order solution at $z = 0$ (i.e., wellbore) and $z = 62.7$ m (i.e., the fracture center) where we observe a very good agreement at all times between the two models. The oscillations observed in the proposed reduced-order model arise due to the presence of the leak-off term in the system. The leak-off term is not dissipative and adds an error to the approximation in the reduced-order model. This type of behavior is not observed when we applied the proposed method to a fluid system governed by one-dimensional Burgers' equation, the results of which are not presented in this manuscript.

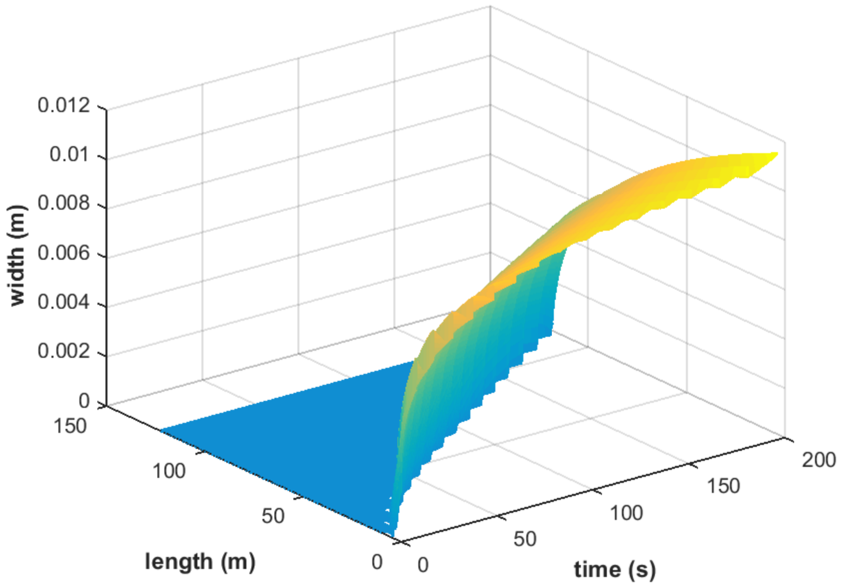
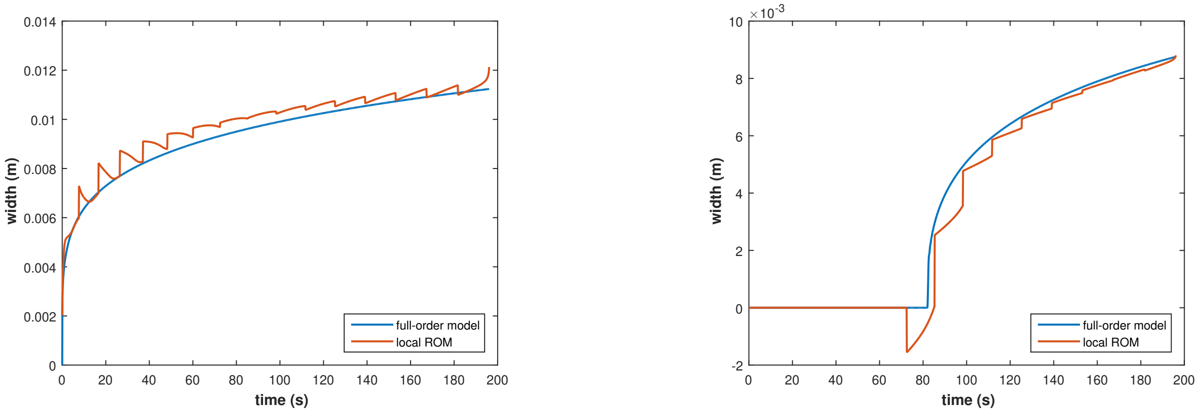


Figure 3.6: Approximate width profile computed from the ROM based on temporally-local eigenfunctions ($d^k = 2, \forall k = 1, \ldots, 16$). Reprinted with permission from [107].

For the comparison purpose, we also constructed a low-dimensional model using the ROM based on the temporally-global eigenfunctions. The solutions obtained from the temporally-global eigenfunctions are presented in Figures 3.8 and 3.9. We note that the number of eigenfunctions required to successfully describe the dominant spatial patterns is $d = 16$ when a set of temporally-global eigenfunctions is used for computing the low-order model. By comparing Figures 3.6 and 3.7 with Figures 3.8 and 3.9, respectively, we observe that the temporally-local eigenfunctions, when used to approximate the nonlinear PDE system provide a better result in terms of accuracy. Even though the distinction is difficult to visualize in Figures 3.6 and 3.8, it becomes more apparent



(a) W(t) at the wellbore, $z = 0$

(b) W(t) at the center $z = 62.7$ m

Figure 3.7: Comparison of width profiles obtained at two different spatial locations, $z = 0$ and $z = 62.7$, from the full-order model and the ROM based on temporally-local eigenfunctions at two different spatial locations. Reprinted with permission from [107].

in Figures 3.7 and 3.9. At both spatial locations, $z = 0$ and $z = 62.7$ m, we can clearly notice that the reduced order solution computed via the proposed technique is closer to the original full-order solution.

To further illustrate the significance of the ROM based on the temporally-local eigenfunctions, we compared the two methods with respect to their relative errors. The relative error is calculated in the following manner [103]:

$$E(t) = \frac{||\mathbf{x}_{full} - \mathbf{x}_{rom}||_{fro}}{||\mathbf{x}_{rom}||_{fro}} \tag{3.28}$$

where $||\mathbf{x}_{full}||_{fro}$ and $||\mathbf{x}_{rom}||_{fro}$ are the Frobenius norms of the high-order discretization of the PDE and the reduced-order model, respectively. Figure 3.10 shows the relative error profiles $E(t)$ between the low-dimensional model obtained from the temporally-global eigenfunctions and temporally-local eigenfunctions. From the plot we observe that initially the deviation is high, but the error decreases with time. This is attributed to the fact that the parabolic PDE system is characterized by fast initial dynamics followed by the slow (dominant patterns) dynamics and the computed eigenfunctions only capture the dominant spatial patterns (we neglect the fast dynam-

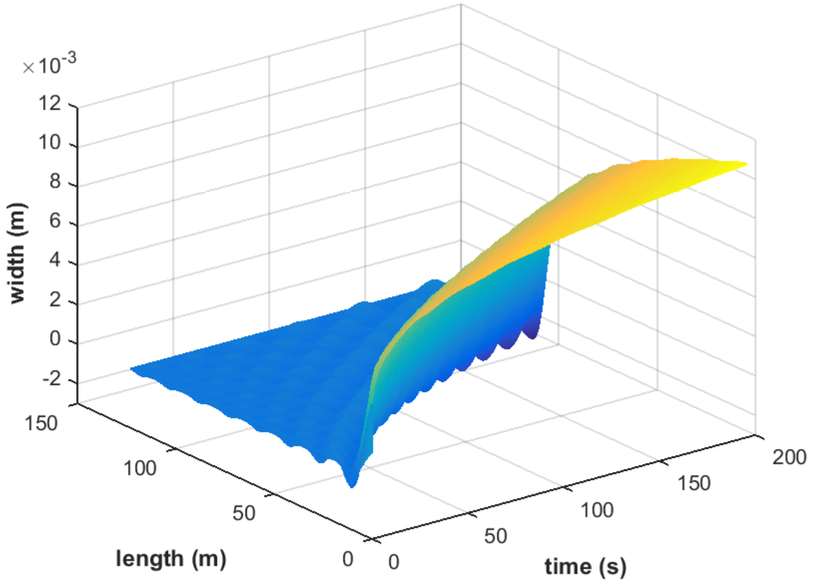
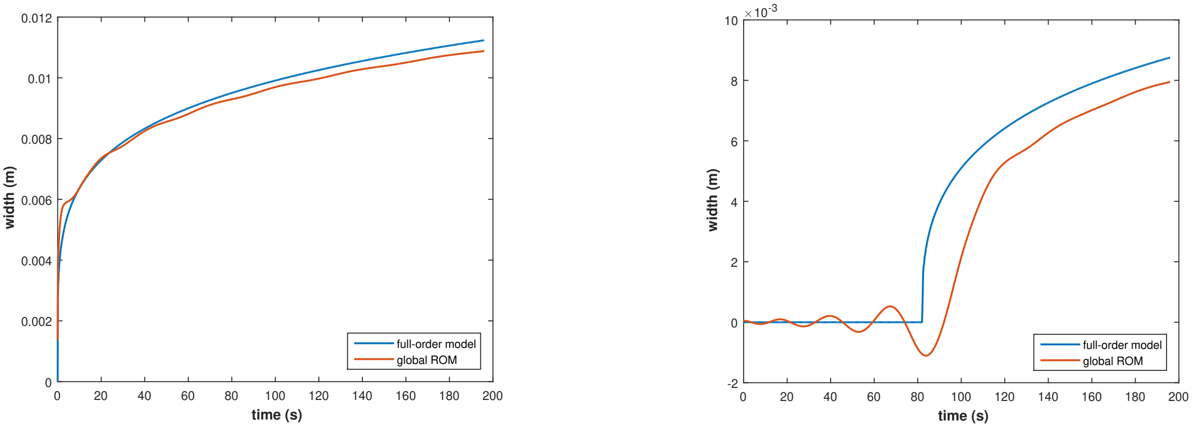


Figure 3.8: Approximate width profile computed from the ROM based on temporally-global eigenfunctions (d = 16). Reprinted with permission from [107].



(a) W(t) at boundary $z = 0$

(b) W(t) at the center $z = 62.7$ m

Figure 3.9: Comparison of width profiles obtained at two different spatial locations, $z = 0$ and $z = 62.7$, from the full-order model and the ROM based on temporally-global eigenfunctions. Reprinted with permission from [107].

ics in the Galerkin's method) after the fast dynamics of the system becomes less significant. The presented error profiles clearly show that the ROM constructed using 2 temporally-local eigenfunctions provides a better approximation to the original solution profile than the ROM derived using 16 temporally-global eigenfunctions. This is a significant reduction in the number of eigenfunctions required to accurately represent the full-order solution.

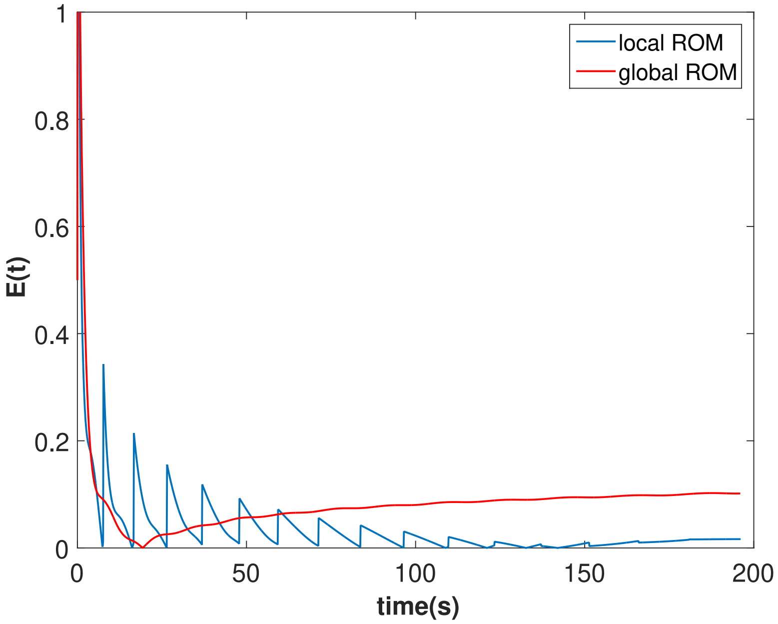


Figure 3.10: Profiles of the relative error with time for approximate solutions constructed from the ROM based on temporally-local and temporally-global eigenfunctions. Reprinted with permission from [107].

We also compared the computational time taken by the two models to generate the approximated solution. In this regard, we expanded the ensemble size and computed the approximate solutions using the two models. Figure 3.11 shows the computational time with respect to different number of snapshots considered as part of the solution ensemble. The figure depicts the significant advantage of the temporally-local ROM over the global model. In general, the POD method

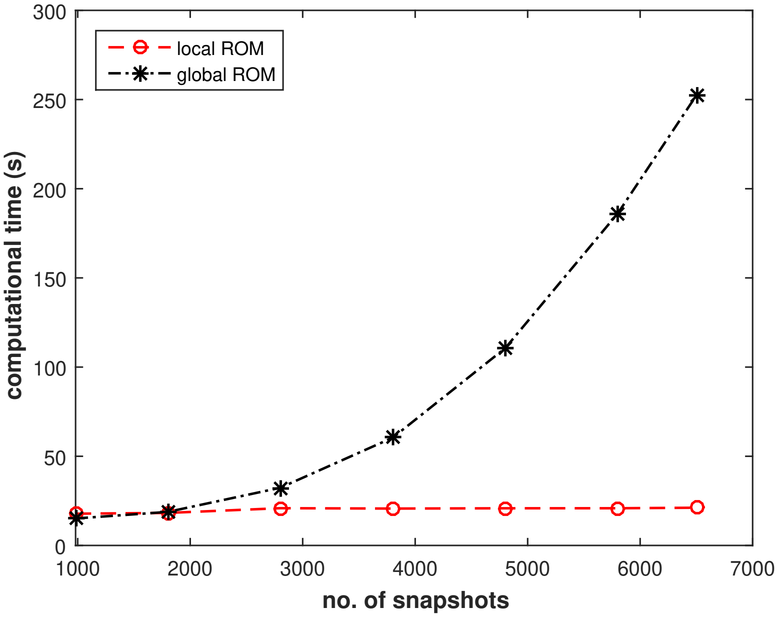


Figure 3.11: Computational time profiles of the local and global reduced order models with respect to different sizes of the snapshot set. Reprinted with permission from [107].

involves the computation of an $N \times N$ eigenvalue problem, where N is the number of snapshots. The proposed method instead requires the computation of several small eigenvalue problems because each cluster contains only a fraction of the total number of snapshots. Although both of the methods start off with similar computational times when less number of snapshots are used, the difference becomes significant as the number increases ($\approx 92\%$ reduction in computational time when $N = 6501$ snapshots). The times taken by the ROM based on the temporally-local eigenfunctions remains almost same because even though the total number of snapshots increases the snapshots per cluster does not change significantly for the considered data. Please note all the calculations were performed using MATLAB on a Dell workstation, powered by Intel(R) Core(TM) i7-4790 CPU@3.60GHz, running the Windows 8 operating system.

We note here that the computational time presented does not include the time required for data clustering. The comparison between the two methods has been performed only with respect to the

POD and Galerkin steps of the algorithm. Even though we implement the proposed model order reduction algorithm online for the purpose of, for example controller design, the data clustering step will be performed offline. Thus, it will not affect the computational requirements for the implementation of the proposed model-order reduction technique online.

As we notice from the error results presented, the MOR technique with temporally-local eigenfunctions performs favorably in terms of accuracy, computational efficiency, and the number of POD modes required to capture the dominant spatial patterns embedded in the set of snapshots. This can be justified by the fact that dominant spatial patterns obtained over the entire temporal and spatial domains often include undesired information from the standpoint of local domains. Therefore, it is desirable to first decompose the temporal and/or spatial domain of the original high-order model into multiple subdomains, and then subsequently develop low-order models that better capture the dominant local dynamics of the system.

3.4.2 Local Dynamic Mode Decomposition

The preliminary results presented in Chapter 3.4.1 are very encouraging in terms of choosing the proposed local MOR framework over its global counterpart when dealing with complex systems. However, one of the main drawbacks of POD is that the energetically optimal modes do not necessarily represent the system *dynamics* accurately. In fact, when the order of the snapshots is changed the computed POD modes do not change meaning that the modes are not dynamically optimal. Moreover, the subsequent Galerkin projection requires a priori knowledge of the system equations which are not always available readily. In Chapter 3.4.2, we aim to address these issues by moving to a completely data-driven architecture that captures the dynamics accurately.

Developed specifically for analyzing the dynamics of nonlinear systems, DMDc provides a linear time invariant (LTI) type reduced-order models by explicitly accounting for the system inputs. Since the overall objective of this work is to develop approximate models for control synthesis, DMDc is an intuitive choice to be incorporated into the proposed framework. It is very attractive because it does not require any knowledge on the system equations and provides linear dynamical models with the potential to enable prediction, estimation and control of nonlinear systems using linear systems theory. Motivated by this, we apply the concept of local DMDc to approximate the nonlinear hydraulic fracturing system with an emphasis on process control. Basically, POD is replaced by DMDc in the local MOR framework shown in Figure 3.1. This also reveals the flexibility of the proposed framework where the model reduction technique can be replaced based on its suitability with respect to the application of interest.

3.4.2.1 Dynamic Mode Decomposition with control

DMD is a method that can extract dynamically relevant spatial structures solely from the data of a high-dimensional complex system. These structures, called dynamic modes, are equivalent to a linear tangent approximation and describe the dominant dynamic behavior of a nonlinear data sequence. In Chapter 3.4.2.1, we present a short mathematical description of DMD and the formulations required for DMDc.

We start with a general description of state vector fields $\mathbf{x}$ collected at regular time intervals Δt, either by direct numerical simulations or from experimental data, $\mathbf{x}_i = \mathbf{x}(t_i) \in \mathbb{R}^n$ where $t_i = i\Delta t$ and n represents the number of available spatial measurements. Through the course of this paper, we will refer to these measurements as snapshots. The fundamental assumption in DMD is that we seek a linear operator $\mathbf{A}$ that approximately connects the snapshot $\mathbf{x}_i$ to the subsequent snapshot $\mathbf{x}_{i+1}$, which is given by

$$\mathbf{x}_{i+1} = \mathbf{A}\mathbf{x}_i, \tag{3.29}$$

and that this mapping is approximately the same over the full sampling interval $t \in [0, (m-1)\Delta t]$. We collect an ensemble of trajectories containing m snapshots and arrange them into two matrices,

$$\mathbf{X}_1 = [\mathbf{x}_1 \, \mathbf{x}_2 \, \ldots \, \mathbf{x}_{m-1}], \quad \mathbf{X}_2 = [\mathbf{x}_2 \, \mathbf{x}_3 \, \ldots \, \mathbf{x}_m] \tag{3.30}$$

Based on the above description the relation between the two snapshot matrices is given by

$$\mathbf{X}_2 = \mathbf{A}\mathbf{X}_1 \tag{3.31}$$

The primary objective now is to solve for an approximation of the system matrix $\mathbf{A}$. In [19], the authors presented a mathematical expression for computing the matrix $\mathbf{A}$ using a companion matrix based on a variant of the Arnoldi algorithm. A more well-conditioned algorithm, which we will follow here, based on a similarity transformation achieved using SVD is presented by [54]. The computation of $\mathbf{A}$ follows from applying the pseudo-inverse of $\mathbf{X}_1$ to both sides of Eq. (3.31) as shown below:

$$\mathbf{A} = \mathbf{X}_2\mathbf{X}_1^{\dagger} \tag{3.32}$$

where $\mathbf{X}_1^{\dagger}$ is the Moore-Penrose pseudo-inverse of $\mathbf{X}_1$. Computationally, the most efficient way to perform the pseudo-inverse is via SVD.

$$\mathbf{X}_1 = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^* \approx \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}^* \tag{3.33}$$

where $\mathbf{U} \in \mathbb{R}^{n \times n}$, $\mathbf{\Sigma} \in \mathbb{R}^{n \times m-1}$, $\mathbf{V}^* \in \mathbb{R}^{m-1 \times m-1}$, $\hat{\mathbf{U}} \in \mathbb{R}^{n \times r}$, $\hat{\mathbf{\Sigma}} \in \mathbb{R}^{r \times r}$, $\hat{\mathbf{V}}^* \in \mathbb{R}^{r \times m-1}$ and * denotes the complex conjugate transpose.

We recognize that in the above decomposition the left singular vectors, $\mathbf{U}$, contains proper orthogonal modes of the snapshot set and hence amounts to a projection of the linear operator $\mathbf{A}$ onto a set of POD bases. Furthermore, Eq. (3.33) results in a low-dimensional representation of $\mathbf{A}$ by accounting for the rank-deficiency in the snapshot sequence via a limited number of projection modes given by r. The appropriate value of r can be given by non-zero singular values (or more practically singular values above a prescribed threshold) [66, 126, 127, 128]. The approximation for the linear operator $\mathbf{A}$ can thus be computed from SVD of the snapshot set by plugging Eq. (3.33) into Eq. (3.32) as follows:

$$\mathbf{A} \approx \mathbf{X}_2\mathbf{X}_1^{\dagger} = \mathbf{X}_2\hat{\mathbf{V}}\hat{\mathbf{\Sigma}}^{-1}\hat{\mathbf{U}}^* \tag{3.34}$$

The reduced-order model can therefore be derived by taking a basis transformation, $\hat{\mathbf{U}}^*\mathbf{X} = \hat{\mathbf{X}}$,

$$\begin{aligned}\hat{\mathbf{X}}_2 =& \hat{\mathbf{U}}^*\mathbf{A}\hat{\mathbf{U}}\hat{\mathbf{X}}_1 \\ =& \hat{\mathbf{U}}^*\mathbf{X}_2\hat{\mathbf{V}}\hat{\mathbf{\Sigma}}^{-1}\hat{\mathbf{X}}_1 \\ =& \hat{\mathbf{A}}\hat{\mathbf{X}}_1\end{aligned} \tag{3.35}$$

where $\hat{\mathbf{X}} \in \mathbb{R}^r$ represents the states in the reduced-order model obtained by projecting the original states onto a new subspace with a reduced dimension r, and $\hat{\mathbf{A}}$ denotes an approximation to the linear operator $\mathbf{A}$.

The eigendecomposition of $\hat{\mathbf{A}}$ defined by $\hat{\mathbf{A}}\mathbf{W} = \mathbf{\Lambda}\mathbf{W}$ yields the eigenvalues and eigenvectors. According to the property of similarity transformation, the eigenvalues of $\hat{\mathbf{A}}$ are also the eigenvalues of the full matrix $\mathbf{A}$. The eigenvectors of the full matrix $\mathbf{A}$ are called dynamic modes, denoted by $\mathbf{\Phi}$, and are related to the eigenvectors of $\hat{\mathbf{A}}$ via the following transformation,

$$\mathbf{\Phi} = \mathbf{P}\mathbf{W} = \mathbf{X}_2\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{W} \tag{3.36}$$

where $\mathbf{P} = \mathbf{X}_2\mathbf{V}\mathbf{\Sigma}^{-1}$ is the required linear transformation.

Building on the DMD theory, [73] developed the DMDc method in order to ascertain the dynamic characteristics of a system that depend both on inherent dynamics (e.g., states) as well as applied external inputs. In addition to the sequence of state snapshots in DMD, we now collect the sequence of input snapshots as

$$\mathbf{\Gamma} = [\mathbf{u}_1 \; \mathbf{u}_2 \; \ldots \; \mathbf{u}_{m-1}] \tag{3.37}$$

where $\mathbf{u}_i \in \mathbb{R}^l$ and l denotes the number of input variables. The system description can therefore be rewritten in an augmented form as

$$\mathbf{X}_2 = \mathbf{A}\mathbf{X}_1 + \mathbf{B}\mathbf{\Gamma} = [\mathbf{A} \; \mathbf{B}] \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{\Gamma} \end{bmatrix} = \mathbf{G}\mathbf{\Omega} \tag{3.38}$$

where $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{B} \in \mathbb{R}^{n \times l}$, $\mathbf{\Gamma} \in \mathbb{R}^{l \times (m-1)}$, $\mathbf{\Omega} \in \mathbb{R}^{(n+l) \times (m-1)}$ and $\mathbf{G} \in \mathbb{R}^{n \times (n+l)}$.

In contrast to DMD, SVD is performed on the augmented data matrix $\mathbf{\Omega} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}^*$ and the approximations for $\mathbf{A}$ and $\mathbf{B}$ are derived as follows:

$$\begin{aligned} \mathbf{A} &\approx \mathbf{X}_2 \hat{\mathbf{V}} \hat{\mathbf{\Sigma}}^{-1} \hat{\mathbf{U}}_1^* \\ \mathbf{B} &\approx \mathbf{X}_2 \hat{\mathbf{V}} \hat{\mathbf{\Sigma}}^{-1} \hat{\mathbf{U}}_2^* \end{aligned} \tag{3.39}$$

where $\hat{\mathbf{U}}_1^* \in \mathbb{R}^{n \times p}$, $\hat{\mathbf{U}}_2^* \in \mathbb{R}^{l \times p}$, $[\hat{\mathbf{U}}_1 \ \hat{\mathbf{U}}_2]^T = \hat{\mathbf{U}}$ and p denotes the reduced-order of the augmented system.

In the next step, we project the state onto a subspace on which it evolves using a basis transformation. However, in DMDc, the truncated left singular vectors $\hat{\mathbf{U}}$ of the augmented matrix, $\mathbf{\Omega}$, can not be used to define this transformation because they include the inputs as well as states; it is a key difference from DMD. To find a linear transformation for the states alone, we utilize a reduced-order subspace of the shifted snapshot sequence, $\mathbf{X}_2$. This fundamental observation allows for DMDc to discover a reduced-order representation of the dynamics $\mathbf{A}$ and the input matrix $\mathbf{B}$. Computationally, SVD of the shifted snapshot sequence ($\mathbf{X}_2 = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^*$ where $\tilde{\mathbf{U}} \in \mathbb{R}^{n \times r}$, $\tilde{\mathbf{\Sigma}} \in \mathbb{R}^{r \times r}$, $\tilde{\mathbf{V}} \in \mathbb{R}^{m-1 \times r}$ and r denotes the dimension of the subspace) is used to define the linear transformation required for projecting the state. Using the new transformation, a low-dimensional representation of the system matrices can be computed as follows:

$$\begin{aligned} \hat{\mathbf{A}} &\triangleq \tilde{\mathbf{U}}^* \mathbf{A} \tilde{\mathbf{U}} = \tilde{\mathbf{U}}^* \mathbf{X}_2 \hat{\mathbf{V}} \hat{\mathbf{\Sigma}}^{-1} \hat{\mathbf{U}}_1^* \tilde{\mathbf{U}} \\ \hat{\mathbf{B}} &\triangleq \tilde{\mathbf{U}}^* \mathbf{B} = \tilde{\mathbf{U}}^* \mathbf{X}_2 \hat{\mathbf{V}} \hat{\mathbf{\Sigma}}^{-1} \hat{\mathbf{U}}_2^* \end{aligned} \tag{3.40}$$

where $\hat{\mathbf{A}} \in \mathbb{R}^{r \times r}$ and $\hat{\mathbf{B}} \in \mathbb{R}^{r \times l}$. In general, the dimension p of the reduced SVD for augmented data is greater than that of the dimension r of the subspace onto which the states are projected. Please note that one could equivalently use SVD of $\mathbf{X}_1$ to obtain the required basis transformation. Depending on the choice of $\mathbf{X}_1$ or $\mathbf{X}_2$ the corresponding linear transformation will be unique.

3.4.2.2 Numerical implementation

Recall, the basic assumption that drives DMDc is that the snapshots are approximately related by a linear operator. Therefore, DMDc may fail to capture the local dynamics if there is significant non linearity in the local dynamics. To handle this issue, in this work, we use a two-step process where, in the first stage, the snapshots are partitioned into clusters by applying the GOS algorithm based on the local dynamics exhibited by the solution, and in the second stage, we apply DMDc to each cluster and compute the low-dimensional representations of $\mathbf{A}$ and $\mathbf{B}$ matrices. To achieve this, we first compute SVD of the augmented matrix, $\mathbf{\Omega}^k = \hat{\mathbf{U}}^k \hat{\mathbf{\Sigma}}^k \hat{\mathbf{V}}^{*^k}$, and the shifted snapshot sequence, $\mathbf{X}_2^k = \tilde{\mathbf{U}}^k \tilde{\mathbf{\Sigma}}^k \tilde{\mathbf{V}}^{*^k}$, where k denotes the current cluster number. Next, using the transformation basis $\tilde{\mathbf{U}}^k$, i.e., the left singular vectors of the shifted snapshot sequence, we apply Eq. (3.17) to identify the input-to-output relationship within each cluster. The algorithm detailing the application of the proposed local DMDc method is presented below.

Remark 10. *Recall, the fundamental form of DMDc is as follows:*

$$\mathbf{x}_{i+1} = \mathbf{A}\mathbf{x}_i + \mathbf{B}\mathbf{u}_i \tag{3.41}$$

The above equation is analogous to a general n^{th} order discrete-time linear state-space model. Hence, we recognize that the DMDc method can be used for system identification of a linear time-invariant state-space model that captures the underlying dynamics in the data obtained from high-fidelity models.

3.4.2.3 Simulation results

Similar to the previous case, we first collect the high-fidelity data by solving the dynamic model of hydraulic fracturing. However, this case study considers simultaneous fracture propagation and proppant transport to provide a detailed simulation of the process. As described in Chapter 2.5.1, proppant concentration measurements inside the fracture are very difficult to obtain and are usually determined with empirical correlations using microseismic data. So, out of all the spatial locations,

Algorithm 2 Nonlinear model order-reduction using temporally-local dynamic modes

```
1: Collect the state and input snapshots and construct the data matrices, $\mathbf{X}_1 = [\mathbf{x}_1\ \mathbf{x}_2\ \dots\ \mathbf{x}_{m-1}], \mathbf{X}_2 = [\mathbf{x}_2\ \mathbf{x}_3\ \dots\ \mathbf{x}_m]$ and $\mathbf{\Gamma} = [\mathbf{u}_1\ \mathbf{u}_2\ \dots\ \mathbf{u}_{m-1}]$.
2: Apply the GOS algorithm to generate clusters and assign the snapshots to each cluster. For the $k^{th}$ cluster, compute the cluster centers $\mathbf{c}_k$ and the optimal number of clusters $m$ that minimizes the clustering balance $\epsilon$. The clustered snapshots $\mathcal{C}^k = \{\mathbf{x}_1^k, \dots, \mathbf{x}_{n^k}^k\}$ for $k = 1, \dots, m$, where $n^k$ denotes the number of snapshots in the $k^{th}$ cluster, will satisfy:
       $\bigcup_{k=1}^{m} \mathcal{C}^k = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$
3: For every cluster, stack the state and input matrices to construct the augmented matrix $\mathbf{\Omega}^k$
       $\mathbf{\Omega}^k = \begin{bmatrix} \mathbf{X}_1^k \\ \mathbf{\Gamma}^k \end{bmatrix}$
4: Compute the reduced SVD of the augmented matrix and the shifted snapshot sequence. Choose values for the reduced-orders $p^k$ and $r^k$ based on the singular values and the corresponding singular vectors to obtain
       $\mathbf{\Omega}^k = \hat{\mathbf{U}}^k \hat{\mathbf{\Sigma}}^k \hat{\mathbf{V}}^{*^k}$
       $\mathbf{X}_2^k = \tilde{\mathbf{U}}^k \tilde{\mathbf{\Sigma}}^k \tilde{\mathbf{V}}^{*^k}$
5: Compute the low-dimensional representations of the operators $\mathbf{A}^k$ and $\mathbf{B}^k$ as described in Eq. (3.40)
       $\hat{\mathbf{A}}^k = \tilde{\mathbf{U}}^{*^k} \mathbf{X}_2^k \hat{\mathbf{V}}^k \hat{\mathbf{\Sigma}}^{k^{-1}} \hat{\mathbf{U}}_1^{*^k} \tilde{\mathbf{U}}^k$
       $\hat{\mathbf{B}}^k = \tilde{\mathbf{U}}^{*^k} \mathbf{X}_2^k \hat{\mathbf{V}}^k \hat{\mathbf{\Sigma}}^{k^{-1}} \hat{\mathbf{U}}_2^{*^k}$
```

6 were selected with uniform spacing inside the fracture and it was assumed the real-time concentration measurements are available at these locations. The test input profile has been designed to mimic the practically-viable inlet proppant concentration profile in the field. Specifically, a simple uniformly increasing staircase input profile has been considered to solve the dynamics of the process, which is presented in Figure 3.12. The corresponding response of the open-loop system is then obtained by solving the high-fidelity model and stored along with the manipulated input trajectory. The stored vectors then comprise the columnar entries in the data matrices.

The GOS algorithm, then, has been applied to the generated data to cluster the snapshot vectors based on the similarity between their local dynamic behaviors. The GOS algorithm has been implemented using the modeling language GAMS, which was interfaced with CPLEX (an MILP

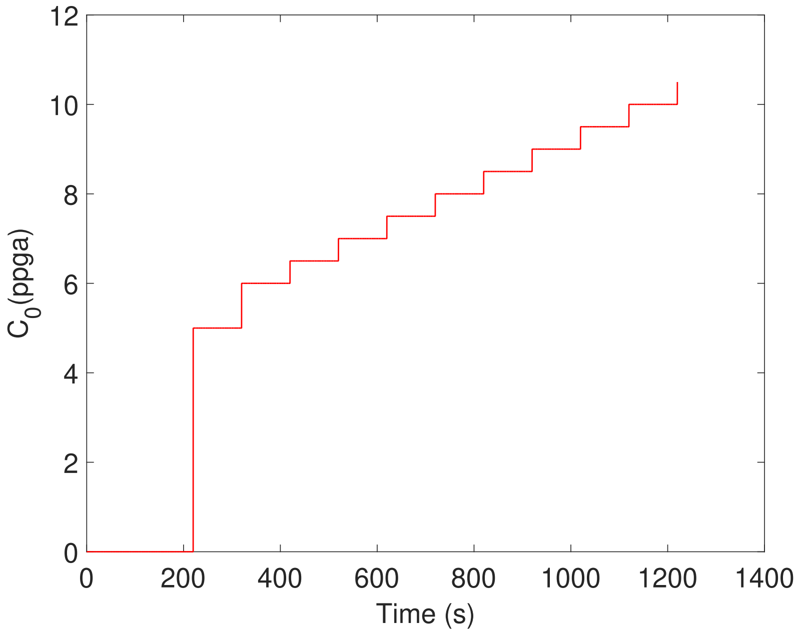


Figure 3.12: The input profile applied to generate the open-loop simulation data of the process. Reprinted with permission from [129].

solver) to determine solutions to the primal and master problems. Figure 3.13 shows how the clustering balance changes with the total number of clusters considered. We can predict the optimal number of clusters by selecting the turning point in the curve. The GOS algorithm determined the optimal number of clusters as $m = 100$ and the corresponding cluster configuration, $\mathcal{C}^k$, has been extracted. Once clustering has been completed, using the data matrices, the DMDc computation has been performed within each cluster to find an approximation of the system matrices. These reduced system matrices are subsequently used to compute approximate solutions to the full-order model.

To illustrate the superior performance of the proposed local model reduction methodology, a comparison with its global counterpart (i.e., applying DMDc over the entire time domain) has been performed. Figure 3.14 illustrates one such comparison arising from the numerical realizations of the fracture width at the wellbore, fracture length and the proppant concentration at two different

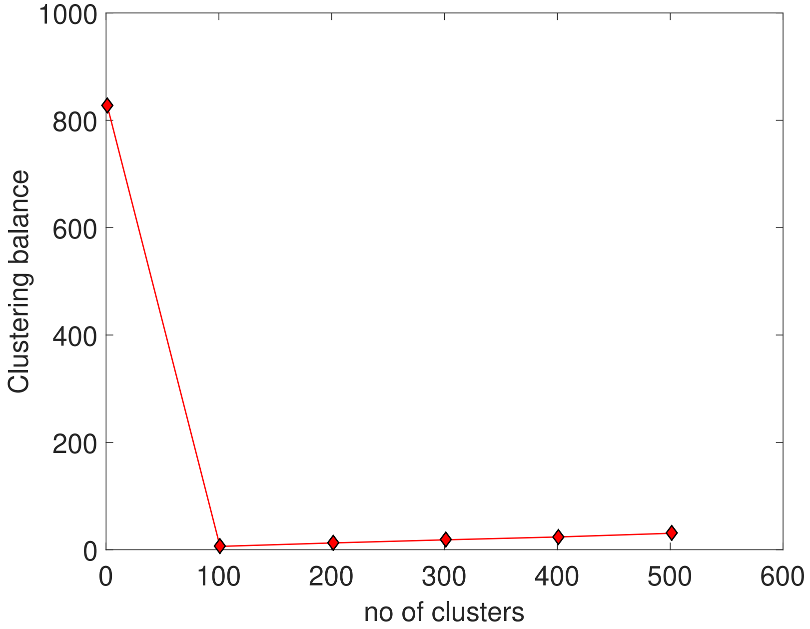
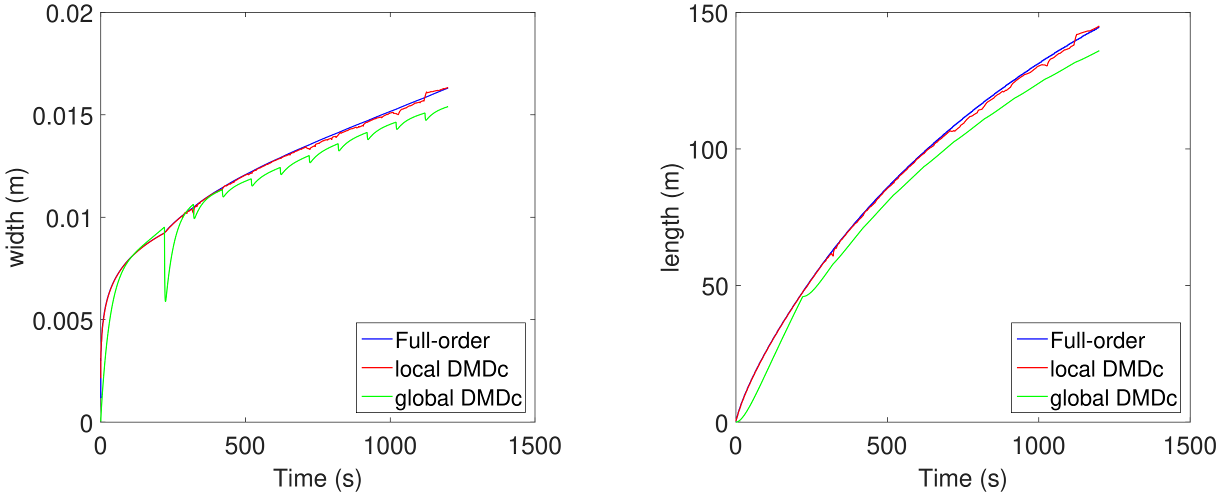


Figure 3.13: Clustering balance curve predicting the optimal number of clusters, which is $m = 100$. Reprinted with permission from [129].

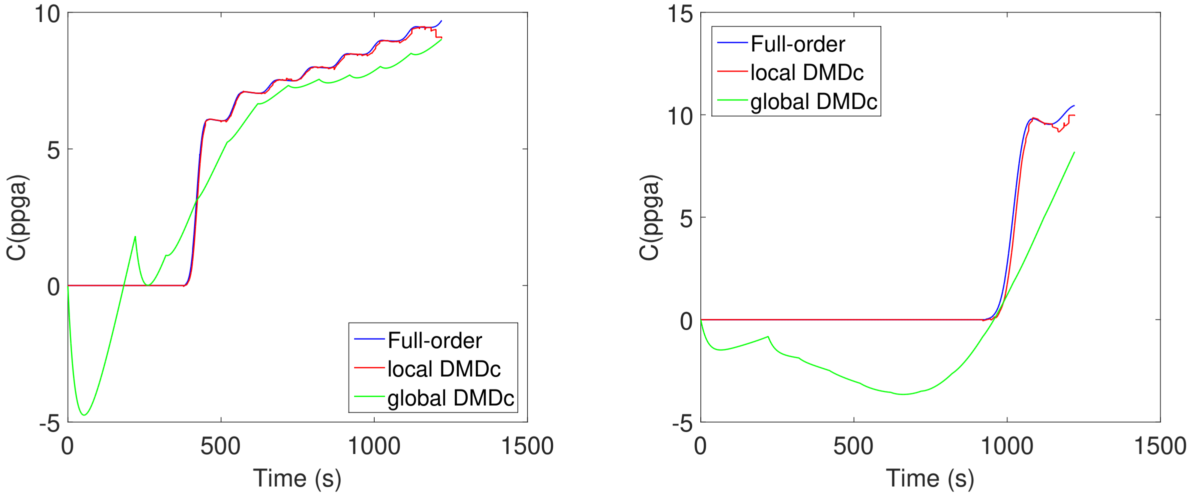
spatial locations. It can be observed from the plot that the approximate solution computed using the proposed local model reduction method mimics the full-order solution more accurately than the one obtained from its global counterpart.

To quantify the significant performance difference between the two reduced-order models, we compared the two methods with respect to their relative errors. The relative error has been calculated using the Frobenius norms of the state vectors using Eq. (3.28). The relative error profiles for approximate solutions constructed from the local and global DMDc are presented in Figure 3.15. From the plot we observe that the proposed local reduced-order model provides a precise approximation to the original solution, thus exhibiting its superior performance and warranting its application in closed-loop operation of the hydraulic fracturing process.

Convergence behavior. Before attempting to design feedback control systems, we study various factors that influence the accuracy of the reduced-order models developed based on global and



(a) W(t) at the wellbore (b) L(t) of the fracture



(c) C(t) at z = 36 m (d) C(t) at z = 108 m

Figure 3.14: Comparison of the approximate solutions computed using local and global DMDc methods. Reprinted with permission from [129].

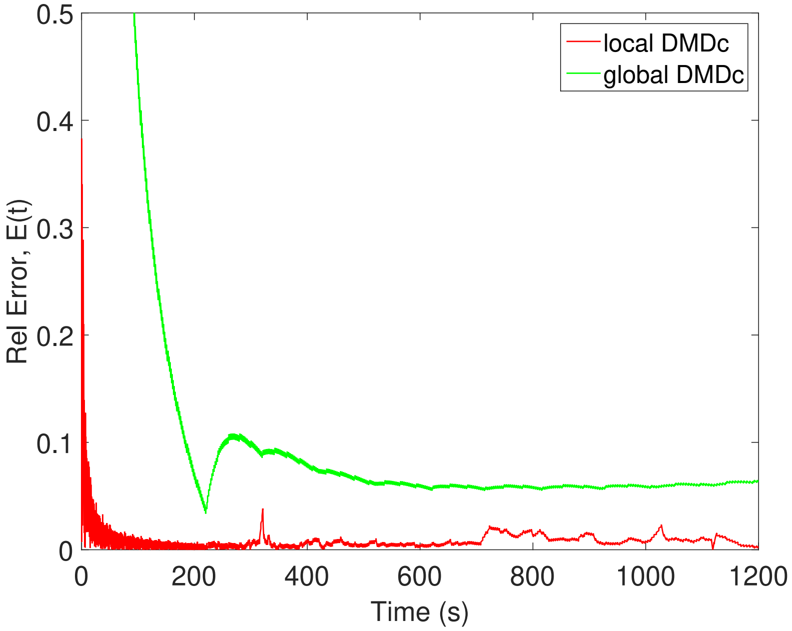


Figure 3.15: Profiles of the relative error with time for approximate solutions constructed from the reduced-order models based on the proposed local and global DMDc methods. Reprinted with permission from [129].

local DMDc methods. Such a study is essential in gaining knowledge that will help us identify reduced-order models that accurately capture the dynamically relevant spatial behavior contained in the process data sequence. For the case of global DMDc, as the number of spatial points (i.e., the number of concentration measurements at different locations) increases, we expect convergence of the reduced-order model towards the high-fidelity model. This is observed in Figure 3.16 which shows that the relative error decreases with an increase in the number of spatial measurements.

Next, we studied the influence of clusters on the convergence behavior of the reduced-order models developed by the proposed local DMDc. Intuitively, we expect an increase in the accuracy of the approximate solutions computed using local reduced-order models with an increase in the number of clusters. In other words, as the number of partitions increases, the local dynamic behavior of the underlying system can be better captured by affine subspaces constructed by the selected dynamic modes and hence will result in more accurate approximations. A demonstration

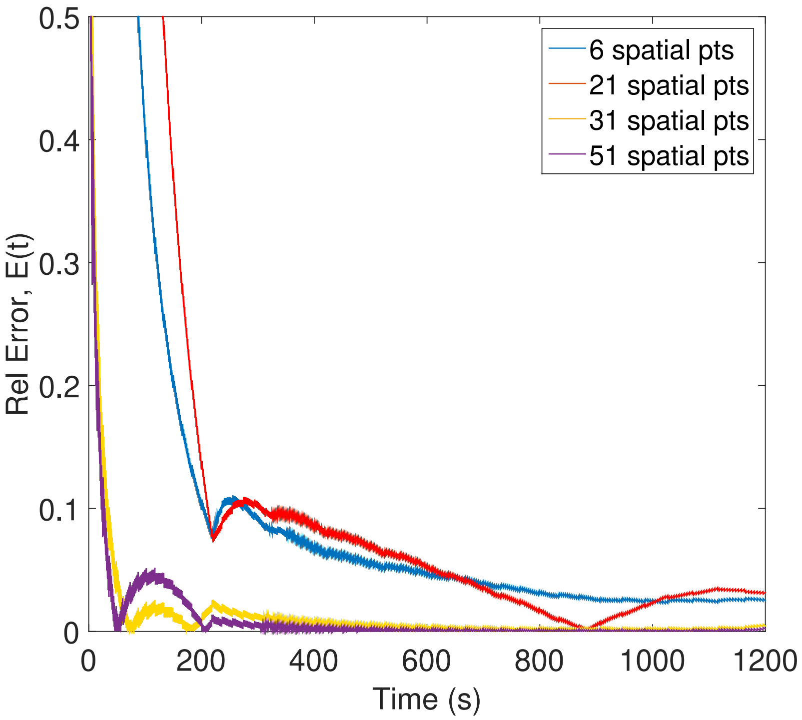


Figure 3.16: Effect of the number of spatial points on the convergence of approximate soluions (to the full-order solution) computed using the global DMDc method. The relative error measures are shown. Reprinted with permission from [129].

of the convergence of reduced-order models based on the proposed local DMDc methodology is presented in Figure 3.17, where we see an overall decrease in the relative error with respect to an increase in the number of clusters.

Another critical factor that dictates the level of accuracy of the reduced-order models is the truncation value r in the SVD step of the DMDc algorithm. Again, one expects an increase in the r value should result in a more accurate reduced-order model. We compare the convergence behavior of the local reduced-order models (both partitioned into 100 clusters) with two different r values in Figure 3.18. From the plot it is clear that the addition of the 3^{rd} mode has greatly improved the accuracy of the low-order approximation. However, as we further increased the value of r, a "blow-up" of the numerical values occurred (i.e., infinite error). Figure 3.19 shows that the

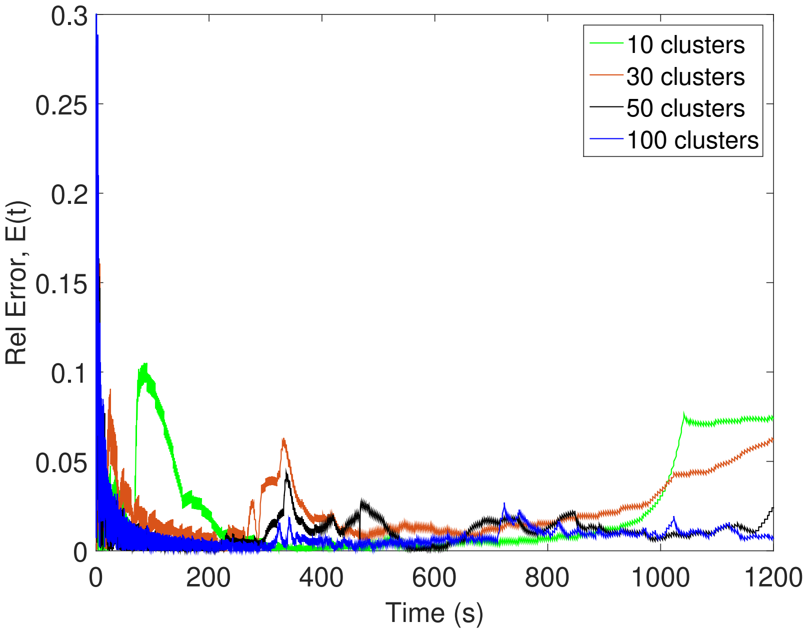


Figure 3.17: Effect of the number of clusters on the convergence of approximate solutions (to the full-order solution) computed using the local DMDc method. The relative error measures are shown. Reprinted with permission from [129].

magnitude of the eigenvalues of the reduced-order model describing cluster 1 is less than unity, $|\lambda| < 1$, implying that this "blow-up" is not due to the numerical instability but probably due to the inherent instability in the system alone. Please note that cluster 1 is used as a representative but this has been observed to be true for all the clusters. Therefore, we suspect as the r value increases it results in the inclusion of very small singular values that in turn introduce fluctuations into the system and results in a "blow-up".

3.4.2.4 Controller design

Equipped with the above results, we used the local reduced-order models developed (with $m = 100$ clusters and truncation value $r = 3$) by applying the proposed methodology to design a MPC framework for the hydraulic fracturing process. The MPC formulation is introduced first, followed by closed-loop simulation results.

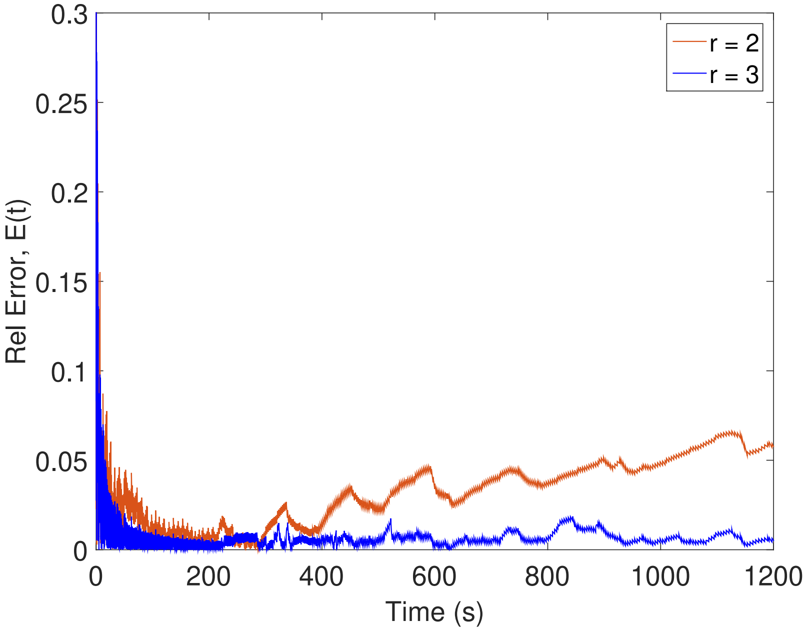


Figure 3.18: Effect of the truncation value 'r' on the convergence of approximate solutions (to the full-order solution) computed using the local DMDc method. The relative error measures are shown. Reprinted with permission from [129].

In practice, the ultimate goal of hydraulic fracturing is to increase the productivity of a stimulated (i.e., fractured) well. In summary, achieving uniform proppant concentration across the fracture at the end of pumping, which strongly influences the propped fracture height and geome-

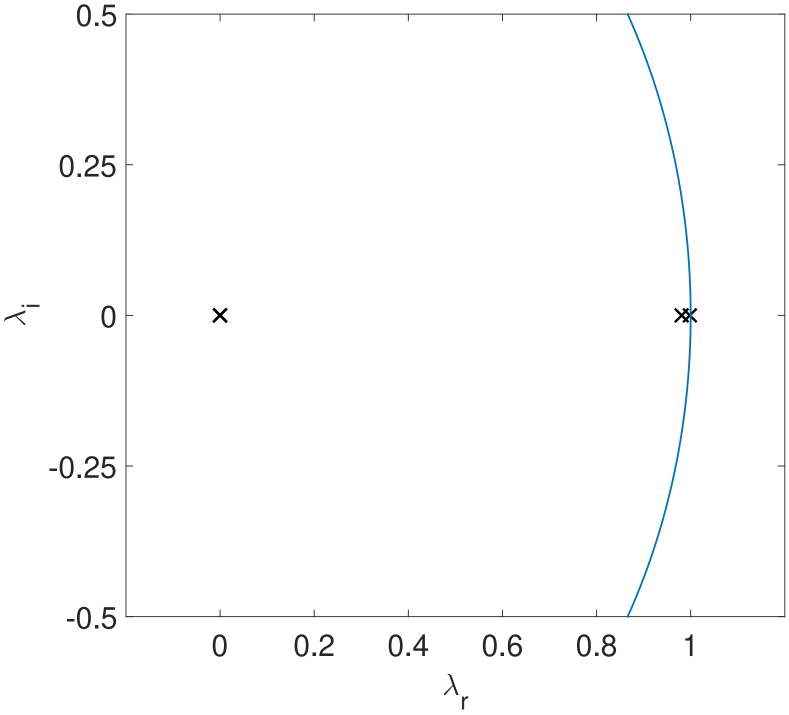


Figure 3.19: The approximate eigenvalues of the system matrix $\hat{\mathbf{A}}$ describing the local dynamics in the 1^{st} cluster, where λ_r and λ_i denote the real and imaginary parts. Note $|\lambda| < 1$ as shown by a unit circle. Reprinted with permission from [129].

try, is paramount for effective oil and gas extraction.

$$\underset{C_{0,k}}{Minimize} \qquad (\bar{\mathbf{C}}(t_f) - C_{target})^T \mathbf{Q_c} (\bar{\mathbf{C}}(t_f) - C_{target}) \tag{3.42a}$$

$$\text{s.t} \qquad \bar{\mathbf{x}}(t_{k+1}) = \bar{\mathbf{A}}_i \bar{\mathbf{x}}(t_k) + \bar{\mathbf{B}}_i C_{0,k}, \qquad \forall t_k \in \mathcal{C}^i \tag{3.42b}$$

$$\bar{\mathbf{y}}(t_k) = \bar{\mathbf{x}}(t_k) \tag{3.42c}$$

$$\bar{\mathbf{x}}(t_k) = \mathbf{x}(t_k) \tag{3.42d}$$

$$\mathbf{C}_{min} \leq \bar{\mathbf{C}}(t_k + j\Delta) \leq \mathbf{C}_{max}, \qquad \forall j = 0, \ldots, 10 - k \tag{3.42e}$$

$$C_{0,k-1+m} \leq C_{0,k+m} \leq C_{0,k-1+m} + 4, \qquad m = 1, \ldots, 10 - k \tag{3.42f}$$

$$2Q_0 \Delta \Big(\sum_k C_{0,k} \Big) = M_{prop} \tag{3.42g}$$

$$\bar{L}(t_f) = L_{opt}, \qquad \bar{W}_0(t_f) \geq W_{opt} \tag{3.42h}$$

where $\bar{(\cdot)}$ indicates the predicted state trajectory, $\mathbf{Q_c}$ is a positive definite matrix used to compute the weighted norm, t_f denotes the total treatment time, Δ is the sampling time, t_k is the current time, $\bar{\mathbf{C}}$ is the vector of proppant concentrations at 6 locations, and $C_{0,k}$ is the inlet proppant concentration (i.e., manipulated input) corresponding to k^{th} time interval, i.e., $t \in [t_k, t_{k+1})$.

In the above optimization problem, Eq. (3.42a) is a quadratic function which penalizes the squared deviation of the proppant concentration from the set-point at 6 different locations at the end of pumping. The predicted state trajectory $\bar{\mathbf{x}}(t)$ is computed using an approximate linear model, Eq. (3.42b), under the piecewise constant input profile computed by the MPC framework. This approximate model is available to us by applying the proposed local DMDc methodology and inverse transformation as detailed in Algorithm 2. Since temporal clustering is performed on the process data offline, we have to use different local in time state-space models at different time instants. Therefore, the corresponding approximate model, Eq. (3.42b), is used to predict the state trajectory at a particular time interval in which the calculation is being performed. More specifically, at any time $t_k \in \mathcal{C}^i$, $\bar{\mathbf{A}}_i$ and $\bar{\mathbf{B}}_i$ matrices corresponding to the i^{th} cluster (i.e., i^{th} temporal subdomain) will be selected to predict the state trajectory. Eq. (3.42c) specifies that the outputs of the process are equal to the states as stated before. The initial conditions are given in Eq. (3.42d) which are obtained at each sampling time from the current full-state measurements. Eq. (3.42e) imposes limits on the concentration profiles to avoid premature termination of the process. Eq. (3.42f) demands a monotonic increase in the input concentration profiles with a maximum increase of 4 ppga/stage, where ppga is a concentration unit used in petroleum engineering that refers to 1 pound of proppant added to a gallon of water. Lastly, Eq. (3.42g) specifies the total mass of the proppant to be injected, M_{prop}, and Eq. (3.42h) imposes the terminal constraints on the fracture geometry at the end of pumping. Please note that the performance index in the above MPC formulation does not include a penalty term on the process inputs. In the specific example presented here, this is implicitly addressed by introducing the input constraint, Eq. (3.42f), specifying that the rate of change of inlet proppant concentration at the wellbore should not exceed 4 ppga/stage. Additionally, in practice, there is no need for penalizing control actions on proppant concentration, because it is not the major source of high production expenses required for hydraulic fracturing.

The MPC block diagram that describes the closed-loop operation of hydraulic fracturing processes is presented in Figure 3.20. The process outputs are used as a feedback to the controller in which the developed local in time reduced-order models predict the state trajectory and compute the optimal input profile that will be applied to the process.

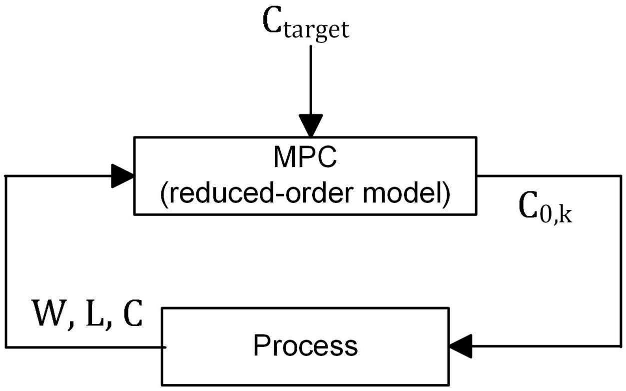


Figure 3.20: Schematic representation of the closed-loop operation of hydraulic fracturing. Reprinted with permission from [129].

3.4.2.5 Closed-loop simulation results

The values used for various process parameters are the same as given earlier. In the closed-loop simulation, the sampling time Δ and the total time of operation t_f were chosen to be 100 s and 1220 s, respectively during which a total of $M_{prop} = 48,000$ kg of proppant has been injected into the fracture. In the field, real-time measurements of the fracture width at the wellbore and the fracture length are available through downhole pressure analysis and microseismic monitoring techniques respectively. This raw data will be analyzed to remove outliers and ensured they are chosen correctly. Therefore, the time required for the data to be prepared typically ranges from 1-5 minutes. Due to this reason, the sampling time was chosen to be around 2 minutes. During the initial stages of the hydraulic fracturing process, a high-pressure fluid (called pad) consisting mostly of water is pumped to break the rock and propagate fractures in the formation at perforated sites. The duration of this pad stage, t_p, has been fixed at 220 s to reach the desired fracture length of $L_{opt} = 135$ m without tip-screen out (premature termination of the process). The feedback control system was initialized after the injection of pad (i.e., $t_k \geq t_p$). The proppant pumping schedule was partitioned into 10 substages (i.e, $k = 1, \dots, 10$) and the duration of each substage is given by the sampling time, Δ. We assumed that at the beginning of each substage, the full-state measurements of $\mathbf{x}(t_k)$ are available, which are used to predict the estimates of the future states via the developed approximate models. For illustration purposes, the approximate model for the 1^{st} cluster computed by the proposed approach has been presented below.

$$\bar{\mathbf{A}}_1 = \begin{bmatrix} 0 & 0.00012 & 0.00059 & -0.00091 & 0.00042 & 0.00082 & -0.00220 & 0 \\ -0.00193 & 0.98919 & -0.15894 & 0.25095 & -0.11058 & -0.22025 & 0.61124 & 0 \\ 0.00061 & 0.00288 & 0.04775 & -0.07333 & 0.03343 & 0.06585 & -0.17973 & 0 \\ -0.00043 & 0.00233 & -0.07444 & 0.11438 & -0.05211 & -0.10268 & 0.28031 & 0 \\ 0.00042 & 0.00271 & 0.03331 & -0.05117 & 0.02333 & 0.04595 & -0.12539 & 0 \\ 0.00084 & 0.002945 & 0.06602 & -0.10141 & 0.04622 & 0.09106 & -0.24853 & 0 \\ -0.00231 & 0.00198 & -0.18183 & 0.27936 & -0.12729 & -0.25079 & 0.68460 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\bar{\mathbf{B}}_1 = \begin{bmatrix} 0.00088 \\ 3.56998 \\ 0.04725 \\ -0.04819 \\ 0.03557 \\ 0.06147 \\ -0.13161 \\ 0 \end{bmatrix} \tag{3.43}$$

The predicted state estimates are then used in the optimization problem to compute the control inputs and the corresponding process behavior that minimizes the squared deviation from its pre-specified target value. The uniform target proppant concentration desired at the end of pumping is $C_{target} = 10$ ppga. Please refer to [130] for details on how to calculate this set-point value. The positive definite matrix, $\mathbf{Q_c}$, containing the weights on the penalty function of the MPC optimization problem is considered to be unity. However, one can use appropriate weights depending on the process nature. The first step, $C_{0,1}$, of the input profile, $C_{0,k}$, obtained by solving the optimization problem over a prediction horizon length of N_p was applied to the high-fidelity model in a sample-and-hold fashion, and this procedure was repeated at every sampling time until the end of treatment. In this application, we used a shrinking horizon which gives $N_p = t_f - t_k$. We solve an optimization problem to compute the control inputs while regulating the output values at the final time step, which is what we desire in this specific application.

In order to gauge the performance of the proposed model reduction method in designing a closed-loop control scheme, we compared it with MOESP, one of the well known subspace identification methods. We developed a 6^{th}-order model by applying the MOESP algorithm to regress a linear time-invariant state-space model of the hydraulic fracturing process using the simulation results from the high-fidelity model. The developed reduced-order model was then used in the MPC framework to achieve the same control objective. Figure 3.21 presents the generated spatial proppant concentration at the end of pumping. As can be observed from the plot, the controller based on the reduced-order model developed by local DMDc drives the concentration closer to the target value thus achieving the control objective to a desirable level compared to the MOESP one, thus establishing the superiority of the proposed method (note that the order of the two reduced-order models is the same). The pumping schedule (i.e., the input concentration profile required to achieve uniform proppant concentration) for the corresponding process parameters and the overall operation time considered is presented in Figure 3.22.

Remark 11. *Please note that due to the presence of moving boundaries and the highly nonlinear*

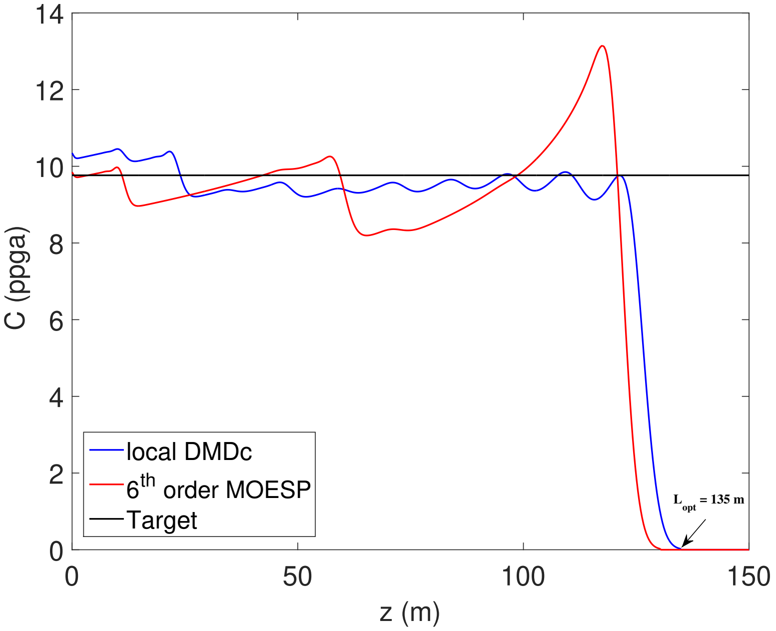


Figure 3.21: Spatial profiles of the proppant concentration at the end of pumping under MPC based on the reduced-order models developed by the proposed local DMDc and the MOESP methods. Reprinted with permission from [129].

nature of the problem, an extra effort should be made in selecting the test input profile so that the overall range and trend of the test input profile is similar to the optimal input profile, especially since we are approximating the system using a linear model. Therefore, for significantly different process set-point values, the local in time reduced-order models should be recomputed carefully with different test input profiles to obtain good approximate models.

3.5 Conclusions

In Chapter 3, we tackled the problem of approximating highly complex dynamics encountered in moving boundary systems such as hydraulic fracturing via spectral decomposition based methods by tailoring the basis functions to capture the local dynamic behavior. Specifically, we developed a local model reduction framework that uses data clustering within projection-based methods to accurately capture the local behavior of every portion of the solution trajectory in the

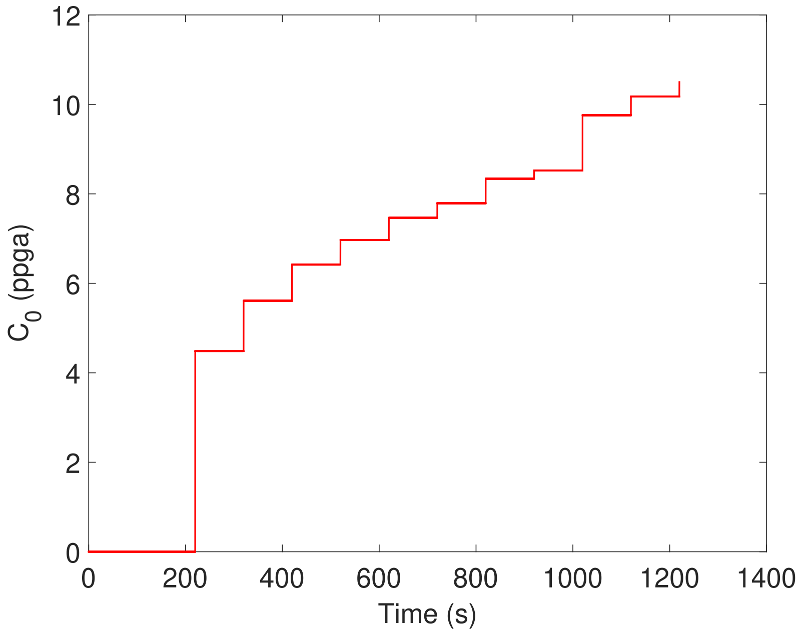


Figure 3.22: Pumping schedule (inlet concentration profile) required for achieving uniform proppant concentration at the end of pumping under MPC based on the reduced-order models developed by the proposed local DMDc method. Reprinted with permission from [129].

state-space. Intuitively, since moving boundary systems are associated with varying spatial features, a time-domain partitioning is performed as a first step. To achieve this, a novel clustering strategy based on Mixed Integer Nonlinear Programming is implemented on the time-series data associated with the system of interest. This data can be obtained either from numerical simulations of the high-fidelity model or from experimental runs and should be able to represent a large portion of the solution state space (i.e., the data is collected at different initial conditions, inputs etc.). The motive of this step is that each cluster is implicitly characterized by the similarity of its dynamic behavior, and therefore, represents a particular portion of the solution. In the next step, the reduced bases are derived within each cluster, using methods such as POD and DMD.

In the case of hydraulic fracturing process, a full-order discretization resulted in a large system of 450 spatial and $20,000$ temporal nodes. The local technique when applied to reduce this system resulted in a 95% decrease in the dimensionality and computation time while having superior approximation accuracy in comparison with the global counterparts. Moreover, local DMDc resulted in linear state-space models, which were easily incorporated in an MPC framework to design an optimal pumping schedule that achieved uniform proppant concentration throughout the fracture. Overall, the results are very encouraging and illustrate the potential of the proposed method in approximating the prevalent dynamics and designing a feedback control system from just snapshot measurements alone, even though the entire spatiotemporal evolution data is only partially captured by the measurements (the fact that we considered the concentration measurements at only 6 different locations).

4. HANDLING SPATIAL HETEROGENEITY IN RESERVOIR PARAMETERS USING POD-BASED EnKF FOR MODEL-BASED FEEDBACK CONTROL OF HYDRAULIC FRACTURING[*]

In Chapter 3, we presented a novel framework based on local model reduction to accurately approximate complex dynamics associated with hydraulic fracturing process. However, we employed reservoir models where the rock mechanical properties (eg. Young's modulus, permeability etc.) were assumed to be known and spatially homogeneous. However, field data indicates that even within the same rock formation, the performance of hydraulic fracturing can be notably different [131, 132]. This variability can, in part, be attributed to the spatially varying rock mechanical properties such as the Young's modulus. As a consequence, one of the key tasks, which has to be performed prior to the model-based controller design, is the characterization of spatially varying rock mechanical properties. We have seen that a large number of discrete elements are required to provide a detailed description of the reservoir in a hydraulic fracturing process. This results in the formulation of a large-scale inverse problem where the number of parameters to be estimated far exceeds the number of available field measurements, resulting in an ill-posed problem.

To deal with this unidentifiability issue, the original problem must be reformulated and one way to achieve this is to reduce the number of unknown model parameters by parameterization. Additionally, it is of critical importance to preserve the spatial features in the geological properties during the parameterization process. Therefore, in Chapter 4 we extend the idea of model reduction to the parameter space to develop an integrated framework to parameterize the unknown spatially varying Young's modulus profile via POD, keeping the key spatial pattern in the geological properties intact, and to update the statistical information using the available process measurements via a Monte Carlo data assimilation technique, the Ensemble Kalman Filter.

---

[*] Reprinted with permission from"Handling of spatial heterogeneity using POD-based EnKF in model-based feedback control of hydraulic fracturing," by A. Narasingam, P. Siddhamshetty and J. S. Kwon, Industrial & Engineering Chemistry Research, 2018, 57, 3977–3989. Copyright 2018 by American Chemical Society.

4.1 Computational methods

Chapter 4.1.1 provides a brief description of the particular parameterization and data assimilation techniques essential for characterizing the spatially varying Young's modulus profile of the reservoir followed by the detailed numerical algorithm. A schematic of the proposed method is shown below.

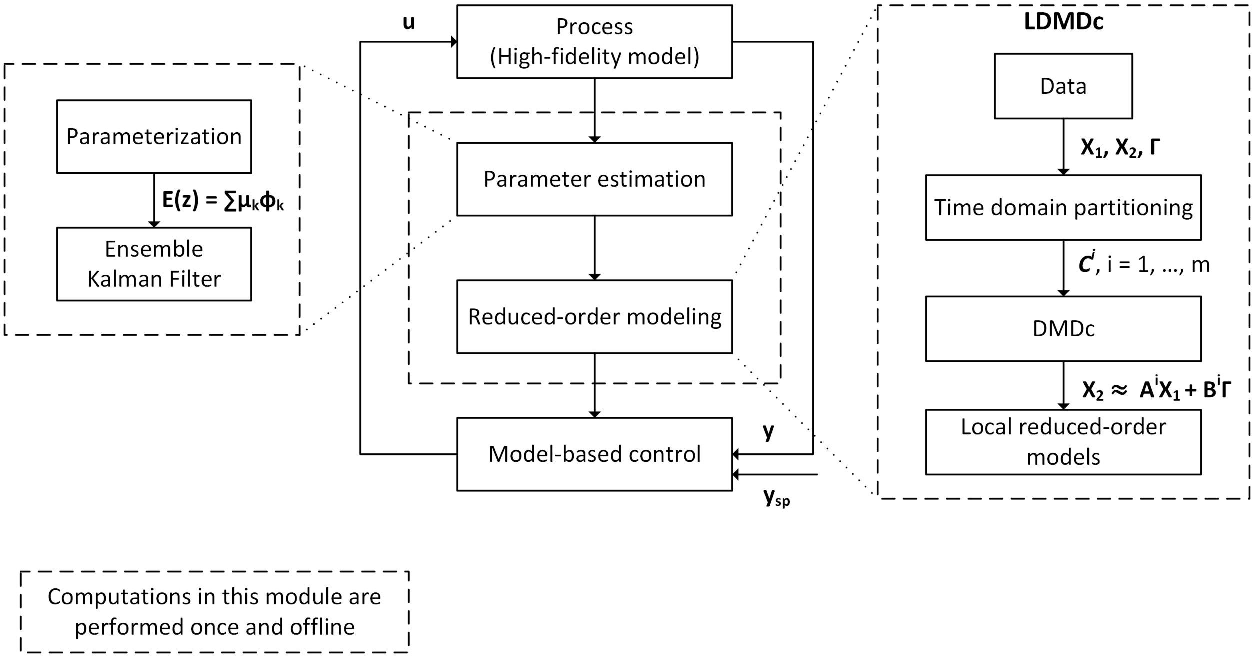


Figure 4.1: Overall flow integrating the POD-based EnKF parameter estimation scheme with LDMDc-based MPC. Reprinted with permission from [133].

4.1.1 POD-based EnKF

In this work, we employ the EnKF, which is a statistical Monte-Carlo simulation where the ensemble of process model states is evolved forward in time with the ensemble mean as the best state estimate [134] and the ensemble spread as the error variance. It combines a stochastic model with a prior assumption about the states, process noise, and measurement noise to derive the first two statistical moments of the states' posterior distribution. In other words, it is a sequential state estimation method for dynamical systems. It has also been widely used to estimate process model parameters [135, 136, 137]. In this work, we use it for solving the inverse problem to estimate the spatially varying Young's modulus profile of the reservoir.

In a practical implementation, the EnKF involves a forecast step which predicts the mean and covariance of the process model parameters at the next time step using the high-fidelity process model of hydraulic fracturing described by Eqs. (2.1)-(2.11). While the true values of the spatially varying parameters are unknown, we can assume that we have access to the first two statistical moments (i.e., mean and covariance) of the underlying geological features of the formation based on historical data. Please note, in reservoir characterization problems, the input reservoir properties are unknown, so two-point geostatistical assumptions are made regarding the reservoir properties and the initial ensemble is generated as random samples drawn from a multivariate Gaussian distribution defined by a variogram or covariance function. If historical data of nearby reservoirs or already induced fractures is available, such data can be used as a starting point for making assumptions regarding the mean and covariance function for generating the initial ensemble of realizations of the unknown reservoir properties. Such data can also be available, to an extent, from the mini-frac test performed prior to the hydraulic fracturing operation. Using this a priori knowledge, an initial ensemble of R realizations is constructed by adding R different perturbations to the mean of the spatially varying parameter distribution. In the next step, called the assimilation step, the cross covariance between the predicted observations and the unknown parameters is updated by means of a filter gain using the available true measurements.

In a hydraulic fracturing process, these available measurements are the estimates of fracture length and fracture width at the wellbore which are provided via the processed micro-seismic and downhole pressure data, respectively. Now, since the number of parameters to be estimated is huge compared to the available measurements, the inverse problem becomes ill-posed. One can increase the identifiability of this inverse problem by reducing the number of parameters to be estimated by transforming the original high-dimensional parameter space to a low-dimensional subspace.

Also, since the spatially varying geological properties follow certain patterns, it is required to preserve the key spatial features observed in the high-dimensional parameter space. To achieve this, we apply POD to extract the most dominant geological features and provide a low-dimensional representation of the parameters, which is then incorporated in the EnKF parameter estimation routine. Below, we present a mathematical formulation for the steps involved in the POD-based EnKF estimation methodology.

Let us consider a horizontal fracture geometry containing I grid points in the discretized system. Therefore, the representation of the spatially heterogeneous parameters (e.g. Young's modulus profile) will be in the form of an $I \times 1$ vector. One can generate R realizations of the spatially varying Young's modulus profile as described above, and construct a representative ensemble of this data in the form of an $I \times R$ matrix. Now, we calculate the correlation matrix using the realizations and solve the integral eigenvalue problem to obtain the basis functions. The members of the ensemble can be approximately represented by a set of basis functions as follows:

$$\mathbf{E} \approx \sum_{i=1}^{d} \mu_i \varphi_i \tag{4.1}$$

Here $\mathbf{E} \in \mathbb{R}^{I \times 1}$ denotes the spatially varying Young's modulus, which is an I-dimensional vector, $\varphi_i \in \mathbb{R}^I$ denote the orthogonal basis functions, $\mu_i \in \mathbb{R}$ are real-valued coefficients that define the appropriate linear combination of basis functions to approximate the true spatially varying Young's modulus profile, and d is the dimension of the reduced parameter space obtained by examining the energy (in a mean square sense) of the singular values. The expression in Eq. (4.1) is a reduced-order representation of the parameter space where the basis functions, shared by all the ensemble members, contain the key spatial patterns of the unknown parameters. These basis functions induce a parameterization of the spatially varying Young's modulus profile if we allow the coefficients μ_i to be free. Therefore, these coefficients will constitute the actual variables to be estimated via the EnKF, thereby effectively reducing the number of parameters to be estimated from I to d. Let the vector $\boldsymbol{\mu}_k \in \mathbb{R}^d$ be the collection of all the free variables used in Eq. (4.1) at time t_k. Now, at time step k, let us consider the forward (high-fidelity) model:

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, \boldsymbol{\mu}_k) \tag{4.2}$$

where $\mathbf{f}$ is a nonlinear function given in Eqs. (2.1)-(2.11) of the state $\mathbf{x}_k$, the coefficients $\boldsymbol{\mu}_k$ and the input $\mathbf{u}_k$. Please note that we have used a subscript k for $\boldsymbol{\mu}$ simply to denote that its value updates at every data assimilation step, but the model parameters do not change with time. Let $\mathbf{y}_k \in \mathbb{R}^p$ be the predicted data (i.e., system outputs) at time t_k. In this work, the measurements are the fracture length and the fracture width at the wellbore, which are a subset of the state vector $\mathbf{x}_k$. The p- dimensional vector of predicted outputs is therefore generated as $\mathbf{y}_k = \mathbf{A}\mathbf{x}_k$, where $\mathbf{A}$ is a linear operator which contains zeroes and ones, and picks out the components of $\mathbf{x}_k$ that are measured at time t_k. However, since the states are a nonlinear function of the model parameters, the relation between the predicted outputs and the model parameters can be represented by a nonlinear operator, $\mathbf{h}$, as $\mathbf{y}_k = \mathbf{h}(\boldsymbol{\mu}_k, \mathbf{u}_k)$.

Since the measurements are a nonlinear function of the model parameters, we can consider an augmented state vector, $\mathbf{X}_k \in \mathbb{R}^{(d+p)\times 1}$, that includes the output measurements, $\mathbf{y}_k$, along with the unknown model parameters, $\boldsymbol{\mu}_k$:

$$\mathbf{X}_k = \begin{bmatrix} \boldsymbol{\mu}_k \\ \mathbf{y}_k \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu}_k \\ \mathbf{h}(\boldsymbol{\mu}_k, \mathbf{u}_k) \end{bmatrix} \tag{4.3}$$

With this definition, the relation between the predicted output vector, $\mathbf{y}_k$, and the augmented state vector, $\mathbf{X}_k$, is given by the following linear expression:

$$\mathbf{y}_k = \mathbf{H}\mathbf{X}_k = \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{\mu}_k \\ \mathbf{h}(\boldsymbol{\mu}_k, \mathbf{u}_k) \end{bmatrix} \tag{4.4}$$

where $\mathbf{H} \in \mathbb{R}^{p\times(d+p)}$, $\mathbf{0} \in \mathbb{R}^{p\times d}$ and $\mathbf{I} \in \mathbb{R}^{p\times p}$. In this work, the state variables, $\mathbf{x}_k$, include the fracture width and proppant concentrations at various locations across the fracture and the fracture length, the model parameters include spatially varying Young's modulus, although it is parameterized via POD and the coefficients, $\boldsymbol{\mu}_k$, are considered as the actual variables to be estimated. Additionally, the observations, $\mathbf{y}_k$, include the fracture length and fracture width at the wellbore that are processed from the microseismic monitoring and downhole pressure analysis. The inputs to the system include the flow rate and the proppant concentration at the wellbore. Algorithm 3 details the sequence of steps involved in the estimation of spatially varying parameters using the POD-based EnKF scheme.

We refer the readers to [138] for rigorous details regarding the practical EnKF implementation.

4.2 Simulation results

For the high-fidelity simulation, we generated a synthetic spatially varying Young's modulus profile which is shown in Figure 4.2 (black). This is taken to be the true parameter profile. Eqs. (2.1)-(2.11) constitute the high-fidelity model and they are solved using a higher-order discretization scheme to describe the fracture propagation, fluid flow, and proppant transport simultaneously. The fracture propagation is terminated at 135 m and the spatial domain is discretized with each grid point having a size 0.1 m, resulting in a total of $I = 1351$ points. The fluid injection rate is fixed at $0.03 \ m^3/s$. The fracture width at the wellbore and the fracture length obtained from the simulation are considered as the true process measurements, giving $p = 2$. The values of the key process parameters used in the simulation are summarized in Table 2.1.

To estimate the true Young's modulus distribution generated above, we construct an initial ensemble of $R = 100$ parameter realizations. To achieve this, we consider a mean that is different from the true parameter profile to represent the initial distribution. In practice, as noted earlier, the first two statistical moments of the spatially varying Young's modulus profile are known a priori based on available historical data. The mean of the initial ensemble is shown in Figure 4.2 (green). A covariance function is then used to generate 100 parameter realizations by adding 100 different perturbations to the initial ensemble mean. Figure 4.2 (blue) shows two representative realizations selected from the initial ensemble that are at the extremes of the ensemble range. Each realization consists of $I = 1351$ points describing the spatial variability in the Young's modulus profile. The

Algorithm 3 Parameter estimation using integrated POD-based EnKF

1: Generate R realizations of the unknown parameter fields $\hat{\mathbf{E}} = \{\mathbf{E}^1, \mathbf{E}^2, \ldots, \mathbf{E}^R\} \in \mathbb{R}^{I \times R}$ using the mean and covariance functions available from the prior knowledge.

2: Parameterize the ensemble of realizations via POD, and compute the spatial basis functions $\boldsymbol{\Phi} = \{\varphi_1(z), \varphi_2(z), \ldots, \varphi_d(z)\} \in \mathbb{R}^{I \times d}$, leaving the coefficients, $\mathbf{M} = \{\boldsymbol{\mu}_k^1, \boldsymbol{\mu}_k^2, \ldots, \boldsymbol{\mu}_k^R\} \in \mathbb{R}^{d \times R}$, to be free.

3: Given a vector of true measurements $\mathbf{d}_k \in \mathbb{R}^p$ available at every sampling time t_k, generate R artificial observations for data assimilation.

$$\mathbf{d}_k^j = \mathbf{d}_k + \epsilon_k^j, \;\; j = 1, \ldots, R; \;\; \epsilon_k^j \in N(0, \mathbf{C}_D) \tag{4.5}$$

which can be stored in the columns of $\mathbf{D}_k = \{\mathbf{d}_k^1, \ldots, \mathbf{d}_k^R\} \in \mathbb{R}^{p \times R}$. Here ϵ is a random perturbation drawn from a normal distribution such that $\epsilon \in N(0, \mathbf{C}_D)$, where $\mathbf{C}_D \in \mathbb{R}^{p \times p}$ is the covariance matrix of the measurement noise.

4: *Forecast* : At time t_k, for every realization of the spatially varying parameters in the set $\mathbf{M}$, the forecast of the ensemble of augmented states, $\mathbf{X}_k^f = \{\mathbf{X}_k^{1,f}, \ldots, \mathbf{X}_k^{R,f}\} \in \mathbb{R}^{(d+p) \times R}$, is generated according to:

$$\mathbf{X}_k^{j,f} = \begin{bmatrix} \boldsymbol{\mu}_k^{j,f} \\ \mathbf{y}_k^{j,f} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu}_{k-1}^{j,a} \\ \mathbf{h}(\boldsymbol{\mu}_{k-1}^{j,a}, \mathbf{u}_{k-1}) \end{bmatrix} \tag{4.6}$$

where the superscripts f and a denote the forecast and assimilation steps, respectively.

5: *Assimilation*: The covariance matrix at time t_k, $\mathbf{C}_k \in \mathbb{R}^{(d+p) \times (d+p)}$, associated with the ensemble of augmented states is computed as:

$$\bar{\mathbf{X}}_k^f = \frac{1}{R} \sum_{j=1}^{R} \mathbf{X}_k^{j,f} \tag{4.7}$$

$$\mathbf{C}_k = \frac{1}{R-1} \big(\mathbf{X}_k^f - \bar{\mathbf{X}}_k^f \cdot \mathbf{1}_R\big)\big(\mathbf{X}_k^f - \bar{\mathbf{X}}_k^f \cdot \mathbf{1}_R\big)^T \tag{4.8}$$

where $\bar{\mathbf{X}}_k^f$ denotes the ensemble average of the augmented states and $\mathbf{1}_R \in \mathbb{R}^{1 \times R}$ is a row vector whose elements are all equal to 1. The update to the predicted state and outputs is then computed as:

$$\mathbf{K}_k = \mathbf{C}_k \mathbf{H}^T \big(\mathbf{H} \mathbf{C}_k \mathbf{H}^T + \mathbf{C}_D\big)^{-1} \tag{4.9}$$

$$\mathbf{X}_k^a = \mathbf{X}_k^f + \mathbf{K}_k \big(\mathbf{D}_k - \mathbf{H} \mathbf{X}_k^f\big) \tag{4.10}$$

where $\mathbf{K}_k \in \mathbb{R}^{(d+p) \times R}$ is the Kalman gain of the ensemble at time t_k.

6: The sequence of forecast and assimilation steps is repeated until all the available measurements are assimilated.

7: The mean of the free coefficients in the final ensemble, $\bar{\boldsymbol{\mu}}^a \in \mathbb{R}^d$, along with the previously obtained basis functions, $\boldsymbol{\Phi}$, is used to compute the unknown spatially varying Young's modulus profile, $\mathbf{E}_{true}$, as follows:

$$\mathbf{E}_{true} \approx \sum_{i=1}^{d} \bar{\mu}_i^a \varphi_i = \boldsymbol{\Phi} \bar{\boldsymbol{\mu}}^a, \qquad \bar{\boldsymbol{\mu}}^a = \frac{1}{R} \sum_{j=1}^{R} \boldsymbol{\mu}^{j,a} \tag{4.11}$$

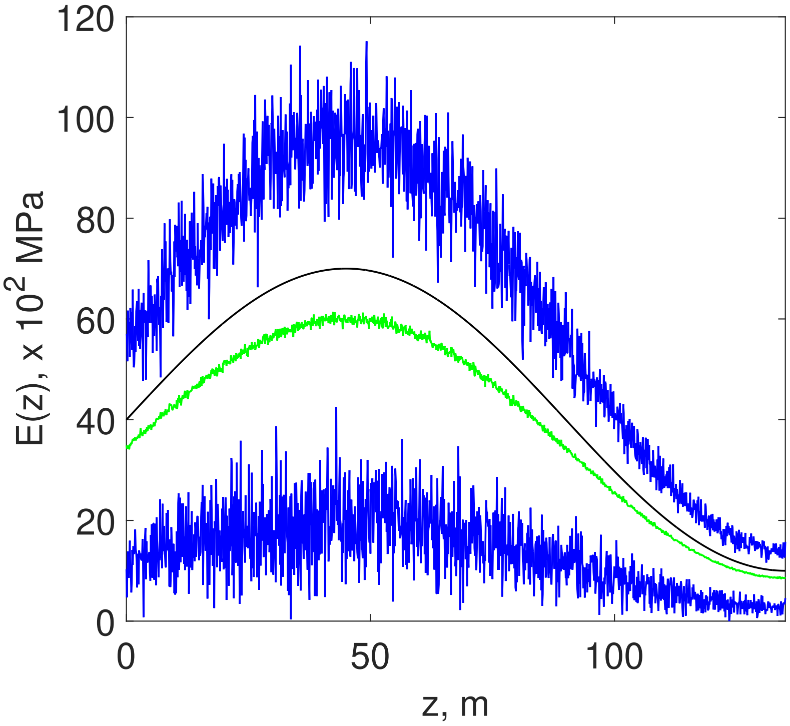


Figure 4.2: The true Young's modulus profile (black) along with two representative realizations (blue) and the mean (green) of the initial ensemble. Reprinted with permission from [133].

high-fidelity model of the hydraulic fracturing process is solved with each realization that represents a spatially varying Young's modulus profile to predict the fracture width at the wellbore and the fracture length. The predicted outputs are compared against the true measurements at every 25 s. The total duration of the process is 1000 s, which gives $k = 40$ time steps. Consequently, every parameter realization utilizes a set of 2×40 distinct measurements. In the numerical experiments, measurement errors are modeled as white noise (Gaussian random samples with zero mean) with the covariance $\mathbf{C}_D$, the value of which is taken as $diag(0.005, 0.008)$, where the first term in the matrix corresponds to the error in the wellbore width and the second term to the fracture length, respectively. The units of the above matrix $\mathbf{C}_D$ are meters and the above values have been determined based on the order of the wellbore width whose maximum value is approximately 15 mm (the variance of the error is taken as 5 mm). During the forward simulations, the predicted outputs are assimilated against the true measurement values with added noise to update the ensemble at each time step. Therefore, we performed a total of 100×40 high-fidelity simulations.

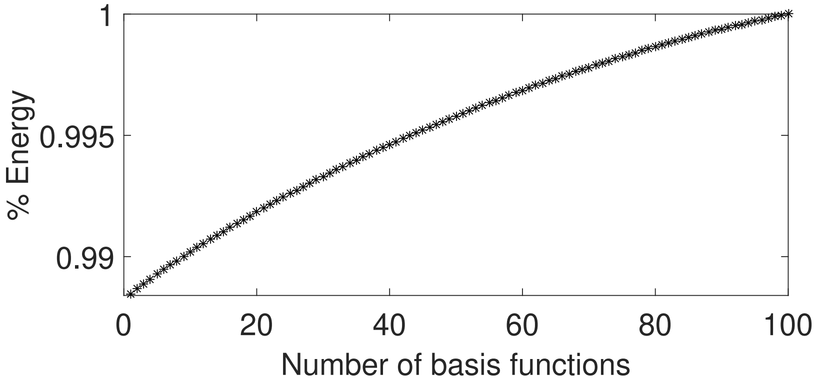
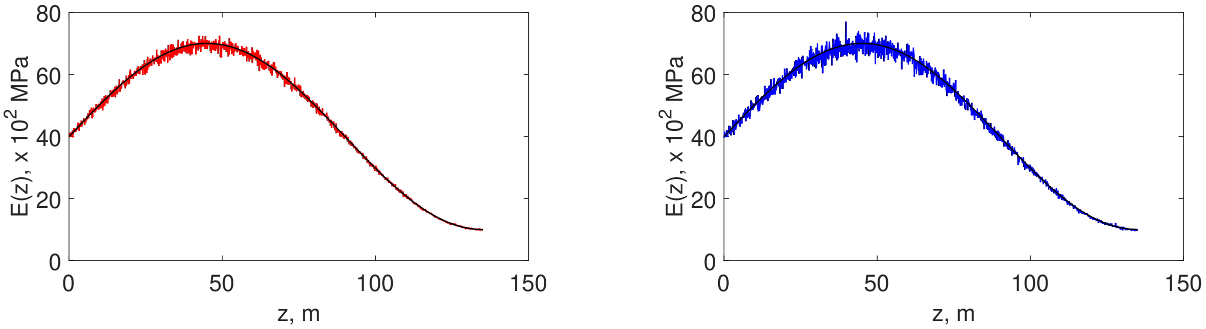


Figure 4.3: Cumulative energy captured by the basis functions as represented by their normalized singular values. Reprinted with permission from [133].

We first performed POD on the initial ensemble where 100 different spatially varying Young's modulus profiles are sampled from the known first two statistical moments. The computed spatial basis functions are then employed to reduce the number of parameters to be estimated. Figure 4.3 shows the cumulative energy content of the singular values associated with the ensemble. Based on this we chose the first $d = 10$ basis functions which correspond to 99% of the total energy captured by the singular values. The coefficients ($\boldsymbol{\mu} \in \mathbb{R}^{10\times 100}$) that describe the linear combination of these basis functions to reconstruct the parameter profiles are allowed to be free in the EnKF algorithm which leads to the parameterization of the spatially varying Young's modulus profile. We emphasize that the number of parameters to be estimated in this problem was reduced to 10 from 1351, which alleviates the ill-posed nature of the original problem.

Remark 12. *Please note that the reduction in the number of parameters via POD depends on the covariance function chosen to sample the initial ensemble. This is because the rate of decay of singular values depends on the covariance function used.*



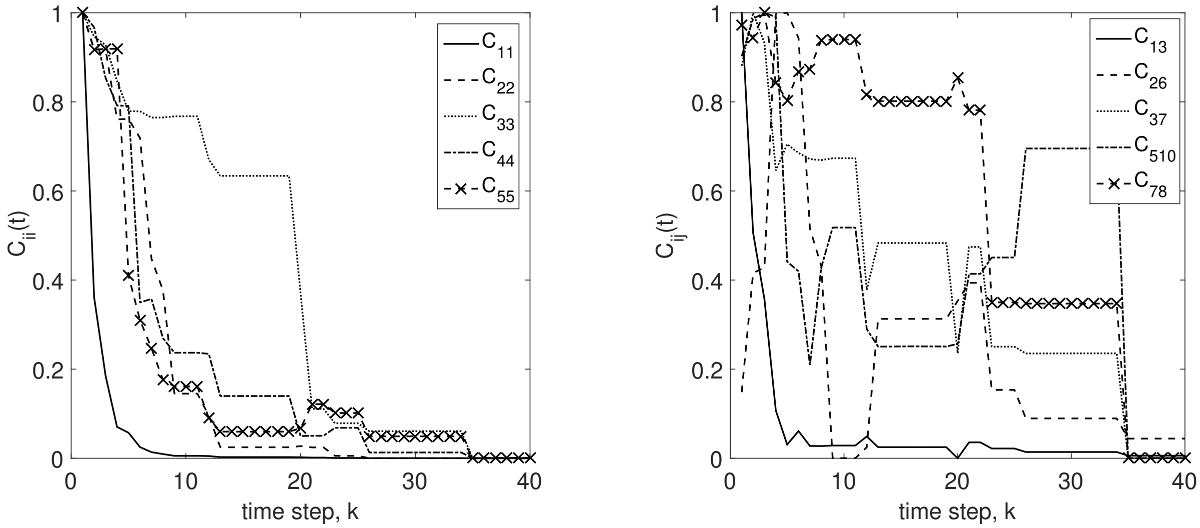
(a) Final ensemble mean (red) and the true parameter profile (black).

(b) A realization from the final ensemble (blue) and the true profile (black).

Figure 4.4: A comparison between the true and the final estimated Young's modulus profile through POD-based EnKF. Reprinted with permission from [133].

Once the parameterization step is completed, we implemented the EnKF algorithm to obtain the best estimate of the true spatially varying parameter profile. Since a major computational requirement is associated with the high-fidelity simulations used in the forecast step, the EnKF algorithm was conveniently parallelized to speed up the computations. In our work, the high-fidelity simulations and the EnKF programs are developed using Parallel Computing Toolbox™ in MATLAB® and then scaled up to computing clusters by running it on MATLAB Distributed Computing Server™. Each assimilation step gives a set of new coefficient vectors that are used with the basis functions to regenerate the estimated spatially varying Young's modulus profile. This set of new parameters is further used in the forecast of output variables at the next time step. The forecast and the assimilation steps are repeated until all the measurements are integrated into the high-fidelity simulation.

Remark 13. *We note that ill-conditioning might occur when computing the matrix inverse in the Kalman gain formulation, in Eq. (3.23). This potential singularity of the inverse computation requires the use of pseudo inverse via Truncated Singular Value Decomposition (TSVD). We would like to note that we have followed the practical implementation suggested in [138, 139, 140] in computing the pseudo inverse and using the singular vectors to complete the analysis step in the EnKF algorithm.*



(a) Evolution of the error variances, C_{ii}, during data assimilation via the EnKF.

(b) Evolution of the error covariances, C_{ij}, during data assimilation via the EnKF.

Figure 4.5: Profiles showing the evolution of the error covariances, $\mathbf{C}_k$, during the data assimilation steps. Reprinted with permission from [133].

The results of parameter estimation are shown in Figure 4.4. Specifically, Figure 4.4a shows an ensemble member, which represents a candidate Young's modulus profile, after the final update step and Figure 4.4b shows the final ensemble mean. In a typical EnKF implementation, the mean of the final ensemble is taken as the approximation for the unknown parameter profile. Therefore, it is evident from the figure that the POD-based EnKF method provided a very good estimate for the true spatially varying Young's modulus profile after assimilating all the available measurement data. The estimated parameter profiles in Figure 4.4 retain the major spatial trend of the true Young's modulus profile. This is attributed to the basis functions which capture the dominant spatial patterns in the underlying data of the prior distribution. Figure 4.5 shows the evolution of error covariances during the data assimilation process. One would expect the covariances to approach to zero as more data is being assimilated. This can be observed in Figure 4.5 where both variances (C_{ii}) and covariances (C_{ij}) decline to zero with time. Additionally, this also shows that the ensemble estimates converge to the true values at the end of data assimilation. Although the figures show a few elements taken from the covariance matrix to describe their time evolution, a

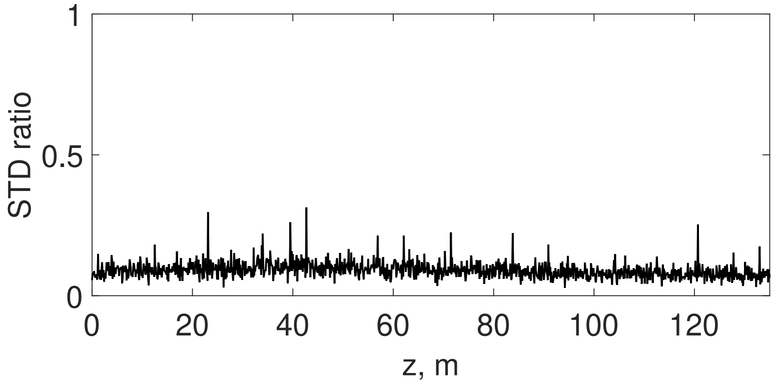


Figure 4.6: Posterior to prior standard deviation ratio of the spatially varying Young's modulus profile. Reprinted with permission from [133].

similar behavior is observed throughout the covariance matrix. Please note that the covariances have been normalized prior to the plots.

One way to measure the performance of the EnKF method is the reduction in the final ensemble spread which is evident by comparing Figures 4.4a and 4.4b. From the figure, it can be seen that the difference between the final ensemble member and the mean is small. Note that for illustration purposes, only one representative member is selected randomly from the final ensemble. However, this reduction in the spread is observed with respect to all the final ensemble members. Furthermore, by comparing Figures 4.2 and 4.4, it can be observed that the uncertainty in the ensemble members is also reduced between the prior and posterior distributions. In order to highlight this point more clearly, we show the ratio of the standard deviation between the final and initial ensemble in Figure 4.6. Please note that a value close to 0 indicates a large reduction in uncertainty whereas a value close to 1 indicates a small reduction. As can be seen from the figure, a significant reduction in uncertainty is achieved throughout the fracture. Figure 4.7 shows the mean square error (MSE) plot between the true parameter values and the mean of the ensemble posterior distribution at each time step. Note that the MSE values have been normalized with respect to the maximum. As can be seen from the figure, a significant reduction in the error is observed between

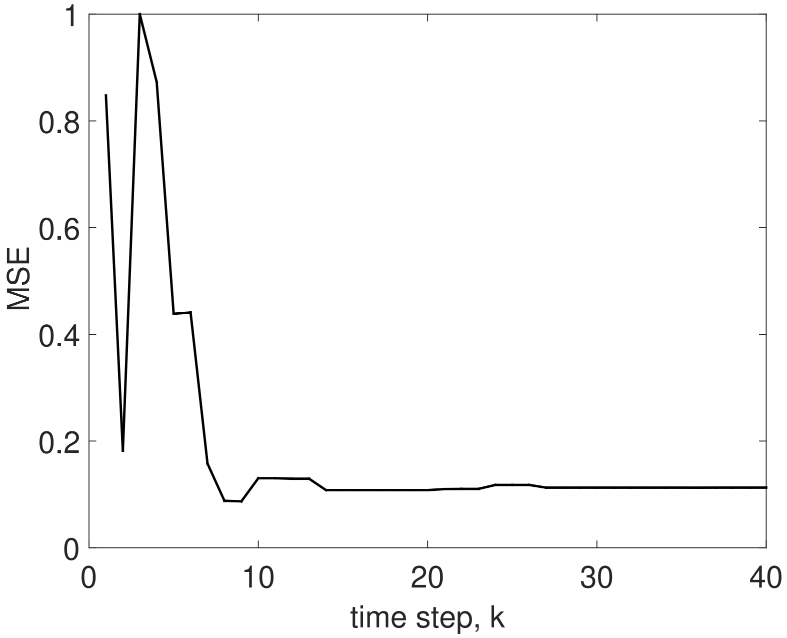
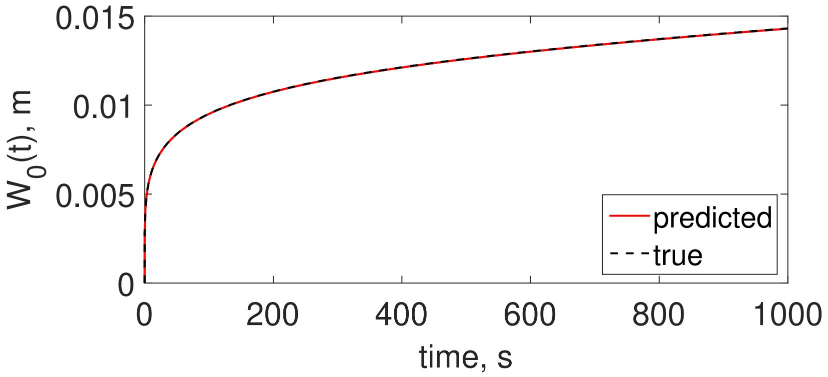


Figure 4.7: The profile of MSE with time. Reprinted with permission from [133].

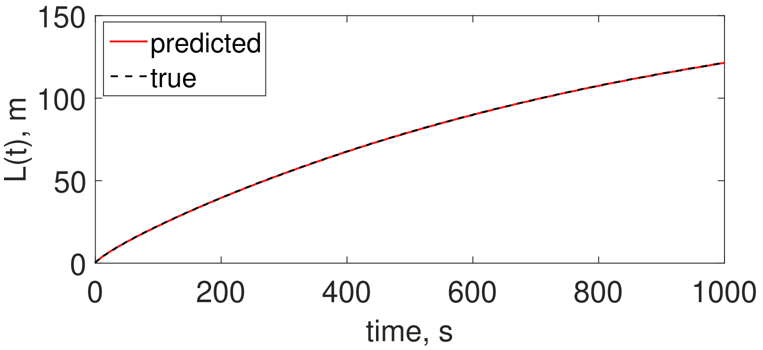
the initial and final update steps. As more data are integrated into the high-fidelity model, the EnKF estimate converges to the original value, resulting in a decrease in the MSE .

In addition to finding a good approximation between the true and estimated parameter values, one of the primary objectives of parameter estimation of dynamic models is to accurately predict the process dynamics. This is an important feature in order to design model-based control systems or perform dynamic optimizations of the process. The comparison between the process dynamics predicted by the updated model, based on the estimated parameters, and the true values is shown in Figure 4.8. The predictive capability of the updated model is evident from the figure, which shows a good match between the true and predicted values of the fracture width at the wellbore and the fracture length through Figure 4.8a and 4.8b, respectively.

Remark 14. *Please note that although the EnKF handles the nonlinear dynamics correctly during the forecast step, it sometimes fails in the analysis step [141, 142, 143, 144]. More specifically, for a highly nonlinear problem, the current state estimates obtained from EnKF and the current*



(a) Fracture width at the wellbore, $W_0(t)$, predicted by the updated model (red) and its true values (black).



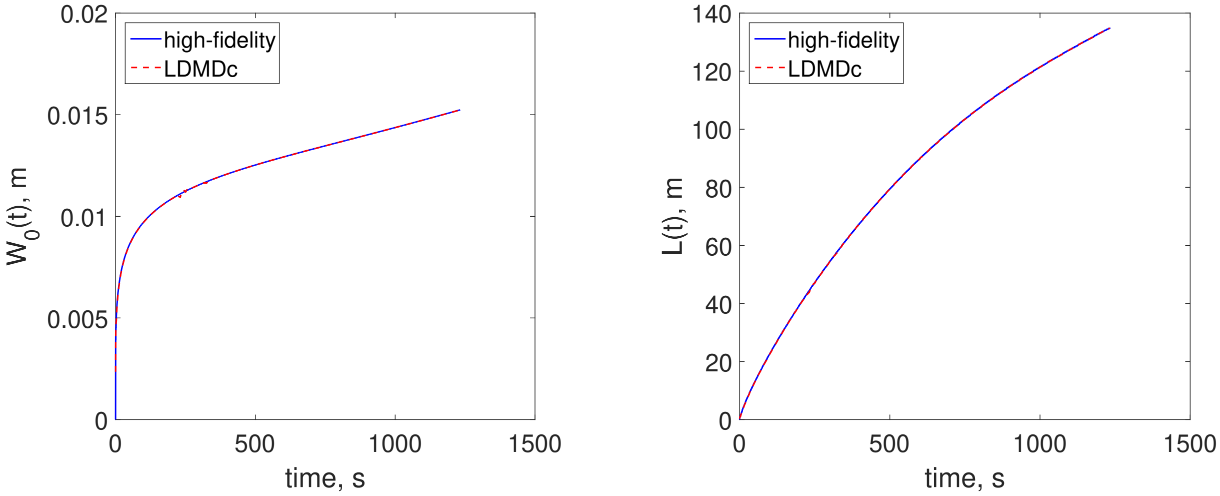
(b) Fracture length, $L(t)$, predicted by the updated model (red) and its true values (black).

Figure 4.8: Profiles of the predicted system outputs and their true measurements. Reprinted with permission from [133].

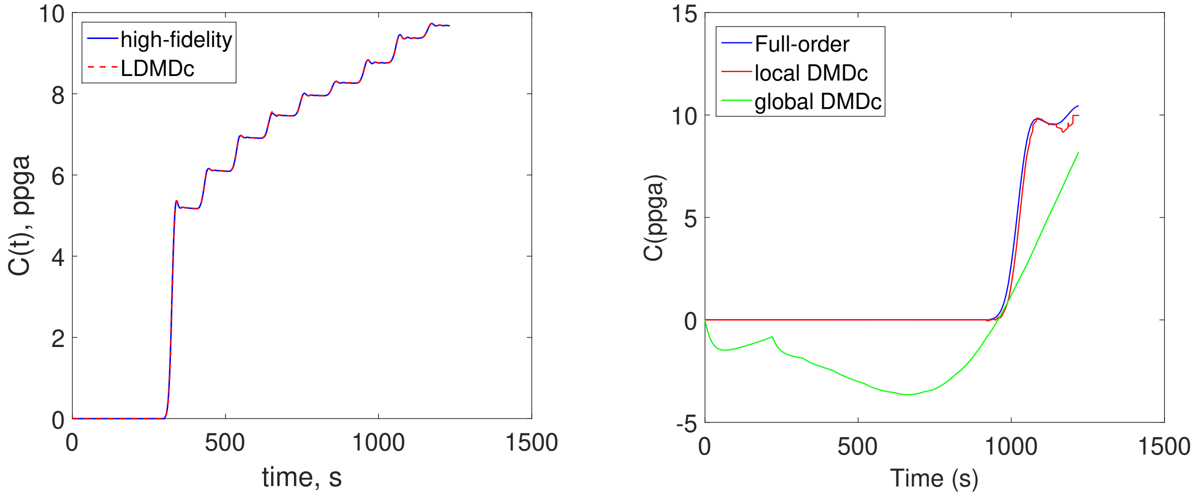
state values computed by re-solving the forward model with the updated parameters may not be identical, implying the production of unrealistic parameter estimates. However, we emphasize that in this manuscript we only estimated the model parameters via the EnKF and the predicted outputs are obtained from the forward model with the updated model parameters.

4.3 Closed-loop results

With the updated spatially varying Young's modulus profile, open-loop simulations of the high-fidelity model are performed using a set of training inputs to generate the data required for developing the reduced-order models. The LDMDc methodology described in Chapter 3.4.2 determined the optimal number of clusters as $m = 100$ and the corresponding reduced-order models are generated within each temporal subdomain. Figure 4.9 shows the comparison between the actual state



(a) $W_0(t)$ at the wellbore (b) $L(t)$ of the fracture



(c) $C(t)$ at $z = 18$ m (d) $C(t)$ at $z = 90$ m

Figure 4.9: Comparison of the approximate solutions computed using local in time reduced-order models with the true measurements of the fracture width at the wellbore, fracture length and proppant concentration at two different positions. Reprinted with permission from [133].

values and their corresponding approximate solutions of the fracture width at the wellbore, fracture length, and proppant concentration profiles at two different positions. It can be observed that the

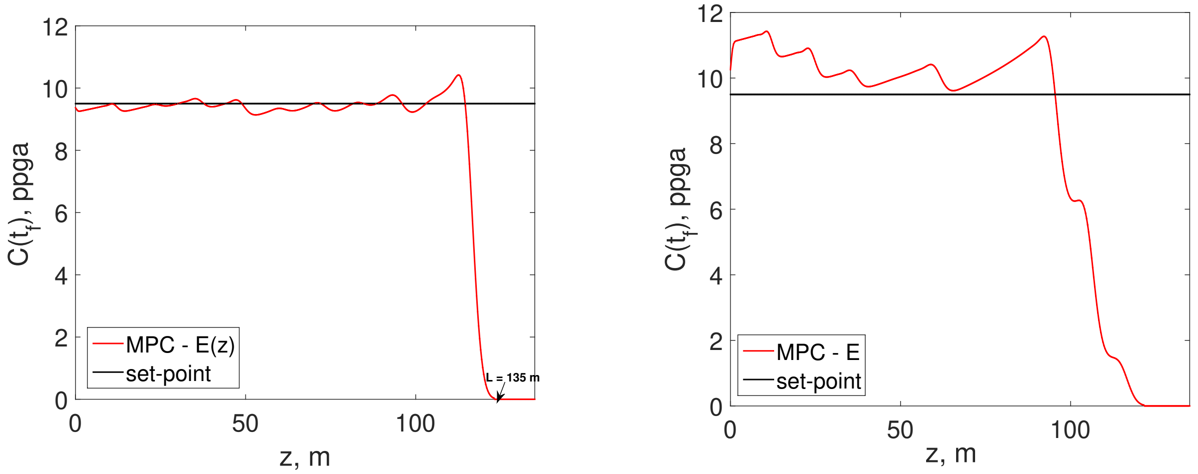
LDMDc technique provides accurate approximations to the true process state values obtained from the high-fidelity model.

Now that we have accurate reduced-order models, a model-based feedback controller of Eqs. (3.42a)-(3.42h) is designed to achieve a uniform final proppant concentration profile across the fracture along with a desired final fracture length and width to enhance the production of the fractured well. The values used for various process parameters are the same as the ones given in Table 2.1. In the closed-loop simulation, the sampling time Δ and the total time of operation t_f were chosen to be 100 s and 1220 s, respectively, during which a total of $M_{prop} = 48,000$ kg of proppant has been injected into the fracture. The sampling time was chosen to be around 100 s to accommodate the practical time required for processing the raw microseismic data and downhole pressure analysis to obtain the wellbore fracture width and fracture length measurements, respectively. During initial stages of the hydraulic fracturing process, a high-pressure fluid (called pad) consisting mostly of water is pumped to break the rock and propagate fractures in the formation at perforated sites. The duration of this pad stage, t_p, has been fixed at 220 s to reach the desired fracture length of $L_{opt} = 135$ m without tip-screen out (premature termination of the process). The feedback control system is initialized after the injection of pad (i.e., $t_k \geq t_p$). The proppant pumping schedule is partitioned into 10 substages (i.e, $k = 1, \ldots, 10$) and the duration of each substage is given by the sampling time, Δ. We assumed that at the beginning of each substage, the full-state measurements of $\mathbf{x}(t_k)$ are available ($\mathbf{x} = [W_0, L, \mathbf{C}]$), which are used to predict the future states via the developed local in time reduced-order models. The predicted state trajectories are then used in the optimization problem to compute the control inputs, and the corresponding process behavior that minimizes the squared deviation from its pre-specified target value. The uniform target proppant concentration desired at the end of pumping is $C_{target} = 10$ ppga. Please refer to [85] for details on the calculation of this set-point value. The positive definite matrix, $\mathbf{Q_c}$, containing the weights on the penalty function of the MPC optimization problem is considered to be an identity matrix. However, one can use appropriate weights depending on the process nature. The first step of the input profile, $C_{0,k}$, obtained by solving the optimization problem over a prediction horizon length of N_p was applied to the high-fidelity model in a sample-and-hold fashion, and this procedure was repeated at every sampling time until the end of pumping. In this application, we used a shrinking horizon which gives $N_p = t_f - t_k$. We solve the optimization problem to compute control inputs while regulating the output value to be close to the set-point at the end of the final time step, which is what we desire in this specific application.

The results are presented in Figure 4.10a which shows the final concentration profile of the closed-loop system under MPC. Clearly, the designed controller drives the final proppant concentration (red) towards the target value (black) while propagating the fracture to the desired length (arrow) specified by the geometry constraints. For the sake of illustrating the effect of not characterizing the spatially varying Young's modulus profile prior to the design of feedback control systems, we have also designed a closed-loop system under MPC based on a constant Young's modulus profile. Here, the value of the Young's modulus is fixed at $E = 50 \times 10^2$ MPa across the fracture. The closed-loop performance under the MPC designed based on the constant Young's modulus is shown in Figure 4.10b. As can be seen, when the rock properties are not characterized accurately, it can result in a poor controller performance, i.e, non-uniform final concentration profiles, thereby diminishing the production capabilities of the fracture well.

4.4 Conclusions

Accurate characterization of reservoir properties is of central importance to achieve a desired fracture geometry during a hydraulic fracturing process. However, the estimation of spatially varying geological properties in hydraulic fracturing is inherently ill-posed due to a limited number of measurements. In this work, parameterization is performed to reduce the dimensionality via a combined POD based EnKF method for the estimation of spatially varying Young's modulus profile. Parameterizing via POD resulted in a significant increase in the identifiability of the original parameter estimation problem, i.e., the number of variables to be estimated in the data assimilation framework is reduced from 1351 to 10. The method gives good results after all the available data has been integrated, and it is demonstrated that a significantly better match is obtained for the Young's modulus estimate relative to the synthetic true Young's modulus profile that is used for



(a) MPC based on the updated spatially varying Young's modulus $E(z)$.

(b) MPC based on the constant Young's modulus assumption.

Figure 4.10: Final proppant concentration profiles of the closed-loop system under MPC of hydraulic fracturing. Reprinted with permission from [133].

testing purposes. Additionally, we observed significant predictive capabilities from the updated model as evident from the close match to both the fracture length and the fracture width at the wellbore. The designed closed-loop controller based on the updated parameter profile, is able to drive the proppant concentration profile at each spatial location to a uniform target value at the end of pumping. Comparison of closed-loop simulation results shows the significance of accurately characterizing the spatial variability in the reservoir parameters.

5. APPLICATION OF KOOPMAN OPERATOR FOR MODEL-BASED CONTROL OF FRACTURE PROPAGATION AND PROPPANT TRANSPORT IN HYDRAULIC FRACTURING OPERATION*

In Chapter 3, we focused on generating accurate approximations to highly nonlinear processes such as hydraulic fracturing by tailoring the basis functions to capture local dynamics of every portion of the system trajectory. Although the local model reduction framework is superior in performance, sometimes it may lead to unstable models due to discontinuity between different clusters. Therefore, a single model that is accurate within a larger domain in the state-space would be beneficial in some cases. However, as discussed earlier, using model reduction methods directly will not yield good results. In Chapter 5, we investigate an alternative approach where we employ Koopman operator theory to generate data-driven models that approximate the hydraulic fracturing dynamics. It is particularly attractive because of its ability to provide (nearly) globally valid linear models. To be more specific, the Koopman linear models are often valid in a larger domain unlike local models whose validity is, in general, limited by the training data [7]. It is important to note here that this is a natural extension to the previous work due to the fact that DMDc is indeed a special case of Koopman theory which will be mathematically detailed below. Although counter intuitive, it is due to this close connection that DMDc is successful in approximating nonlinear dynamics using a linear system.

Operator theoretic approaches have recently sparked a lot of research interest, however, it is still in its infancy and has not been applied to large scale systems. Our contribution in Chapter 5 is demonstrating, on a non-trivial engineering application of hydraulic fracturing, the data-driven Koopman method for constructing linear models that can be readily used within a predictive control

---

*Reprinted with permission from "Application of Koopman operator for model-based control of fracture propagation and proppant transport in hydraulic fracturing operation," by A. Narasingam and J. S. Kwon, Journal of Process Control, 2020, 91, 25-36. Copyright is retained by the authors for all articles published in Elsevier journals.

*Reprinted with permission from "Data-driven identification of interpretable reduced-order models using sparse regression," by A. Narasingam and J. S. Kwon, Computers & Chemical Engineering, 2018, 119, 101-111. Copyright is retained by the authors for all articles published in Elsevier journals.

framework to maximize the productivity of a fractured well. To do so, we first apply the Koopman-based system identification method to create a dynamic model of the fracturing process and verify that it captures the system's true dynamic behavior. Then, we design a model predictive controller (MPC) using the model obtained in the previous step. Since the developed models are linear in the space of observable functions, the predictive controller when formulated in this space will lead to a standard convex optimization problem, which allows for extremely fast solutions compared to the original nonlinear problem. We consider two case studies of varying complexities. First, we apply the approach to regulate fracture geometry by considering only the fracture propagation dynamics governed by a PDE. Second, as a more sophisticated case study, we consider the simultaneous fracture propagation and proppant transport to determine the optimal pumping schedule that achieves uniform proppant concentration throughout the fracture at the end of operation. The overall dynamic model describing rock deformation, fluid flow, and proppant transport is a set of highly-coupled nonlinear equations defined over a time-dependent spatial domain making it considerably more complex than fracture propagation alone. Therefore, this calls for careful curating of the Koopman basis and we show how a priori system knowledge can be incorporated during system identification in choosing appropriate observable functions.

5.1 Preliminaries

In Chapter 5.1, we present some preliminaries on the Koopman operator theory and its application for model identification and controller synthesis.

5.1.1 Koopman Operator

Consider a discrete controlled nonlinear dynamical system given below:

$$x_{k+1} = F(x_k, u_k) \tag{5.1}$$

where $x_k \in \mathcal{X} \subseteq \mathbb{R}^n$ is the state of the system, $u_k \in \mathcal{U} \subseteq \mathbb{R}^m$ is the input, F is the flow map that evolves the system states forward in time, and $k \in \mathbb{Z}$ represents the time step such that $t_k = k\Delta$ where Δ is the sampling time.

Let us define a set of scalar-valued *observables* that are functions of the states and the inputs, $g : \mathcal{X} \times \mathcal{U} \rightarrow \mathbb{R}$. Each observable is an element of an infinite-dimensional function space $\mathcal{G}$ which, for example, can be defined by the Lebesgue square-integrable functions, $\mathcal{G} = \ell^2(\mathcal{X} \times \mathcal{U}, \mathbb{R})$, or other appropriate spaces [145]. In this infinite-dimensional function space, the flow of the system is governed by the Koopman operator, $\mathcal{K} : \mathcal{G} \rightarrow \mathcal{G}$ that defines the dynamics of observables $g \in \mathcal{G}$ along the trajectories of the system as:

$$\mathcal{K}g \triangleq g \circ F \tag{5.2}$$

By definition, the Koopman operator is linear even though the underlying dynamical system is nonlinear. For all $g_1, g_2 \in \mathcal{G}$ and all $\alpha, \beta \in \mathbb{R}$, it satisfies

$$\begin{aligned} \mathcal{K}(\alpha g_1 + \beta g_2) &= (\alpha g_1 + \beta g_2) \circ F \\ &= (\alpha g_1 \circ F) + (\beta g_2 \circ F) \\ &= \alpha \mathcal{K} g_1 + \beta \mathcal{K} g_2 \end{aligned} \tag{5.3}$$

The system identification method described below exploits the fact that any finite-dimensional nonlinear system can be equivalently represented using an infinite-dimensional linear system by transforming the traditional state-space to the space of functions (observables) of the system's states and inputs. Please note that the Koopman operator has originally been proposed for autonomous dynamical systems, and it can be adopted to controlled systems by considering joint observations of state and input, i.e., $\mathcal{K}$ for the above system Eq. (5.1) can be considered as the classical Koopman operator for the augmented system, $x^+ = F(x, u)$ and $u^+ = \mathcal{S}(u)$ where $\mathcal{S}(u)$ indicates a forward shift operator of a known input signal. This is just one way to adopt the Koopman operator to controlled systems, and for more generalizations the readers can refer to [146, 147, 148].

5.1.2 System identification using EDMD

The Koopman operator theory is conceptually developed on the infinite-dimensional function space $\mathcal{G}$. However, it is not practically feasible unless we can determine finite-dimensional approximations to the Koopman operator without a great loss in accuracy. To this end, we consider a finite-dimensional subspace $\bar{\mathcal{G}} \subset \mathcal{G}$ spanned by a set of basis functions $\phi(x,u) = [\phi_1(x,u), \cdots, \phi_{N\phi}(x,u)]^T$. Now, any observable function $\psi \in \bar{\mathcal{G}}$ can be represented as a linear combination of these basis functions as follows:

$$\psi = c_1\phi_1 + c_2\phi_2 + \cdots + c_{N_\phi}\phi_{N_\phi} = c^T\phi \tag{5.4}$$

For these functions in $\bar{\mathcal{G}}$, we seek to generate the finite-dimensional approximation of the Koopman operator, denoted as $K \in \mathbb{R}^{N_\phi \times N_\phi}$. Because, typically $\bar{\mathcal{G}}$ is not invariant with respect to $\mathcal{K}$, there is a residual term which is minimized in the L_2-norm sense via linear regression [149]. Intuitively, if these observables ψ represent the system's outputs (or equivalently, sensor measurements), a cost function to be minimized in an optimal control problem, or nonlinear constraints, then the evolution of these "nonlinear" observables can now be analyzed using a linear system via K. This is done by carrying a nonlinear transformation of the system states (outputs) to the so-called "lifted" space using the set of N_ϕ basis functions.

In its most general form, the Koopman linear system is presented as follows:

$$\phi(x_{k+1}, u_{k+1}) = K(x_k, u_k)\phi(x_k, u_k) \tag{5.5}$$

where $\phi \in \mathbb{R}^{N_\phi}$ and usually $N_\phi \gg n$. For the augmented state (x,u), the above formulation will admit any nonlinear dynamical system. However, as the dimension of the original system increases, the number of basis functions to be considered can grow to infeasible levels. So, to make the computations tractable and amenable in the case of predictive controller design (please see Chapter 5.2 for more details), the simplifications described below are introduced in the structure of the nonlinear transformation [10].

Specifically, the objective is to identify a linear dynamical system of the following form (similar to a linear time invariant state-space model) in the space of observables using time-series data generated by the control system of Eq. (5.1).

$$\phi(x_{k+1}) = A\phi(x_k) + Bu_k \tag{5.6}$$

For this purpose, we use the recently developed EDMD algorithm. To construct a finite dimensional approximation to the Koopman operator for the controlled system in Eq. (5.1), the EDMD algorithm requires

1. a time-series data set of N_t snapshot pairs satisfying the dynamical system in Eq. (5.1), which can be organized in the following matrices.

   $$\begin{gathered} X = [x_1, x_2, \cdots, x_{N_t}], \quad Y = [y_1, y_2, \cdots, y_{N_t}], \\ U = [u_1, u_2, \cdots, u_{N_t}] \end{gathered} \tag{5.7}$$

   Note that we use y instead of x_{k+1} here because the data above need not be temporally ordered as long as it satisfies $y_k = F(x_k, u_k)$.

2. a library of nonlinear basis functions $\{\phi_1, \phi_2, \ldots, \phi_{N_\phi}\}$ whose span is $\bar{\mathcal{G}} \subset \mathcal{G}$.

The EDMD algorithm then seeks to solve a least-squares problem to obtain K which is the transpose of the finite-dimensional approximation to the Koopman operator, $\mathcal{K}$:

$$\min_{K} \quad \sum_{k=1}^{N_t} \|\phi(y_k, u_k) - K\phi(x_k, u_k)\|_2^2 \tag{5.8}$$

In order to obtain the form described in Eq. (5.6), the following simplification can be introduced into the formulation. Since we are not interested in predicting the future values of inputs, without loss of generality we can assume that

$$\phi(x,u) = \begin{bmatrix} \phi(x) \\ \\ u \end{bmatrix}, \qquad \phi(y,u) = \begin{bmatrix} \phi(y) \\ \\ u \end{bmatrix} \tag{5.9}$$

which is to say that the nonlinear observable functions are applied to the system states alone and not the inputs. Suppose that N nonlinear functions are used to lift the states to the observable space and the m system inputs are added separately to the basis as shown in Eq. (5.9), we have $N_\phi = N + m$. To obtain the evolution of lifted states, we can then disregard the last m components of each of the terms in Eq. (5.8), decompose the Koopman matrix as $K = [A, B]$, and use only the first N rows in K, which leads to the following minimization problem

$$\min_{A,B} \quad \sum_{k=1}^{N_t} \|\phi(y_k) - A\phi(x_k) - Bu_k\|_2^2 \tag{5.10}$$

The practical solution to the above equation is obtained by using regularization via a TSVD and the value of K that minimizes Eq. (5.10) is given by

$$K_N := [A, B] = \phi_{XY}\phi_{XX}^{\dagger} \tag{5.11}$$

where † denotes the pseudo inverse, and the matrices are computed as

$$\begin{gathered} \phi_{XX} = \begin{bmatrix} \phi_X \\ \\ U \end{bmatrix} \begin{bmatrix} \phi_X \\ \\ U \end{bmatrix}^T, \quad \phi_{XY} = \phi_Y \begin{bmatrix} \phi_X \\ \\ U \end{bmatrix}^T \\ \phi_X = [\phi(x_1), \cdots, \phi(x_{N_t})], \quad \phi_Y = [\phi(y_1), \cdots, \phi(y_{N_t})] \end{gathered} \tag{5.12}$$

with

$$\phi(x_k) = \begin{bmatrix} \phi_1(x_k) \\ \vdots \\ \phi_N(x_k) \end{bmatrix} \tag{5.13}$$

Please note that any solution to Eq. (5.11) is a solution to Eq. (5.10), and the formulation of Eq. (5.11) has an advantage of being independent of the number of data samples N_t. Therefore, the Koopman linear system obtained using the above algorithm will be in the form of the controlled linear dynamical system given by Eq. (5.6), where $\phi \in \mathbb{R}^N$ is the lifted state in the observable space, and $A \in \mathbb{R}^{N\times N}$ and $B \in \mathbb{R}^{N\times m}$ are the matrices that describe the system dynamics in the lifted space.

The Koopman linear system Eq. (5.6), derived using the above algorithm, governs the evolution of the basis ϕ in the lifted space. With the use of these basis functions, all observables of interest $\psi \in \mathbb{R}^{n_\psi}$ can be determined (refer to Eq. (5.4)) by simply using a matrix of coefficients as $\psi(x_k) = C\phi(x_k)$, where $C \in \mathbb{R}^{n_\psi \times N}$. In summary, the basis functions are first evolved linearly using the Koopman operator, and the value of (required) observable is then computed by linearly combining these basis functions. In reality, not all observables can be contained in the span of the chosen subspace (recall, we are only using a finite-dimensional truncated function space). Thus, the matrix C must be chosen such that the projection of ψ onto the span$\{\phi_1, \cdots, \phi_N\}$ is minimized in the L_2-norm sense as below:

$$\min_{C} \quad \sum_{k=1}^{N_t} \|\psi(x_k) - C\phi(x_k)\|_2^2 \tag{5.14}$$

When dealing with dynamical systems, since the goal is stability analysis or controller synthesis, it is advantageous to reproduce the state dynamics. Therefore, the observable of interest here is the state itself, $\psi = x$. In such cases, we can assume that the basis also contains the state observable, i.e., $[\phi_1, \cdots, \phi_n]^T = x$. In particular, such a set of basis functions is said to be *state-inclusive*, and typically the solution to matrix C is trivial and can be obtained by $C = [I_n, 0_{N-n}]$.

The concept of linearizing dynamics is not new, but unlike local linearization around equilibrium points, the Koopman operator provides a global linearization of the original dynamics provided the set of (observable) basis functions is "rich enough." So, the main source of error stems from our choice and the finite size of the set of basis functions used in the approximation of the Koopman operator. Nevertheless, the convergence of EDMD approximation has been recently established under the assumption that the dimension of the subspace N approaches infinity [149]. Therefore, this error can be made arbitrarily small by considering a large set of basis functions. However, making a shrewd choice of the basis functions for unknown systems is still an active research area and a few studies suggest that canonical choices like radial basis functions, Hermite polynomials, etc., are a good starting point [10, 72]. Several other approaches propose using deep learning methods to discover these dictionaries [150, 151]. Nevertheless, when the system model is known or a limited knowledge on the functional forms present in the model is available, it can be readily incorporated in the basis function dictionary as will be shown with an illustrative example below.

5.2 Koopman Model Predictive Control

In Chapter 5.2, we briefly describe how the Koopman linear model obtained in Eq. (5.6) is used within an MPC scheme. In an MPC controller, an optimization problem is solved repeatedly over finite prediction horizons with respect to control inputs and predicted outputs of the system and a feedback behavior is achieved by taking process measurements as the initial condition. Typically, for nonlinear systems this is a non convex optimization problem due to the nonlinear dynamics. For the system in Eq. (5.1), the MPC problem at sampling time step k is given by

$$\begin{aligned}
\underset{u_k,\cdots,u_{k+N_p-1}}{\text{Minimize}} \quad & \sum_{i=1}^{N_p-1} x_{k+i}^T Q x_{k+i} + u_{k+i}^T R u_{k+i} + x_{k+N_p}^T Q_{N_p} x_{k+N_p} \\
\text{s.t} \quad & x_{k+i} = F(x_{k+i-1}, u_{k+i-1}), \quad i = 1, \cdots, N_p \\
& x_k = x_k^{measured} \\
& c(x_{k+i}) \leq b, \quad i = 1, \cdots, N_p \\
& \underline{u} \leq u \leq \overline{u}
\end{aligned} \tag{5.15}$$

where the cost matrices $Q, Q_{N_p} \in \mathbb{R}^{n \times n}$ and $R \in \mathbb{R}^{m \times m}$ are positive semi-definite matrices, N_p is the prediction horizon, $x_k^{measured}$ is the state measured at time $t_k = k\Delta$ where Δ is the sampling time during the closed-loop operation, $c : \mathbb{R}^n \to \mathbb{R}^{n_c}$ are nonlinear functions used to define the state constraints, $b \in \mathbb{R}^{n_c}$ is a constant vector, and $\underline{u}$ and $\overline{u}$ denote the lower and upper bounds, respectively, on the manipulated inputs.

In contrast to the above nonlinear optimization problem, in the Koopman MPC framework, a convex quadratic optimization problem (QP) is solved instead via lifting to the observable functions space. At time step k, the predictions of the system trajectory are initialized from the lifted state $\phi_k = \phi(x_k)$. Similarly, the objective function and the state constraints are all transformed to the lifted space. In this lifted space, the original nonlinear equation is replaced by the linear dynamics obtained in Eq. (5.6). Additionally, any nonlinear constraints associated with the system can be absorbed in the basis and therefore predicted in a linear manner through Eq. (5.6). For simplicity, if we consider the following output mapping ϕ

$$\phi = \begin{bmatrix} \bar{\phi}(x) \\ c(x) \end{bmatrix} \tag{5.16}$$

where $\bar{\phi} \in \mathbb{R}^{N-n_c}$ are some nonlinear functions in the lifted space, the nonlinear state constraints in x then translate to linear constraints in ϕ, i.e., $c(x) = E\phi(x)$ where $E = [0_n \quad I_{n_c}]^T$. Based on the above transformations, the Koopman MPC, in discrete formulation, solves the following QP problem at time step k of the closed-loop operation.

$$\begin{aligned} \underset{u_k,\cdots,u_{k+N_p-1}}{\text{Minimize}} \quad & \sum_{i=1}^{N_p-1} \phi_{k+i}^T Q \phi_{k+i} + u_{k+i}^T R u_{k+i} + \phi_{k+N_p}^T Q_{N_p} \phi_{k+N_p} \\ \text{s.t} \quad & \phi_{k+i} = A\phi_{k+i-1} + Bu_{k+i-1},\ i = 1, \cdots, N_p \\ & \phi_k = \phi(x_k^{measured}) \\ & E\phi_{k+i} \leq b,\ i = 1, \cdots, N_p \\ & \underline{u} \leq u \leq \overline{u} \end{aligned} \tag{5.17}$$

This feature of incorporating all nonlinearities within the observable basis is one of the main attractions of the Koopman operator approach. For example, in our previous work, the same idea was used to propose a stabilizing Lyapunov-based MPC formulation for nonlinear systems [152]. It can be further extended to translate any nonlinear objective functions such as those observed in Economic MPC formulations [153] to convex quadratic functions.

Remark 15. *It is emphasized that the convex MPC presented in Eq.* (5.17) *is only an approximation of the original nonlinear MPC problem in Eq.* (5.15). *But, nonlinear MPC problems are typically np-hard whereas the presented convex QP is polynomial time solvable and as long as the predictions are accurate, we expect the solution of the linear MPC problem to be close to the optimal solution of the nonlinear MPC problem. Please note that the rigorous quantification of approximation errors in the Koopman operator is still an actively researched area with several results available for the convergence of the EDMD algorithm under some assumptions [149].*

Remark 16. *Regarding the stability of the closed-loop system, the Koopman methodology presents an exciting avenue to develop stabilizing feedback controllers. Specifically, the Lyapunov theory can be naturally extended to analyze the stability of the resulting closed-loop system. Several successful results exploring this can be found in the following works [8, 152, 154]. One idea is to design a stabilizing control framework by explicitly including Lyapunov constraints within the MPC problem and show that the stability properties of the linear system (in the observable space) are inherited by the original nonlinear system under certain assumptions which is the subject of the following chapters.*

5.3 Regulating fracture geometry

To demonstrate and evaluate the performance of the Koopman system identification method outlined in Chapter 5.1.2, we first applied it to regulate the fracture propagation during a hydraulic fracturing process. In the following, we describe in detail the model identification, model performance evaluation and controller design.

5.3.1 Koopman model identification

The first step in the system identification method is to collect and construct the required data matrices. The fracture propagation dynamics can be obtained by solving part of the high-fidelity model described by Eqs. (2.1)-(2.6). The fracture propagation was terminated at 135 m, and the spatial domain was discretized with each grid point having a size of 0.3 m, resulting in a total of $n = 451$ points. In this work, we assumed only a partially observed system; the average fracture width, $W_{avg}(t)$, fracture length, $L(t)$ and fracture width at six uniformly spaced locations, $[W_1(t), \cdots, W_6(t)]$, obtained from the numerical experiment were considered as the true process measurements, i.e., $x(t) = [W_{avg}(t), L(t), W_1(t), \cdots, W_6(t)]^T \in \mathbb{R}^8$.

The data required to construct the Koopman linear model was collected from 100 simulated trajectories over varying operating time periods with each numerical experiment terminated when the fracture propagates to a total length of 135 m. A total of 1000 (synchronous) time samples per trajectory were used to populate the required data matrices. Each trajectory starts with the same initial condition and was subjected to an input signal generated randomly over each sampling time. The manipulated inputs were bounded as $\{\underline{u}, \overline{u}\} = \{0.03, 0.06\}$ m^3/s. This results in the data matrices X and Y of size 8×10^5 and the matrix U of size 1×10^5. A 50% *train-test split* was then created to partition the collected data set into training and validation sets.

Remark 17. *Please note that in the case of a real hydraulic fracturing process, the unmeasurable states such as local fracture width, proppant concentration, etc. can be estimated by using the high-fidelity model and a state observer such as Kalman Filter (see [84] for reference). Alternatively, one can design a state estimator directly for the Koopman linear model using linear observer design methodologies [14, 15]. When the high-fidelity model is unknown, one can use classical system identification techniques such as time-delayed measurements to construct the data matrices required for Koopman operator approximation (see [12, 65]).*

Once the data has been collected, the next step is to choose a set of basis functions that span the Koopman subspace to which the system dynamics are lifted. In this case, we chose a basis of monomials of the system states with the total degree less than or equal to 2 (all possible linear and bilinear combinations) as follows:

$$\begin{aligned} &\phi_i(x) \in \Big\{ \prod_{j=1}^{8} x_j^{m_j} \mid (m_1, \cdots, m_8) \in \mathbb{Z}_0^+, \sum_{j=1}^{8} m_j \leq 2 \Big\} \\ \implies &\phi(x) = [1,\ W_{avg},\ L,\ W_1,\ \cdots,\ W_6,\ W_{avg}^2,\ W_{avg}L,\ \cdots,\ W_6^2]^T \end{aligned} \tag{5.18}$$

where $\mathbb{Z}_0^+$ denotes the set of non-negative integers. Therefore, the total number of functions in the basis $\{\phi_i\}_{i=1}^N$ is equal to $N = 45$. This choice was motivated by the presence of polynomial terms in the governing equations. One observation here is that when the order of the monomials is 1, they represent the states themselves; in other words, this basis is *state-inclusive*. Consequently, (any of) the original physical states can be recovered by a trivial linear transformation of the lifted states back to the state space as discussed in Chapter 5.1.2. For unknown systems, there is no established way to select the dictionary although some canonical choices like radial basis functions have been proposed as a good starting point [72]. As a matter of fact, in the next example we will see that a canonical basis like the polynomial basis selected here fails to accurately capture the dynamics. In such cases, system knowledge becomes paramount.

Using the collected data and the constructed basis ϕ, we determined approximate linear predictors for the fracture propagation model described in Eqs. (2.1)-(2.6). We then evaluated its accuracy by comparing the Koopman model predictions to each of the validation data sets, one of which is shown in Figure 5.1. From the figure it can be seen that a relatively good agreement between the true model and the identified linear model was achieved with respect to the test data. To assess this comparison quantitatively, goodness of fit for the trajectory of system states starting from the same initial condition subjected to random input profiles was calculated and averaged over $N_{total} = 100$ simulations using the average relative root mean squared error (RMSE) defined

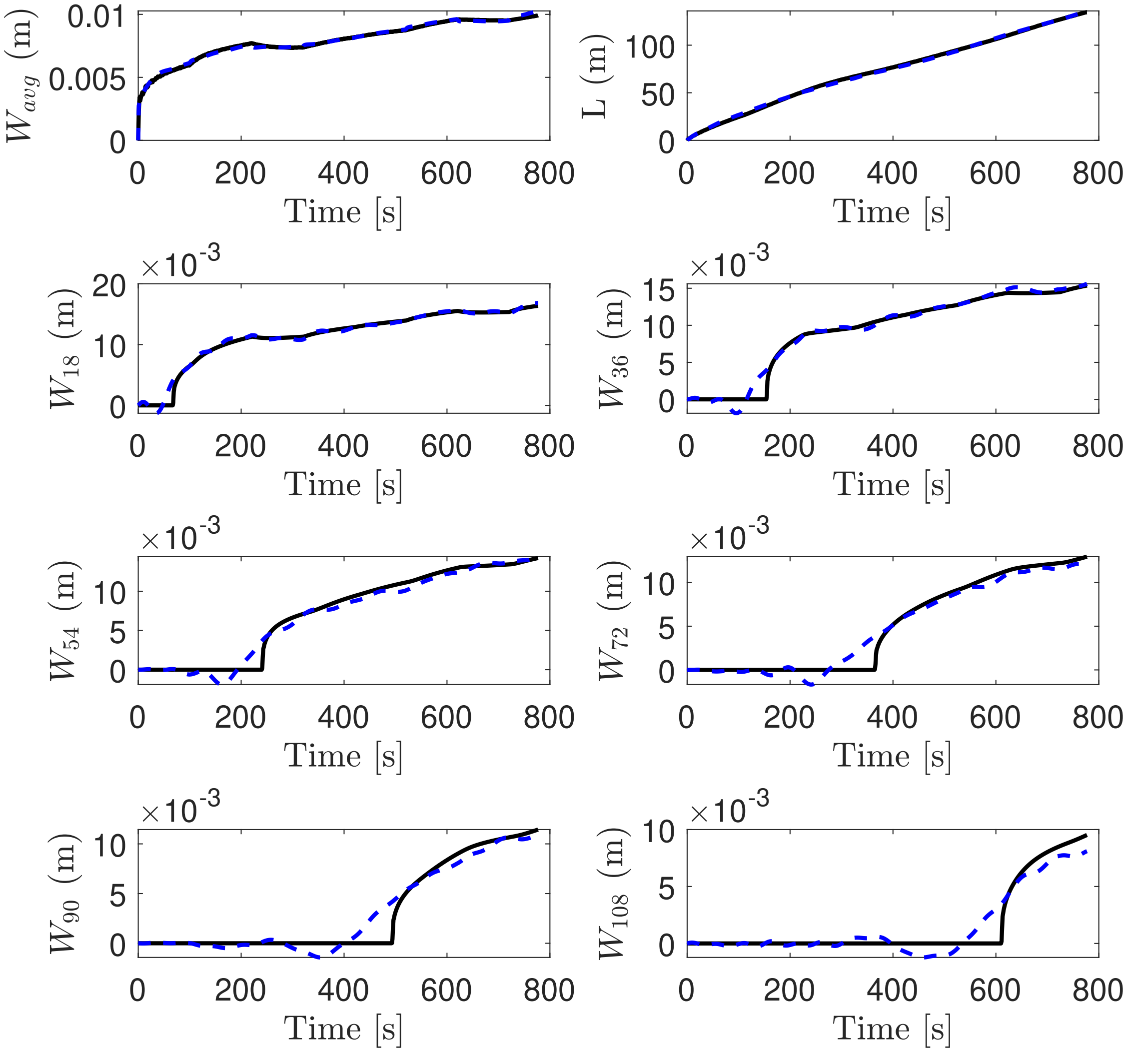


Figure 5.1: The measured output of the system (black) superimposed with the predictions of the Koopman-based model (blue dotted) given the same initial condition and control inputs. Reprinted with permission from [109].

as:

$$\begin{aligned} \text{RMSE} &= \frac{\|x - \hat{x}\|_{fro}}{\|x\|_{fro}} \\ \text{Avg. RMSE} &= \frac{1}{N_{total}} \sum_{i=1}^{N_{total}} \text{RMSE}_i \end{aligned} \tag{5.19}$$

where $\|\cdot\|_{fro}$ is the Frobenius norm. The RMSE details of the validation experiments are presented in Table 5.1. Based on these results, we can say that the Koopman model consistently captures the real behavior of all eight measured states of the system.

Please note that the developed Koopman models are valid for the fixed values of model parameters presented in the manuscript. When the system parameters change, one has to determine a new model in the new parameter space. For example, in the case of a hydraulic fracturing process, when modeling two different reservoirs, the parameters that are subject to change are the rock properties such as Young's modulus, porosity, permeability, etc. Since the underlying flow physics remain the same, the basis functions will be kept constant, and the Koopman matrices will be computed independently for the reservoirs using the data collected from each reservoir.

Case	Max. RMSE	Min. RMSE	Avg. RMSE
Fracture Width	6.82	0.78	3.79
Proppant Concentration	9.77	4.77	7.19

Table 5.1: RMSE values of validation experiments. Reprinted with permission from [109].

5.3.2 Closed-loop results

Now that an accurate model is identified, the Koopman-based MPC presented in Chapter 5.2 was formulated to achieve the desired geometry whose objective is to minimize the squared deviation of the fracture length from its set-point at the end of propagation. To prevent early termination of hydraulic fracturing due to tip-screen out, the average fracture width at the end of the operation must be greater than a pre-specified target value; this is considered as the state constraint. So, the MPC controller solves the following optimization problem of the closed-loop operation at sampling time step k:

$$\begin{aligned}
&\underset{u_k,\cdots,u_{T-1}}{Minimize} && (D\phi_T - L_{target})^2 \\
&\text{s.t} && \phi_{k+j} = A\phi_{k+j-1} + Bu_{k+j-1}, \quad j = 1,\cdots,T-k \\
& && \phi_k = \phi(x_k) \\
& && E\phi_T \leq -W_{target} \\
& && \underline{u} \leq u \leq \overline{u}
\end{aligned} \tag{5.20}$$

where $D \in \mathbb{R}^{1\times N}$ and $E \in \mathbb{R}^{1\times N}$ are matrices that project the lifted state back to the original state space to obtain fracture length and average width, respectively, $\underline{u}$ and $\overline{u}$ denote the lower and upper bounds, respectively, on the manipulated input (i.e., the injection flow rate), L_{target} and W_{target} are the desired fracture length and average width, respectively, k is the current time step, and T denotes the total sampling time steps. The solution of this problem defines a feedback control law $u = u^\star(x_k)$ where only the first value is applied to the closed-loop system for the next sampling time period $t \in [t_k, t_{k+1})$, and the procedure is repeated until the end of operation.

To design the above MPC, the optimal fracture geometry which will maximize the productivity of a stimulated well for a given amount of proppant particles was obtained using the UFD scheme. Specifically, the total mass of proppant to be injected was taken as $48,000$ kg over the entire treatment. For this specified amount, the corresponding optimal fracture length and width determined by UFD are $L_{target} = 135$ m and $W_{target} = 5.4$ mm, respectively. Please refer to [85] for more details on these target values. The control objective is to generate a fracture having a total length equal to L_{target} while keeping the average fracture width greater than the optimal width W_{target} at the end of the treatment.

Since the basis was state-inclusive, the matrices D and E used in the objective function and state constraints are trivial. Specifically, based on Eq. (5.18), $D \in \mathbb{R}^{1\times N} = [0\ 1\ 0_{1\times N-2}]$ and $E \in \mathbb{R}^{1\times N} = [0_{1\times 2}\ 1\ 0_{1\times N-3}]$, where $0_{1\times j}$ is a row matrix whose elements are equal to 0, were used to determine fracture length and average width, respectively. In the closed-loop simulation, the sampling time, Δ, between consecutive measurements was considered to be 100 s and the total process duration to be 800 s; thus, the injection schedule was partitioned into a total of $T = 8$ stages (i.e., $k = 1, \cdots, 8$) with the duration of each step given by the sampling time, Δ. In the beginning of each stage, the state measurements were assumed to be available, which were lifted to the Koopman subspace and used to predict the estimates of the future states via the developed approximate model. The optimization problem was then solved in a shrinking horizon manner (since the control objective was to regulate the output at final time) in this lifted space and the control inputs were computed.

Starting from the initial point, the closed-loop simulation results from $t = 0$ to $t = 800$ s are shown in Figure 5.2. Using Koopman-based MPC controller, the closed-loop trajectory of the fracture length was able to converge to its set-point at the end of the treatment. Additionally, the average fracture width was able to satisfy the state constraint to prevent tip-screen out.

5.4 Learning the EDMD dictionary using sparse regression

In Chapter 5.5, we consider the design of an optimal pumping schedule by considering the coupled dynamics of fracture propagation as well as proppant transport. The high fidelity model is now described by Eqs. (2.1)-(2.11) and is considerably more complex than the case of fracture propagation alone. Due to increased complexity in the dynamics, the choice of basis functions for this case study is not trivial. As a matter of fact, a key challenge in implementing Koopman theory is identifying observables that provide the best finite-dimensional approximation of the linear operator which is the subject of ongoing research. The choice of these observable functions is closely related to the form of nonlinearity in the dynamics. In fact, for the prediction of the developed linear models to be accurate, it should be satisfied that the states of the system are included in the space spanned by the selected observable functions.

Although canonical functions such as Hermite polynomials and radial basis functions are shown to work for simple systems [72], it is critical to identify optimal observables especially for a system such as hydraulic fracturing which is characterized by a large number of state variables compared to the number of measurements. Several numerical experiments (the results of

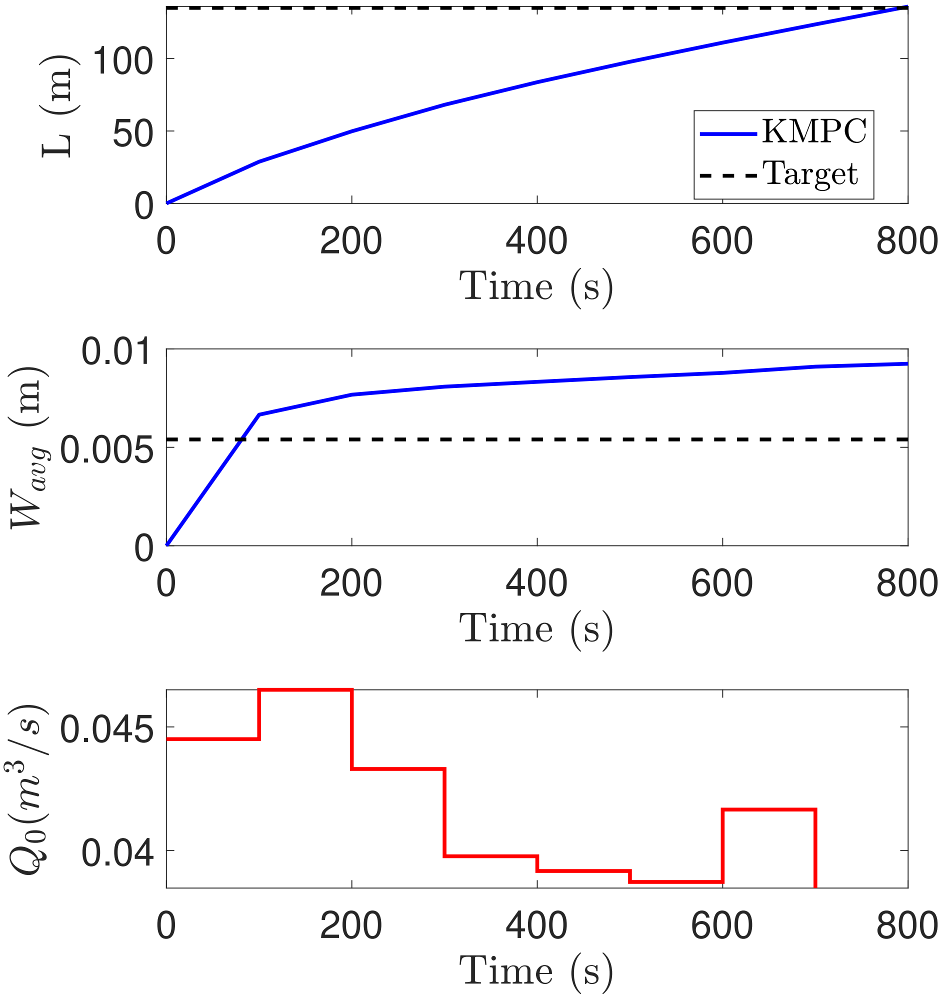


Figure 5.2: Closed-loop trajectories of fracture length, average width and inputs determined by the MPC. Reprinted with permission from [109].

which are not presented here) revealed that using a canonical basis failed to accurately reproduce the concentration dynamics observed here. Specifically, polynomial basis with an order up to 5 and gaussian radial basis functions have been observed to fail when used as Koopman basis functions. In such cases, it is really helpful to incorporate any system knowledge to populate the basis. Motivated by this, we propose a strategy for data-driven identification of the observable functions using sparse regression. Recently, a lot of emphasis is being placed on developing techniques that allow for the identification of physics-based models purely from measurements of the system. In

Chapter 5.4, we explore the idea of using sparse regression techniques to identify the observable functions that span the Koopman subspace.

5.4.1 Sparse Identification

Let us suppose that the time-series data is obtained from the simulation of a high-fidelity process model whose state-space form is given by

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \tag{5.21}$$

where $\mathbf{x}(t) = [x_1(t), \cdots, x_n(t)] \in \mathbb{R}^n$ denotes the vector of state variables at time t, $\mathbf{u}(t) = [u_1(t), \cdots, u_l(t)] \in \mathbb{R}^l$ denotes the vector of inputs applied to system at time t, and the nonlinear vector function $\mathbf{f}(\cdot, \cdot)$ represents the governing equations that define the dynamics of the system. We seek to learn the functional form of $\mathbf{f}$ in the above dynamics by determining an approximate system shown below.

$$\frac{d}{dt}\hat{\mathbf{x}}(t) = \hat{\mathbf{f}}(\hat{\mathbf{x}}(t), \mathbf{u}(t)) \tag{5.22}$$

where $\hat{\mathbf{x}}(t) = [\hat{x}_1(t), \cdots, \hat{x}_d(t)] \in \mathbb{R}^d$ denotes the vector of approximate state variables at time t, and the nonlinear function $\hat{\mathbf{f}}(\cdot, \cdot)$ represents the governing equations that approximate the dynamics of the system. Now, we formulate a regression problem to solve for which functional forms are present in $\hat{\mathbf{f}}$. In this work, we assume that $\hat{\mathbf{f}}$ contains only a few terms and therefore formulate this as a sparse regression problem. This is a reasonable assumption which holds for many physical systems. More specifically, we induce sparsity in the right-hand-side dynamics selection by constructing an over-full library of possible functions. In other words, if we select the over-full library, since the number of relevant terms in $\hat{\mathbf{f}}$ associated with the response is relatively small, while the number of irrelevant ones is large, it makes the model selection problem sparse in the space of potential candidate functions.

To determine the function $\hat{\mathbf{f}}$, the time-series data of $\hat{\mathbf{x}}(t)$ is collected at m time instants $t_1, \cdots, t_m$ and used to construct the matrix $\hat{\mathbf{X}} \in \mathbb{C}^{m \times n}$. The left-hand-side of Eq. (5.22) is formed by computing the derivatives of the state (either numerically using the sampled time-series data or by experimental measurement) and arranged into a matrix $\dot{\hat{\mathbf{X}}} \in \mathbb{C}^{m\times n}$ as follows:

$$\begin{aligned}
\hat{\mathbf{X}} &= \begin{bmatrix} \hat{\mathbf{x}}^T(t_1) \\ \hat{\mathbf{x}}^T(t_2) \\ \vdots \\ \hat{\mathbf{x}}^T(t_m) \end{bmatrix} = \begin{bmatrix} \hat{x}_1(t_1) & \hat{x}_2(t_1) & \cdots & \hat{x}_n(t_1) \\ \hat{x}_1(t_2) & \hat{x}_2(t_2) & \cdots & \hat{x}_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{x}_1(t_m) & \hat{x}_2(t_m) & \cdots & \hat{x}_n(t_m) \end{bmatrix} \\
\dot{\hat{\mathbf{X}}} &= \begin{bmatrix} \dot{\hat{\mathbf{x}}}^T(t_1) \\ \dot{\hat{\mathbf{x}}}^T(t_2) \\ \vdots \\ \dot{\hat{\mathbf{x}}}^T(t_m) \end{bmatrix} = \begin{bmatrix} \dot{\hat{x}}_1(t_1) & \dot{\hat{x}}_2(t_1) & \cdots & \dot{\hat{x}}_n(t_1) \\ \dot{\hat{x}}_1(t_2) & \dot{\hat{x}}_2(t_2) & \cdots & \dot{\hat{x}}_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \dot{\hat{x}}_1(t_m) & \dot{\hat{x}}_2(t_m) & \cdots & \dot{\hat{x}}_n(t_m) \end{bmatrix}
\end{aligned} \tag{5.23}$$

In the above matrices, the columns correspond to the time evolution of states and the rows correspond to the snapshots, respectively. Please note when the state measurements are corrupted by measurement noise, the derivatives can be computed numerically either by filtering the data first or using smoothing functions like Gaussian kernels or polynomial interpolation [155] which can provide more robust differentiation. Proper evaluation of the numerical derivatives is a critical task for the success of the method.

In the next step, a library of p potential candidate functions is constructed using the columns of $\hat{\mathbf{X}}$. This library is not limited to linear terms and may contain several nonlinear functions, which makes the proposed model identification method more suitable for complex nonlinear models. For example, the library which we denoted here by $\mathbf{\Theta}(\hat{\mathbf{X}}) \in \mathbb{C}^{m\times p}$ may consist of linear, polynomial, trigonometric, and other functions as shown below:

$$\mathbf{\Theta}(\hat{\mathbf{X}}) = \begin{bmatrix} \mathbf{1} & \hat{\mathbf{X}} & \hat{\mathbf{X}}^{P_2} & \hat{\mathbf{X}}^{P_3} & \cdots & \sin(a\hat{\mathbf{X}}) & \cos(b\hat{\mathbf{X}}) & \cdots \end{bmatrix} \tag{5.24}$$

Here, $\hat{\mathbf{X}}^{P_2}$ and $\hat{\mathbf{X}}^{P_3}$ denote second- and third-order polynomials in $\hat{\mathbf{x}}(t)$. For example, the quadratic nonlinearities in the state variables are denoted by $\hat{\mathbf{X}}^{P_2}$ and constructed as follows:

$$\hat{\mathbf{X}}^{P_2} = \begin{bmatrix} \hat{x}_1^2(t_1) & \hat{x}_1(t_1)\hat{x}_2(t_1) & \cdots & \hat{x}_2^2(t_1) & \hat{x}_2(t_1)\hat{x}_3(t_1) & \cdots & \hat{x}_n^2(t_1) \\ \hat{x}_1^2(t_2) & \hat{x}_1(t_2)\hat{x}_2(t_2) & \cdots & \hat{x}_2^2(t_2) & \hat{x}_2(t_2)\hat{x}_3(t_2) & \cdots & \hat{x}_n^2(t_2) \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \hat{x}_1^2(t_m) & \hat{x}_1(t_m)\hat{x}_2(t_m) & \cdots & \hat{x}_2^2(t_m) & \hat{x}_2(t_m)\hat{x}_3(t_m) & \cdots & \hat{x}_n^2(t_m) \end{bmatrix} \tag{5.25}$$

Therefore, each column of the library $\mathbf{\Theta}(\hat{\mathbf{X}})$ contains the values of a particular candidate function choice for the right-hand-side of Eq. (5.22) at all time instants. It is clear that there is a high degree of freedom in choosing which candidate functions to be included in the function library $\mathbf{\Theta}$. However, since we assume that only a few terms are required to approximate the dynamics of the original system, this provides us an opportunity to set up the problem as a sparse regression in the space of possible functions. In general, $m \gg p$, i.e., there are more time-series data samples than the number of candidate functions. Thus, we seek a sparse solution to the over determined system. More specifically, the proposed model reduction method can now be formulated as the following regression problem for which we seek the sparsest solution.

$$\dot{\hat{\mathbf{X}}} = \mathbf{\Theta}(\hat{\mathbf{X}})\mathbf{\Sigma} \tag{5.26}$$

Here, $\mathbf{\Sigma} \in \mathbb{R}^{p\times n}$ represents the matrix of regression coefficients that determine which candidate functions are active in the right-hand-side dynamics. Since we seek a sparse solution, the columns of $\mathbf{\Sigma}$ are sparse vectors with a majority of the elements equal to zeros. Once $\mathbf{\Sigma}$ has been determined, a model of the low-dimensional system may be constructed with the respective functional forms used to generate data in the library. Please note that the states have not undergone any linear or nonlinear transformation to determine the approximate dynamics. We emphasize here that $\mathbf{\Theta}$ has a sufficiently rich column space in order for it to represent the dynamics within its range. In other words, enough candidate functions are to be chosen in the library that the ROM can be written as a weighted sum of a few terms. From a practical point of view, one may test many

Algorithm 4 Learning the EDMD dictionary using sparse regression

```
Collect the time-series data and construct the reduced state matrix, $\hat{\mathbf{C}}$
Numerically differentiate $\hat{\mathbf{C}}$ to construct $\dot{\hat{\mathbf{C}}}$
Select the candidate functions and populate the library, $\mathbf{\Theta}$
Solve the least-squares problem $\dot{\hat{\mathbf{C}}} = \mathbf{\Theta}\Sigma$    ▷ this gives $\Sigma$ with all non-zero coefficients
Initialize $\lambda$    ▷ assign desired value to the threshold parameter
for i = 1:iter do
    bigcoeffs = $\{j : |\Sigma[j]| \geq \lambda\}$    ▷ select coefficients greater than threshold
    $\Sigma$[bigcoeffs] = 0    ▷ apply hard threshold
    $\dot{\hat{\mathbf{C}}} = \mathbf{\Theta}\Sigma$[bigcoeffs]    ▷ solve least-squares problem for new non-zero coefficients
Perform cross-validation to determine $\lambda$ in the algorithm    ▷ optional
```

different functions and use the trade-off between complexity and accuracy of the resulting model as a diagnostic tool to determine the correct basis to represent the dynamics.

To determine solutions to the above regression problem, one can adopt any available sparsity promoting algorithms. For example, the LASSO developed in statistics works well in providing a sparse solution to a regression problem with minimum residue. However, when the data matrix has high correlations between columns due to the addition of spatially correlated noise, variable selection may fail to be consistent when using path dependent regression methods like the LASSO. In such cases, an ℓ^2 regularized variation of the least-squares, called the ridge regression [156], can be used to avoid the issue due to correlations [157]. So, depending on the nature of the regression problem and the available time-series data, one can incorporate any regression method to find the sparse solution to determine the required ROM. In our case, we used the ordinary least-squares method with an additional hard thresholding step to force some of the coefficients to zero. More specifically, we used an iterative process where the coefficients with magnitudes smaller than a threshold tolerance, λ, are forced to become zero and the process is repeated till the remaining coefficients converged. Since, each value of the threshold tolerance, λ, will give a different level of sparsity in the final solution, we used cross-validation to find the best tolerance. Algorithm 4 presented below summarizes the proposed method to develop physics-based interpretable ROMs.

Please note that several pre-processing steps may be required to obtain a good solution. For example, depending on the magnitude of the data, if the higher powers are either very large or very small, it will be extremely useful to normalize all the columns of $\boldsymbol{\Theta}$ to unit length before solving for sparse regression coefficients $\boldsymbol{\Sigma}$. Also, if a high measurement error is expected to exist in the system, it may be necessary to perform data filtering for a proper evaluation of the numerical derivatives before solving for $\boldsymbol{\Sigma}$.

5.4.2 Simulation results

We outline the numerical experiments carried out on a nonlinear hydraulic fracturing process model and the simulation results for comparing the proposed method to the available model reduction methods with respect to the model accuracy and future state prediction for designing a model-based feedback controller. All the calculations presented in this work were carried out in Matlab version $8.5.0.197613$ installed on an Intel (R) core (TM) i7-4790 CPU @ 3.60 GHz Dell workstation. The data required for constructing the state matrices was obtained by solving the high-fidelity model. We first collected the time-series data of proppant concentration across the fracture to construct the state, $\hat{\mathbf{X}}$, and the corresponding time derivative, $\dot{\hat{\mathbf{X}}}$, matrices, respectively. The derivatives are computed using implicit finite difference schemes. Note that in this work, we assume the data is clean with no measurement noise added to it. In the presence of noisy data, one can use a more robust numerical differentiation to compute the time derivatives accurately.

The next step is to create a large library of candidate terms that may appear in the approximate model. From the proppant concentration data available from the high-fidelity simulations, we realize that the system closely resembles an empirical first-order plus dead time (FOPDT) model. Because of the simultaneous fracture propagation and proppant transport, there is a time delay in the input variable, i.e., it takes some time for proppant particles injected at the wellbore to reach a particular spatial location. Based on this observation, we constructed the library, $\boldsymbol{\Theta}$, using linear functions of the concentrations at the six locations. Additionally, the time delay in the input variable is incorporated into the model implicitly by constructing six inputs (one for each location) based on the time it takes for the inlet concentration to reach the specific location. This is one of the main advantages of the proposed method because any a priori knowledge of the system can be readily incorporated. Since we consider only linear functions in the library, the total number of columns in $\mathbf{\Theta}$ is $p = 12$. More specifically, the library of the candidate functions is constructed as shown below.

$$\mathbf{\Theta}(\hat{\mathbf{C}}, \mathbf{u}) = \begin{bmatrix} C_1(t_1) & \cdots & C_6(t_1) & C_{0,1}(t_1) & \cdots & C_{0,6}(t_1) \\ C_1(t_2) & \cdots & C_6(t_2) & C_{0,1}(t_2) & \cdots & C_{0,6}(t_2) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ C_1(t_m) & \cdots & C_6(t_m) & C_{0,1}(t_m) & \cdots & C_{0,6}(t_m) \end{bmatrix}_{m\times 12} \tag{5.27}$$

where $C_i(t), i = 1, \cdots, 6$ is the proppant concentration at the six locations considered inside the fracture, and $C_{0,i}(t), i = 1, \cdots, 6$ is the time-delayed input proppant concentration directly acting on the specific location.

Once the library is populated, the final step in the method is to solve a sparse regression problem, Eq. (5.26), to obtain the regression coefficients, $\mathbf{\Sigma}$, that provides a sparse representation of the dynamics.The algorithm converged to the final solution in a small number of iterations. The final solution of the regression problem is presented in Table 5.2.

	C_1	C_2	C_3	C_4	C_5	C_6
C_1	-0.01450	0	0	0	0	0
C_2	-0.00399	-0.00881	0	0	0	0
C_3	0	-0.00458	-0.00623	0	0	0
C_4	0	0	-0.00258	-0.00487	0	0
C_5	0	0	0	-0.00144	-0.00451	0
C_6	0	0	0	0	-0.00180	0.00350
C_{01}	0.02041	0	0	0	0	0
C_{02}	-0.00150	0.01430	0	0	0	0
C_{03}	0	0	0.01205	0	0	0
C_{04}	0	0	-0.00214	0.01091	0	0
C_{05}	0	0	0	-0.00352	0.01075	0
C_{06}	0	0	0	0	-0.00398	0.00855

Table 5.2: Regression coefficients for the ROM of the hydraulic fracturing process. Reprinted with permission from [158].

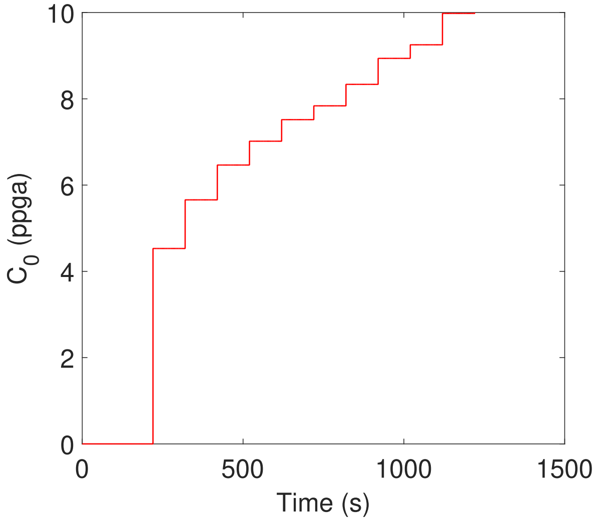
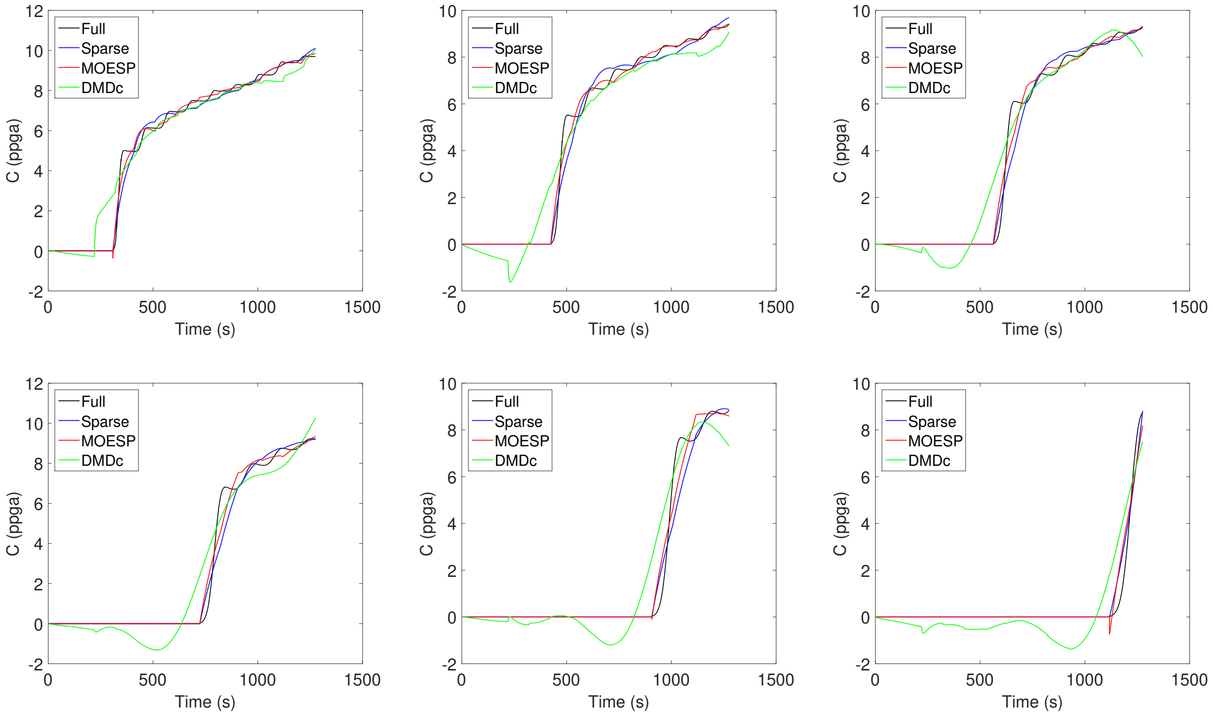


Figure 5.3: Inlet proppant concentration, $C_0(t)$, profile used to train the developed reduced-order models. Reprinted with permission from [158].

We compared the proposed method with other model reduction techniques with respect to model accuracy and future state prediction using a set of training and validation data. To this end, we selected MOESP and DMD techniques. The training input-output data required to develop ROMs has been obtained from the open-loop simulations of the hydraulic fracturing process. The training input profile has been designed to mimic the practically viable inlet proppant concentration profile in the field. Specifically, a simple uniformly increasing staircase-like input profile has been considered to solve the dynamics of the process, which is presented in Figure 5.3. The input-output data is then used to obtain ROMs using both the MOESP and DMD methods. To make a consistent comparison, 6^{th} order models are constructed using both MOESP and DMD methods similar to the order of the ROM developed using the proposed method. Figure 5.4 illustrates this comparison with respect to the proppant concentration profiles at six different locations.

It can be observed from the plot that all the ROMs perform comparably in approximating the solutions to the original high-fidelity model. This observation becomes more apparent in Figure 5.5, which represents the error profiles for all the ROMs. The error is calculated using the ℓ_2-norm of the residuals in the following manner:



(d) z = 72 m (e) z = 90 m (f) z = 108 m

Figure 5.4: Approximate solutions of the proppant concentration profiles showing the comparison between different reduced-order models with the true measurements at six different locations inside the fracture. Reprinted with permission from [158].

$$E(t) = \|\mathbf{x}(t) - \hat{\mathbf{x}}(t)\|_2 \tag{5.28}$$

where $\|.\|_2$ denotes the ℓ_2-norm, $\mathbf{x}(t)$ and $\hat{\mathbf{x}}(t)$ are state vectors obtained from the high-order discretization of the PDE and the reduced-order model, respectively.

To validate the performance of the generated ROMs, approximate solutions were computed using open-loop simulations in response to a different input profile. A simple step input profile shown in Figure 5.6, is given to the system as the validation input. The results of the model validation using all the different methods is shown in Figure 5.7. As is evident from the plots, the approximate solution generated using the proposed ROM is more accurate and closer to the original solution compared to DMD method. However, the performance of the proposed method and MOESP are comparable. This is also shown clearly in Figure 5.8, which represents the approximation error

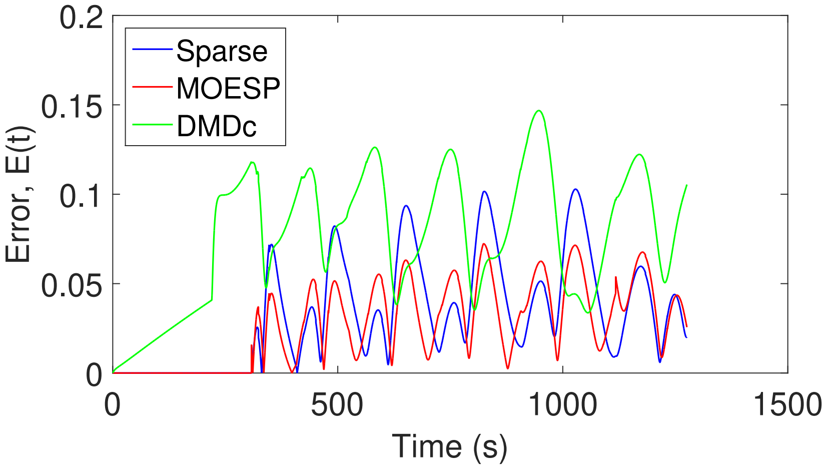


Figure 5.5: Profile of the ℓ_2 norm error for training data. Reprinted with permission from [158].

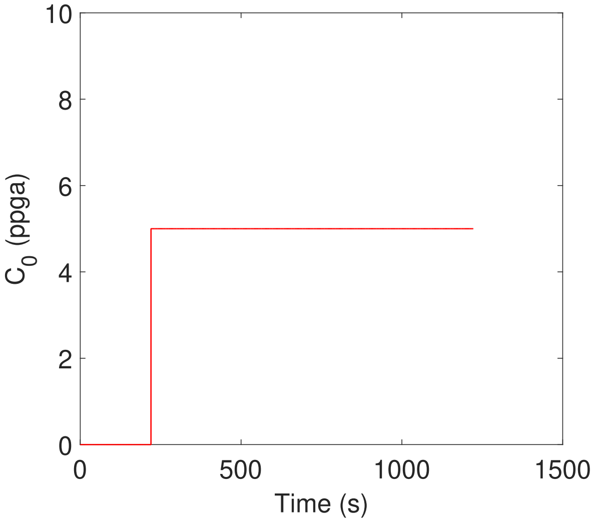
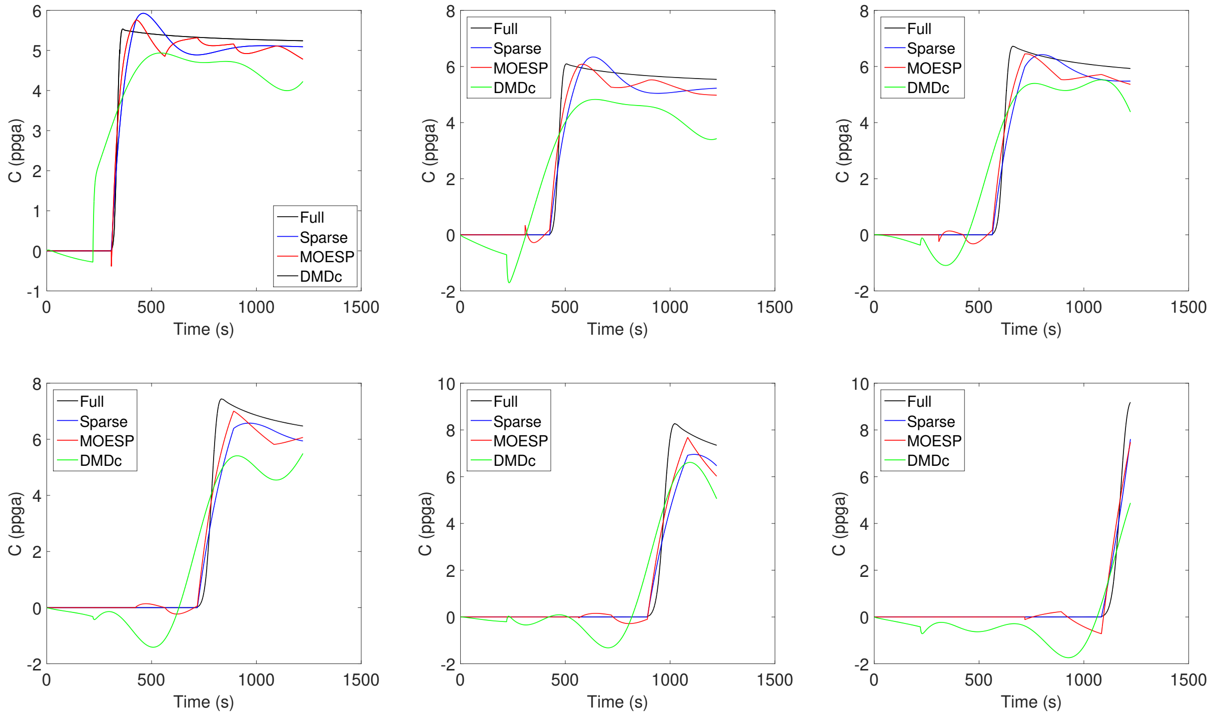


Figure 5.6: Inlet proppant concentration, $C_0(t)$, profile used to validate the developed reduced-order models. Reprinted with permission from [158].

computed for the validation input.

Although there is no significant difference between the two methods in terms of model accuracy and future state prediction, the proposed method has an added benefit of determining physics-based and interpretable functions that provide valuable insights into the underlying physical phenomena



(d) z = 72 m (e) z = 90 m (f) z = 108 m

Figure 5.7: Approximate solutions of the proppant concentration profiles for the testing data set showing the comparison between different reduced-order models with the true measurements at six different locations inside the fracture. Reprinted with permission from [158].

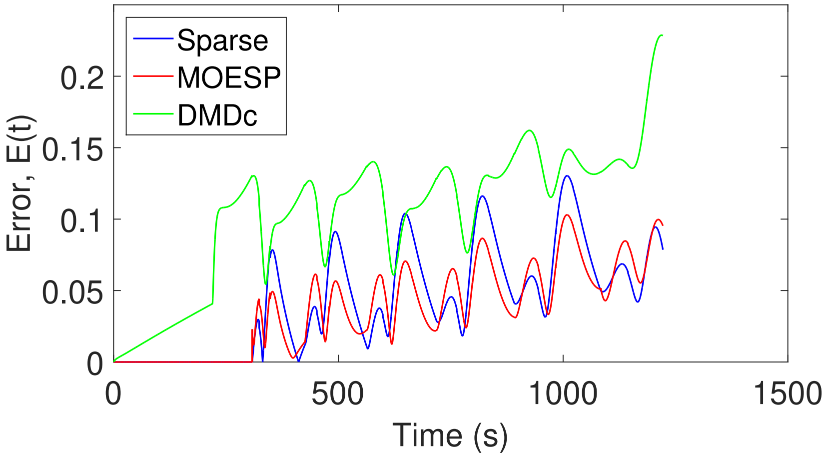


Figure 5.8: Profile of the ℓ_2 norm error for testing data. Reprinted with permission from [158].

of the system. As can be seen from the generated ROM shown in Table 5.2, the proppant concentration at a spatial location inside the fracture is a function of its neighboring locations. This is an accurate representation because in a hydraulic fracturing process, due to the simultaneous fracture propagation and fluid leak-off into the surrounding reservoir, the concentration at a location depends on the adjacent locations. Let us suppose the fracture has propagated to a spatial node n. As the proppant is transported to that location, its concentration increases with time. Additionally, a further increase in concentration is also observed due to the fluid leaking off into the reservoir. However, due to simultaneous fracture propagation, a new spatial location, say $n + 1$ becomes available which facilitates the transport of the proppant away from the location n. Therefore, the concentration at n is slightly decreased or remains constant depending on the magnitude of the input. This phenomenon can be clearly seen in the generated ROM where the states (concentration) at a spatial location are direct functions of the neighboring states. Because the proposed method does not transform the states in any manner, it results in a ROM that can provide physical interpretations and insights into some of the significant underlying phenomena. Therefore, by reformulating the learning problem as a sparse regression problem, we were able to determine the functional form of the dynamics. These functions can now be considered as observables to construct the Koopman basis functions.

5.5 Regulating spatial proppant concentration distribution

In Chapter 5.4, we developed a dictionary learning algorithm by reformulating the problem as a sparse regression problem. Based on these results, the Koopman basis was chosen using linear functions of the concentrations at the six locations. Therefore, the total number of functions in the basis $\{\phi_i\}_{i=1}^{N}$ is equal to $N = 6$. Note that this is again a *state-inclusive* basis.

$$\begin{aligned} &\phi_i(x) \in \left\{ \prod_{j=1}^{6} x_j^{m_j} \mid (m_1, \cdots, m_6) \in \mathbb{Z}_0^+, \sum_{j=1}^{6} m_j = 1 \right\} \\ \implies &\phi(x) = [C_1 \;\; C_2 \;\; \cdots \;\; C_6]^T \end{aligned} \tag{5.29}$$

However, the complexity of this system actually lies in the way that the input is handled, not in the states themselves. If a simple linear basis was used without proper handling of the input, the obtained result failed to accurately represent the system. Since it takes different times for the inlet concentration to reach different locations inside the fracture, this time delay in the input variable was incorporated into the model implicitly by constructing 6 auxiliary inputs (i.e., one for each location) based on the time it takes for the inlet concentration to reach the specific location, $u \in \mathbb{R}^6$. The numerical values for input time-delays were determined from the simulation data.

The validation results of the identified Koopman linear model are shown in Figure 5.9. It can be seen that the Koopman model was able to identify the measured concentration dynamics fairly accurately. This was further verified from the low Avg. RMSE value of 7.19% when 100 trajectories were validated with different inputs. Please refer to Table 5.1 for more details.

5.5.1 Closed-loop results

Based on the optimal length and width computed in the previous example in Chapter 5.3, a target concentration can be computed as follows:

$$C_{target} = \frac{M_{prop}}{HL_{target}W_{target}} = 9.5 \text{ ppga} \tag{5.30}$$

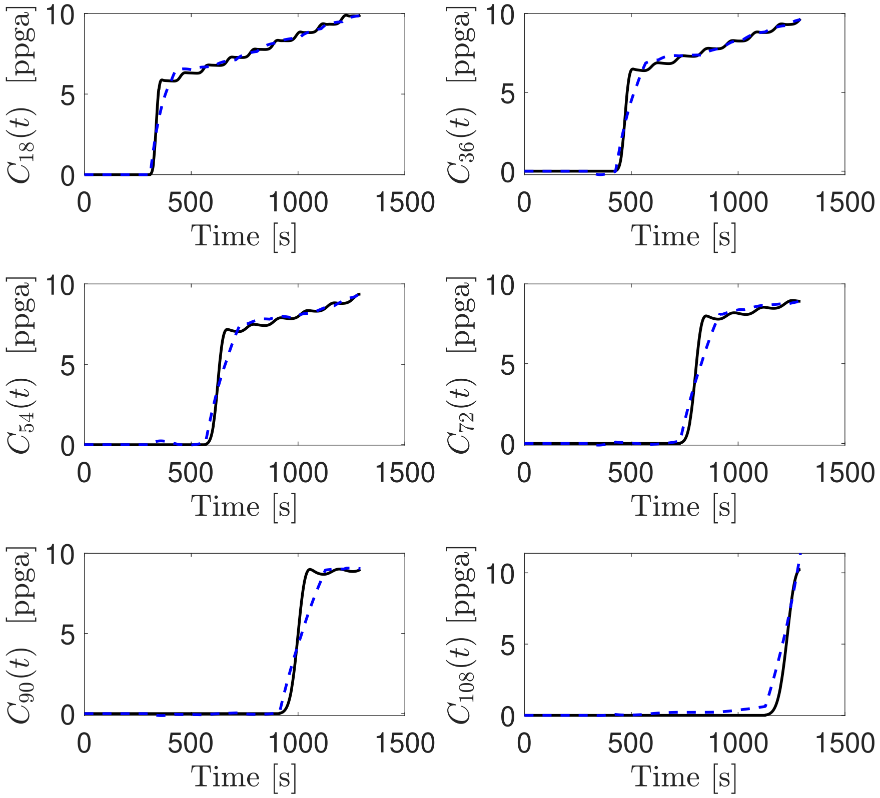


Figure 5.9: The measured concentration output of the system (black) superimposed with the predictions of the Koopman-based model (blue dotted) given the same initial condition and control inputs. Reprinted with permission from [109].

The control objective is to achieve the above concentration C_{target} uniformly throughout the fracture. There are several constraints associated with the problem. The input profile should increase monotonically to follow the practical implementation in the field ($C_{0,k} \leq C_{0,k+1}$). A material constraint given by $2Q_0\Delta \sum C_{0,k} = M_{prop}$ is also required to ensure the required amount of proppant is injected into the fracture. All these constraints are introduced into the MPC problem as linear constraints in a similar manner as described in Chapter 5.2. The following optimization problem was then solved over the prediction horizon, and the first step of the solution was applied to the high-fidelity model in a sample-and-hold fashion, and the procedure repeated at every sampling time until the end of treatment.

$$
\begin{aligned}
\underset{u_k,\cdots,u_{T-k}}{Minimize} \quad & (D\phi_T - C_{target})^T Q (D\phi_T - C_{target}) \\
\text{s.t} \quad & \phi_{k+j} = A\phi_{k+j-1} + Bu_{k+j-1}, \quad j = 1,\cdots,T-k \\
& \phi_k = \phi(x_k) \\
& G_u u \leq 0 \\
& G_m u = M_{prop} \\
& \underline{u} \leq u \leq \overline{u}
\end{aligned}
\tag{5.31}
$$

where D is the identity matrix in this example. The cost matrix Q was taken to be identity. The input constraint matrix $G_u \in \mathbb{R}^{T-k \times T-k}$ is an upper bi-diagonal matrix with $G_u(j,j) = 1, G_u(j,j+1) = -1 \; \forall j = 1,\cdots,T-k$. Essentially, this constraint specifies that the injected concentration at a time step $k+1$ is greater than or equal to that of k, i.e., an increasing input profile. This constraint was used to obtain a practical pumping schedule, one that closely resembles the field. The material constraint vector G_m is given by $G_m \in \mathbb{R}^{1 \times T} = 2Q_0\Delta * 1_{1\times T}$ where $1_{1\times j}$ denotes a row vector whose elements are equal to 1. This constraint ensures that the total mass of injected proppant, given by the cumulative input concentration, is equal to the pre-specified value of M_{prop}.

The results of the Koopman-based MPC are presented in Figure 5.10. From the figure it can be seen that the derived controller was able to regulate the final proppant concentration to the pre-specified target value throughout the fracture. The pumping schedule (i.e., the input concentration profile required to achieve the desired proppant concentration) for the corresponding process parameters over the entire operation time is shown in Figure 5.10b. Please note that the developed approximate models considered only measurements at 6 (out of 451) locations; in the control problem, we do not necessarily require knowledge of the entire system state but only of some observations which are used in the controller. We see that even by using partial observations, the linear model was able to control the PDE fairly accurately. Moreover, the dimension of the optimization problem is reduced because instead of using a high-dimensional discretization and a higher order time integrator, we use a matrix vector product to predict the dynamics of the observed states.

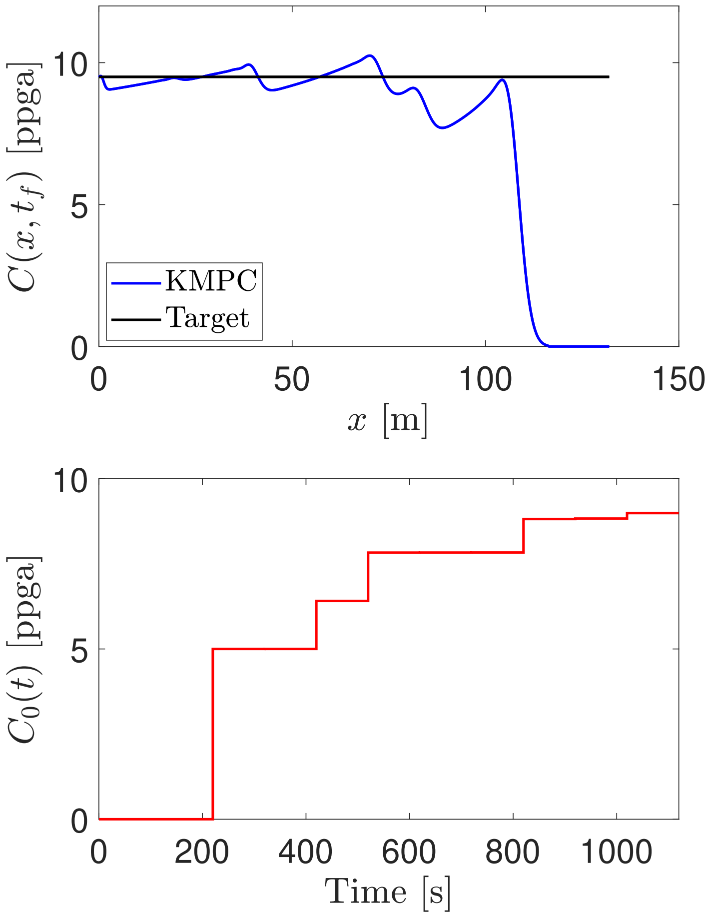


Figure 5.10: Final proppant concentration and input trajectories of the closed-loop process determined by the Koopman MPC. Reprinted with permission from [109].

5.6 Conclusions

In Chapter 5, we successfully applied the Koopman operator theory for system identification and feedback control of a hydraulic fracturing process. We studied two cases: (1) fracture propagation and (2) proppant transport. The results showed that in both the examples presented, the generated linear models were able to accurately predict the evolution of all the observed states and resulted in a very small Avg. RMSE value (3.79% and 7.19%) over 100 trajectories with different inputs. One major difference in the examples is in the choice of the Koopman basis functions. In the case of fracture propagation, since the dynamics were relatively simple, a canonical basis like the bilinear basis functions was able to accurately capture the observed dynamics. However, the case of proppant concentration is more sophisticated where the dynamics are highly coupled and nonlinear. In such cases, it is observed that a priori process knowledge can be leveraged to select the required basis of observables. Specifically, sparse regression based dictionary identification was used as a precursor to obtain the most relevant functions in the basis.

The obtained Koopman models for both the cases were then used to design feedback controllers to regulate the fracture geometry and proppant concentration. Due to the superior predictive capabilities of the linear models, the derived controllers achieved the final desired fracture length while satisfying the constraint on average fracture width and uniform proppant concentration at the end of the fracturing treatment. Moreover, because of its linear structure, the Koopman model resulted in a convex quadratic MPC problem that is amenable to be solved using any of the available linear MPC solvers. Although future work in this direction will need to certify these controllers, providing guarantees on stability, and a rigorous quantification of error and uncertainty bounds, the current work shows the potential of the operator-theoretic framework for approximation and control of chemical and petroleum engineering processes.

6. KOOPMAN LYAPUNOV-BASED MODEL PREDICTIVE CONTROL OF NONLINEAR CHEMICAL PROCESS SYSTEMS[*]

In Chapter 5, we introduced a new operator theoretic viewpoint of dynamical systems and obtaining linear representations for nonlinear systems. By virtue of linearity, Koopman operator facilitates the use of existing linear control methodologies on nonlinear systems and promotes convex optimization based solutions for predictive control formulations with the computational complexity comparable to that of linear systems (with the same number of inputs and states). However, in Chapter 5 as well as in several recent research works [10, 11, 12, 13], the tightness of the linear predictors is not established and the closed-loop guarantees on stability are not well studied. This is critical because, in practice, EDMD models are known to have closure issues as there is no guarantee that the selected observables form Koopman invariant subspaces [72, 147, 159]. This is especially true for the systems with multiple steady states (as a linear system can only characterize a single steady state). This may lead to spurious dynamics which may subsequently lead to unstable controllers.

Motivated by these considerations, in Chapter 6, we propose a systematic approach for the design of a stabilizing feedback predictive controller for nonlinear systems. To this end, we propose to integrate Koopman based linear predictors with Lyapunov-based model predictive control (LMPC) scheme [160]. LMPC is known for its explicit characterization of stability properties and guaranteed closed-loop stabilization of nonlinear systems in the presence of state and input constraints [161]. LMPC is a powerful tool that combines the complementary properties of the CLF (stability) and predictive control (optimality) approaches. Recently, researchers have also successfully designed LMPC that guarantees closed-loop stability even when (linear) empirical models are used in the design as long as a set of assumptions is satisfied [162]. Consequently, in this work, we integrate Koopman linear predictors with LMPC for the stable control of nonlinear dynamical

---

[*] Reprinted with permission from "Koopman Lyapunov-based model predictive control of nonlinear chemical process systems," by A. Narasingam, and J. S. Kwon, AIChE Journal, 2017, 63, e16743. Copyright 2019 by John Wiley and Sons.

systems. Specifically, we perform a lifting (nonlinear transformation) of the system states to an observable space using nonlinear functions and obtain linear representations of the system in the observable space. We then design LMPC based controllers in the same observable space that stabilize the nonlinear system. The proposed method introduces an attractive modification that exploits the construction of linear predictors and the structure of LMPC to yield a completely standard convex (quadratic) optimization problem (provided the state and input constraints are linear) within the LMPC framework. Therefore, the large library of efficient solvers available for linear MPC can be readily used to solve the proposed Koopman based LMPC scheme for the control of nonlinear systems.

6.1 Feedback control design

The methodology presented in Chapter 5.1.2 allows us to construct accurate linear predictors of the dynamical system in the form of Eq. (5.6). Here, we will utilize these predictors to design a model based control scheme to control the original nonlinear system. To this end, the LMPC is a powerful tool for the design of a stabilizing feedback controller which is also optimal with respect to the state and input constraints. This is particularly attractive in this context because the linear predictors obtained using regression methods for approximating the Koopman operator are known to have closure issues which may lead to unstable controllers. More specifically, the premise of EDMD is that if the set of observable functions is "rich enough" we can then reconstruct the original system states from these observables. However, since the dictionary depends on our choice of variables, in most applications, it is possible to miss some elements and therefore the Koopman operator cannot be fully represented in the chosen dictionary. As a result, the subspace spanned by the dictionary of observable functions is not guaranteed to be forward invariant, which may lead to closure issues [72, 147, 159]. Moreover, as described in Chapter 5.1.2, we numerically compute a projection of the Koopman operator K onto the finite dimensional subspace rather than the infinite dimensional Koopman operator, $\mathcal{K}$, itself. Therefore, the tightness of these linear predictors is not guaranteed and this may lead to a closed-loop design that is unstable. To address this issue, in this work, we use the Koopman based linear predictors in an LMPC framework which allows for an explicit characterization of the stability region and provides guarantees on controller feasibility and closed-loop stability.

6.1.1 Lyapunov-based model predictive control

In Chapter 6.1.1, we briefly describe the LMPC design proposed in [160]. We recall that LMPC is a control strategy that is designed based on an explicit stable (albeit not optimal) control law $\mathbf{h}(\mathbf{x})$ and a Lyapunov constraint by virtue of which the controller is able to stabilize the closed-loop system. Before we go into the formulation, the LMPC framework requires certain assumptions that need to be satisfied by the nonlinear dynamical system considered in Eq. (5.1).

Assumption 1. *The dynamics* $\mathbf{f}$ *that govern the evolution of system states is assumed to be locally Lipschitz on* $\mathbb{R}^n \times \mathbb{R}^m$ *such that* $\mathbf{f}(0,0) = 0$*. This implies that the origin is an equilibrium point for the nominal system. Note that this assumption on* $\mathbf{f}$ *is reasonable which holds true for many chemical processes.*

Assumption 2. *The control input space* $\mathcal{U}$ *is restricted to be nonempty and convex.*

Assumption 3. ***Stabilizability assumption****. The nonlinear systems considered are restricted to a class of stabilizable systems which implies the existence of a feedback control law* $\mathbf{u}(t) = \mathbf{h}(\mathbf{x})$ *that satisfies input constraints for all* $\mathbf{x}(t)$ *inside a given stability region and renders the origin of the closed-loop system asymptotically stable. This is equivalent to assuming that there exists a Lyapunov function for the nominal system (Lyapunov theorem).*

Based on the above assumptions, the predictive control of the system of Eq. (5.1) under LMPC is formulated as follows:

$$\min_{\mathbf{u}\in\mathcal{S}(\Delta)} \quad \int_{t_k}^{t_{k+N_p}} [\|\mathbf{x}(\tau)\|_W + \|\mathbf{u}(\tau)\|_R] d\tau, \tag{6.1a}$$

$$\text{s.t} \quad \dot{\hat{\mathbf{x}}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \tag{6.1b}$$

$$\hat{\mathbf{x}}(t_k) = \mathbf{x}(t_k) \tag{6.1c}$$

$$V(\mathbf{x}(t)) \leq r \tag{6.1d}$$

$$\frac{\partial V(\mathbf{x}(t_k))}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}(t_k), \mathbf{u}(t_k)) \leq \frac{\partial V(\mathbf{x}(t_k))}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}(t_k), \mathbf{h}(\mathbf{x}(t_k))) \tag{6.1e}$$

where $\mathcal{S}(\Delta)$ is the family of piece-wise constant functions with sampling period Δ, N_p is the prediction horizon, $W \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{m\times m}$ are positive definite weighting matrices, $\hat{\mathbf{x}}$ is the predicted state trajectory, with initial (measured) state $\mathbf{x}(t_k)$, due to the application of the control input $\mathbf{u}$, and $V(\mathbf{x})$ is the Lyapunov function associated with the explicit control law $\mathbf{h}(\mathbf{x})$. The manipulated input (optimal solution) of the above system under the LMPC control law is defined as

$$\mathbf{u}(t) = \mathbf{u}^\star(t|t_k), \qquad \forall t \in [t_k, t_{k+N_p}) \tag{6.2}$$

where $\mathbf{u}^\star(t|t_k) = [\mathbf{u}^\star(t_k), \cdots, \mathbf{u}^\star(t_{k+N_p})]$. The first value of $\mathbf{u}^\star(t|t_k)$ is applied to the closed-loop system for the next sampling time period $t \in [t_k, t_{k+1})$ and the procedure is repeated until the end of operation.

In the LMPC formulation of Eqs. (6.1a)- (6.1c), Eq. (6.1a) denotes a performance index that is to be minimized, Eq. (6.1b) is the nominal model of the nonlinear system used to predict the future evolution of the states, and Eq. (6.1c) provides the initial state which is obtained as a measurement of the actual system state. In addition to these constraints, the LMPC formulation considers the Lyapunov constraints, Eq. (6.1d) and (6.1e). Eq. (6.1d) ensures that the closed-loop system stays within the stability region $\Omega_r = \{\mathbf{x} \in \mathbb{R}^n : V(\mathbf{x}) \leq r\}$ and Eq. (6.1e) guarantees that the rate of change of the Lyapunov function, $V(\mathbf{x})$, at time t_k is smaller than or equal to that of the value obtained if the explicit control law $\mathbf{h}(\mathbf{x})$ is applied to the closed-loop system in a sample-and-hold fashion. These constraints allow the LMPC controller to inherit the stability properties, i.e., it possesses the same stability region Ω_r as the controller $\mathbf{h}(\mathbf{x})$. This implies that the (equilibrium point of) closed-loop system is guaranteed to be stable for any initial state inside the region Ω_r provided the sampling time Δ is sufficiently small. Note that because of this property, the LMPC does not require the terminal constraint generally used in a traditional MPC setting. Additionally, the feasibility of LMPC is also guaranteed because $\mathbf{u} = \mathbf{h}(\mathbf{x})$ is always a feasible solution to the above optimization problem. Even though the above formulation does not explicitly consider the state and input constraints, they can be readily incorporated.

6.1.2 Integrating EDMD with LMPC

In Chapter 6.1.2, we show how the linear predictors based on EDMD can be embedded into the most general LMPC formulation described above. Additionally, we propose a simple modification which will result in a standard convex quadratic optimization problem. This attractive feature has already been alluded to in Chapter 5, specifically via Eq. (5.16)

In order to use the Koopman based model in the LMPC framework, the problem must be formulated in the observable (lifted) space. Therefore, at each time step t_k the predictions of the system trajectory are initialized from the lifted state $\mathbf{z}_k = \boldsymbol{\phi}(\mathbf{x}_k)$. Similarly, the objective function and the state constraints are all transformed to the lifted space. In addition, we propose to include the Lyapunov function in the dictionary of nonlinear observable functions by setting for some $j, \phi_j = V(\mathbf{x})$. By including the Lyapunov function as one of the observable functions, we can effectively transform the nonlinear Lyapunov constraint in Eq. (6.1d) to a linear one. For example, if we set $\phi_1 = V(\mathbf{x})$ we can simply define a vector $\mathbf{D} = [1, 0, \cdots, 0]^T \in \mathbb{R}^N$ such that $\mathbf{D}\mathbf{z}_k = V(\mathbf{x}_k)$. Now, writing the Lyapunov constraint of Eq. (6.1d) in discrete formulation, we have

$$V(\mathbf{F}(\mathbf{x}_k, \mathbf{u}_k)) - V(\mathbf{x}_k) \leq V(\mathbf{F}(\mathbf{x}_k, \mathbf{h}(\mathbf{x}_k))) - V(\mathbf{x}_k) \tag{6.3}$$

Substituting $V(\mathbf{x}_k) = \mathbf{D}\mathbf{z}_k$ into the above equation, we get

$$\mathbf{D}\mathbf{z}_{k+1} - \mathbf{D}\mathbf{z}_k \leq V(\mathbf{F}(\mathbf{x}_k, \mathbf{h}(\mathbf{x}_k))) - V(\mathbf{x}_k) \tag{6.4}$$

We exploit the linear structure of the nonlinear system in the observable space to transform the nonlinear LMPC formulation to a standard convex quadratic optimization problem. Intuitively, we can see that a Lyapunov function is indeed a particular observable function, the decrease of which dictates the stability of a nonlinear system. The motivation behind this comes from the fact that operator-theoretic approaches have been successfully applied (albeit implicitly) to nonlinear systems for global stability analysis and control [8]. Particularly, Lyapunov's stability criterion relies on the operator-theoretic framework which uses a (Lyapunov) function description in the infinite dimensional function space to analyze the stability of a system, rather than a point wise description that traditionally studies stability with respect to equilibrium points in the finite-dimensional state space.

Based on the above modifications, the proposed Koopman-LMPC framework, in discrete formulation, solves the following optimization problem at each time step k of the closed-loop operation.

$$\min_{\mathbf{u}_i} \quad \sum_{i=0}^{N_p} \Big((\mathbf{C}\mathbf{z_i})^T W (\mathbf{C}\mathbf{z_i}) + \mathbf{u_i}^T R \mathbf{u_i} \Big) \tag{6.5a}$$

$$\text{s.t} \quad \mathbf{z}_{i+1} = \mathbf{A}\mathbf{z}_i + \mathbf{B}\mathbf{u}_i, \qquad i = 0, \cdots, N_p - 1 \tag{6.5b}$$

$$\mathbf{z}_0 = \boldsymbol{\phi}(\mathbf{x}_k) \tag{6.5c}$$

$$\mathbf{E}_i\mathbf{z}_i + \mathbf{H}_i\mathbf{u}_i \leq \mathbf{b}_i, \qquad i = 0, \cdots, N_p - 1 \tag{6.5d}$$

$$\mathbf{D}\mathbf{z}_i \leq r, \qquad i = 0, \cdots, N_p - 1 \tag{6.5e}$$

$$\mathbf{D}\mathbf{z}_1 - \mathbf{D}\mathbf{z}_0 \leq V(\mathbf{F}(\mathbf{x}_k, \mathbf{h}(\mathbf{x}_k))) - V(\mathbf{x}_k) \tag{6.5f}$$

In the above formulation, Eq. (6.5a) represents the quadratic cost function which has been transformed to the observable state-space, Eq. (6.5b) denotes the linear predictors based on EDMD that describe the evolution of the lifted states with matrices $\mathbf{A}$ and $\mathbf{B}$ constructed as described in Chapter 5.1.2, Eq. (6.5c) determines the initialization $\mathbf{z}_0$ using the observable functions mapping $\boldsymbol{\phi}$, Eq. (6.5d) gives the state and input polyhedral constraints using the matrices $\mathbf{E}_i \in \mathbb{R}^{n_c \times N}, \mathbf{H}_i \in \mathbb{R}^{n_c \times m}$ and the vector $\mathbf{b}_i \in \mathbb{R}^n_c$, and Eqs. (6.5e) and (6.5f) correspond to the Lyapunov constraints (Eqs. (6.1d) and (6.1e)) which guarantee the closed-loop stability of the system. The above optimization problem of Eq. (6.5) is parameterized by the current state $\mathbf{x}_k$ of the nonlinear system (see Eqs. (6.5c) and (6.5f)). Please note that Eq. (6.5f), although appears nonlinear, is still a linear constraint because it requires computation of the Lyapunov function derivative of the system under the nonlinear control law at the current state value alone, which is available from the measurement. The above optimization problem defines a feedback controller $\mathbf{u}_k = \mathbf{u}_0^\star$ where $\mathbf{u}_0^\star$ denotes the first component of the optimal solution to Eq. (6.5) parameterized by the current state $\mathbf{x}_k$. This problem is solved at each time step k of the closed-loop operation.

Several important features of the proposed method are summarized below:

1. The optimization problem Eq. (6.5) is a convex quadratic programming problem even when the original dynamic model is nonlinear. Therefore, it avoids solving difficult non-convex optimization problems and allows for a fast evaluation of the control input.

2. The evaluated controller possesses the same stability region, Ω_r, as that of the nonlinear control law $\mathbf{h}(\mathbf{x})$, thereby mitigating the closure issues that otherwise plague the linear predictors developed using the EDMD algorithm that solves a regression problem.

3. As previously shown in Chapter 5, if nonlinear state constraints are present in the system, they can be absorbed into the observable function dictionary $\boldsymbol{\phi}$. For example, if the state constraints are given as $\mathbf{c}_i(\mathbf{x}_i) \leq 0, i = 1, \cdots, N_p$ where $\mathbf{c}_i$ are scalar-valued, by simply considering $\boldsymbol{\phi} = [\mathbf{x}, V(\mathbf{x}), \mathbf{c}_i, \cdots, \phi_{N-n-1-n_c}]$ we can obtain the matrices $\mathbf{E}_i = [0_{1\times n+1}, 1, 0_{N-n-1-i}]$ such that the nonlinear state constraints can be rendered linear. This is also true if the objective function is defined by nonlinear functions (other than quadratic costs) such as those prevalent in economic MPC design [162].

6.2 Application to chemical process example

In Chapter 6.2, we illustrate the proposed method on a well mixed, non isothermal continuously stirred tank reactor (CSTR) example. We consider an irreversible exothermic reaction, $A \rightarrow B$, where the conversion of reactant A to product B is governed by second-order kinetics. The feed to the reactor is pure A at a constant flow rate F, inlet temperature T_{A0} and molar concentration of C_{A0}. Since the process is non isothermal, heat is either removed or provided to the system through the reactor jacket. The dynamic model of the system can be easily obtained by applying mass and energy balances as given below:

$$\begin{aligned}\frac{dC_A}{dt} &= \frac{F}{V_r}(C_{A0} - C_A) - k_0 e^{\frac{-E}{RT_r}} C_A^2 \\ \frac{dT_r}{dt} &= \frac{F}{V_r}(T_{A0} - T_r) - \frac{\Delta H}{\rho C_p} k_0 e^{\frac{-E}{RT_r}} C_A^2 + \frac{Q}{\rho C_p V_r}\end{aligned} \tag{6.6}$$

where C_A and T_r are the concentration of reactant A and the reactor temperature, respectively (system states), Q is the rate of heat input to the reactor (manipulated input), V_r is the volume of the reactor, $k_0, E, \Delta H$ denote the pre-exponential factor, activation energy and the enthalpy of the reaction, respectively, and ρ, C_p denote the density of the fluid and its heat capacity, respectively.

The above system is characterized by three steady states (two asymptotically stable and one unstable) for $Q_s = 0$ KJ/h which is the steady-state value of the manipulated input. The control objective is to operate the CSTR in a compact state-space around the steady-state given by $[C_{As}, T_{rs}] = [1.22\ kmol/m^3, 438.2\ K]$. The given steady-state is open-loop stable. The reason for choosing this steady-state for operation is that it has been shown to maximize the time-averaged production rate of product B [162]. Throughout the rest of Chapter 6.2, the dynamic model in Eq. (6.6) is used in its deviation form that defines the state and input vectors as $\mathrm{x} = [C_A - C_{As},\ T_r - T_{rs}]^T$ and $u = [Q - Q_s]$, respectively. Therefore, the origin is a stable equilibrium point for the new system.

Property	**Unit**	**Value**
Inlet temperature, T_0	K	300
Volume, V	m^3	1.0
Pre-exponential factor, k_0	$m^3/hr/kmol$	8.46×10^6
Pure fluid density, ρ	kg/m^3	1000
Heat capacity, C_p	$kJ/kg/K$	0.231
Flow rate, F	m^3/hr	5.0
Activation energy, E	$kJ/kmol$	5.0×10^4
Enthalpy of reaction, ΔH	$kJ/kmol$	-1.15×10^4
Gas constant, R	$kJ/kmol/K$	8.314

Table 6.1: Parameters of the CSTR process. Reprinted with permission from [152].

6.2.1 Model identification and validation

To demonstrate the application of EDMD, we assume that the nonlinear dynamic model for the CSTR, Eq. (6.6), is not available and a linear time invariant state-space model (in the observable space) will be identified and validated using the algorithm described in the Chapter 5.1.2. In order to obtain the predictors, we first collect the data by solving the set of ODEs using the Matlab solver `ode45` with an integration time step of $h = 1 \times 10^{-4}$ hr. The values of the process parameters used in the simulations are presented in Table 6.1. Since the goal is to design a controller, we used closed-loop simulation data to build the linear predictors. The data required to construct these predictors using the EDMD algorithm is obtained from 250 simulated trajectories over an operating period of 1 hr. A total of 1000 time samples per trajectory are used to populate the required data matrices. Each trajectory starts with an initial condition generated randomly over an interval around the operating steady-state, and is subjected to an input signal determined by a feedback controller applied to the process. The manipulated inputs are bounded as $-5 \times 10^5 \leq Q \leq 5 \times 10^5$ kJ/hr. We assume that the output is equal to the state (i.e., a state feedback problem). This results in the data matrices $\mathbf{X}$ and $\mathbf{Y}$ of size $2 \times 25 \cdot 10^4$ and the matrix $\mathbf{U}$ of size $1 \times 25 \cdot 10^4$.

For unknown systems, there is no established way to select the dictionary although some canonical choices like radial basis functions have been proposed as a good starting point. However, in this example, we wanted to show the flexibility of the method where any available knowledge of the system can be easily incorporated into the dictionary. Specifically, the observable functions ϕ_i are chosen as the state itself, quadratic and exponential functions of the state, and a quadratic Lyapunov function so the dimension of the observable state-space is $N = 7$.

$$\boldsymbol{\phi} = \begin{bmatrix} x_1 & x_2 & x_1^2 & x_2^2 & e^{1/x_1} & e^{1/x_2} & \mathbf{x}^T P \mathbf{x} \end{bmatrix}^T \tag{6.7}$$

where

$$P = \begin{bmatrix} 1060 & 22 \\ \\ 22 & 0.52 \end{bmatrix} \tag{6.8}$$

Using the collected data and the constructed library $\boldsymbol{\phi}$, we determined approximate linear predictors for the CSTR model described in Eq. (6.6).

Remark 18. *The choice of this library is highly dependent on the specific process of interest. For example, in the case of hydraulic fracturing where only the proppant concentration at different spatial locations are important it was shown that considering monomials of the concentration terms yields accurate reduced-order models that can be used in the design of feedback controllers [158]. Therefore, it is intuitive to use these monomials as the observable functions in the EDMD algorithm.*

In order to evaluate the performance of the Koopman based linear predictors, we compared its prediction performance with two commonly used methods for obtaining linear models.

1. Linear model obtained using local linearization of the dynamics at the stable steady-state.
2. Linear model obtained using a subspace identification method, MOESP [21].

The identified matrices for the localized linear model of the CSTR in continuous-time are given below:

$$A = \begin{bmatrix} -27.76 & -0.473 \\ \\ 1130 & 18.1 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ \\ 0.0043 \end{bmatrix} \tag{6.9}$$

where as to obtain the MOESP based model, we used the input/output data to regress a second-order linear model using `n4sid` function in Matlab.

Figures 6.1 and 6.2 show the output predictions of the above described linear models for two different initial conditions (randomly) chosen from the training data set. From the figure it can be seen that a relatively good agreement between the true model and the identified linear models is achieved with respect to the training data. The associated input trajectories are depicted in Figure 6.3. To validate this model, we used additional responses generated by applying a series of step, impulse and random input signals to the process initiated from a given initial condition. Specifically, (a) a step of magnitude $Q = 5000$ kJ/hr was given to the heat rate input starting at $t = 0.5$ hr, (b) an impulse was numerically simulated using a rectangular pulse of magnitude $Q = -10000$ kJ/hr for one sampling period starting at $t = 0.5$ hr, and (c) a random heat rate profile satisfying the input bounds was generated using `rand` function in Matlab, which was provided to the system throughout the operating period.

The results of the model validation are shown in Figures 6.4-6.6. As can be seen from the figures, the best prediction performance is achieved by Koopman based linear predictors followed by the local linearization model. Please note that the prediction quality of the MOESP based model can be enhanced by using a higher-order model. It is well known that for a CSTR process, the local linearization around the stable steady-state provides a very good approximation to the original system. This is clearly reflected in the above figures. Nevertheless, the Koopman based linear models perform even better by virtue of enriching the data with nonlinear observable functions. To assess this quantitatively, the relative root mean squared errors (RMSE) of all the models starting from the same initial condition subjected to random input profiles are averaged over 100 simulations and presented in Table 6.2. The average RMSE is computed as

$$\begin{aligned} RMSE &= \frac{||\mathbf{x}_{approx} - \mathbf{x}_{true}||_{fro}}{||\mathbf{x}_{true}||_{fro}} \\ Average\ RMSE &= \frac{1}{100}\sum_{i=1}^{100} RMSE_i \end{aligned} \tag{6.10}$$

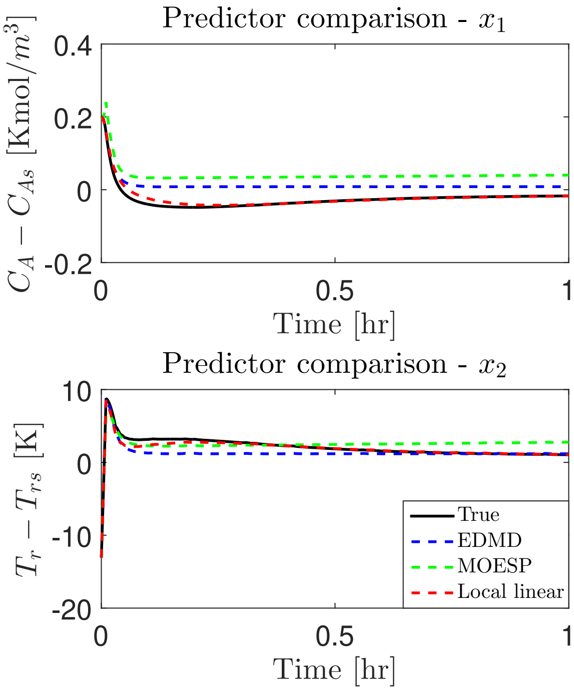


Figure 6.1: Prediction comparison of the CSTR response with respect to the input profile shown in Figure 6.3a that was randomly chosen from the set of training data. Reprinted with permission from [152].

where $\| \cdot \|_{fro}$ is the Frobenius norm. From the table, we observe that the Koopman based linear model is superior to both the local and MOESP models in terms of prediction accuracy. To test the closed-loop performance of these predictors, we utilize them in the design of a feedback controller below.

6.2.2 Closed-loop simulation results

The formulation used in Eq. (6.5) needs an explicit controller (see Eq. (6.5e)) to be constructed based on which the Lyapunov constraint and the stability region of the closed-loop operation are characterized. To this end, we design a Lyapunov-based controller developed initially by Son-

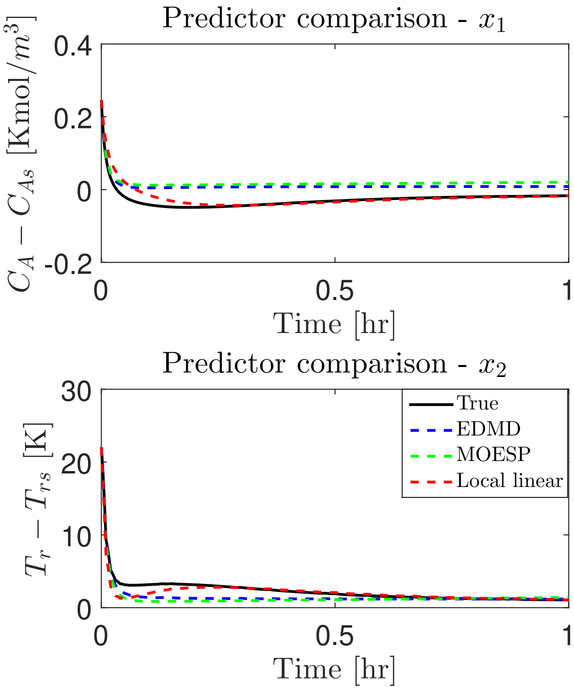


Figure 6.2: Prediction comparison of the CSTR response with respect to the input profile shown in Figure 6.3b that was randomly chosen from the set of training data. Reprinted with permission from [152].

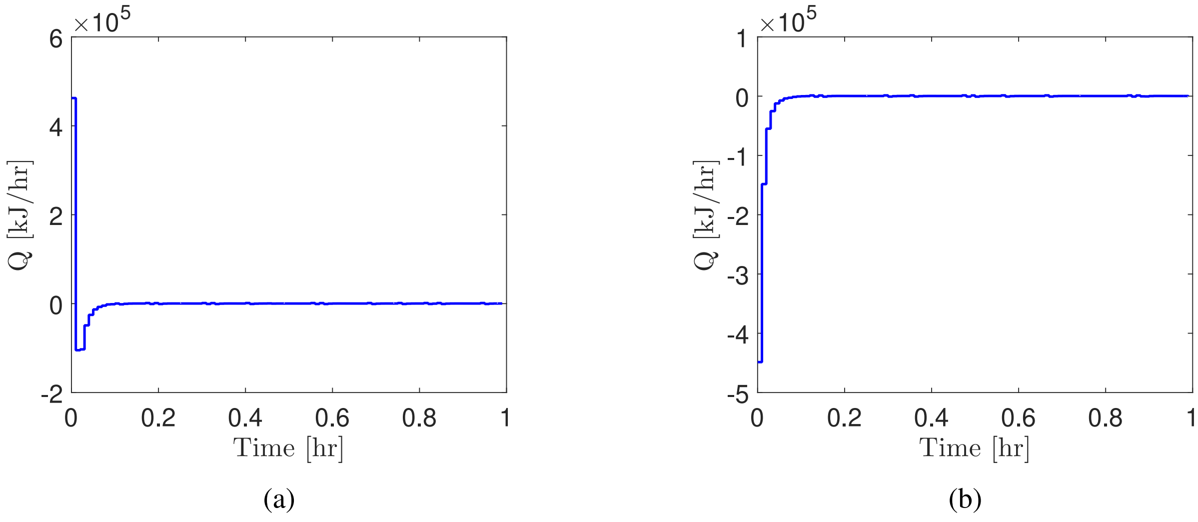
tag [163] for a class of input affine nonlinear dynamical systems of the form:

$$\dot{x} = f(x) + g(x) \cdot u \tag{6.11}$$

Utilizing the above notation, the nonlinear controller for the rate of heat input is given as

$$h(x) = \begin{cases} -\frac{L_fV + \sqrt{L_fV^2 + L_gV^4}}{L_gV}, & \text{if } L_gV \neq 0 \\ 0, & \text{if } L_gV = 0 \end{cases} \tag{6.12}$$

where $f(x)$ and $g(x)$ are the terms in the ODE corresponding to the temperature of the reactor in Eq. (6.6), and $L_fV,\ L_gV$ are the Lie derivatives of the Lyapunov function along the trajectories of



(a)

(b)

Figure 6.3: The associated input profiles selected from the training data sets. (a) input profile corresponding to Figure 6.1. (b) input profile corresponding to Figure 6.2. Reprinted with permission from [152].

Predictor	**Average RMSE**
Koopman	0.5549
Local linear	0.6323
MOESP	0.6331

Table 6.2: Average RMSE of different predictors over 100 random input profiles. Reprinted with permission from [152].

the functions $f(x)$ and $g(x)$, respectively (for example, $L_f V = \frac{\partial V}{\partial x} f(x)$). The quadratic Lyapunov function considered in this example is given in Chapter 6.2.1. Following [162], the stability region $\Omega_r = \{\mathbf{x} \in \mathbb{R}^n : V(\mathbf{x}) \leq r\}$, which will be used in the LMPC formulation, is taken as $r = 64.3$.

Using the nonlinear control law shown above, we designed the proposed LMPC formulation in the form of Eq. (6.5) using the Koopman based linear predictors and the Lyapunov constraint. The sampling time and the prediction horizon of the optimization problem are taken as $\Delta = 0.01$ hr and $N_p = 10$ (i.e., 0.1 hr), respectively. The weighting matrices are taken as $W = diag(10^2, 1)$ and $R = 10^{-6}$ based on the magnitudes of x_1, x_2 and u. In this specific example, we did not consider any explicit state constraints so that $\mathbf{E}_i, \mathbf{H}_i, \mathbf{b}_i = 0$. At the beginning of each sampling interval,

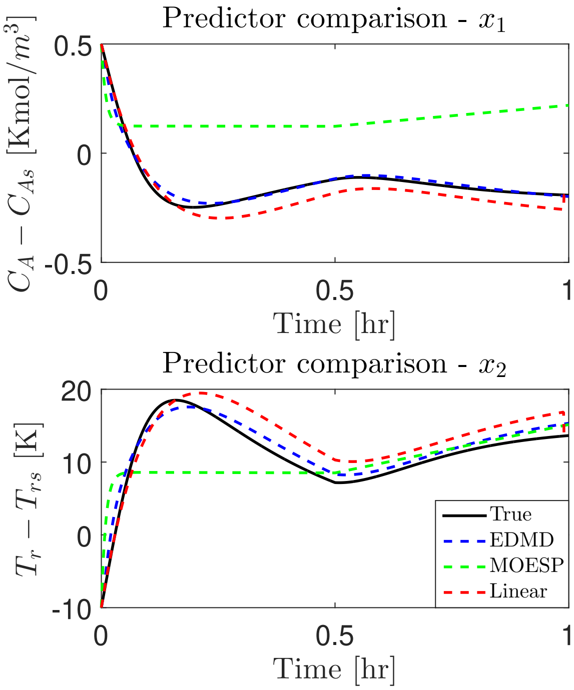


Figure 6.4: Prediction comparison of the CSTR response with respect to a impulse input profile used to validate the linear models. Reprinted with permission from [152].

the state measurements are assumed to be available, which are used to initialize the linear model, as shown in Eq. (6.5c), and perform future state predictions. Additionally, the state measurements are used to calculate $h(x)$ which enforces the Lyapunov constraint of Eq. (6.5e) numerically by imposing it at the beginning of each sampling period. Please note that the input $h(x)$ was applied with input saturation at the bounds and setting it to zero at the steady-state according to Eq. (6.12). The predicted state values are then used in the optimization problem to compute the control inputs and the corresponding process behavior that minimizes the squared deviation from the origin. At any time step, k, the optimization is performed over the prediction horizon length of N_p, and the first step, $u_0^\star$, of the optimal input profile, $[u_0^\star, \cdots, u_{N_p-1}^\star]$, is applied to the process in a sample-and-hold fashion, and this procedure is repeated at every sampling time over the entire operating period of 1 hr.

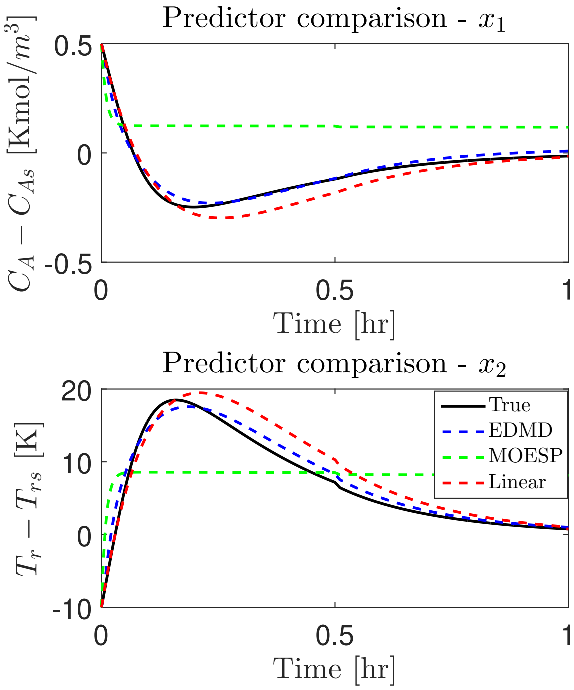


Figure 6.5: Prediction comparison of the CSTR response with respect to a step input profile used to validate the linear models. Reprinted with permission from [152].

The constrained optimization problem was solved using the interior point solver within `fmincon` function in MATLAB R2016b. The default settings of the solver were used with the steady-state value of the input chosen as the initial guess at every sampling time. The CSTR was initialized from $[x_1, x_2] = [0.5, -18]$ and the simulations were performed using an Intel(R) Core(TM) i7-4790 CPU at 3.60 GHz with a 16 GB RAM and an x64-based processor running Windows 8.1 Enterprise.

The closed-loop trajectories of the CSTR under the proposed linear LMPC scheme are shown in Figure 6.7. As can be seen from the figure, starting from the initial condition, successful stabilization of the closed-loop system was achieved and the controller was able to drive the system to its steady-state (dotted line). The proposed controller was able to accomplish this due to accurate predictions of the Koopman based linear model over the prediction horizon of the optimization

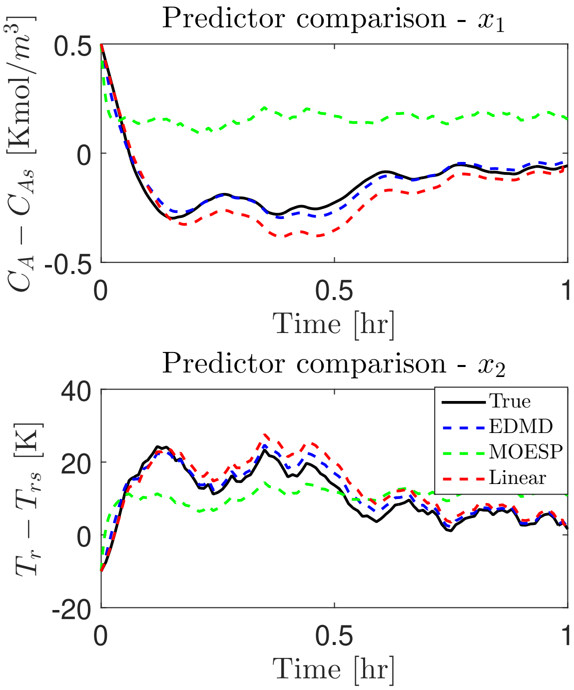


Figure 6.6: Prediction comparison of the CSTR response with respect to a random input profile used to validate the linear models. Reprinted with permission from [152].

problem. The associated input profile determined by solving the optimization problem is shown in Figure 6.8. Note that, in addition to good closed-loop performance, the proposed control scheme has the benefit of being completely data-driven requiring only state (output) measurements. Also, the average computation time required to evaluate the control input from the proposed method is low because it solves a convex quadratic programming problem (when compared to using nonlinear models within the LMPC framework).

6.3 Conclusions

In Chapter 6, we presented a data-driven method for the design of stabilizing controllers for nonlinear dynamical systems by integrating Koopman linear predictors within the LMPC framework. The key idea is to compute finite-dimensional approximations to the Koopman operator which yields linear models that are valid on the entire state space or at least a larger subset of

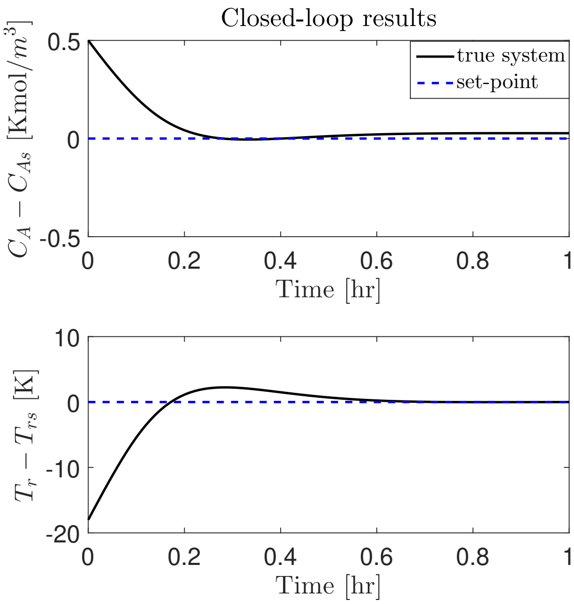


Figure 6.7: The closed-loop response of the CSTR under the proposed Koopman based LMPC scheme for the initial condition of $x_1 = 0.5$ kmol/m^3 and $x_2 = -18$ K. Reprinted with permission from [152].

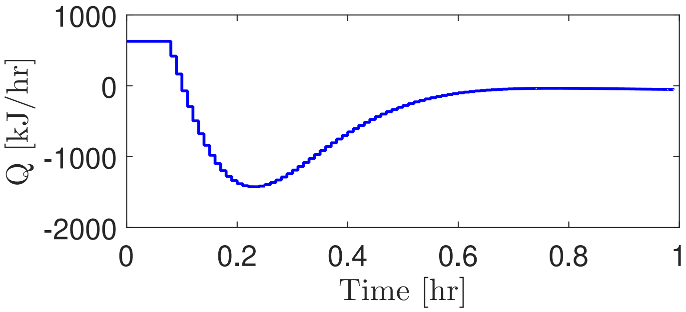


Figure 6.8: The optimal input profile of the CSTR determined by the proposed Koopman based LMPC scheme for the initial condition of $x_1 = 0.5$ kmol/m^3 and $x_2 = -18$ K. Reprinted with permission from [152].

it compared to a small neighborhood around an equilibrium point. These linear models can be used to predict future state evolution in the design of feedback controllers. Specifically, we proposed the use of these predictors in combination with LMPC to obtain closed-loop stability of the time-varying nonlinear operation. Moreover, we showed that the integration of Koopman linear predictors within the LMPC framework leads to a convex (quadratic) optimization problem which can be solved using any of the extensive array of solvers suitable for linear MPC. We demonstrated the application of the proposed method on a nonlinear chemical process example and extensive simulation results were presented. The average RMSE taken for 100 simulations subjected to random input profiles showed the superior performance of Koopman linear predictors compared to local linearization and subspace identification methods. The resultant linear model was then used in the design of a feedback controller using the LMPC approach. In the simulations, the LMPC formulation with the Koopman linear predictors successfully achieved closed-loop stability.

Please note that in order to use the proposed Koopman based LMPC formulation, a CLF for the nominal system must be available a priori. This is a limitation of the proposed approach; however, for simple systems such CLFs can be obtained by solving the Riccati equation using the matrices of the associated linearized system. Furthermore, the stability region used in the LMPC formulation should be characterized using extensive closed-loop simulations under the nonlinear explicit controller $h(x)$. Under this control law the stability region is determined as a sufficiently large level set where the time-derivative of the Lyapunov function along the closed-loop state trajectories is negative.

7. DATA-DRIVEN FEEDBACK STABILIZATION OF NONLINEAR SYSTEMS: KOOPMAN-BASED MODEL PREDICTIVE CONTROL

Although successfully implemented on a broad range of applications, the true potential of Koopman approach can only be realized by certifying that the controllers will guarantee closed-loop stability and robustness. Unlike systems characterized by unforced dynamics, providing stability analysis for forced (input dependent) systems has proven to be difficult because the predictive capability of the Koopman operator can be significantly impacted unless the role of actuation (i.e., the manipulated inputs) is appropriately accounted. To deal with this, [148] redefined the Koopman operator as a function of both states and the inputs. In [146], a modification of EDMD was presented that compensates for the effect of inputs. In [15], a bilinear representation was provided in the Koopman space that is tight and theoretically justified. Using this representation, the authors in [154] proposed a stabilizing feedback controller which relies on control Lyapunov function (CLF) and thus achieves stabilization of the bilinear system.

However, the method in [154] neither solved an optimal control problem nor accounted for explicit state and input constraints. Moreover, it did not comment on the stability analysis of the original nonlinear system under the implementation of the designed controller in the Koopman space. To address this, CLFs were employed in Chapter 6 where a feedback controller was designed for the Koopman space (i.e., lifted domain) using Lyapunov constraints within a MPC formulation. The linear structure of the Koopman models was exploited to transform the original nonlinear MPC problem to a convex quadratic MPC problem that is computationally attractive. However, the limitation of the method presented in Chapter 6 is that the CLF was derived for the original system which requires an explicit mathematical expression of the original nonlinear dynamics; it is particularly challenging when we have limited a priori knowledge of the original nonlinear system. Additionally, even though we have a good understanding of the nonlinear system, it is in practice computationally demanding to determine its corresponding CLFs.

To address these issues, in Chapter 7, we seek to improve upon Chapter 6 by proposing a stabilizing feedback controller applicable to a general system. The major difference here is that the Koopman linear system is represented in terms of the eigenfunctions of the Koopman operator rather than observable functions - for control affine systems this results in a bilinear model in the (lifted) function space. To do so, first, a modified Koopman system identification is applied to derive a bilinear representation of the dynamics. Then, a CLF is determined for the bilinear system in the Koopman eigenfunction space which is employed in the LMPC formulation. Then, a stability criterion is presented that guarantees stability of the original closed-loop system in the $\epsilon - \delta$ sense based on stability of the Koopman bilinear system. Unlike Chapter 6, the feedback control design proposed in this work is completely data-driven and does not require any a priori knowledge of the original system. Moreover, deriving CLFs for the Koopman bilinear system is much more computationally affordable than the original nonlinear system. In fact, the search for CLFs can be focused on a class of quadratic functions which are known to effectively characterize the stability region of simpler systems like the (Koopman) bilinear systems.

7.1 Linear embedding using Koopman eigenfunctions

In Chapter 7.1, we provide background on the Koopman operator for continuous dynamical systems (unlike Chapter 6) and its relation to forced dynamical systems. Subsequently, we present a new modified system identification method over Koopman eigenfunctions, which yields a practical training procedure for embedding nonlinear systems to a bilinear model (for control affine systems) from data.

7.1.1 Koopman eigenfunctions

Let $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$ be the vector of state variables of a continuous-time nonlinear dynamical system whose evolution is governed by the function

$$\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}) \tag{7.1}$$

where $\mathbf{F} : \mathcal{X} \rightarrow \mathcal{X}$ is the nonlinear operator that maps the system states forward in time. It is assumed that the vector field $\mathbf{F}$ is continuously differentiable. The solution to Eq. (7.1) is given by the flow field $\mathbf{\Phi}^t(\mathbf{x})$. Typically, an analytic form for $\mathbf{\Phi}^t(\mathbf{x})$ is impossible to determine and we resort to numerical solutions for Eq. (7.1), which can become computationally intractable.

Now, let $\mathcal{G}$ be a Hilbert space of complex-valued functions on $\mathcal{X}$. The elements of $\mathcal{G}$ are often called *observables* as they may correspond to measurements taken during an experiment or the output of a simulation. In his seminal work, Koopman realized an alternative description of Eq. (2.1) in terms of the evolution of these observables denoted as $g(\mathbf{x})$ with $g : \mathcal{X} \to \mathbb{C}$. Specifically, Koopman theory asserts that the nonlinear system in Eq. (7.1) can be mapped to a *linear* system using an infinite dimensional linear operator $\mathcal{K}^t$ that advances these observables forward in time.

Definition 1 (Koopman operator)**.** *For a given space $\mathcal{G}$ of observables, the Koopman (semi)group of operators $\mathcal{K}^t : \mathcal{G} \to \mathcal{G}$ associated with system Eq. (7.1) is defined by*

$$[\mathcal{K}^t g](\mathbf{x}) = g \circ \mathbf{\Phi}^t(\mathbf{x}) \tag{7.2}$$

By definition, the Koopman operator is linear even though the underlying dynamical system is nonlinear, i.e., it satisfies

$$[\mathcal{K}^t(\alpha g_1 + \beta g_2)](\mathbf{x}) = \alpha[\mathcal{K}^t g_1](\mathbf{x}) + \beta[\mathcal{K}^t g_2](\mathbf{x}) \tag{7.3}$$

The linearity of the Koopman operator allows it to be characterized by its eigenvalues and eigenfunctions. An eigenfunction $\psi \in \mathcal{G} : \mathcal{X} \to \mathbb{C}$ of the Koopman operator is defined to satisfy

$$\begin{aligned} &[\mathcal{K}^t \psi](\mathbf{x}) = e^{\lambda t}\psi(\mathbf{x}) \\ &\frac{d}{dt}\psi(\mathbf{x}) = \lambda\psi(\mathbf{x}) \end{aligned} \tag{7.4}$$

where $\lambda \in \mathbb{C}$ is the associated eigenvalue. These eigenfunctions can be used to predict the time evolution of an observable, in relation with the state dynamics, as long as the given observable lies within the span of these eigenfunctions. Applying chain rule to Eq. (7.4),

$$\frac{d}{dt}\psi(\mathbf{x}) = \nabla\psi(\mathbf{x}) \cdot \mathbf{F}(\mathbf{x}) \triangleq L_{\mathbf{F}}\psi(\mathbf{x}) = \lambda\psi(\mathbf{x}) \tag{7.5}$$

where the Lie derivative with respect to the vector field $\mathbf{F}$, denoted as $L_{\mathbf{F}} = \mathbf{F} \cdot \nabla$, is the infinitesimal generator of the Koopman operator $\mathcal{K}^t$, i.e., $\lim\limits_{t\to 0}(\mathcal{K}^t - I)/t$. Hence, the time varying observable $\tilde{g}(t, \mathbf{x}) = \mathcal{K}^t g(\mathbf{x})$ can be obtained as a solution to the partial differential equation,

$$\begin{aligned} &\frac{\partial}{\partial t}\tilde{g} = \mathbf{F} \cdot \nabla\tilde{g} \triangleq L_{\mathbf{F}}\tilde{g} \\ &\tilde{g}(0, \mathbf{x}) = g(\mathbf{x}) \end{aligned} \tag{7.6}$$

Any finite subset of the Koopman eigenfunctions naturally forms an invariant subspace and discovering these eigenfunctions enables globally linear representations of strongly nonlinear systems.

7.1.2 Modeling forced dynamics

The Koopman operator theory has been conceptually developed for uncontrolled systems. To adopt it for the purposes of control, consider a control affine system as follows:

$$\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}) + \sum_{i=1}^{m} \mathbf{G}_i(\mathbf{x})u_i \tag{7.7}$$

where $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$, $u_i \in \mathcal{U}$ for $i = 1, \cdots, m$, and $\mathbf{G}_i : \mathcal{X} \to \mathcal{X}$ denotes the control vector fields that dictate the effect of input on the system. It is assumed that the vector fields are locally Lipschitz continuous. This is a reasonable assumption which holds true for many physical systems. The evolution of the observable functions for the controlled system of Eq. (7.7) is given, by applying chain rule similar to Eq. (7.6), as

$$\begin{aligned} &\frac{\partial}{\partial t}\tilde{g} = L_{\mathbf{F}}\tilde{g} + \sum_{i=1}^{m} u_i L_{\mathbf{G}_i}\tilde{g} \\ &\tilde{g}(0, \mathbf{x}) = g(\mathbf{x}) \end{aligned} \tag{7.8}$$

where $L_{\mathbf{F}}$ and $L_{\mathbf{G}_i}$ denote the Lie derivatives with respect to the vector fields $\mathbf{F}$ and $\mathbf{G}_i$ for $i = 1, \cdots, m$, respectively. The system Eq. (7.8) is analogous to a bilinear system except for the fact that the operators $L_{\mathbf{F}}$ and $L_{\mathbf{G}_i}$ are infinite dimensional, operating on the function space $\mathcal{G}$. However, if there exist a finite number of observable functions $\bar{g}_1, \cdots, \bar{g}_N$ that span a subspace $\bar{\mathcal{G}} \subset \mathcal{G}$ such that $\mathcal{K}\bar{g} \in \bar{\mathcal{G}}$ for any $\bar{g} \in \bar{\mathcal{G}}$, then $\bar{\mathcal{G}}$ is said to be an invariant subspace and the Koopman operator becomes a finite-dimensional matrix, $\mathbf{K}$. For practical implementation, the Koopman eigenfunctions can be used for $\bar{g}$ such that the finite-dimensional approximation can be determined by projecting on the subspace spanned by these eigenfunctions [164]. The choice of using eigenfunctions as basis is intuitive because an action of the infinitesimal generator of the Koopman operator on these eigenfunctions is dictated simply by a scalar, i.e., the corresponding eigenvalue (see Eq. (7.5)).

7.1.3 Koopman bilinear system identification

To obtain a bilinear form of system of Eq. (7.7) in the Koopman eigenfunction coordinates, we use the Koopman canonical transform (KCT) [15]. Such a transformation is given by

$$\begin{gathered}
\mathbf{z} = \mathbf{\Psi}(\mathbf{x}) = [\psi_1(\mathbf{x}), \cdots, \psi_N(\mathbf{x})]^T, \text{where} \\
\psi_j(\mathbf{x}) = \tilde{\psi}_j(\mathbf{x}), \text{if } \tilde{\psi}_j : \mathcal{X} \to \mathbb{R} \\
[\psi_j(\mathbf{x}),\ \psi_{j+1}(\mathbf{x})]^T = [2Re(\tilde{\psi}_j(\mathbf{x})),\ -2Im(\tilde{\psi}_j(\mathbf{x}))]^T, \\
\text{if } \tilde{\psi}_j,\ \tilde{\psi}_{j+1} : \mathcal{X} \to \mathbb{C} \\
\text{and assuming } \tilde{\psi}_{j+1} = \tilde{\psi}_j^\star
\end{gathered} \tag{7.9}$$

where * denotes the complex conjugate. Applying the above transformation to Eq. (7.7) yields

$$\dot{\mathbf{z}} = \Lambda \mathbf{z} + \sum_{i=1}^{m} u_i L_{\mathbf{G}_i} \mathbf{\Psi} \tag{7.10}$$

where Λ is a block-diagonal matrix constructed using the Koopman eigenvalues $\lambda_j, j = 1, \cdots, N$, which are corresponding to the Koopman eigenfunctions shown in Eq. (7.9), i.e.,

$$\begin{aligned} &\Lambda_{j,j} = \lambda_j, \quad \text{if } \tilde{\psi}_j : \mathcal{X} \to \mathbb{R} \\ &\begin{bmatrix} \Lambda_{j,j} & \Lambda_{j,j+1} \\ \Lambda_{j+1,j} & \Lambda_{j+1,j+1} \end{bmatrix} = |\lambda_j| \begin{bmatrix} cos(\angle\lambda_j) & sin(\angle\lambda_j) \\ -sin(\angle\lambda_j) & cos(\angle\lambda_j) \end{bmatrix}, \\ &\qquad \text{if } \tilde{\psi}_j, \tilde{\psi}_{j+1} : \mathcal{X} \to \mathbb{C} \\ &\qquad \text{and assuming } \tilde{\psi}_{j+1} = \tilde{\psi}_j^\star \end{aligned} \tag{7.11}$$

Assumption 4. $\exists\, \psi_j, j = 1, \cdots, N$ *such that*

$$L_{\mathbf{G}_i}\mathbf{\Psi} = \sum_{j=1}^{N} b_j^{\mathbf{G}_i}\psi_j(\mathbf{x}) = B_i\mathbf{\Psi}$$

where $b_j^{\mathbf{G}_i} \in \mathbb{R}^n$ and $\psi_j(\mathbf{x})$ are defined in Eq. (7.9). In other words, it is assumed that $L_{\mathbf{G}_i}\mathbf{\Psi}$ lies in the span of the eigenfunctions $\psi_j,\ j = 1, \cdots, N$ so that it can be represented using a constant matrix, $B_i \in \mathbb{R}^{N\times N}$.

Based on this assumption, the system of Eq. (7.10) becomes the following bilinear control system in the Koopman space,

$$\dot{\mathbf{z}} = \Lambda\mathbf{z} + \sum_{i=1}^{m} u_i B_i \mathbf{z} \tag{7.12}$$

The objective of the system identification method is to determine the continuous bilinear system of Eq. (7.11) using time-series data generated by the controlled dynamical system of Eq. (7.7). This is done in two parts. First, we calculate the system matrix Λ using the eigenfunctions of the Koopman operator for the uncontrolled part of Eq. (7.7). Although there are several methods available in the literature that can achieve this, the EDMD algorithm is utilized in this work. The algorithm is detailed below.

Calculating Λ:

1. The time-series data of N_t snapshot pairs satisfying the dynamical system of Eq. (7.1) are

generated and organized in the following matrices:

$$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{N_t}], \quad \mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{N_t}], \tag{7.13}$$

where $\mathbf{x}_k \in \mathcal{X}$, $\mathbf{y}_k = \mathbf{F}(\mathbf{x}_k)\Delta t + \mathbf{x}_k \in \mathcal{X}$ and Δt is the discretization time. Note $\mathbf{y}_k$ is used here instead of $\mathbf{x}_{k+1}$ because the data need not necessarily be temporally ordered as long as the corresponding pairs $(\mathbf{x}_k, \mathbf{y}_k)$ are obtained as shown above.

2. A library of nonlinear observable functions $\mathcal{D} = \{\phi_1, \phi_2, \ldots, \phi_N\}$ is selected to define the vector-valued function $\boldsymbol{\phi} : \mathcal{X} \rightarrow \mathbb{R}^N$

$$\boldsymbol{\phi}(\mathbf{x}) = [\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \cdots, \phi_N(\mathbf{x})]^T \tag{7.14}$$

where $\boldsymbol{\phi}$ is used to lift the system from a state space to a function space of observables.

3. A least-squares problem is solved over all the data samples to obtain $\mathbf{K} \in \mathbb{R}^{N \times N}$ which is the transpose of the finite dimensional approximation to the Koopman operator, $\mathcal{K}^t$:

$$\min_{\mathbf{K}} \quad \sum_{i=1}^{N_t} \|\boldsymbol{\phi}(\mathbf{y}_i) - \mathbf{K}\boldsymbol{\phi}(\mathbf{x}_i)\|_2^2 \tag{7.15}$$

The value of $\mathbf{K}$ that minimizes Eq. (7.15) can be determined analytically as:

$$\mathbf{K} = \boldsymbol{\phi}_{XY}\boldsymbol{\phi}_{XX}^{\dagger} \tag{7.16}$$

where † denotes the pseudo inverse, and the data matrices are given by

$$\boldsymbol{\phi}_{XX} = \boldsymbol{\phi}_X\boldsymbol{\phi}_X^T, \quad \boldsymbol{\phi}_{XY} = \boldsymbol{\phi}_Y\boldsymbol{\phi}_X^T \tag{7.17}$$

where

$$\boldsymbol{\phi}_X = [\boldsymbol{\phi}(\mathbf{x}_1), \cdots, \boldsymbol{\phi}(\mathbf{x}_{N_t})],$$

$$\boldsymbol{\phi}_Y = [\boldsymbol{\phi}(\mathbf{y}_1), \cdots, \boldsymbol{\phi}(\mathbf{y}_{N_t})]$$

It has been previously shown that the matrix $\mathbf{K}$ asymptotically approaches the Koopman operator as we increase N_t [149], and hence approximates the evolution of observables.

4. An eigen decomposition of $\mathbf{K}$ is performed to determine the eigenvalues $\tilde{\lambda}_j$ and eigenvectors e_j for $j = 1, \cdots, N$.

5. The eigenvalues are converted to continuous time as $\lambda_j = \log(\tilde{\lambda}_j)/\Delta t$, and the eigenfunctions, ψ_j, are computed, using $\tilde{\psi}_j = \boldsymbol{\phi}^T e_j$, according to the procedure described in Eq. (7.9).

6. The system matrix Λ is constructed using the block-diagonalization described in Eq. (7.11).

Calculating control matrix B_i:

In the next step, the control matrix B_i is calculated using the eigenfunctions. Specifically using *Assumption 4* and the fact that $\boldsymbol{\Psi}(\mathbf{x}) = E^T\boldsymbol{\phi}(\mathbf{x})$ where $E = [e_1, \cdots, e_N]$ is the matrix containing the eigenvectors, we have

$$\begin{aligned} B_i\boldsymbol{\Psi}(\mathbf{x}) &= L_{\mathbf{G}_i}\boldsymbol{\Psi}(\mathbf{x}) \\ &= L_{\mathbf{G}_i}(E^T\boldsymbol{\phi}(\mathbf{x})) = E^T\frac{\partial\boldsymbol{\phi}}{\partial\mathbf{x}}\mathbf{G}_i(\mathbf{x}) \end{aligned} \tag{7.18}$$

The control matrix B_i can be obtained by equating the coefficients of right- and left-hand-side functions of the above equation. Once the system matrices Λ and B_i are determined, a bilinear system of Eq. (7.12) can be constructed using the Koopman eigenfunctions and can be used for the task of designing feedback controllers.

7.2 Koopman Lyapunov-based MPC

In Chapter 7.2, we detail how Koopman operator theory can be integrated with Lyapunov-based predictive control scheme to stabilize the system of Eq. (7.7).

7.2.1 Lyapunov-based predictive control

For simplicity, let us consider the control affine system of Eq. (7.7) with $i = 1$, i.e., a single input. All the results can be generalized to the case of multiple inputs. Without loss of generality, we assume $\mathbf{F}(0) = 0$ and that the origin is an unstable equilibrium point of the uncontrolled system. Then, the closed-loop stabilization problem associated with Eq. (7.7) seeks a state-dependent control law of the form $u = h(\mathbf{x}), h : \mathbb{R}^n \to \mathbb{R}$ which renders the origin stable within some domain $\mathscr{D} \subset \mathbb{R}^n$ for the closed-loop form of Eq. (7.7).

One of the widely used approaches to design state feedback controllers is via the use of CLFs as they facilitate explicit consideration of the stability prior to the controller design. CLF is a continuously differentiable positive definite function $V : \mathscr{D} \to \mathbb{R}_+$ such that for all $\mathbf{x} \in \mathscr{D}/\{0\}$, $\dot{V} := L_{\mathbf{F}}V + uL_{\mathbf{G}}V < 0$. Once this CLF is constructed, design of a feedback law can be straightforward [163].

LMPC is a powerful tool that uses CLFs for the design of an optimal stabilizing feedback controller for nonlinear dynamical systems [160], particularly those characterized by a set of constraints. Essentially, LMPC is a control strategy that possesses all the advantages of a standard MPC and is designed based on an explicit, stable (albeit not optimal) control law $h(\cdot)$. By explicitly adding a Lyapunov constraint to a standard MPC formulation, the controller is able to stabilize the closed-loop system. Additionally, LMPC explicitly characterizes a set of initial conditions starting from where the closed–loop stability is guaranteed. Hence, it ensures stability irrespective of the prediction horizon, i.e., the computational time can be made smaller by decreasing the prediction horizon (reducing the size of the optimization problem). However, the main bottleneck to the success of this method lies in the construction of CLFs for a general nonlinear system. To avoid this, in the proposed method, the system of Eq. (7.7) is first transformed into a bilinear control system of Eq. (7.11), using the procedure described above, for which determining a CLF is much easier. Particularly, the search for a CLF of a bilinear system can now be limited to the class of quadratic functions and an optimization problem can be solved to determine the required CLF [154]. Then, one can apply LMPC in the Koopman eigenspace to determine a stabilizing input for the bilinear system of Eq. (7.11).

7.2.2 Bounded explicit control $h(\mathbf{z})$

Let us consider the Koopman bilinear system of Eq. (7.11) with $i = 1$, i.e., a single input obtained using the system identification method described in Chapter 7.1.3. This system is assumed to be stabilizable, which implies the existence of a feedback control law $u(t) = h(\mathbf{z}(t))$ that satisfies input constraints for all $\mathbf{z}$ inside a given stability region and renders the origin of the closed-loop system asymptotically stable. This is equivalent to assuming that there exists a CLF for the system of Eq. (7.11). Due to the bilinear structure of the system, the CLF can be limited to a class of quadratic functions, i.e., $V(\mathbf{z}) = \mathbf{z}^T P\mathbf{z}$. The necessary and sufficient conditions for the symmetric positive definite matrix P such that the system of Eq. (7.11) is stabilizable are provided in [154]. The theorem is stated below.

Proposition 1 (see [154], Theorem 2)**.** *The bilinear system of Eq.* (7.11) *is stabilizable if and only if there exists an* $N \times N$ *symmetric positive definite matrix* P *such that for all* $\mathbf{z} \neq 0 \in \mathbb{R}^N$ *with* $\mathbf{z}^T(P\Lambda + \Lambda^T P)\mathbf{z} \geq 0$*, we have* $\mathbf{z}^T(PB + B^T P)\mathbf{z} \neq 0$.

In other words, for $\dot{V}(\mathbf{z}) = \mathbf{z}^T(P\Lambda + \Lambda^T P)\mathbf{z} + u(\mathbf{z}^T(PB + B^T P)\mathbf{z})$ to be negative, given that the first term on the right hand side is positive, then the second term cannot be zero so that the control action u can render $\dot{V} < 0$. Once the conditions of *Proposition 1* are satisfied, one way to determine the explicit control law $h(\mathbf{z})$, required to stabilize the bilinear system, is provided by the following formula by Sontag [165]:

$$
\begin{gathered}
b(\mathbf{z}) = \begin{cases} -\frac{L_\Lambda V + \sqrt{L_\Lambda V^2 + L_B V^4}}{L_B V}, & \text{if } L_B V \neq 0 \\ 0, & \text{if } L_B V = 0 \end{cases} \\
h(\mathbf{z}) = \begin{cases} u_{min}, & \text{if } b(\mathbf{z}) < u_{min} \\ b(\mathbf{z}), & \text{if } u_{min} \leq b(\mathbf{z}) \leq u_{max} \\ u_{max}, & \text{if } b(\mathbf{z}) > u_{max} \end{cases}
\end{gathered}
\tag{7.19}
$$

where $L_\Lambda V = \mathbf{z}^T(P\Lambda + \Lambda^T P)\mathbf{z}$, $L_B V = \mathbf{z}^T(PB + B^T P)\mathbf{z}$, and $h(\mathbf{z})$ represents the saturated control law that accounts for the input constraints $u_{min} \leq u(t) \leq u_{max} \in \mathcal{U}$. For the above controller, one can show, using a standard Lyapunov argument, that if the closed-loop state evolves within a level set of V, the time-derivative of the CLF is negative definite ensuring asymptotic stability. Let the largest level set of V be given by

$$\Omega_r = \{\mathbf{z} \in \mathbb{R}^N : V(\mathbf{z}) \leq r\} \tag{7.20}$$

where r is the largest number for which $\Omega_r \subseteq \Omega$. Ω is the complete stability region, starting from which the origin of the bilinear system under Eq. (7.19) is guaranteed to be stable. In practice, the entire region of attraction, Ω, is very difficult to estimate even for simple systems.

7.2.3 Koopman Lypaunov-based predictive control

Now that we have the explicit control law, the idea is to stabilize the bilinear system using the Lyapunov-based predictive control scheme as below:

$$\min_{u \in \mathcal{S}(\Delta)} \quad \int_{t_k}^{t_k+N_p\Delta} [\mathbf{z}^T(\tau) W \mathbf{z}(\tau) + u^T(\tau) R u(\tau)] d\tau, \tag{7.21a}$$

$$\text{s.t} \quad \dot{\mathbf{z}}(t) = \Lambda \mathbf{z}(t) + u(t) B \mathbf{z}(t) \tag{7.21b}$$

$$\mathbf{z}(t_k) = \mathbf{\Psi}(\mathbf{x}(t_k)) \tag{7.21c}$$

$$u_{min} \leq u(t) \leq u_{max}, \ \forall t \in [t_k, t_k + N_p\Delta) \tag{7.21d}$$

$$V(\mathbf{z}(t)) \leq \hat{r}, \qquad \forall t \in [t_k, t_k + N_p\Delta] \quad \text{if } \mathbf{x}(t_k) \in \Omega_{\hat{r}} \tag{7.21e}$$

$$\dot{V}(\mathbf{z}(t_k), \mathbf{u}(t_k)) \leq \dot{V}(\mathbf{z}(t_k), h(\mathbf{z}(t_k))), \quad \text{if } \mathbf{x}(t_k) \in \Omega_r / \Omega_{\hat{r}} \tag{7.21f}$$

where $\mathcal{S}(\Delta)$ is the family of piece-wise constant functions with sampling period Δ, N_p is the prediction horizon, and $W \in \mathbb{R}^{N \times N}$ and $R \in \mathbb{R}$ are positive definite weighting matrices. The manipulated input (solution to the optimization problem) of the above system under the LMPC control law is defined as

$$u = u^\star(t|t_k), \qquad \forall t \in [t_k, t_k + N_p\Delta) \tag{7.22}$$

where $u^\star(t|t_k) = [u^\star(t_k), \cdots, u^\star(t_k + N_p\Delta)]$. The first value of $u^\star(t|t_k)$ is applied to the closed-loop system for the next sampling time period $t \in [t_k, t_k + \Delta)$ and the procedure is repeated until the end of operation.

In the LMPC formulation of Eqs. (7.21a) -(7.21c), Eq. (7.21a) denotes a performance index that is to be minimized, Eq. (7.21b) is the Koopman bilinear model of the system of Eq. (7.11) used to predict the future evolution of the states, and Eq. (7.21c) provides the initial condition which is obtained as a transformation of the actual state measurement. In addition to these constraints, the LMPC formulation considers two Lyapunov constraints, Eq. (7.21e) and Eq. (7.21f). In the design of LMPC, one important factor we need to consider is the sample-and-hold implementation of the control law. To explicitly deal with the sampled system, we consider a region $\Omega_{\hat{r}}$, where $\hat{r} < r$. Specifically, when $\mathbf{z}(t_k)$ is received at a sampling time t_k, if $\mathbf{z}(t_k)$ is within the region $\Omega_{\hat{r}}$, the LMPC minimizes the cost function within the region $\Omega_{\hat{r}}$; however, if $\mathbf{z}(t_k)$ is in the region $\Omega_r/\Omega_{\hat{r}}$, i.e., $\mathbf{z}(t_k) \in \Omega_r$ but $\mathbf{z}(t_k) \notin \Omega_{\hat{r}}$, the LMPC first drives the system state to the region $\Omega_{\hat{r}}$ and then minimizes the cost function within $\Omega_{\hat{r}}$. In other words, due to the sample-and-hold implementation of the control law, the region $\Omega_{\hat{r}} \subset \Omega_r$ is chosen as a 'safe' zone to make Ω_r invariant. Please note that this is not a limitation of the LMPC formulation but of the discrete-time implementation of the control action to a continuous-time dynamical system. Ultimately, the size of the safe set $\Omega_{\hat{r}}$ depends on the hold time (i.e., sampling time), Δ (details given below in *Proposition 2*).

Therefore, Eq. (7.21e) is only active when $\mathbf{z}(t_k) \in \Omega_{\hat{r}}$ and ensures that the sampled state is maintained in the region $\Omega_{\hat{r}}$ (so that the actual state of the closed-loop system is in the stability region Ω_r). The constraint of Eq. (7.21f) is only active when $\hat{r} < V(\mathbf{z}(t_k)) \leq r$ and ensures the rate of change of the Lyapunov function is smaller than or equal to that of the value obtained if the explicit control law $h(\mathbf{z})$ is applied to the closed-loop system in a sample-and-hold fashion. These constraints allow the LMPC controller to inherit the stability properties of $h(\mathbf{z})$, i.e., it possesses at least the same stability region Ω_r as the controller $h(\mathbf{z})$. This implies that the (equilibrium point of) closed-loop system of Eqs. (7.21a) -(7.21f) is guaranteed to be stable for any initial state inside the region Ω_r provided that the sampling time Δ is sufficiently small. Note that because of this property, the LMPC does not require a terminal constraint used in a traditional MPC setting. Additionally, the feasibility of Eqs. (7.21a) -(7.21f) is guaranteed because $u = h(\mathbf{z})$ is always a feasible solution to the above optimization problem. Even though the above formulation does not explicitly consider the state constraints, they can be readily incorporated.

Proposition 2. *Consider the system of Eq. (7.11) under the MPC control law of Eqs. (7.21a)-(7.21f), which is designed using a CLF, V, that has a stability region Ω_r under continuous implementation of the explicit controller $h(\mathbf{z})$. Then, given any positive real number d, $\exists$ positive real numbers $\Delta^\star$ such that if $\mathbf{z}(0) \in \Omega_r$ and $\Delta \in (0, \Delta^\star]$, then $\mathbf{z}(t) \in \Omega_r, \forall t \geq 0$ and $\lim_{t\to\infty} \|\mathbf{z}(t)\| \leq d$.*

Proof. The proof is divided into three parts. In *Part 1*, the robustness of the explicit controller is shown which preserves the closed-loop stability when the control action is implemented in a sample-and-hold fashion with a sufficiently small hold time (Δ). In *Part 2*, the controller of Eqs. (7.21a)-(7.21f) is shown to be feasible for all $\mathbf{z}(0) \in \Omega_r$. Subsequently, in *Part 3*, it is shown that the stability region Ω_r is invariant under the predictive controller of Eqs. (7.21a)-(7.21f).

Part 1. *To prove the robustness of the explicit controller, we need to show the existence of a positive real number $\Delta^\star$ such that all state trajectories originating in Ω_r converge to the level set of $\Omega_{\hat{r}}$ for any value of $\Delta \in (0, \Delta^\star]$. To achieve this, we need to consider different cases for $\mathbf{z}(0)$ inside the stability region, i.e., we consider arbitrary regions $\mathcal{Z}$ and $\Omega_{r'}$ inside Ω_r. Figure 7.1 represents a schematic of the different cases considered in the following proof.*

First, consider a small region close to the boundary of the stability region denoted as $\mathcal{Z} := \{\mathbf{z} : (r - r') \leq V(\mathbf{z}) \leq r\}$, for some $0 < r' < r$. Now, let $h(0) = h_0$ be computed for $\mathbf{z}(0) = \mathbf{z}_0 \in \mathcal{Z}$

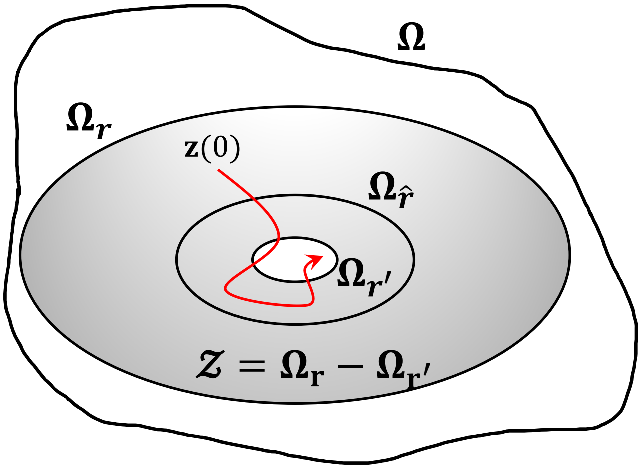


Figure 7.1: A schematic representing the stability region of the bounded controller Ω_r, together with the sample-and-hold constrained set, $\Omega_{\hat{r}}$, and the overall stability region of the system, Ω. The gray shaded part represents the ring, $\mathcal{Z}$, close to the boundary of the stability region, Ω_r.

and held constant until a time $\hat{\Delta}$ *such that* $h(t) := h_0\ \forall t \in (0, \hat{\Delta}]$. *Then,*

$$\begin{aligned}\dot{V}(\mathbf{z}(t)) &= L_\Lambda V(\mathbf{z}(t)) + L_B V(\mathbf{z}(t))h_0 \\ &= L_\Lambda V(\mathbf{z}_0) + L_B V(\mathbf{z}_0)h_0 \\ &\quad + (L_\Lambda V(\mathbf{z}(t)) - L_\Lambda V(\mathbf{z}_0)) \\ &\quad + (L_B V(\mathbf{z}(t))h_0 - L_B V(\mathbf{z}_0)h_0).\end{aligned} \tag{7.23}$$

Since the initial state $\mathbf{z}_0 \in \mathcal{Z} \subseteq \Omega_r$, *and* h_0 *are computed based on the stabilizing control law Eq.* (7.19), *it follows that* $\dot{V}(\mathbf{z}_0) := L_\Lambda V(\mathbf{z}_0) + L_B V(\mathbf{z}_0)h_0 \leq -\rho V(\mathbf{z}_0)$ *(this can be shown by substituting Eq.* (7.19) *in* $\dot{V}$*). Combining this with the definition of* $\mathcal{Z}$, *we have* $L_\Lambda V(\mathbf{z}_0) + L_B V(\mathbf{z}_0)h_0 \leq -\rho(r - r')$.

We also need the following properties to complete the proof.

Property 1. *Since the evolution of* $\mathbf{z}$ *is continuous,* $\|u\| \leq u_{max}$ *and* $\mathcal{Z}$ *are bounded, one can find,* *for all* $\mathbf{z}_0 \in \mathcal{Z}$ *and a fixed* $\hat{\Delta}$*, a positive real number* k_1 *such that* $\|\mathbf{z}(t) - \mathbf{z}_0\| \leq k_1\hat{\Delta}$ *for all* $t \leq \hat{\Delta}$*.*

Property 2. *Additionally, since* $L_\Lambda V(\cdot)$ *and* $L_B V(\cdot)$ *are continuous functions, the following properties hold:*

$$\begin{aligned} \|L_\Lambda V(\mathbf{z}(t)) - L_\Lambda V(\mathbf{z}_0)\| &\leq k_2\|\mathbf{z}(t) - \mathbf{z}_0\| \leq k_1 k_2 \hat{\Delta} \\ \|L_B V(\mathbf{z}(t))h_0 - L_B V(\mathbf{z}_0)h_0\| &\leq k_3\|\mathbf{z}(t) - \mathbf{z}_0\| \leq k_1 k_3 \hat{\Delta} \end{aligned} \tag{7.24}$$

where the second inequality in each equation holds because of Property 1. *Using all the above inequalities in Eq.* (7.23),

$$\dot{V}(\mathbf{z}(t)) \leq -\rho(r - r') + (k_1 k_3 + k_2 k_3)\hat{\Delta} \tag{7.25}$$

Now, if we choose $\hat{\Delta} < (\rho(r - r') - c)/(k_1 k_3 + k_2 k_3)$ *where* $c < \rho(r - r')$ *is a positive number, we get* $\dot{V}(\mathbf{z}(t)) \leq -c < 0$ *for all* $t \leq \hat{\Delta}$*. This implies that, given a* $\hat{r}$*, if we find an* r' *such that* $r - r' < \hat{r}$ *and determine the corresponding* $\hat{\Delta}$*, then the control action computed for any* $\mathbf{z} \in \mathcal{Z}$ *and held for a time period less than* $\hat{\Delta}$ *will ensure that the state does not escape* Ω_r *(because* $\dot{V} < 0$ *during this time).*

Now, we need to show the existence of a Δ' *such that for all* $\mathbf{z}_0 \in \Omega_{r'} := \{\mathbf{z}_0 : V(\mathbf{z}_0) \leq r - r'\}$ *we have* $\mathbf{z}_0 \in \Omega_{\hat{r}} := \{\mathbf{z}_0 : V(\mathbf{z}_0) \leq \hat{r}\}$*. Consider* Δ' *such that*

$$\hat{r} = \underset{\mathbf{z}_0 \in \Omega'_r, h \in \mathcal{U}, t \in [0, \Delta']}{max} V(\mathbf{z}(t)) \tag{7.26}$$

This is possible because both V *and* $\mathbf{z}$ *are continuous functions, and therefore for any* $r' < r$*, one can find a sufficiently small* Δ' *such that Eq.* (7.26) *holds. All that remains now is to show that for all* $\mathbf{z}_0 \in \Omega_{\hat{r}}$ *if* $\Delta \in (0, \Delta^\star]$ *where* $\Delta^\star = min\{\hat{\Delta}, \Delta'\}$*, then* $\mathbf{z}(t) \in \Omega_{\hat{r}}$ $\forall t \geq 0$*.*

Consider all $\mathbf{z}_0 \in \Omega_{\hat{r}} \cap \Omega_{r'}$*. Then by definition,* $\mathbf{z}(t) \in \Omega_{\hat{r}}$ *for* $t \in [0, \Delta^\star]$ *since* $\Delta^\star \leq \Delta'$*. On the other hand, for all* $\mathbf{z}_0 \in \Omega_{\hat{r}}/\Omega_{r'}$*, i.e.,* $\mathbf{z}_0 \in \mathcal{Z}$*, it was shown that* $\dot{V} < 0$ *for* $t \in [0, \Delta^\star]$ *since* $\Delta^\star \leq \hat{\Delta}$*. Therefore,* $\Omega_{\hat{r}}$ *is an invariant set under the control law of Eq.* (7.19).

Hence, all trajectories originating in Ω_r *converge to* $\Omega_{\hat{r}}$ *with a hold time less than* $\Delta^\star$*. That is, for all* $\mathbf{z}_0 \in \Omega_r$*, lim sup*$_{t \to \infty} V(\mathbf{z}(t)) \leq \hat{r}$*. Since,* $V(\cdot)$ *is a continuous function, one can always find* *a finite, positive number d such that $V(\mathbf{z}) \leq \hat{r} \implies \|\mathbf{z}\| \leq d$. Therefore, lim sup$_{t\to\infty}V(\mathbf{z}(t)) \leq \hat{r} \implies$ lim sup$_{t\to\infty}\|\mathbf{z}(t)\| \leq d$.*

Part 2. *Let us consider some $\mathbf{z}(0) \in \Omega_r$ under the predictive controller of Eqs.* (7.21a)-(7.21f) *with a prediction horizon N_p denoting the number of prediction steps. There are two cases. If $\mathbf{z}_0 \in \Omega_r/\Omega_{\hat{r}}$, the feasibility of constraint of Eq.* (7.21f) *is guaranteed by the control law of Eq.* (7.19) *as shown in* Part 1. *Additionally, if $V(\mathbf{z}(0)) \leq \hat{r}$, once again the control input trajectory under the explicit controller of Eq.* (7.19)*, given by $u(t) = h(\mathbf{z}(t))$, $\forall t \in [t_k, t_k + N_p\Delta]$, provides a feasible initial guess to constraint of Eq.* (7.21e) *because it was designed to stabilize the system, i.e., $V(\mathbf{z}(t)) \leq \hat{r}$. This shows that for all $\mathbf{z}(0) \in \Omega_r$ the Koopman LMPC of Eqs.* (7.21a)-(7.21f) *is feasible.*

Part 3. *To prove the last part, please note that since constraint of Eq.* (7.21f) *is feasible, upon implementation it ensures that the value of the Lyapunov function under the predictive controller $u(t)$ decreases at each sampling time. Since Ω_r is a level set of V, and $\dot{V}$ decreases, the state trajectories cannot escape Ω_r. Additionally, satisfying constraint of* (7.21e) *means that $\Omega_{\hat{r}}$ continues to remain invariant under the implementation of the predictive controller of Eqs.* (7.21a)-(7.21f). *The recursive feasibility of Eqs.* (7.21d)-(7.21f) *implies that $V \leq r$ and $\dot{V} < 0$ for all $\mathbf{z}(t)$ under the Lyapunov-based controller given by Eqs.* (7.21a)-(7.21f)*. However, since it is implemented in a sample-and-hold fashion there exists a maximum sampling time $\Delta^\star$, given in* Part 1*, such that when $\Delta \in (0, \Delta^\star)$ it is guaranteed that for all $\mathbf{z}(\mathbf{0}) \in \Omega_r$, lim$_{t\to\infty}\|\mathbf{z}(t)\| \leq d$.*

This completes the proof. □

Remark 19. *Please note that in practice, one can characterize the values of $r, \hat{r}, \Delta^\star$ and d by performing several closed-loop simulations where the controller defined in Eqs.* (7.21a)-(7.21f) *is continuously applied to the system. However, the estimate of the stability region Ω_r determined using explicit controllers such as Eq.* (7.19) *does not necessarily equate the entire domain Ω, which remains a difficult problem even for linear systems. Nevertheless, these estimates can be improved by considering multiple CLFs.*

Proposition 2 formalizes that the stability properties of the Koopman bilinear system under the Lyapunov–based predictive controller are inherited from the explicit (bounded) controller under discrete implementation. Now, when there is no mismatch between the Koopman model and the original system, the stability properties will be easily translated to the original system. Obviously, we can derive an exact model without any model-plant mismatch if we can implement the infinite dimensional Koopman operator. However, as described previously, only a finite dimensional approximation based on the projection of these operators on a subspace is commonly used for practical implementation. In this regard, since the model-plant mismatch between the Koopman model and the original system is inevitable, we additionally study and derive the bound on the prediction error between the original state and the predicted state from the Koopman model in the following theorem.

In order to extend the stability results to the original nonlinear system of Eq. (7.7), we make the following assumption.

Assumption 5. *Let the inverse mapping from the Koopman space,* $\mathbf{z}$*, to the original state space,* $\mathbf{x}$*, be continuously differentiable, i.e.,* $\exists\, \boldsymbol{\xi}(\mathbf{z}) = [\xi_1(\mathbf{z}), \cdots, \xi_n(\mathbf{z})]^T \in C^1 : \mathbb{R}^N \to \mathbb{R}^n$ *such that* $\hat{x}_i = \xi_i(\mathbf{z}), i = 1, \cdots, n$ *where* $\hat{\mathbf{x}} = [\hat{x}_1, \cdots, \hat{x}_n]$ *is the predicted state vector obtained from the inverse mapping defined above.*

Then, the stability properties of the closed-loop system of Eqs. (7.21a)-(7.21f) of $\mathbf{z}$ can be shown to be inherited to the original nonlinear system of $\mathbf{x}$ under the above assumption and is formalized in the following theorem.

Theorem 1. *Suppose that system Eq. (7.7) satisfies* Assumptions 4-5. *Let* $\mathbf{x}(t)$ *and* $\hat{\mathbf{x}}(t)$ *denote the original state and the predicted state values, respectively. The solutions for* $\mathbf{x}(t)$ *and* $\hat{\mathbf{x}}(t)$ *are* *given by the following dynamic equations:*

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), u(t)), \qquad \mathbf{x}(0) = \mathbf{x}_0 \tag{7.27}$$

$$\hat{\mathbf{x}}(t) = \xi(\mathbf{z}(t)), \qquad \hat{\mathbf{x}}(0) = \mathbf{x}_0 \tag{7.28}$$

$$\dot{\mathbf{z}}(t) = \Lambda\mathbf{z}(t) + u(t)B\mathbf{z}(t), \; \mathbf{z}(0) = \boldsymbol{\phi}(\hat{\mathbf{x}}(0)) \tag{7.29}$$

Then, the difference between $\mathbf{x}(t)$ *and* $\hat{\mathbf{x}}(t)$ *is bounded by*

$$\|\mathbf{x}(t) - \hat{\mathbf{x}}(t)\| \leq \frac{\nu}{l_x}(e^{l_x t} - 1) \tag{7.30}$$

where ν *denotes the modeling error which bounds the difference between*

$$\|\mathbf{f}(\hat{\mathbf{x}}, u) - \hat{\mathbf{f}}(\hat{\mathbf{x}}, u)\| \leq \nu \tag{7.31}$$

where $\mathbf{f}(\cdot) = \mathbf{F}(\cdot) + \mathbf{G}(\cdot)u$ *is the original nonlinear dynamical system, and* $\hat{\mathbf{f}}(\hat{\mathbf{x}}, u) = \frac{\partial \xi}{\partial \mathbf{z}}\dot{\mathbf{z}}$ *denotes the solution to* $\dot{\hat{\mathbf{x}}}(t)$*. Under this condition, the stabilizing feedback control input* $u^{\star}(t)$ *obtained from the Lyapunov-based predictive control law of Eqs. (7.21a)-(7.21f) for the Koopman linear system of Eq. (7.9) also stabilizes the original system of Eq. (7.6), i.e., the origin of the closed-loop system of Eq. (7.6) is Lyapunov stable.*

Proof. The proof is divided into two parts. First, we show that the predicted state $\hat{\mathbf{x}}(t)$ is stable under the application of the Koopman LMPC controller of Eqs. (7.21a)-(7.21f) to the Koopman bilinear system. In the second part, we show that the evolution of the error between the original state and the predicted state is bounded under *Assumption 5* and the Lipschitz property of the vector fields, $\mathbf{F}$ and $\mathbf{G}$.

Part 1. *Let us consider any initial condition* $\mathbf{x}(0)$ *such that* $\mathbf{x}(0) = \hat{\mathbf{x}}(0) = \mathbf{x}_0$ *and* $\|\mathbf{x}_0\| \leq \delta$*. Recall from* Proposition 2 *that the predictive controller of Eqs.* (7.21a)-(7.21f) *ensures that the lifted states do not escape the stability region* Ω_r*, i.e.,* $V(\mathbf{z}(t)) \leq r, \dot{V} < 0 \; \forall t$*. Therefore,* *lim* $sup_{t\to\infty}\|\mathbf{z}(t)\| \leq d$. *Now, from* Assumption 5, *since the inverse mapping* $\boldsymbol{\xi}(\mathbf{z})$ *is assumed to be continuous (differentiable), the following holds true:*

$$\begin{aligned}&\|\boldsymbol{\xi}(\mathbf{z}(t))\| := \|\hat{\mathbf{x}}(t)\| \leq \epsilon_z\|\mathbf{z}(t)\|\\ &\textit{lim sup}_{t\to\infty}\|\hat{\mathbf{x}}(t)\| \leq \hat{d}\end{aligned} \tag{7.32}$$

where $\hat{d} = \epsilon_z d$. *In other words, since the controller ensures asymptotic stability of the lifted state, it implies that* $\|\mathbf{z}(t)\|$ *is bounded at all times and eventually converges to* d. *This in turn implies that* $\hat{\mathbf{x}}(t)$ *is bounded at all times, albeit by different constants at different sampling times. Now, if we choose* $\hat{\epsilon}$ *to be the maximum of all these bounds, then* $\|\hat{\mathbf{x}}(t)\| < \hat{\epsilon}$, $\forall t$. *Hence, for any initial condition* $\|\mathbf{x}_0\| \leq \delta$, *the implementation of the predictive controller of Eqs.* (7.21a)-(7.21f) *guarantees that* $\|\hat{\mathbf{x}}(t)\| \leq \hat{\epsilon}, \forall t$. *This implies that the predicted states of the original system starting close enough to the equilibrium (at a distance* δ*) will be maintained close to the equilibrium at all times.*

Part 2. *Now, it remains to prove that the modeling error between the original state vector and the predicted states is bounded at all times for all* $\|\mathbf{x}_0\| \leq \delta$. *Let us consider the modeling error* $e(t) = \mathbf{x}(t) - \hat{\mathbf{x}}(t)$, *then the evolution of the error is given as*

$$\begin{aligned}\|\dot{e}(t)\| &= \|\dot{\mathbf{x}}(t) - \dot{\hat{\mathbf{x}}}(t)\|\\ &= \|\mathbf{f}(\mathbf{x}, u) - \hat{\mathbf{f}}(\hat{\mathbf{x}}, u)\|\end{aligned} \tag{7.33}$$

where $\mathbf{f}(\mathbf{x}, u) = \mathbf{F}(\mathbf{x}) + \mathbf{G}(\mathbf{x})u$ *is the nonlinear dynamical system, and* $\hat{\mathbf{f}}(\hat{\mathbf{x}}, u)$ *denotes the evolution of the predicted state* $\hat{\mathbf{x}}$, *which can be determined from the following Koopman bilinear system:*

$$\hat{\mathbf{f}}(\hat{\mathbf{x}}, u) = \frac{\partial\boldsymbol{\xi}}{\partial\mathbf{z}}\dot{\mathbf{z}} \tag{7.34}$$

By adding and subtracting $\mathbf{f}(\hat{\mathbf{x}}, u)$ *to Eq.* (7.33)*, we get*

$$\begin{aligned} \|\dot{e}(t)\| &= \|\mathbf{f}(\mathbf{x}, u) - \mathbf{f}(\hat{\mathbf{x}}, u) + \mathbf{f}(\hat{\mathbf{x}}, u) - \hat{\mathbf{f}}(\hat{\mathbf{x}}, u)\| \\ &\leq \|\mathbf{f}(\mathbf{x}, u) - \mathbf{f}(\hat{\mathbf{x}}, u)\| + \|\mathbf{f}(\hat{\mathbf{x}}, u) - \hat{\mathbf{f}}(\hat{\mathbf{x}}, u)\| \end{aligned} \tag{7.35}$$

The Lipschitz property of $\mathbf{f}(\cdot)$*, combined with the bounds on* u*, implies that there exists a positive constant* l_x *such that the following inequality holds for all* $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ *and* $u \in \mathcal{U}$*:*

$$\|\mathbf{f}(\mathbf{x}, u) - \mathbf{f}(\mathbf{x}', u)\| \leq l_x \|\mathbf{x} - \mathbf{x}'\| \tag{7.36}$$

Additionally, since $\hat{\mathbf{x}}$ *is bounded (see* Part 1 *in the proof of* Theorem 1*),* $\mathbf{f}$ *is Lipschitz, and the mapping* $\boldsymbol{\xi}$ *is continuously differentiable, there exists a positive constant* ν *such that the second term on the right hand side of the inequality in Eq.* (7.35) *is bounded by* ν*. Combining it with Eq.* (7.36) *we have*

$$\begin{aligned} \|\dot{e}(t)\| &\leq l_x \|\mathbf{x} - \hat{\mathbf{x}}\| + \nu \\ &\leq l_x \|e(t)\| + \nu \end{aligned} \tag{7.37}$$

Therefore, given the zero initial condition (i.e., $e(0) = 0$*), the upper bound for the norm of the error vector can be determined by integrating Eq.* (7.37) *as*

$$\int_0^t \frac{\|\dot{e}(\tau)\|}{l_x \|e(\tau)\| + \nu} \leq t \tag{7.38}$$

and solving for $\|e(t)\|$

$$\|e(t)\| = \|\mathbf{x}(t) - \hat{\mathbf{x}}(t)\| \leq \frac{\nu}{l_x}(e^{l_x t} - 1) \tag{7.39}$$

Finally, since the error between the original and predicted vectors is bounded and that the Koopman LMPC controller of Eqs. (7.21a)-(7.21f) *stabilizes the predicted state vector* $\|\hat{\mathbf{x}}(t)\| \leq \hat{\epsilon}$*, there exists a positive constant* ϵ *such that* $\|\mathbf{x}(t)\| \leq \epsilon$ *for all* t*.*

Therefore, for all $\|\mathbf{x}_0\| \leq \delta$*, the implementation of the predictive controller of Eqs.* (7.21a)-(7.21f) *ensures that* $\|\mathbf{x}(t)\| \leq \epsilon$ *for all* t*, thereby rendering the original nonlinear system stable.*

This completes the proof. □

Remark 20. *Please note that one cannot guarantee asymptotic stability of the original nonlinear system under the proposed controller because there is always loss of information when transforming the system to a different space.*

Remark 21. Assumption 5 *seems restrictive in selecting the types of basis functions to determine the Koopman bilinear models as the inverse of the eigenfunctions is required to be* C^1*. However, in practice, one can numerically obtain a separate mapping from the Koopman space to the original space without actually inverting the eigenfunctions. One example would be to assume the system states* $\mathbf{x}$ *be contained in the span of* $\bar{\mathcal{G}}$*, the finite subset of the observable space. This implies that there exists a constant matrix* $C \in \mathbb{R}^{n \times N}$ *such that* $\mathbf{x} = C\mathbf{z}$*. Then, a convex optimization problem can be solved to determine the relation* C *[10]. In this case, the error of the optimization problem must be certified to be bounded to ensure that the proposed controller successfully stabilizes the closed-loop system.*

Remark 22. *Please note that in this work we do not consider model-plant mismatch due to uncertainties. In the presence of disturbances, to ensure the robust closed-loop stability of the original system, we have to show the inherent robustness of the KLMPC law of Eq.* (7.21) *by guaranteeing the robust feasibility and robust positive invariance of the control system (such as in [166, 167]) under a specific prediction error bound. This robust closed-loop stability of KLMPC will be studied as a future work, and the prediction error bound between the original state and the predicted state based on the Koopman model, which derived in* Theorem 1*, would be a great starting point.*

7.3 Numerical experiments

We applied our results on two illustrative examples: Van der Pol oscillator and a simple pendulum system, showing the performance of our provably-stable Lyapunov-based predictive controller designed in the Koopman function space. Each example produced closed-loop results that are stable with respect to the original state-space.

7.3.1 Van der Pol oscillator

In our first example, we consider the Van der Pol oscillator, which is described by the following equations:

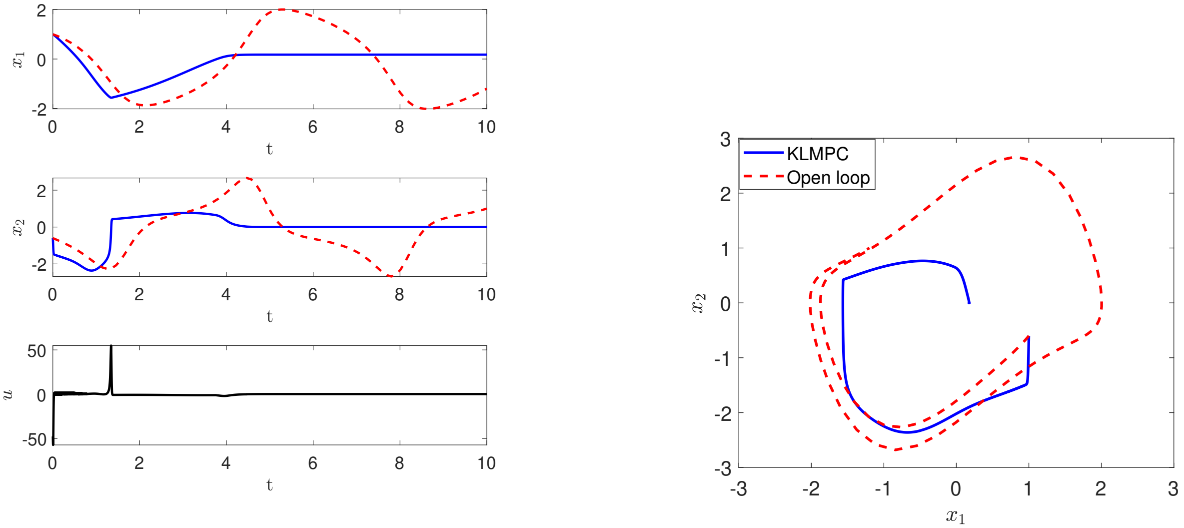
$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= (1 - x_1^2)x_2 - x_1 + u\end{aligned} \tag{7.40}$$

At $u = 0$, the unforced dynamics of the Van der Pol oscillator are characterized by a limit cycle with an unstable equilibrium point at the origin. We will see whether the proposed Koopman LMPC is able to stabilize the system at the origin. First, the data required to build the Koopman bilinear model is generated by simulating the unforced system of Eq. (7.40). The simulations were initialized uniformly over a circle around the origin, and a number of trajectories for 10 s were collected with a sampling time of $\Delta = 0.01$ s, i.e., 10^3 time-series samples per trajectory. In the next step, the states were lifted to the high-dimensional space by using monomials of degree 5 as the dictionary functions $\boldsymbol{\phi}(\mathbf{x}(t))$, i.e., $\boldsymbol{\phi}(\mathbf{x}(t)) = [1, x_1, x_2, x_1^2, x_1x_2, \cdots, x_2^5]^T$. This results in a lifted system of dimension $\mathbf{z} \in \mathbb{R}^{21}$, and the system matrix Λ was constructed using the algorithm described in Chapter 7.1.3. To determine the B matrix in the controlled setting, the relation between the Koopman eigenfunctions and dictionary functions was used as shown in Eq. (7.18). The derivatives of the eigenfunctions were computed using the symbolic toolbox in MATLAB. This completes the identification of the Koopman bilinear model of Eq. (7.11).

Next, the Koopman LMPC developed in Chapter 7.2 was applied to control the system of Eq. (7.40) with $N = 21$ eigenfunctions as the new states, $\mathbf{z}$, in the transformed space. The initial condition was chosen randomly around the unstable equilibrium and the control objective was to stabilize the system at the origin. The CLF used to define the explicit stable controller $h(\mathbf{z})$ was obtained by solving the following optimization problem as defined in [154]:

$$\begin{aligned} \min_{\sigma>0, P=P^T} \quad & \sigma - \gamma \text{trace}(PB) \\ \text{s.t} \quad & \sigma I - (PA + A^T P) \geq 0 \\ & c^L I \leq P \leq c^U I \end{aligned} \tag{7.41}$$

where σ represents the epigraph form of the largest singular value of $(PA + A^T P)$, and $c^L, c^U > 0$ are two positive scalars used to bound the eigenvalues of P. The weighting parameter $\gamma > 0$ was chosen as 2 in this example. The explicit controller, $h(\mathbf{z})$, was determined by using the obtained CLF, $V = \mathbf{z}^T P \mathbf{z}$, within the Sontag's formula as shown in Eq. (7.19). The matrices W and R in Eq. (7.21a) were chosen to be $W = I \in \mathbb{R}^{21\times 21}$ and $R = 1$, respectively. The prediction horizon was set to 1 s, i.e., $N_p = 1/\Delta = 100$. Figure 7.2 shows the comparison between open and closed loop results. It can be observed from Figure 7.2 that the system was stabilized at the origin as desired.



(a) (b)

Figure 7.2: Comparison of open-loop and closed-loop trajectories for the Van der Pol oscillator with u from Eqs. (7.21a) - (7.21f).

7.3.2 Simple pendulum

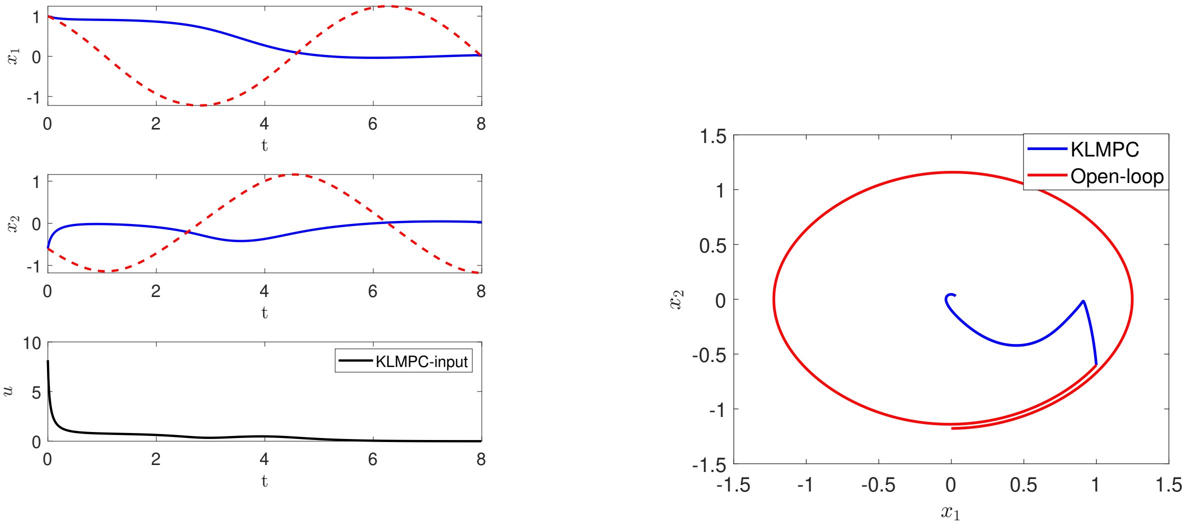
The next example we considered is the controlled two dimensional pendulum oscillator given by the following dynamics:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= 0.01x_2 - sin(x_1) + u \end{aligned} \tag{7.42}$$

where $[x_1, x_2] = [\theta, \dot{\theta}] \in \mathbb{R}^2$ denote the angular displacement and angular velocity of the pendulum, respectively. The system of Eq. (7.42) is characterized by a unique unstable equilibrium point at the origin. We considered the system dynamics near the unique unstable equilibrium point at the origin all the way until the limit cycle (shown in Figure 7.3). The Koopman models have to make predictions over this range of initial conditions, and the control objective is to stabilize the system at the origin.

The training data were generated by simulating the unforced pendulum equation from uniform random initial conditions $(x_1(0), x_2(0)) \in [-2, 2] \times [-2, 2]$. From each trajectory, 10^3 samples were recorded at $\Delta = 0.01$ s apart. Similar to the previous example, the dictionary of observable functions required for nonlinear transformation was considered to be monomials of degree up to 5, i.e., $\mathbf{z} \in \mathbb{R}^{21}$. The approximation of the Koopman operator and eigenfunctions was then performed by lifting the time-series data samples using the selected dictionary. The system matrices, Λ, and the control matrix, B, were then used to design the feedback controller proposed in Eq. (7.21). The CLF used in the explicit control design was determined by solving the optimization problem of Eq. (7.41) using the `cvx` package, a MATLAB-based modeling system for solving disciplined convex optimization problems and is much suitable for semidefinite matrix optimization problems like Eq. (7.41).

It is worth mentioning that the proposed Koopman LMPC controller design is not restricted to using a specific form of control law for $h(\mathbf{z})$. In fact, besides Sontag's formula, there are several other possible choices for the explicit controller $h(\mathbf{z})$. Provided we are not constrained to specifications on the amplitude of feedback, we can use the following simple feedback law to define the control law: $h(\mathbf{z}) = -kL_BV(\mathbf{z}) = -k\mathbf{z}^T(PB+B^TP)\mathbf{z}$. In this example, the value of k was chosen to be $k = 10$. The matrices W and R in Eq. (7.21a) were chosen to be $W = I \in \mathbb{R}^{21\times 21}$ and $R = 1$, respectively. The prediction horizon was set to 1 s, i.e., $N_p = 1/\Delta = 100$. For the closed-loop simulation, we randomly selected initial points within $[-1, 1] \times [-1, 1]$ and solved the closed-loop system with `ode45` solver in MATLAB. Figure 7.3 shows the comparison between open and closed loop results for one such initial condition. It can be observed from Figure 7.3 that the controller forced the trajectory of the closed-loop system to the origin as desired. Moreover, in the case of pendulum system, the limit cycle of the open loop system corresponds to the boundary of the basin of attraction and the proposed Koopman LMPC controller forced the states to remain inside this stability region (limit cycle) at all times before the trajectories slide to the origin.



(a) (b)

Figure 7.3: Comparison of open-loop and closed-loop trajectories for the simple pendulum oscillator with u from Eqs. (7.21a) - (7.21f).

7.4 Conclusions

In Chapter 7, we introduced a new approach for designing stabilizing feedback controllers for nonlinear dynamical systems. Leveraging Koopman operator theory, nonlinear dynamics are lifted to a function space where they are embedded in bilinear models that are computed using finite-dimensional approximations to the Koopman operator and its eigenfunctions. A feedback controller is then designed using LMPC that uses explicit Lyapunov constraints to characterize closed-loop stability of the Koopman bilinear system. Due to the bilinear structure of the Koopman model, the CLF can be obtained easily by limiting the search to the class of quadratic functions via an optimization problem. Furthermore, universal control approaches like Sontag's formula readily provides the explicit control law required in the LMPC formulation which is typically a bottleneck for general nonlinear systems. Most importantly, we demonstrated, based on the stability of the Koopman model, that the proposed controller was capable of stably regulating nonlinear dynamics in the original state-space provided that a continuously differentiable inverse mapping exists. The numerical examples indicated that the proposed feedback controller was able to successfully force unstable dynamics to the origin. This was observed from the closed-loop plots presented. Future work will focus on certifying the proposed approach in terms of robustness in the presence of uncertainties. Furthermore, we hope to apply the proposed approach to other flow control problems, studying whether it can provide similar insight into how to design stabilizing feedback controllers for other applications.

8. SUMMARY AND FUTURE WORK

8.1 Summary

Nonlinear systems abound in nature. In fact, many chemical processes are characterized by complex nonlinear models that describe their dynamics. However, a nonlinear trajectory can explore many regions of the state space, and therefore, different portions of such a trajectory can be associated with distinct regimes especially in the case of systems with moving boundaries. In such systems, since the spatial domain of interest as well as the dominant spatial patterns of the system changes with time, traditional model approximation methods may fail to accurately capture the dynamics. This poses challenges for optimization, estimation and synthesis of practically implementable controllers that achieve optimal (economic) performance. To this end, my work focuses on developing novel methods based on linear operator theory by combining spectral decomposition, model reduction and machine learning techniques to obtain computationally efficient yet highly accurate models that aid in prediction and control. The application of interest throughout this research work is hydraulic fracturing. Specifically, we consider simultaneous fracture propagation and multi-phase proppant flow within a single fracture based on the widely used PKN model. The hydraulic fracturing process is inherently a moving boundary problem associated with highly-coupled dynamics which makes obtaining approximate models extremely difficult. The problem we seek to solve is to obtain a desired final fracture conductivity, via model predictive control (MPC), in order to maximize the productivity of a fractured well.

Specifically, this research work tackles the hydraulic fracturing control problem on two fronts. First, by developing a local model reduction framework that uses data clustering within projection-based methods to accurately capture the local behavior of every portion of the solution trajectory in the state-space. Intuitively, since moving boundary systems are associated with varying spatial features, a time-domain partitioning is performed as a first step. To achieve this, a novel clustering strategy based on Mixed Integer Nonlinear Programming is implemented on the time-series data associated with the system of interest. This data, which can be obtained either from numerical simulations of the high-fidelity model or from experimental runs, should be able to represent a large portion of the solution state space (i.e., the data is collected at different initial conditions, inputs, etc.). The motivation of this step is that each cluster is implicitly characterized by the similarity of its dynamic behavior, and therefore, represents a particular portion of the solution. In the next step, the reduced bases are derived within each cluster, using methods such as Proper Orthogonal Decomposition and Dynamic Mode Decomposition. In the case of fracking, a full-order discretization resulted in a large system of 450 spatial and 20,000 temporal nodes. The local technique, when applied to reduce this system, resulted in a 95% decrease in the dimensionality and computation time while having superior approximation accuracy in comparison with the global methods.

Second, by adopting an operator-centric perspective of dynamical systems to develop linear models. Although the local model reduction framework is superior in performance, sometimes it may lead to unstable models due to discontinuity between different clusters. Therefore, a single model that is accurate within a larger domain in the state-space would be beneficial in some cases. However, as discussed earlier, using model reduction methods directly will not yield good results. To avoid this, an alternative approach is explored which relies on nonlinear transformation. Specifically, Koopman operator theory is relied on developing linear predictors (usually high-dimensional) that approximate nonlinear systems. In order to achieve this, the first step is to learn the finite-dimensional approximation of the Koopman operator from data using a dictionary of observable functions. For example, the dictionary can be populated with polynomial functions of proppant concentration, or in an unsupervised fashion by using sparse regression from machine learning. A large number of basis functions must be constructed such that the observable quantities of interest (states) lie in the span of these functions and hence can be predicted in a linear fashion. The problem of dictionary learning is re-envisioned from the perspective of sparse regression. Specifically, a sparse regression problem is solved over a large set of candidate functional forms. By imposing sparsity, only the functional forms that contribute the most towards the system dynamics are selected to determine the underlying structure of the model. Since no transformation of the state variables is performed, the obtained model will be able to infer some key underlying physical phenomena. On application to a hydraulic fracturing process with simultaneous fracture propagation and proppant transport, the sparse regression based method was able to successfully reveal important physical phenomena such as proppant transport and fracture propagation inside a fracture. It also highlights how a priori knowledge can be incorporated easily into the algorithm. The functional forms thus obtained from sparse regression are then taken as observables to span the Koopman subspace. In the final step, the control part is incorporated by solving a multi-step error minimization problem which is simply a least-squares regression. The final model will be in the form of a linear time invariant system which results in a standard convex optimization problem in the MPC formulation allowing it to be solved faster than in the case of nonlinear MPC problems. Moreover, we can integrate these linear predictors within a stabilizing control framework and show that the stability properties of the linear system (in the observable space) are inherited by the original nonlinear system under certain assumptions. Based on numerical simulations it is demonstrated that the developed linear predictors perform favorably in terms of prediction accuracy compared to several standard techniques. Additionally, it is shown that the Koopman MPC framework successfully achieves the control objective and has superior performance compared to feedback strategies based on local linearization.

Overall, this work enhances the capabilities of traditional methods to approximate more complex dynamical systems such as those characterized by moving boundaries and provides a platform for future innovation within data-based model identification with potential applications to many real world industrial systems.

8.2 Future work

The latter part of this research work has shown the tremendous potential of operator theory in designing universal and principled feedback controllers for general nonlinear systems. Despite the tremendous progress in operator-based control, there are a number of theoretical and applied questions that must be addressed in the future. In line with this, the following is a list of natural extensions and potential improvements in this direction that are currently under consideration in Dr. Kwon's research lab.

1. *Deep auto encoders for learning Koopman invariant subspaces*

   As described earlier, one of the key challenges in Koopman system identification is the requirement to select, a priori, a suitable dictionary of basis functions for the nonlinear transformation, which is usually left to the discretion of the user. Although we proposed a sparse regression based learning algorithm to deal with this, it can quickly become computationally intractable especially for systems with a large number of states, which results in a combinatorially large search space for potential functions. Moreover, the effect of actuation must be appropriately accounted for to describe a general nonlinear system. The challenge of including actuation is that since the input affects the Koopman operator and its eigenfunctions in a nonlinear way, it would be necessary to include many nonlinear functions of the input, in addition to the state variables within the dictionary of EDMD. For this reason, the function space, once again, may quickly become prohibitively large to cover a sufficiently large range of the dynamics. One way to deal with this issue is to leverage the power of deep learning to discover parsimonious representations of the nonlinear dictionary. Specifically, the proposed architecture will be made up of a deep auto encoder neural network that has three high-level requirements: (1) the dictionary function values are given as the outputs of an encoder network, i.e., the states and inputs are lifted using the encoder; (2) a linear time evolution of these nonlinear functions is given by the finite dimensional approximation of the Koopman operator, parameterized by the state and inputs; and (3) a nonlinear decoder reconstructs the original state values from the dictionary values. The three requirements are associated with different loss functions namely, the reconstruction error of the encoder/decoder, the error associated with enforcing linear dynamics with the Koopman operator, and the future state prediction error with respect to a fixed horizon. A regularization term can also be added for generality and to avoid over fitting as is the standard practice. Simultaneous learning of the operator with the encoder and decoder networks in this fashion has the capability to extract more information about the original states from fewer features provided by the encoder. This enables nonlinear reconstruction of the states from descriptive dictionary functions while still keeping the dictionary size to be extremely small. This is actually the same principle that has led to the incredible success of auto encoders for feature extraction, manifold learning, and dimensionality reduction in many applications. In this regard, several works utilized the power of machine learning to train the dictionary employed in EDMD [150, 151, 168, 169]. However, all these works consider unforced systems and, at this point, these advanced embedding techniques have not been adapted for controlled dynamics, which is an exciting avenue of ongoing work.

2. *Switched-system control of nonlinear systems using multiple Koopman models*

   Obtaining global bilinearization for a nonlinear system (with moving boundaries) is practically impossible owing to the time-varying spatial domain. In such cases, one way to improve the control performance of Koopman LMPC is to use multiple Koopman models (this is in line with the local model reduction idea presented in Chapter 3). While it is computationally difficult to estimate the entire stability region of a controlled dynamical system, the computational requirement can be substantially improved by using multiple CLFs. Motivated by this, it is beneficial to develop multiple model-based controllers. In this case, the control design problem transforms to a switching problem between multiple control problems based on different Koopman models. In fact, this idea of quantization has been explored in the context of Koopman operator based control in [170] where the control system is replaced by a set of autonomous systems with constant inputs. Specifically, in [170], within the set of admissible inputs, a finite number of constant input profiles are predefined and the corresponding autonomous Koopman linear models are generated and used within the optimal control problem. The problem then becomes a (computationally very demanding) mixed integer optimal problem where the set of feasible inputs (decision variables) is restricted to the finite set of constant inputs used for model generation. The proposed approach considers a linear interpolation or Euclidean distance based online switching between models that will eliminate the combinatorial nature of the optimization problem introduced in [170]. Also, for each Koopman model of the switched system, a CLF will be designed to estimate the stability region based on each CLF. This will improve the stability region estimates and will ensure that the underlying closed-loop system remains stable. Moreover, the proposed method should not assume fixed constant input profiles, and therefore, the feasibility set of inputs and their optimality will be significantly increased compared to [170] and will enable controller designs for continuous dynamical systems.

REFERENCES

[1] I. Mezić, “Spectral properties of dynamical systems, model reduction and decompositions,” *Nonlinear Dynamics*, vol. 41, no. 1-3, pp. 309–325, 2005.

[2] B. O. Koopman, “Hamiltonian systems and transformation in hilbert space,” *Proc. Natl. Acad. Sci. USA*, vol. 17, no. 5, p. 315, 1931.

[3] B. O. Koopman and J. V. Neumann, “Dynamical systems of continuous spectra,” *Proc. Natl. Acad. Sci. USA*, vol. 18, no. 3, pp. 255–263, 1932.

[4] M. Budišić, R. Mohr, and I. Mezić, “Applied koopmanism,” *Chaos*, vol. 22, no. 4, p. 047510, 2012.

[5] A. Mauroy and I. Mezić., “On the use of fourier averages to compute the global isochrons of (quasi) periodic dynamics,” *Chaos interdiscip. J. Nonlin. Sci*, vol. 22, no. 3, p. 033112, 2012.

[6] A. Mauroy, I. Mezić, and J. Moehlis, “Isostables, isochrons, and koopman spectrum for the action–angle representation of stable fixed point dynamics,” *Physica D*, vol. 261, pp. 19–30, 2013.

[7] Y. Lan and I. Mezić, “Linearization in the large of nonlinear systems and koopman operator spectrum,” *Physica D*, vol. 242, pp. 42–53, 2013.

[8] A. Mauroy and I. Mezić, “Global stability analysis using the eigenfunctions of the koopman operator,” *IEEE Trans. Automat. Control*, vol. 61, no. 11, pp. 3356–3369, 2016.

[9] I. Mezić, “Koopman operator spectrum and data analysis.” *arXiv preprint*, *arXiv:1702.0759*, 2017.

[10] M. Korda and I. Mezić, “Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control,” *Automatica*, vol. 93, pp. 149–160, 2018.

[11] M. Korda, Y. Susuki, and I. Mezić, “Power grid transient stabilization using koopman model predictive control,” *arXiv preprint*, *arXiv:1803.10744*, 2018.

[12] H. Arbabi, M. Korda, and I. Mezić, “A data-driven koopman model predictive control framework for nonlinear partial differential equations,” in *2018 IEEE Conference on Decision and Control (CDC)*, Miami Beach, FL, Dec 17-19 2018, pp. 6409–6414.

[13] S. Hanke, S. Peitz, O. Wallscheid, S. Klus, J. Böcker, and M. Dellnitz, “Koopman operator based finite-set model predictive control for electrical drives,” *arXiv preprint*, *arXiv:1804.00854*, 2018.

[14] A. Surana, “Koopman operator based observer synthesis for control-affiine nonlinear systems,” in *55th IEEE Conf. Decision and Control (CDC)*, Las Vegas, USA, 2016, pp. 6492–6499.

[15] A. Surana and A. Banaszuk, “Linear observer synthesis for nonlinear systems using koopman operator framework,” *IFAC-PapersOnLine*, vol. 49, no. 18, pp. 716–723, 2016.

[16] A. Surana, M. O. Williams, M. Morari, and A. Banaszuk, “Koopman operator framework for constrained state estimation,” in *56th IEEE Conf. Decision and Control (CDC)*, Melbourne, Australia, 2017, pp. 94–101.

[17] P. Holmes, J. L. Lumley, and G. Berkooz, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. New York: Cambridge University Press,, 1996.

[18] P. J. Schmid and J. Sesterhenn, “Dynamic mode decomposition of numerical and experimental data.” *In Bull. Amer. Phys. Soc. 61st APS Meeting, San Antonio, Texas.*, vol. 208, 2008.

[19] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D. S. Henningson, “Spectral analysis of nonlinear flows.” *J. Fluid Mech.*, vol. 641, pp. 115–127, 2009.

[20] K. Willcox and J. Peraire, “Balanced model reduction via the proper orthogonal decomposition.” *AIAA J*, vol. 40, pp. 2323–2330, 2002.

[21] M. Verhaegen and P. Dewilde, "Subspace model identification part 1. the output-error state-space model identification class of algorithms," *Int. J. Control*, vol. 56, no. 5, pp. 1187–1210, 1992.

[22] P. Van Overschee and B. De Moor, "N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems." *Automatica*, vol. 30, pp. 75–93, 1994.

[23] P. Van Overschee and B. De Moor, *Subspace Identification for Linear Systems: Theory - Implementation - Applications*. New York: Springer, 1996.

[24] J. N. Juang, M. Phan, L. G. Horta, and R. W. Longman, "Identification of observer/kalman filter markov parameters: Theory and experiments," NASA, Technical Memorandum 104069, 1991.

[25] J. Juang and R. Pappa, "An eigensystem realization algorithm for modal parameter identification and model reduction." *Journal of guidance, control, and dynamics*, vol. 8, no. 5, pp. 620–627, 1985.

[26] R. Gonzalez-Garcia, R. Rico-Martinez, and I. G. Kevrekidis, "Identification of distributed parameter systems: A neural net based approach," *Comput. Chem. Eng.*, vol. 22, pp. 965–968, 1998.

[27] O. Ogunmolu, S. Gu X., Jiang, and N. Gans, "Nonlinear systems identification using deep dynamic neural networks," *arXiv preprint arXiv:1610.01439*, 2016.

[28] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine*, vol. 2, no. 1, pp. 559–572, 1901.

[29] M. Loève, *Probability Theory II.*, 4th ed. Springer-Verlag, 1978.

[30] E. N. Lorenz, "Empirical orthogonal functions and statistical weather prediction." Massachusetts Institute of Technology" Technical report, December 1956.

[31] J. L. Lumley, *Stochastic Tools in Turbulence.* New York: Academic Press, 1970.

[32] L. Sirovich, “Turbulence and the dynamics of coherent structures i.” *Q. Appl. Math*, vol. 45, pp. 561–571, 1987.

[33] L. Sirovich, “Turbulence and the dynamics of coherent structures ii.” *Q. Appl. Math*, vol. 45, pp. 573–582, 1987.

[34] L. Sirovich, “Turbulence and the dynamics of coherent structures iii.” *Q. Appl. Math*, vol. 45, pp. 583–590, 1987.

[35] A. K. Bangia, P. F. Batcho, I. G. Kevrekidis, and G. E. Karniadakis, “Unsteady 2-d flows in complex geometries: Comparative bifurcation studies with global eigenfunction expansion.” *SIAM J. Sci. Comput*, vol. 18, no. 3, pp. 775–805, 1997.

[36] K. Kunisch and S. Volkwein, “Galerkin proper orthogonal decomposition methods for parabolic problems.” *Numerische mathematik*, vol. 90, no. 1, pp. 117–148, 2001.

[37] S. Volkwein, “Model reduction using proper orthogonal decomposition.” Lecture notes, university of Graz.

[38] G. Berkooz, P. Holmes, and J. Lumley, “The proper orthogonal decomposition in the analysis of turbulent flows.” *Ann. Rev. Fluid. Mech*, vol. 25, pp. 539–575, 1993.

[39] G. Berkooz and E. Titi, “Galerkin projections and the proper orthogonal decomposition for equivariant equations.” *Phys. Lett. A*, vol. 174, pp. 94–102, 1993.

[40] E. Christensen, M. Brons, and J. Sorensen, “Evaluation of proper orthogonal decomposition based decomposition techniques applied to parameter-dependent nonturbulent flows.” *SIAM J. Sci. Comput*, vol. 21, pp. 1419–1434, 2000.

[41] H. M. Park and M. W. Lee, “An efficient method of solving the navier-stokes equations for flow control.” *Int. J. Numer. Methods Eng*, vol. 41, pp. 1133–1151, 1998.

[42] H. M. Park and Y. D. Jang, “Control of burger’s equation by means of mode reduction.” *Int. J. Eng. Sci*, vol. 38, no. 7, pp. 785–805, 2000.

[43] H. M. Park and J. H. Lee, "Solution of an inverse heat transfer problem by means of empirical reduction of modes." *Z. Angew. Math. Phys*, vol. 51, pp. 17–38, 2000.

[44] H. M. Park and W. J. Lee, "A new numerical method for the boundary optimal control problems of the heat conduction equation." *Int. J. Numer. Methods Eng*, vol. 53, pp. 1593–1613, 2002.

[45] S. Ravindran, "Proper orthogonal decomposition in optimal control of fluids." *Int. J. Numer. Methods Fluids*, vol. 34, pp. 425–448, 2000.

[46] S. Ravindran, "Reduced-order adaptive controllers for fluid flows using pod." *J. Sci. Comput*, vol. 15, pp. 457–478, 2000.

[47] J. Rodriguez and L. Sirovich, "Low-dimensional dynamics for the complex ginzburg-landau equations." *Physica D*, vol. 43, pp. 77–86, 1990.

[48] N. Smaoui and D. Armbruster, "Symmetry and the karhunen-loève analysis." *SIAM J. Sci. Comput*, vol. 18, pp. 1526–1532, 1987.

[49] S. Volkwein, "Optimal control of a phase field model using the proper orthogonal decomposition." *Z. Angew. Math Mech*, vol. 81, pp. 83–97, 2001.

[50] L. Lao, M. Ellis, and P. D. Christofides, "Economic model predictive control of parabolic pde systems: Addressing state estimation and computational efficiency," *J. Process Control*, vol. 24, no. 4, pp. 448–462, 2014.

[51] L. Lao, M. Ellis, and P. D. Christofides, "Handling state constraints and economics in feedback control of transport-reaction processes," *J. Process Control*, vol. 32, pp. 98–108, 2015.

[52] B. R. Noack, M. Schlegel, B. Ahlborn, B. Mutschke, M. Morzyński, P. Comte, and G. Tadmor, "A finite-time thermodynamics formalism for unsteady flows." *J. Non-Equilib. Thermodyn.*, vol. 33, pp. 103–148, 2008.

[53] P. J. Schmid, "Dynamic mode decomposition of experimental data." *In 8th International Symposium on Particle Image Velocimetry - PIV09*, 2009.

[54] P. J. Schmid, “Dynamic mode decomposition of numerical and experimental data.” *J. Fluid Mech.*, vol. 656, pp. 5–28, 2010.

[55] P. J. Schmid, L. Li, M. P. Juniper, and O. Pust, “Applications of the dynamic mode decomposition.” *Theor. Comput. Fluid Dyn.*, vol. 25, pp. 249–259, 2011.

[56] A. Seena and H. J. Sung, “Dynamic mode decomposition of turbulent cavity flows for self-sustained oscillations.” *Int. J. Heat Fluid Fl.*, vol. 32, pp. 1098–1110, 2011.

[57] Y. Mizuno, D. Duke, C. Atkinson, and J. Soria, “Investigation of wall-bounded turbulent flow using dynamic mode decomposition.” *J. Phys. Conf. Ser.*, vol. 318, p. 042040, 2011.

[58] T. W. Muld, G. Efraimsson, and D. S. Henningson, “Flow structures around high-speed train extracted using proper orthogonal decomposition and dynamic mode decomposition,” *Comput. Struct.*, vol. 57, pp. 87–97, 2012.

[59] P. J. Schmid, K. E. Meyer, and O. Pust, “Dynamic mode decomposition and proper orthogonal decomposition of flow in a lid-driven cylindrical cavity.” *In 8th International Symposium on Particle Image Velocimetry - PIV09*, 2009.

[60] P. J. Schmid, “Application of the dynamic mode decomposition to experimental data.” *Exp. Fluids.*, vol. 50, pp. 1123–1130, 2011.

[61] C. Pan, D. Yu, and J. Wang, “Dynamical mode decomposition of gurney flap wake flow.” *Theor. Appl. Mech. Lett.*, vol. 1, p. 012002, 2011.

[62] O. Semeraro, G. Bellani, and F. Lundell, “Analysis of time-resolved PIV measurements of a confined turbulent jet using POD and Koopman modes.” *Exp. Fluids*, vol. 53, no. 5, pp. 1203–1220, 2012.

[63] F. Lusseyran, F. Gueniat, J. Basley, C. L. Douay, L. R. Pastur, T. M. Faure, and P. J. Schmid, “Flow coherent structures and frequency signature: Application of the dynamic modes decomposition to open cavity flow,” *J. Phys. Conf. Ser.*, vol. 318, p. 042036, 2011.

[64] D. Duke, J. Soria, and D. Honnery, “An error analysis of the dynamic mode decomposition.” *Exp. Fluids*, vol. 52, pp. 529–542, 2012.

[65] J. H. Tu, D. M. Luchtenburg, and C. W. Rowley, “On dynamic mode decomposition: Theory and Applications,” *J. Comput. Dyn.*, vol. 1, pp. 391–421, 2014.

[66] K. K. Chen, J. H. Tu, and C. W. Rowley, “Variants of dynamic mode decomposition: Connections between Koopman and Fourier analyses.” *J. Nonlinear Sci.*, vol. 22, no. 6, pp. 897–915, 2012.

[67] I. Mezić, “Analysis of fluid flows via spectral properties of the Koopman operator.” *Ann. Rev. Fluid Mech.*, vol. 45, pp. 357–378, 2013.

[68] S. Bagheri, “Koopman-mode decomposition of the cylinder wake.” *J. Fluid Mech.*, vol. 726, pp. 596–623, 2013.

[69] P. J. Goulart, A. Wynn, and D. S. Pearson, “Optimal mode decomposition for high dimensional systems.” *In Proceedings of the 51st IEEE Conference on Decision and Control. Maui, Hawaii.*, vol. 4965-4970, 2012.

[70] A. Wynn, D. S. Pearson, B. Ganapathisubramani, and P. J. Goulart, “Optimal mode decomposition for unsteady flows.” *J. Fluid Mech.*, vol. 733, pp. 473–503, 2013.

[71] M. R. Jovanović, P. J. Schmid, and J. W. Nichols, “Sparsity-promoting dynamic mode decomposition.” *Phys. Fluids*, vol. 26, p. 024103, 2014.

[72] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, “A data-driven approximation of the Koopman operator: Extending dynamic mode decomposition.” *J. Nonlinear Sci.*, vol. 25, no. 6, pp. 1307–1346, 2015.

[73] J. L. Proctor, S. L. Brunton, and J. N. Kutz, “Dynamic mode decomposition with control.” *SIAM J. Appl. Dyn. Syst.*, vol. 15, no. 1, pp. 142–161, 2016.

[74] T. Katayama, *Subspace Methods for System Identification*. London: Springer-Verlag, 2005.

[75] J. Qin, “An overview of subspace identification.” *Comput. Chem. Eng.*, vol. 30, pp. 1502–1513, 2006.

[76] P. Stevens, *The Shale Gas Revolution. Developments and Changes*. Chatham House Briefing Paper, August 2012.

[77] M. J. Economides and K. G. Nolte, *Reservoir stimulation*. Chichester: Wiley,, 2000.

[78] M. J. Economides, R. E. Oligney, and P. Valko, *Unified fracture design: bridging the gap between theory and practice*. Orsa Press, 2002.

[79] S. Bhattacharya, M. Nikolaou, and M. J. Economides, “Unified fracture design for very low permeability reservoirs,” *J. Nat. Gas Sci. Eng.*, vol. 9, pp. 184–195, 2012.

[80] K. G. Nolte, “Determination of proppant and fluid schedules from fracturing-pressure decline,” *SPE. Prod. Eng.*, vol. 1, pp. 255–265, 1986.

[81] H. Gu and J. Desroches, “New pump schedule generator for hydraulic fracturing treatment design.” in *SPE Latin American and Caribbean Petroleum Engineering Conference, Port-of-Spain, Trinidad and Tobago, April 27-30, 2003*.

[82] E. V. Dontsov and A. P. Pierce, “A new technique for proppant schedule design.” *Hydraulic Fracturing J.*, vol. 1, pp. 1–8, 2014.

[83] Q. Gu and K. A. Hoo, “Model-based closed-loop control of the hydraulic fracturing process.” *Ind. Eng. Chem. Res*, vol. 54, no. 5, pp. 1585–1594, 2015.

[84] P. Siddhamshetty, S. Liu, P. Valko, and J. S. Kwon, “Feedback control of proppant bank heights during hydraulic fracturing for enhanced productivity in shale formations,” *AIChE J. In press*, vol. 64, no. 5, pp. 1638–1650, 2017.

[85] P. Siddhamshetty, S. Yang, and J. S. Kwon, “Modeling of hydraulic fracturing and designing of online pumping schedules to achieve uniform proppant concentration in conventional oil reservoirs,” *Comput. Chem. Eng.*, vol. 114, pp. 306–317, 2017.

[86] T. K. Perkins and L. R. Kern, “Widths of hydraulic fractures,” *J. Pet. Technol*, vol. 13, pp. 937–949, 1961.

[87] R. Nordgren, “Propagation of a vertical hydraulic fracture,” *Soc. Petrol. Eng. J*, vol. 12, pp. 306–314, 1972.

[88] A. Bao, E. Gildin, A. Narasingam, and J. S. Kwon, “Data-driven model reduction for coupled flow and geomechanics based on dmd methods,” *Fluids (In Press)*, 2019.

[89] J. Adachi, E. Siebrits, A. Pierce, and J. Desroches, “Computer simulation of hydraulic fractures.” *Int. J. Rock Mech. Min. Sci.*, vol. 44, pp. 739–757, 2007.

[90] A. Daneshy, “Numerical solution of sand transport in hydraulic fracturing.” *J. Pet. Technol.*, vol. 30, pp. 132–140, 1978.

[91] R. Barree and M. Conway, “Experimental and numerical modeling of convective proppant transport.” *J. Pet. Technol.*, vol. 47, pp. 216–222, 1995.

[92] E. J. Novotny, “Proppant transport.” *In Proceedings of the 52nd SPE annual technical conference and exhibition, Denver, CO.*, vol. (SPE 6813), 1977.

[93] E. Liu, S. R. Tod, and X. Y. Li, “The effects of stress and pore fluid pressure on seismic anisotropy in cracked rocks.” *Canadian SEG Recorder*, vol. 27, pp. 92–98, 2002.

[94] S. R. Tod, “The effects of stress and fluid pressure on the anisotropy of interconnected cracks.” *Geophys. J. Int.*, vol. 149, pp. 149–156, 2002.

[95] S. Maultzsch, M. Chapman, E. Liu, and X. Y. Li, “Modelling frequency dependent seismic anisotropy in fluid-saturated rock with aligned fractures: implication of fracture size estimation from anisotropic measurements.” *Geophys. Prosp.*, vol. 51, pp. 381–392, 2003.

[96] E. Liu, J. H. Queen, X. Y. Li, M. Chapman, S. Maultzsch, H. B. Lynn, and E. M. Chesnokov, “Observation and analysis of frequency-dependent seismic anisotropy from a multicomponent vsp.” *J. Appl. Geophys.*, vol. 54, pp. 319–333, 2003.

[97] R. W. Zimmerman and G. S. Bodvarsson, “Hydraulic conductivity of rock fractures.” *Transp. Porous Media*, vol. 23, pp. 1–30, 1996.

[98] A. P. Oron and B. Berkowitz, “Flow in rock fractures: the local cubic law assumption reexamined.” *Water Resour. Res.*, vol. 34, pp. 2811–2825, 1998.

[99] C. E. Renshaw, “On the relationship between mechanical and hydraulic apertures in rough-walled fractures.” *J. Geophys. Res.*, vol. 100, pp. 24 629–24 636, 1995.

[100] S. Sisavath, A. Al-Yaaruby, C. C. Pain, and R. W. Zimmerman, “A simple model for deviations from the cubic law for a fracture undergoing dilation or closure.” *Pure Appl. Geophys.*, vol. 160, pp. 1009–1022, 2003.

[101] Y. Méheust and J. Schmittbuhl, “Scale effects related to flow in rough fractures.” *Pure Appl. Geophys.*, vol. 160, pp. 1023–1050, 2003.

[102] R. W. Zimmerman and I. Main, *Mechanics of Fluid-Saturated Rocks*. London: Elsevier Academic Press, 2004, ch. Hydromechanical behaviour of fractured rocks, pp. 361–419.

[103] A. Armaou and P. D. Christofides, “Finite-dimensional control of nonlinear parabolic pde systems with time-dependent spatial domains using empirical eigenfunctions.” *Int. J. Appl. Meth. Comput. Sci*, vol. 11, no. 2, pp. 287–317, 2001.

[104] W. D. Murray and F. L. Landis, “Numerical and machine solutions of transient heat conduction problems involving melting or freezing.” *J. Heat Transfer*, p. 106, 1959.

[105] H. R. Eyres, D. R. Hartree, J. Inghem, R. Jackson, R. J. Sargent, and J. B. Wastaff, “The calculation of variable heat flow in solids.” *Phil. Trans. Royal Soc*, p. 240, 1948.

[106] W. W. Yuen and A. M. Kleinman, “Application of a variable time step finite difference method for the one-dimensional melting problem including the effect of subcooling.” *AIChE. J*, vol. 26, no. 5, pp. 828–832, 1980.

[107] A. Narasingam, P. Siddhamshetty, and J. S. Kwon, "Temporal clustering for order reduction of nonlinear parabolic PDE systems with time-dependent spatial domains: Application to a hydraulic fracturing process," *AIChE J. (in press, DOI: 10.1002/aic.15733)*, 2017.

[108] H. M. Bücker, B. Pollul, and A. Rasch, "On CFL evolution strategies for implicit upwind methods in linearized Euler equations." *Int. J. Numer. Methods Fluids*, vol. 59, pp. 1–18, 2009.

[109] A. Narasingam and J. Kwon, "Application of koopman operator for modell-based control of fracture propagation and proppant transport in hydraulic fracturing operation," *Journal of Process Control*, vol. 91, pp. 25–36, 2020.

[110] N. R. Warpinski, B. P. Engler, C. J. Young, R. Peterson, P. T. Branagan, J. E. Fix, and E. James, "Microseismic mapping of hydraulic fractures using multi-level wireline receivers," in *SPE Annual Technical Conference and Exhibition, Dallas, Texas, October 22-25, 1995*.

[111] J. T. Rutledge and W. S. Phillips, "Hydraulic stimulation of natural fractures as revealed by induced microearthquakes, carthage cotton valley gas field, east texas." *Geophysics*, vol. 68, no. 2, pp. 441–452, 2003.

[112] S. C. Maxwell, J. Rutledge, R. Jones, and M. Fehler, "Petroleum reservoir characterization using downhole microseismic monitoring." *Geophysics.*, vol. 75, no. 5, pp. 129–137, 2010.

[113] M. Dihlmann, M. Drohmann, and B. Haasdonk, "Model reduction of parametrized evolution problems using the reduced basis method with adaptive time partitioning." *Proc. of ADMOS*, p. 64, 2011.

[114] J. S. R. Anttonen, "Techniques for reduced order modeling of aeroelastic structures with deforming grids." Ph.D. dissertation, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio., 2001.

[115] D. Amsallem, Z. M. J., and C. Farhat, "Nonlinear model order reduction based on local reduced-order bases." *Int. J. Numer. Methods Eng*, vol. 92, pp. 891–916, 2012.

[116] M. P. Tan, J. R. Broach, and C. A. Floudas, “A novel clustering approach and prediction of optimal number of clusters: global optimum search with enhanced positioning.” *J. Glob. Optim*, vol. 39, no. 3, pp. 323–346, 2007.

[117] A. Armaou and P. D. Christofides, “Computation of empirical eigenfunctions and order reduction nonlinear parabolic pde systems with time-dependent spatial domains.” *Nonlinear Anal. Theory Methods Appl*, vol. 47, no. 4, pp. 2869–2874, 2001.

[118] C. A. Floudas, *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*. New York: Oxford University Press, 1995.

[119] C. A. Floudas, A. Aggarwal, and A. R. Ciric, “Global optimum search for non convex nlp and minlp problems.” *Comput. Chem. Eng*, vol. 13, no. 10, pp. 1117–1132, 1989.

[120] Y. Jung, H. Park, and B. L. Du, D.and Drake, “A decision criterion for the optimal number of clusters in hierarchical clustering.” *J. Glob. Optim*, vol. 25, pp. 91–111, 2003.

[121] G. Kerschen, J. C. Golinval, A. F. Vakakis, and L. A. Bergman, “The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: An overview.” *Nonlinear Dyn*, vol. 41, pp. 147–169, 2005.

[122] A. Armaou and P. D. Christofides, “Nonlinear feedback control of parabolic partial differential equation systems with time-dependent spatial domains,” *J. Math. Anal. Appl*, vol. 239, no. 1, pp. 124–157, 1999.

[123] P. D. Christofides, *Nonlinear and Robust Control of PDE Systems: Methods and Applications to Transport-Reaction Processes*. New York: Birkhäuser, 2001.

[124] A. Brooke, D. Kendrick, and A. Meeraus, *GAMS: A User’s guide.* San Francisco: The Scientific Press,, 1988.

[125] G. E. I. Paules and C. A. Floudas, “Apros: Algorithmic development methodology for discrete-continuous optimization problems.” *Oper. Res. J*, vol. 37, pp. 902–915, 1989.

[126] C. Eckart and G. Young, "The approximation of one matrix by another of lower rank." *Psychometrika*, vol. 1, no. 3, pp. 211–218, 1936.

[127] L. Mirsky, "Symmetric gauge functions and unitarily invariant norms." *Q. J. Math. Oxford*, vol. 11, no. 11, pp. 50–59, 1960.

[128] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions." *Numer. Math.*, vol. 14, pp. 403–420, 1970.

[129] A. Narasingam and J. S. Kwon, "Development of local dynamic mode decomposition with control: Application to model predictive control of hydraulic fracturing," *Comput. Chem. Eng.*, vol. 106, pp. 501–511, 2017.

[130] S. Yang, P. Siddhamshetty, and J. S. Kwon, "Optimal pumping schedule design to achieve a uniform proppant concentration level in hydraulic fracturing." *Comput. Chem. Eng.*, vol. 101, pp. 138–147, 2017.

[131] J. L. Daniels, G. A. Waters, J. H. Le Calvez, D. Bentley, and J. T. Lassek, "Contacting more of the barnett shale through an integration of real-time microseismic monitoring, petrophysics, and hydraulic fracture design," in *SPE Annual Technical Conference and Exhibition, Anaheim, California, November 11-14, 2007*, no. SPE-110562-MS.

[132] G. E. King, "Thirty years of gas shale fracturing: What have we learned?" in *SPE Annual Technical Conference and Exhibition, Florence, Italy, September 19-22, 2010*, no. SPE-133456-MS.

[133] A. Narasingam, P. Siddhamshetty, and J. S. Kwon, "Handling spatial heterogeneity in reservoir parameters using proper orthogonal decomposition based ensemble kalman filter for model-based feedback control of hydraulic fracturing," *Ind. Eng. Chem. Res*, vol. 57, no. 11, pp. 3977–3989, 2018.

[134] G. Burgers, P. J. van Leeuwen, and G. Evensen, "Analyisis scheme in the ensemble kalman filter." *Mon. Weather Rev.*, vol. 126, pp. 1719–1724, 1998.

[135] G. Li, M. Han, R. Banerjee, and A. C. Reynolds, “Integration of well test pressure data into heterogeneous geological reservoir models.” in *SPE Annual Technical Conference and Exhibition, New Orleans, Louisiana, October 4-7, 2009*, vol. 13, no. 3.

[136] G. Gao, M. Zafari, and A. C. Reynolds, “Quantifying the uncertainty for the PUNQ-S3 problem in a bayesian setting with RML and EnKF.” *SPE J.*, vol. 11, no. 4, pp. 506–515, 2006.

[137] E. Liu, “Effects of fracture aperture and roughness on hydraulic and mechanical properties of rocks: implication of seismic characterization of fractured reservoirs.” *J. Geophys. Eng.*, vol. 2, pp. 38–47, 2005.

[138] G. Evensen, “The ensemble kalman filter: theoretical formulation and practical implementation,” *Ocean Dynamics*, vol. 53, no. 4, pp. 343–367, 2003.

[139] J. A. Skjervheim, G. Evensen, S. I. Aanonsen, B. O. Ruud, and T. A. Johansen, “Incorporating 4D seismic data in reservoir simulation models using ensemble Kalman filter,” *SPE J.*, vol. 12, pp. 282–292, 2007.

[140] J. Rafiee and A. C. Reynolds, “Theoretical and efficient practical procedures for the generation of inflation factors for ES-MDA.” *Inverse Problems*, vol. 33, no. 11, p. 115003, 2017.

[141] K. Thulin, G. Li, S. I. Aanonsen, and A. C. Reynolds, “Estimation of initial fluid contacts by assimilation of production data with EnKF,” in *SPE Annual Technical Conference and Exhibition, Anaheim, California, November 11-14, 2007*.

[142] Y. Gu and D. S. Oliver, “An iterative ensemble Kalman filter for multiphase fluid flow data assimilation,” *SPE J.*, vol. 12, no. 4, pp. 438–446, 2007.

[143] M. Zafari and A. C. Reynolds, “Assessing the uncertainty in reservoir description and performance predictions with the ensemble kalman filter,” *SPE J.*, vol. 12, pp. 382–391, 2007.

[144] Y. Wang, G. Li, and A. C. Reynolds, “Estimation of depths of fluid contacts by history matching using iterative ensemble-Kalman smoothers.” *SPE J.*, vol. 15, pp. 509–525, 2010.

[145] R. Mohr and I. Mezić., “Construction of eigenfunctions for scalar-type operators via laplace averages with connections to koopman operator,” *arXiv preprint , arXiv:1403.6559*, 2014.

[146] M. O. Williams, M. S. Hemati, S. T. Dawson, and I. G. Kevrekidis, “Extending data-driven koopman analysis to actuated systems,” *IFAC-PapersOnLine*, vol. 49, no. 8, pp. 704–709, 2016.

[147] E. Kaiser, J. N. Kutz, and B. S. L, “Data-driven discovery of koopman eigenfunctions for control,” *arXiv preprint*, *arXiv:1707.01146*, 2017.

[148] J. L. Proctor, S. L. Brunton, and J. N. Kutz, “Generalizing koopman theory to allow for inputs and control,” *SIAM J. Appl. Dyn. Syst.*, vol. 17, no. 1, pp. 909–930, 2018.

[149] M. Korda and I. Mezić, “On convergence of extended dynamic mode decomposition to the koopman operator,” *J, Nonlin. Sci*, vol. 28, no. 2, pp. 687–710, 2018.

[150] Q. Li, F. Dietrich, E. M. Bollt, and I. G. Kevrekidis, “Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the koopman operator,” *Chaos*, vol. 27, p. 103111, 2017.

[151] B. Lusch, J. N. Kutz, and S. L. Brunton, “Deep learning for universal linear embeddings of nonlinear dynamics,” *Nat. Commun*, vol. 9, no. 1, p. 4950, 2018.

[152] A. Narasingam and J. S. Kwon, “Koopman lyapunov-based model predictive control of nonlinear chemical process systems,” *AIChE J.*, vol. 65, no. 11, p. e16743, 2019.

[153] M. Ellis, H. Durand, and P. Christofides, “A tutorial review of economic model predictive control methods.” *J. Process Control*, vol. 24, no. 8, pp. 1156–1178, 2014.

[154] B. Huang, X. Ma, and M. Vaidya, “Feedback stabilization using koopman operator,” in *2018 IEEE Conference on Decision and Control (CDC)*, Miami Beach, FL, Dec 17-19 2018, pp. 6434–6439.

[155] I. Knowles and R. J. Renka, “Methods for numerical differentiation of noisy data.” *Electron. J. Differ. Eq.*, vol. 212012, pp. 235–246, 2014.

[156] A. E. Hoerl and R. W. Kennard, “Ridge regression: Biased estimation for nonorthogonal problems.” *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[157] S. Rudy, S. Brunton, J. Proctor, and J. Kutz, “Data-driven discovery of partial differential equations.” *Science Advances*, vol. 3, no. 4, p. e1602614, 2017.

[158] A. Narasingam and J. S. Kwon, “Data-driven identification of interpretable reduced-order models using sparse regression,” *Comput. Chem. Eng*, vol. 119, pp. 101–111, 2018.

[159] S. L. Brunton, B. W. Brunton, J. L. Proctor, and J. N. Kutz, “Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control,” *PloS one*, vol. 11, no. 2, p. e0150171, 2016.

[160] P. Mhaskar, N. H. El-Farra, and P. D. Christofides, “Predictive control of switched nonlinear systems with scheduled mode transitions,” *IEEE Trans. Automat. Control*, vol. 50, no. 11, pp. 1670–1680, 2005.

[161] P. Mhaskar, N. H. El-Farra, and P. D. Christofides, “Stabilization of nonlinear systems with state and control constraints using lyapunov-based predictive control,” *Syst. Control Lett*, vol. 55, no. 8, pp. 650–659, 2006.

[162] A. Alanqar, M. Ellis, and P. D. Christofides, “Economic model predictive control of nonlinear process systems using empirical models,” *AIChE J*, vol. 61, no. 3, pp. 816–830, 2015.

[163] E. D. Sontag, “A ‘universal’ construction of artstein’s theorem on non-linear stabilization,” *Syst. Control Lett*, vol. 13, pp. 117–123, 1989.

[164] D. Goswami and D. A. Paley, “Global bilinearization and controllability of control-affine nonlinear systems: A koopman spectral approach,” in *IEEE 56th Annual Conference on Decision and Control*, Melbourne, Australia, Dec 2017, pp. 6107–6112.

[165] Y. Lin and E. D. Sontag, “A universal formula for stabilization with bounded controls,” *Syst. Control Lett*, vol. 16, no. 6, pp. 393–397, 1991.

[166] D. A. Allan, C. N. Bates, M. J. Risbeck, and J. B. Rawlings, "On the inherent robustness of optimal and suboptimal nonlinear mpc," *Systems & Control Letters*, vol. 106, pp. 68–78, 2017.

[167] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model predictive control: theory, computation, and design*, 2nd ed. Madison, WI: Nob Hill Publishing, 2017.

[168] N. Takeishi, Y. Kawahara, and T. Yairi, "Learning koopman invariant subspaces for dynamic mode decomposition," in *Advances in Neural Information Processing Systems*, 2017, pp. 1130–1140.

[169] E. Yeung, S. Kundu, and N. Hodas, "Learning deep neural network representations for koopman operators of nonlinear dynamical systems," in *2019 American Control Conference (ACC)*, Philadelphia, USA, July 2019, pp. 4832–4839.

[170] S. Peitz and S. Klus, "Koopman operator-based model reduction for switched-system control of pdes." *Automatica*, vol. 106, pp. 184–191, 2019.