Real-Time Reduced Order Modeling Using Time Dependent Bases: Applications and Advances

by

Michael Donello

B.S. Mechanical Engineering, University of Pittsburgh, 2018

Submitted to the Graduate Faculty of

the Swanson School of Engineering in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

Real-Time Reduced Order Modeling Using Time Dependent Bases: Applications and Advances

Michael Donello, PhD

University of Pittsburgh,

In the first part, we present a reduced order modeling (ROM) strategy for computing finite time sensitivities in evolutionary systems, called the *forced optimally time dependent* (f-OTD) decomposition. The approach is used for low-rank approximation of sensitivities governed by forced linear differential equations. The sensitivity fields are approximated using time dependent bases (TDB), that are evolved via closed form evolution equations. We demonstrate the accuracy of f-OTD for computing sensitivities in a variety evolutionary systems.

In the second part, we extend f-OTD to approximate *nonlinear sensitivities*, which we call NL-fOTD. Unlike solving a *linearized* system that assumes infinitesimal perturbations around a base trajectory, this framework allows for *finite* perturbations as *nonlinear* interactions are considered. Similar to f-OTD, we solve low-rank evolution equations that leverage TDB by extracting correlations between sensitivities on-the-fly. The resulting equations are Jacobian-free and leverage the same nonlinear solver that is used to compute the evolution of the base state. For nonlinear sensitivities with arbitrarily time dependent base state, we demonstrate that low-rank structure often exists, and can be accurately extracted in real time directly from the governing equations.

In the third part, we address some of the outstanding challenges of TDB based ROMs, like f-OTD and NL-fOTD. In particular, TDB ROMs are (1) inefficient for solving general nonlinear equations, (2) intrusive to implement, and (3) ill-conditioned in the presence of small singular values. Since these challenges can arise regardless of the governing equations, we develop a new TDB ROM method for solving general nonlinear matrix differential equations (MDEs) that is computationally efficient, minimally intrusive, robust in the presence of small singular values, and rank-adaptive. The new method is based on a sparse sampling strategy for the low-rank approximation of a time discrete MDE. Guided by the discrete empirical interpolation method (DEIM), a low-rank approximation is computed at each iteration of the time stepping scheme. The new method is coined TDB-CUR, since the resulting low-rank approximation is equivalent to a CUR matrix factorization.

Table of Contents

List of Tables

List of Figures

Preface

First, I would like to thank Dr. Hessam Babaee for his support and mentorship over the past five years. Your guidance and reassurance has helped shape me into the person and scientist that I am today. Thank you to my committee: Dr. Peyman Givi, Dr. Inanc Senocak, Dr. Mark Carpenter, and Dr. Amir Alavi for your advice, criticism and feedback over the years. I would also like to thank Dr. Joe Derlaga for his support and intellectual insight throughout the process. To all my labmates, both past and present, thank you for your support over the years. Prerna, Shaghayegh, Hossein, and Grishma: thank you for the laughs and the much needed distractions from reduced order modeling.

To my family and friends, thank you for your support and patience over the past five years, despite my ups and downs. My deepest gratitude goes to my parents, Marc and Cathryn, my brother Matthew, and my three best friends Julia, Callie, and Bella. Without them, none of this would be possible.

1.0 Introduction

1.1 Motivation

Quantifying input-output relationships for systems governed by high dimensional partial differential equations (PDEs) is a central goal amongst a diverse set of fields and applications. For many practical problems of interest, PDEs can depend on a large number of parameters including initial and boundary conditions, geometry, and material properties [18]. While the primary means of investigating these input-output relationships is via numerical simulations, solving these high dimensional PDEs for a large range of parameter values is often cost prohibitive.

One such example is the uncertainty propagation of random parameters into PDEs, which requires solving the PDEs for a large number of random realizations [81, 49]. Discretization of this problem can be formulated as a matrix differential equation (MDE) in the form of $\mathrm{d}\mathbf{V}/\mathrm{d}t = \mathcal{F}(\mathbf{V})$, where $\mathbf{V} \in \mathbb{R}^{n\times s}$ is the solution matrix and $\mathcal{F}(\mathbf{V}) \in \mathbb{R}^{n\times s}$ is obtained by discretizing the PDE in all dimensions except time. Here, the rows of the matrix are obtained by discretizing the PDE in the physical domain and the columns of the matrix are samples of the discretized equation for a particular choice of random parameters. For high-dimensional PDEs subject to high-dimensional random parameters, the resulting MDEs can be massive. For example, uncertainty quantification (UQ) of a 3D time-dependent fluid flow typically requires solving an MDE with $n \sim \mathcal{O}(10^6) - \mathcal{O}(10^9)$ grid points (rows) and $s \sim \mathcal{O}(10^4) - \mathcal{O}(10^7)$ random samples (columns). Therefore, the solution to these massive MDEs is cost prohibitive due to the floating point operations (flops), memory, and storage requirements.

Other problems can also be cast as MDEs. For example, sensitivity analysis (SA), which plays an integral role in gradient-based optimization [39, 42], optimal control [20], grid adaptivity [30], parameter identification [33], skeletal model reduction [70, 59], and stability analysis [61]. For these types of problems, the system output is the solution to an ODE/PDE, which implicitly depends on a set of input parameters. The goal of an SA is to quantify the change in output relative to a change in the input, and is commonly computed via finite difference (FD) or by directly solving a forward sensitivity equation (SE). The computational cost of using FD or an SE scales linearly with the number of parameters – making them impracticable when sensitivities with respect to a large number of parameters are needed. To alleviate this computational cost, an adjoint equation (AE) can be solved for computing the gradient of an objective function that depends on the sensitivity. While the computational cost of solving an AE is independent of the number of parameters, the *forward-backward* workflow of the adjoint solver is problematic for two main reasons: (1) it is not adequate for problems where real-time sensitivities are required, e.g. grid adaptivity for time-dependent problems [30] and (2) for high dimensional dynamical systems, the imposed I/O operations in the AE workflow lead to insurmountable limitations, especially in high performance computing architectures [1]. Motivated by these challenges, there is a critical need for robust and efficient algorithms that can incorporate UQ and SA into computer based models at reasonable computational cost [84].

1.2 Reduced Order Modeling

For many UQ and SA applications, the resulting solution matrix, $\mathbf{V}(t)$, is instantaneously low-rank. As a result, there has been a growing interest to reduce the dimension of these systems by building reduced order models (ROMs) that describe the evolution of the dynamics in a reduced state space of rank $r \ll \min\{n, s\}$. The importance of ROMs is evident – they can be used instead of the full-order model (FOM) for in-loop applications, where many forward model evaluations are required, thereby reducing the computational cost for tasks like UQ and SA. However, the most well-known techniques for building ROMs, e.g. the proper orthogonal decomposition (POD) [19, 29] and dynamic mode decomposition (DMD) [53], are based on extracting a *static*, i.e. time-independent, low-rank basis from observations of the full-dimensional dynamical system. Therefore, the DMD and POD extract correlated structures in a *time averaged* sense, and are limited in application to systems that are stationary or exhibit periodic or quasi-periodic behavior in time. For the highly transient systems targeted in this work, the DMD and POD would require a large number of modes to resolve the system with an acceptable degree of accuracy, defeating the purpose of the reduced order modeling task. Another challenge for static basis ROM strategies is that they incur the cost of generating high-fidelity simulation data in order to extract a suitable basis in the first place. In theory, this upfront cost is paid once to generate a basis that is valid over a large range of operating conditions. However, in practice, this basis is limited in scope and not well suited for highly transient problems, especially if extrapolation is required. See [18] for a recent survey of projection based parametric model reduction.

1.3 Time Dependent Bases

In more recent years, time dependent basis (TDB) ROMs have emerged as a potential solution to alleviate the computational burden of solving the massive MDEs required for UQ and SA [49, 68, 81, 26, 66, 9, 7, 72, 31, 69, 52]. Unlike the static bases used in DMD and POD, TDBs are evolved with the dynamics of the system, making them amenable to the low-rank approximation of highly transient systems. For these systems, a low-rank approximation via TDBs extracts instantaneous correlated structures from the column and row spaces of the solution matrix. A ROM is then constructed by projecting the FOM equations onto the low-rank column and row TDBs. As such, TDB-ROMs can be described as an *on the fly* model compression that extracts instantaneous correlated structures via low-rank evolution equations. An important implication of TDB ROMs is that high-fidelity data generation is not required to extract the basis in an offline training stage. Rather, the basis is updated in real-time, eliminating the offline stage from the TDB ROM workflow.

Low-rank approximation based on TDBs was first introduced in the quantum chemistry field to solve the Schrödinger equation [16], where it is commonly known as the multiconfiguration time-dependent Hartree (MCTDH) method. The MCTDH methodology was later presented for generic matrix differential equations in [49] and is referred to as dynamical low-rank approximation (DLRA). Various TDB ROM schemes have also been developed to solve stochastic partial differential equations (SPDEs). Dynamically orthogonal (DO) decomposition [81], bi-orthogonal (BO) decomposition [26], dual dynamically orthogonal (DDO) decomposition [65], and dynamically bi-orthogonal decomposition (DBO) [72], are all TDB-based low-rank approximation techniques for solving SPDEs. Although these decompositions have different forms and constraints, they are all equivalent, i.e., they produce identical low-rank matrices [27, 72], and their differences lie only in their numerical performance.

1.4 Challenges of Time Dependent Bases

Despite the potential of TDB ROMs to significantly reduce the computational cost of solving massive MDEs, there are still a number of outstanding challenges for most practical problems of interest. We summarize three key challenges below:

(C1) **Computational efficiency:** For specific classes of equations (e.g. homogeneous linear and quadratic nonlinear), rank-r TDB ROMs can be solved efficiently using explicit time integration with operations that scale with $\mathcal{O}(nr)$ and $\mathcal{O}(sr)$ for linear MDEs or scale with $\mathcal{O}(nr^2)$ and $\mathcal{O}(sr^2)$ for quadratic MDEs. However, this computational efficiency is lost for general nonlinearities, requiring operations that scale with the size of the FOM, i.e., $\mathcal{O}(ns)$. While this challenge has been addressed in the context of static basis ROMs like POD, see e.g. missing point estimation (MPE) [5], empirical interpolation method (EIM) [13], discrete empirical interpolation method (DEIM) [25], Gappy POD [34], and Gauss Newton with approximated tensors (GNAT) [23], these techniques are still in their infancy for TDB ROMs.

(C2) **Intrusiveness:** Even in the special cases of homogeneous linear and quadratic nonlinear equations, efficient implementation of TDB ROM evolution equations is an intrusive process [67, Appendix B]. This involves replacing the low-rank approximation in the FOM, projecting the resulting equation onto the tangent manifold, and obtaining low-rank matrices for each term on the right-hand side. The process requires significant effort to derive, implement, and debug the code. This poses a major obstacle for most practitioners, creating a significant barrier to adopting the methodology.

(C3) **Ill-conditioning:** The TDB ROM evolution equations become numerically unstable when the singular values of the low-rank approximation become very small. This is particularly problematic because it is often necessary to retain very small singular values in order to have an accurate approximation. Small singular values lead to ill-conditioned matrices that require inversion in all variations of TDB ROM evolution equations [49, 81, 26, 65, 72], resulting in restrictive time step limitations for numerical integration and error amplification.

Although some of these challenges have been tackled, there is currently no methodology that can address all three. To address the issue of ill-conditioning, a projector-splitting time integration was proposed [60], in which arbitrarily small singular values can be retained. However, this scheme includes a backward time integration substep, which is an unstable substep for dissipative problems. To address this issue, an unconventional robust integrator was recently proposed [24] which retains the robustness with respect to small singular values while avoiding the unstable backward step. The authors also presented an elegant rank adaptive strategy, where the rank of the approximation changes over time to maintain a desired level of accuracy. Despite these advantages, this scheme is first-order in time [24, Theorem 4]. In [9], a pseudo-inverse methodology was presented as a remedy to maintain a well-conditioned system. However, in this approach, it is difficult to determine what singular value threshold must be used. Another projection method was presented in [47] that retains robustness with respect to small singular values and can be extended to high-order explicit time discretizations.

Although the three time-integration schemes presented in [60, 24, 47] and the pseudo-inverse methodology presented in [9] can retain $\mathcal{O}(n+s)$ cost for linear and quadratic MDEs, this speedup comes at the expense of a highly intrusive implementation. However, for generic nonlinear MDEs, an intrusive implementation is not possible, and the computational cost of solving the TDB ROMs using methods presented in [60, 24, 47, 9] scales with $\mathcal{O}(ns)$, which is the same as the cost of solving the FOM. In more recent work, a sparse interpolation algorithm was presented for solving the TDB ROM evolution equations with a computational complexity that scales with $\mathcal{O}(n+s)$ for generic nonlinear SPDEs [67]. However, this methodology still lacks robustness when the singular values become small, as it requires the inversion of the matrix of singular values.

1.5 Objectives and Contributions

This dissertation is comprised of three self contained works, each contributing to the application and improvement of reduced order modeling using time dependent bases. The objectives and contributions of each chapter are summarized as follows:

- In Chapter 2, the objective is to investigate the application of TDB ROMs for computing sensitivities in evolutionary systems. To this end, we present a variational principle and derive low-rank evolution equations for solving forced linear sensitivity equations under the assumption of infinitesimal perturbations. The method is called f-OTD, and the resulting equations are solved forward in time. For a number of systems, we show that low-rank structure exists, and can be accurately extracted on-the-fly by solving low-rank evolution equations forward in time: (1) sensitivity with respect to model parameters in the Rössler system (2) sensitivity with respect to an infinite dimensional forcing parameter in the chaotic Kuramoto-Sivashinsky equation and (3) sensitivity with respect to reaction parameters for species transport in a turbulent reacting flow. In these examples, we perform error and convergence analyses with satisfactory results. We also demonstrate how the f-OTD components directly represent sensitivity information in a simplified and interpretable manner. Specifically, we demonstrate the utility of f-OTD for parameter identification in a turbulent reacting flow. The f-OTD method has also been applied in independent works for skeletal model reduction of chemical kinetics [70, 59].
- In Chapter 4, we present a new methodology based on a CUR factorization of low-rank matrices that addresses the above challenges of TDB ROMs, i.e., (1) the computational cost of the methodology scales with $\mathcal{O}(n+s)$ for generic nonlinear SPDEs both in terms of flops and memory costs, (2) it lends itself to simple implementation in existing codes, and (3) the time-integration is robust in the presence of small singular values, and high-order explicit time integration can be used. The main elements of the presented methodology are a time-discrete variational principle for minimization of the residual due to low-

rank approximation error, and a CUR factorization based on strategic row and column sampling of the time discrete MDE. We also provide a priori error analysis and show how the error can be controlled via a rank-adaptive strategy. The method is applied to solve the stochastic Burgers equation to demonstrate the accuracy and efficiency of the method. Furthermore, we provide a MATLAB script to illustrate the implementation of the method.

2.0 Low-Rank Approximation of Linear Sensitivity

In this chapter, we present a reduced order modeling strategy for computing finite time sensitivities in evolutionary systems, called the *forced optimally time dependent* (f-OTD) decomposition. The approach is used for low-rank approximation of sensitivities governed by forced linear differential equations. The sensitivity fields are approximated using time dependent bases, that are evolved via closed form evolution equations. We demonstrate the accuracy of f-OTD for computing sensitivities in a variety evolutionary systems. The following contains material from the article, "Computing Sensitivities in Evolutionary Systems: A Real-Time Reduced Order Modeling Strategy", published in the SIAM Journal on Scientific Computing [31].

2.1 Notation and Definitions

We denote $\mathbf{u}(x,t)$ to be a time dependent field variable. We denote the spatial domain as $D \subset \mathbb{R}^m$, where $m = 1$, 2, or 3. The spatial coordinate is denoted by $x \in D$ and the function is evaluated at time t. We introduce a quasimatrix notation to represent a set of functions in matrix form, and denote the quasimatrix $\mathbf{U}(x,t) \in \mathbb{R}^{\infty \times r}$ as [15]:

$$\mathbf{U}(x,t) = \Big[\mathbf{u}_1(x,t) \;\Big|\; \mathbf{u}_2(x,t) \;\Big|\; \dots \;\Big|\; \mathbf{u}_d(x,t)\Big]_{\infty \times r},$$

where the first dimension is infinite and represents the continuous state space contained by D and the second dimension is discrete. Similarly, we use the term quasitensor for tensors whose first dimension is infinity. For example, $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{\infty \times r_1 \times r_2}$ is a third-order quasitensor. We define the column-wise inner product of two quasimatrices $\mathbf{U}(x,t) \in \mathbb{R}^{\infty \times r}$ and $\mathbf{V}(x,t) \in \mathbb{R}^{\infty \times d}$ as

$$\mathbf{S}(t) = \langle \mathbf{U}(x,t), \mathbf{V}(x,t) \rangle,$$

where $\mathbf{S}(t) \in \mathbb{R}^{r \times d}$ is a matrix with components

$$S_{ij}(t) = \int_D \mathbf{u}_i(x,t)\mathbf{v}_j(x,t)\, dx,$$

where $\mathbf{u}_i(x,t)$ and $\mathbf{v}_j(x,t)$ are the ith and jth columns of $\mathbf{U}(x,t)$ and $\mathbf{V}(x,t)$, respectively. The discrete analogue of this operation is the matrix multiplication, $\mathbf{U}(t)^T\mathbf{W}\mathbf{V}(t)$, where $\mathbf{U}(t) \in \mathbb{R}^{n\times r}$ and $\mathbf{V}(t) \in \mathbb{R}^{n\times d}$ are space discrete with n grid points and $\mathbf{W} \in \mathbb{R}^{n\times n}$ is a diagonal weight matrix. For the case of single-column quasimatrices $\mathbf{u}(x,t) \in \mathbb{R}^{\infty\times 1}$ and $\mathbf{v}(x,t) \in \mathbb{R}^{\infty\times 1}$, i.e., functions, the above definition reduces to an inner product between two functions, which induces an L_2 norm:

$$\langle \mathbf{u}(x,t), \mathbf{v}(x,t)\rangle = \int_D \mathbf{u}(x,t)\mathbf{v}(x,t)\,dx, \qquad \|\mathbf{u}(x,t)\|_2 = \langle \mathbf{u}(x,t), \mathbf{u}(x,t)\rangle^{\frac{1}{2}}.$$

The Frobenius norm of a quasimatrix is defined as:

$$\left\|\mathbf{U}(x,t)\right\|_F = \sqrt{\text{trace}\langle \mathbf{U}(x,t), \mathbf{U}(x,t)\rangle}.$$

Finally, we define multiplication between a quasimatrix and a vector

$$\mathbf{c}(x,t) = \mathbf{U}(x,t)\mathbf{b}(t),$$

where $\mathbf{b}(t) = (b_1(t), b_2(t), \dots, b_r(t))^T \in \mathbb{R}^{r\times 1}$ is an arbitrary vector and $\mathbf{c}(x,t) \in \mathbb{R}^{\infty\times 1}$ is a function given by $\mathbf{c}(x,t) = b_i(t)\mathbf{u}_i(x,t)$. We use index notation and the repeated index implies summation.

2.2 Mathematical Formulation

We consider the nonlinear PDE for the evolution of $\mathbf{v}(x,t;\boldsymbol{\alpha})$:

$$\frac{\partial \mathbf{v}(x,t;\boldsymbol{\alpha})}{\partial t} = \mathcal{M}\left((\mathbf{v}(x,t;\boldsymbol{\alpha});\boldsymbol{\alpha}\right), \quad t \in [0, T_f] \tag{2.1}$$

where $\mathcal{M}$ is in general a nonlinear differential operator. Our goal is to compute the sensitivity of $\mathbf{v}(x,t;\boldsymbol{\alpha})$ with respect to the design parameters $\boldsymbol{\alpha}$, which can either be infinite-dimensional, i.e., a function $\boldsymbol{\alpha} = \boldsymbol{\alpha}(x,t)$, or finite-dimensional, i.e., a vector $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \dots, \alpha_d)$. For the sake of simplicity in the exposition we consider the finite-dimensional parametric space.

A common approach to computing sensitivities in evolutionary systems is by instantaneous linearization of the dynamical system around a base state. The linear sensitivity equation is obtained by differentiating Equation 2.1 with respect to design parameter α_i, which leads to an evolution equation for the sensitivity of the dynamical system:

$$\frac{\partial \mathbf{v}_i'(x,t)}{\partial t} = \mathcal{L}\left(\mathbf{v}_i'(x,t)\right) + \mathbf{f}_i'(x,t;\boldsymbol{\alpha}), \tag{2.2}$$

where $\mathbf{v}_i' = \partial \mathbf{v}/\partial \alpha_i$ is the sensitivity of $\mathbf{v}(x,t;\boldsymbol{\alpha})$ with respect to α_i, $\mathcal{L}(\sim) = \partial \mathcal{M}/\partial \mathbf{v}(\sim)$ is the linearized operator evaluated at the base state $\mathbf{v}(x,t,;\boldsymbol{\alpha})$ (i.e. the Jacobian once discretized), and $\mathbf{f}_i' = \partial \mathcal{M}/\partial \alpha_i$ is the forcing term. From observation of Equation 2.2, solving for the full system of sensitivities scales with the number of parameters, where a linear PDE must be solved for each parameter of interest. While this might be manageable for a small number of parameters, solving Equation 2.2 will be cost prohibitive as the dimension of the parametric space, d, increases. However, based on our observations, the sensitivities governed by Equation 2.2 tend to be highly correlated at any given time, and thus, have the potential to effectively be approximated by a low rank time-dependent subspace.

2.3 The OTD Decomposition

The idea of approximating linear systems via low-rank time-dependent subspaces is not new. In fact, a new low-dimensional model was recently presented in [11] that can describe transient instabilities in high-dimensional nonlinear dynamical systems via an instantaneous linearization of the governing equations. This approach is based on a TDB known as the optimally time-dependent (OTD) modes. The evolution equations for the OTD modes is obtained by minimizing the functional

$$\mathcal{F}(\dot{\mathbf{u}}_1, \dot{\mathbf{u}}_2, \ldots, \dot{\mathbf{u}}_r) = \sum_{i=1}^{r} \left\| \dot{\mathbf{u}}_i - \mathbf{L}(\mathbf{v}(t),t)\mathbf{u}_i(t) \right\|^2, \tag{2.3}$$

subject to the orthonormality of the OTD modes, i.e. $\mathbf{u}_i^T \mathbf{u}_j = \delta_{ij}$, where $\mathbf{u}_i(t) \in \mathbb{R}^n, i = 1, \ldots, r$ are the OTD modes. In the above functional, $\|\mathbf{u}\|^2 = \mathbf{u}^T\mathbf{u}$ and $\mathbf{L}(\mathbf{v}(t),t) \in \mathbb{R}^{n\times n}$ is the instantaneous linearized operator and $\dot{(\sim)} = \mathrm{d}(\sim)/\mathrm{d}t$. The optimality condition of the above variational principle leads to a closed form evolution equation for the OTD subspace: $\dot{\mathbf{U}} = (\mathbf{I} - \mathbf{U}\mathbf{U}^T)\mathbf{L}\mathbf{U}$, where $\mathbf{U} = [\mathbf{u}_1|\mathbf{u}_2|\ldots|\mathbf{u}_r] \in \mathbb{R}^{n\times r}$ and $\mathbf{I} \in \mathbb{R}^{n\times n}$ is the identity matrix. It was shown later that the OTD subspace converges exponentially fast to the eigendirections of the Cauchy–Green tensor associated with the most intense finite-time instabilities [10]. In this sense, the OTD reduction can be interpreted as a low-rank subspace that approximates the evolution of perturbed initial condition in all directions of the phase space. One of the computational advantages of OTD is that it only requires solving forward equations. Moreover, the computational complexity of solving OTD reduction scales linearly with respect to the number of modes. The OTD reduction has also been used for flow control [21], building precursors for bursting phenomena [35] as well as detection of edge manifolds in infinite-dimensional dynamical systems [17]. Despite its recent success, OTD is not adequate when applied to systems subject to perturbations in a parametric space. These perturbations are governed by Equation 2.2, and in general, the OTD subspace is not an optimal basis for the evolution of $\mathbf{v}'_i$. To this end, we present a new approach based on a time-dependent basis for solving ROMs of time-varying linear systems forced by a high-dimensional function.

2.4 The Forced OTD Decomposition

In this section, we present a real-time reduced order modeling strategy that aims to extract a time-dependent subspace for building sensitivity ROMs. In particular, we present a variational principle, whose first-order optimality conditions lead to evolution equations for a time-dependent subspace and its coefficients, which we call forced OTD (f-OTD). To this end, we estimate the sensitivities using the low-rank decomposition:

$$\mathbf{V}'(x,t) = \mathbf{U}(x,t)\mathbf{Y}(t)^T + \mathbf{E}(x,t), \tag{2.4}$$

where $\mathbf{V}'(x,t) = \big[\mathbf{v}'_1(x,t) \bigm| \mathbf{v}'_2(x,t) \bigm| \ldots \bigm| \mathbf{v}'_d(x,t)\big]_{\infty\times d}$ is a quasimatrix with its i^{th} column corresponding to the sensitivity of α_i, $\mathbf{U}(x,t) = \big[\mathbf{u}_1(x,t) \bigm| \mathbf{u}_2(x,t) \bigm| \ldots \bigm| \mathbf{u}_r(x,t)\big]_{\infty\times r}$ is a quasimatrix representing a rank-r time-dependent orthonormal basis, in which $\langle \mathbf{u}_i(x,t), \mathbf{u}_j(x,t)\rangle =$

δ_{ij}, $\mathbf{Y}(t) = \big[\mathbf{y}_1(t) \;\big|\; \mathbf{y}_2(t) \;\big|\; \dots \;\big|\; \mathbf{y}_r(t)\big]_{d\times r}$ is the coefficient matrix, and $\mathbf{E}(x,t) \in \mathbb{R}^{\infty\times d}$ is the approximation error. The f-OTD decomposition is shown schematically in Figure 1.

We formulate a variational principle with control parameters $\dot{\mathbf{U}}(x,t)$ and $\dot{\mathbf{Y}}(t)$, that seeks to optimally update the subspace $\mathbf{U}(x,t)$ and its coefficients $\mathbf{Y}(t)$ by minimizing the residual of the low-rank approximation of Equation 2.2:

$$\mathcal{F}(\dot{\mathbf{U}}(x,t), \dot{\mathbf{Y}}(t)) = \left\| \frac{\partial(\mathbf{U}(x,t)\mathbf{Y}(t)^T)}{\partial t} - \mathcal{L}\left(\mathbf{U}(x,t)\right)\mathbf{Y}(t)^T - \mathbf{F}'(x,t;\boldsymbol{\alpha}) \right\|_F^2, \tag{2.5}$$

where $\mathbf{F}'(x,t) = \big[\mathbf{f}'_1(x,t) \;\big|\; \mathbf{f}'_2(x,t) \;\big|\; \dots \;\big|\; \mathbf{f}'_d(x,t)\big]_{\infty\times d}$. Taking the time derivative of the orthonormality condition leads to the following constraint for the minimization problem:

$$\langle \dot{\mathbf{u}}_i(x,t), \mathbf{u}_j(x,t)\rangle + \langle \mathbf{u}_i(x,t), \dot{\mathbf{u}}_j(x,t)\rangle = 0. \tag{2.6}$$

We denote $\boldsymbol{\phi}_{ij}(t) = \langle \mathbf{u}_i(x,t), \dot{\mathbf{u}}_j(x,t)\rangle$, in which $\boldsymbol{\Phi}(t) = [\phi_{ij}(t)] \in \mathbb{R}^{r\times r}$. It is easy to see that $\boldsymbol{\Phi}(t)$ must be a skew-symmetric matrix in order to satisfy Equation 2.6, i.e., $\boldsymbol{\phi}_{ji}(t) = -\boldsymbol{\phi}_{ij}(t)$. Incorporating this constraint leads to the following unconstrained optimization problem functional:

$$\begin{aligned}\mathcal{G}(\dot{\mathbf{U}}(x,t), \dot{\mathbf{Y}}(t), \lambda(t)) = &\left\| \frac{\partial(\mathbf{U}(x,t)\mathbf{Y}(t)^T)}{\partial t} - \mathcal{L}\left(\mathbf{U}(x,t)\right)\mathbf{Y}(t)^T - \mathbf{F}'(x,t;\boldsymbol{\alpha}) \right\|_F^2 \\ &+ \sum_{i,j=1}^{r} \lambda_{ij}(t)\big(\langle \mathbf{u}_i(x,t), \dot{\mathbf{u}}_j(x,t)\rangle - \boldsymbol{\phi}_{ij}(t)\big),\end{aligned} \tag{2.7}$$

where $\lambda(t) = [\lambda_{ij}(t)] \in \mathbb{R}^{r\times r}$ are Lagrange multipliers. In the following section, we show that minimizing the above functional leads to close form evolution equations for $\mathbf{U}(x,t)$ and $\mathbf{Y}(t)$.

2.4.1 Optimality Conditions of the Variational Principle

For the sake of brevity, we forgo the explicit written dependencies on x, t, and $\boldsymbol{\alpha}$ in the following derivation. Using index notation, we start by expanding Equation 2.7

$$\begin{aligned}\mathcal{G}(\dot{\mathbf{U}}, \dot{\mathbf{Y}}, \lambda) &= \langle \dot{\mathbf{u}}_i, \dot{\mathbf{u}}_j \rangle \left(\mathbf{y}_i^T \mathbf{y}_j\right) + \langle \mathbf{u}_i, \mathbf{u}_j \rangle \left(\dot{\mathbf{y}}_i^T \dot{\mathbf{y}}_j\right) + 2\langle \dot{\mathbf{u}}_i, \mathbf{u}_j \rangle \left(\mathbf{y}_i^T \dot{\mathbf{y}}_j\right) \\ &- 2\langle \dot{\mathbf{u}}_i, \mathcal{L}(\mathbf{u}_j) \rangle \left(\mathbf{y}_i^T \mathbf{y}_j\right) - 2\langle \mathbf{u}_i, \mathcal{L}(\mathbf{u}_j) \rangle \left(\dot{\mathbf{y}}_i^T \mathbf{y}_j\right) \\ &+ \langle \mathcal{L}(\mathbf{u}_i), \mathcal{L}(\mathbf{u}_j) \rangle \left(\mathbf{y}_i^T \mathbf{y}_j\right) - 2\langle \dot{\mathbf{u}}_i, \mathbf{F}'\mathbf{y}_i \rangle - 2\langle \mathbf{u}_i, \mathbf{F}'\dot{\mathbf{y}}_i \rangle \\ &+ 2\langle \mathcal{L}(\mathbf{u}_i), \mathbf{F}'\mathbf{y}_i \rangle + \left\|\mathbf{F}'\right\|_F^2 + \lambda_{ij} \left(\langle \mathbf{u}_i, \dot{\mathbf{u}}_j \rangle - \phi_{ij}\right).\end{aligned}$$

The first order optimality condition requires the derivative of $\mathcal{G}$ with respect to $\dot{\mathbf{U}}, \dot{\mathbf{Y}}$ and λ vanish. The derivative of $\mathcal{G}$ with respect to λ produces the time derivative of the orthonormality constraint given by Equation 2.6. Provided that the f-OTD modes are orthonormal at $t = 0$, the time integration of Equation 2.6 reproduces the orthonormality condition of the f-OTD modes for $t > 0$: $\langle \mathbf{u}_i, \mathbf{u}_j \rangle = \delta_{ij}$. To take the derivative of $\mathcal{G}$ with respect to $\tilde{\dot{\mathbf{u}}}_k$ we use the Fréchet differential as follows:

$$\mathcal{G}'|_{\dot{\mathbf{U}}} := \lim_{\epsilon \to 0} \frac{\mathcal{G}(\dot{\mathbf{U}} + \epsilon \dot{\mathbf{U}}', \dot{\mathbf{Y}}, \lambda) - \mathcal{G}(\dot{\mathbf{U}}, \dot{\mathbf{Y}}, \lambda)}{\epsilon}.$$

Using the above definition we have:

$$\begin{aligned}\mathcal{G}'|_{\dot{\mathbf{u}}_k} &= 2\langle \dot{\mathbf{u}}', \dot{\mathbf{u}}_j \rangle \left(\mathbf{y}_k^T \mathbf{y}_j\right) + 2\langle \dot{\mathbf{u}}', \mathbf{u}_j \rangle \left(\mathbf{y}_k^T \dot{\mathbf{y}}_j\right) - 2\langle \dot{\mathbf{u}}', \mathcal{L}(\mathbf{u}_j) \rangle \left(\mathbf{y}_k^T \mathbf{y}_j\right) \\ &- 2\langle \dot{\mathbf{u}}', \mathbf{F}'\mathbf{y}_k \rangle + \lambda_{jk} \langle \dot{\mathbf{u}}', \mathbf{u}_j \rangle = 0.\end{aligned}$$

The above equation can be written as $\langle \dot{\mathbf{u}}', \nabla_{\dot{\mathbf{u}}_k} \mathcal{G} \rangle$ and we observe that for any arbitrary direction $\dot{\mathbf{u}}'$, we must satisfy $\nabla_{\dot{\mathbf{u}}_k} \mathcal{G} = \mathbf{0}$. This leads to the following condition:

$$\nabla_{\dot{\mathbf{u}}_k} \mathcal{G} = 2\dot{\mathbf{u}}_j \left(\mathbf{y}_k^T \mathbf{y}_j\right) + 2\mathbf{u}_j \left(\mathbf{y}_k^T \dot{\mathbf{y}}_j\right) - 2\mathcal{L}(\mathbf{u}_j) \left(\mathbf{y}_k^T \mathbf{y}_j\right) - 2\mathbf{F}'\mathbf{y}_k + \lambda_{jk} \mathbf{u}_j = \mathbf{0}. \tag{2.8}$$

To eliminate λ_{jk}, we take the inner product of $\mathbf{u}_l$ with Equation 2.8 to obtain

$$\begin{aligned}\langle \mathbf{u}_l, \nabla_{\dot{\mathbf{u}}_k} \mathcal{G} \rangle &= 2\phi_{lj} (\mathbf{y}_k^T \mathbf{y}_j) + 2\delta_{lj} \left(\mathbf{y}_k^T \dot{\mathbf{y}}_j\right) - 2\langle \mathbf{u}_l, \mathcal{L}(\mathbf{u}_j) \rangle \left(\mathbf{y}_k^T \mathbf{y}_j\right) \\ &- 2\langle \mathbf{u}_l, \mathbf{F}'\mathbf{y}_k \rangle + \lambda_{jk} \delta_{lj} = 0,\end{aligned}$$

where we have used $\langle \mathbf{u}_l, \dot{\mathbf{u}}_j \rangle = \phi_{lj}$ and $\langle \mathbf{u}_l, \mathbf{u}_j \rangle = \delta_{lj}$. Rearranging for λ_{lk} gives

$$\lambda_{lk} = 2\left[-\phi_{lj}(\mathbf{y}_k^T\mathbf{y}_j) - \left(\mathbf{y}_k^T\dot{\mathbf{y}}_l\right) + \langle \mathbf{u}_l, \mathcal{L}(\mathbf{u}_j)\rangle \left(\mathbf{y}_k^T\mathbf{y}_j\right) + \langle \mathbf{u}_l, \mathbf{F}'\mathbf{y}_k\rangle\right].$$

Dividing 2.8 by 2 and substituting λ_{lk} gives

$$[\dot{\mathbf{u}}_j - \mathcal{L}(\mathbf{u}_j) + \langle \mathbf{u}_l, \mathcal{L}(\mathbf{u}_j)\rangle \mathbf{u}_l - \phi_{lj}\mathbf{u}_l]\left(\mathbf{y}_k^T\mathbf{y}_j\right) - \mathbf{F}'\mathbf{y}_k + \langle \mathbf{u}_l, \mathbf{F}'\mathbf{y}_k\rangle \mathbf{u}_l = \mathbf{0}.$$

Rearranging the above Equation for $\dot{\mathbf{u}}_j$ we get

$$\boxed{\dot{\mathbf{u}}_j = \mathcal{L}(\mathbf{u}_j) - \langle \mathbf{u}_l, \mathcal{L}(\mathbf{u}_j)\rangle \mathbf{u}_l + [\mathbf{F}'\mathbf{y}_k - \langle \mathbf{u}_l, \mathbf{F}'\mathbf{y}_k\rangle \mathbf{u}_l]\, C_{kj}^{-1} + \phi_{lj}\mathbf{u}_l,} \tag{2.9}$$

where $\mathbf{C}(t) = [C_{kj}(t)] \in \mathbb{R}^{r\times r}$ is the low-rank *correlation* matrix, in which $C_{ik}(t) = \mathbf{y}_k(t)^T\mathbf{y}_j(t)$. Similarly, the first order optimality condition of $\mathcal{G}$ with respect to $\dot{\mathbf{y}}_k$ requires that

$$\frac{\partial \mathcal{G}}{\partial \dot{\mathbf{y}}_k} = \langle \mathbf{u}_k, \mathbf{u}_j\rangle \dot{\mathbf{y}}_j + \langle \dot{\mathbf{u}}_j, \mathbf{u}_k\rangle \mathbf{y}_j - \langle \mathbf{u}_k, \mathcal{L}(\mathbf{u}_j)\rangle \mathbf{y}_j - \langle \mathbf{F}', \mathbf{u}_k\rangle = \mathbf{0}.$$

Again, we use $\langle \mathbf{u}_k, \mathbf{u}_j\rangle = \delta_{kj}$ and $\langle \dot{\mathbf{u}}_j, \mathbf{u}_k\rangle = -\phi_{jk}$. Rearranging for $\dot{\mathbf{y}}_k$ gives

$$\boxed{\dot{\mathbf{y}}_k = \langle \mathbf{u}_k, \mathcal{L}(\mathbf{u}_j)\rangle \mathbf{y}_j + \langle \mathbf{F}', \mathbf{u}_k\rangle + \phi_{jk}\mathbf{y}_j.} \tag{2.10}$$

Here we have shown that minimizing the above functional with respect to $\dot{\mathbf{U}}(x,t)$ and $\dot{\mathbf{Y}}(t)$ leads to closed form evolution equations for the modes and corresponding sensitivity coefficients (ROM). Equations 2.9 and 2.10 are initialized by solving Equation 2.2 for a single time step and computing the singular value decomposition (SVD) of $\mathbf{V}'(x, t = \Delta t)$, such that $\mathbf{U}(x, t = \Delta t)$ contains the first r left singular vectors and $\mathbf{Y}(t = \Delta t)$ is the matrix multiplication of the first r right singular vectors and singular values; see Section 2.4.4. In Section 2.4.2, we show that the skew symmetric matrix ϕ_{ij} can be taken to be zero, i.e., $\phi_{ij} = 0$.

In the following, we make several observations about Equations 2.9 and 2.10: (i) Equation 2.9 determines the evolution of the f-OTD subspace. For $\phi_{ij} = 0$, the right hand side of Equation 2.9 is equal to the projection of $\mathcal{L}(\mathbf{U}) + \mathbf{FYC}^{-1}$ onto the complement of the space spanned by $\mathbf{U}$. Therefore, if $\mathcal{L}(\mathbf{U}) + \mathbf{FYC}^{-1}$ is in the span of $\mathbf{U}$, the f-OTD subspace does not evolve, i.e., $\dot{\mathbf{U}} = \mathbf{0}$. However, when $\mathcal{L}(\mathbf{U}) + \mathbf{FYC}^{-1}$ is not in the span of $\mathbf{U}$, the f-OTD subspace evolves optimally to follow the right hand side. Equation 2.10 is the f-OTD reduced order model (ROM) that determines the evolution of the sensitivities within the f-OTD subspace. (ii) We observe that if we set $\mathbf{F}'(x,t) = \mathbf{0}$ in the above equations, we recover the OTD evolution equations presented in [11]. However, unlike the OTD equations, where the evolution of the OTD modes are independent of the evolution of the coefficients ($\mathbf{Y}$), there is a two-way nonlinear coupling between the f-OTD evolution equations for $\mathbf{U}$ and $\mathbf{Y}$. (iii) From the above equations, it is clear to see that f-OTD extracts the low-rank approximation directly from the sensitivity evolution equation. In that sense, it is different from data-driven low-rank approximations such as proper orthogonal decomposition [6, 2, 4] or dynamic mode decomposition [83, 55], in which the low-rank subspace is extracted from preexisting data. The need to generate data simply does not exist in the f-OTD workflow. (iv) The computational cost of solving the f-OTD Equations 2.9 and 2.10 is roughly equivalent to that of solving r forward sensitivity equations. This is because the evolution of the f-OTD modes described by Equation 2.9 inherits the same differential operators from the sensitivity equation. In fact, Equation 2.9 can be formulated as a forced linear system $\partial \mathbf{u}_i/\partial t = \mathcal{L}(\mathbf{u}_i)+\mathbf{g}_i$. Assuming the discrete f-OTD modes have the size of $\mathbf{U} \in \mathbb{R}^{n\times r}$ and also assuming that $\mathcal{L}$ represents differential operators that can be represented discretely with a matrix of size $n\times n$, for implicit time integration, the cost of solving a linear system often exceeds that of computing $\mathbf{g}_i$. Evaluating $\mathbf{g}_i$ involves computing: (i) the low-rank matrix $\mathbf{L}_{r_{ij}} = \langle \mathbf{u}_j, \mathcal{L}(\mathbf{u}_i)\rangle$, which has the computational complexity of $\mathcal{O}(r^2 n)$, when $\mathcal{L}$ is sparse and $\mathcal{O}(r^2n^2)$ when $\mathcal{L}$ is a full matrix; (ii) $\langle \mathbf{u}_j, \mathbf{F}'\mathbf{y}_k\rangle = \langle \mathbf{u}_j, \mathbf{F}'\rangle \mathbf{y}_k$ which has computational complexity $\mathcal{O}(nrd + dr^2)$, and (iii) the correlation matrix inversion $\mathbf{C}^{-1}$ which has computational complexity $\mathcal{O}(r^3)$. Since r is often much smaller than n, the cost of inverting $\mathbf{C}$ is negligible. Equation 2.10 is an ODE and therefore its computational cost is negligible compared to the f-OTD modes, which are governed by a PDE. The cost of computing the terms that appear on the right hand side of Equation 2.10 is already accounted for in Equation 2.9. Also, the computational storage requirement of solving r f-OTD modes is equivalent to that of solving r forward sensitivity equations as the storage cost of each f-OTD mode is equivalent to a single sensitivity field and the storage cost of $\mathbf{Y}$ is negligible.

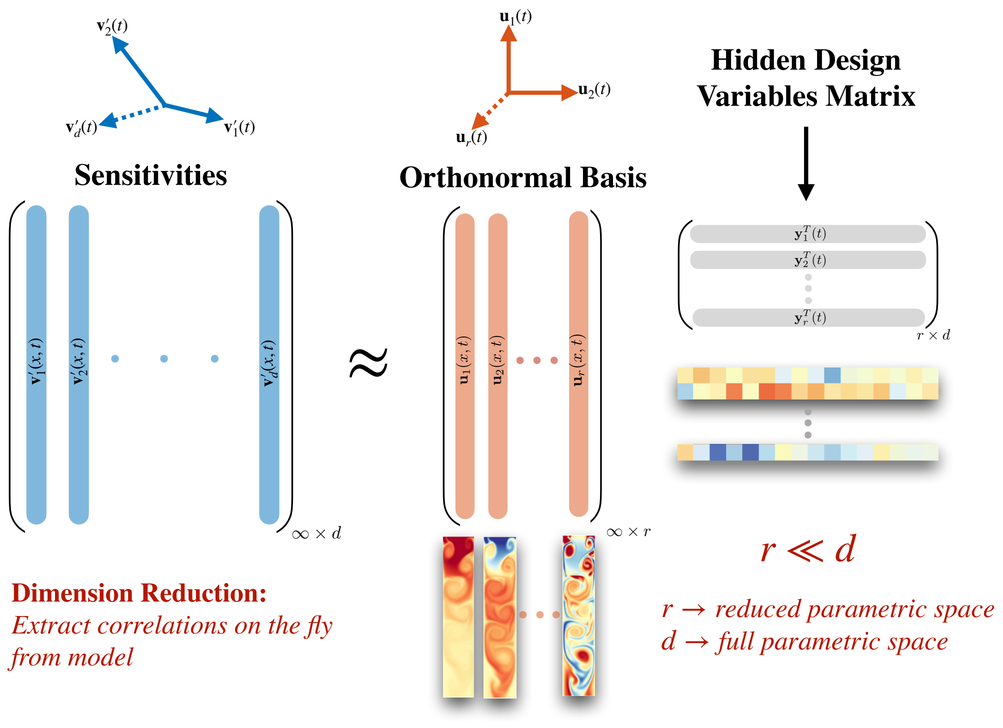


Figure 1: Overview of the reduced order modeling strategy. Shown on left in blue is the full dimensional system of sensitivities that we seek to model using the f-OTD low-rank approximation. Shown on right is the low-rank approximation which consists of a set of temporally evolving orthonormal modes (red) and hidden design variables (gray). The hidden design variables are coefficients that map the orthonormal basis to each sensitivity in the full-dimensional system. That is, each of the d sensitivities are approximated as a linear combination of the r orthonormal modes, where $r \ll d$. It is important to note that the orthonormal basis and hidden design variables are model-driven and evolve based on the linear sensitivity dynamics. Thus, the proposed method only requires solving a system of r PDEs and r ODEs for the modes and coefficients, respectively.

2.4.2 Equivalence

It is important to note that the choice of ϕ_{ij} in Equations 2.9 and 2.10 is not unique, and any skew-symmetric matrix yields an equivalent reduction. Similar to the OTD equations [11], we choose $\phi_{ij} = 0$, which corresponds to the dynamically orthogonal (DO) condition. This property is summarized in the theorem below.

Theorem 2.4.1. *Let $\{\mathbf{U}(x,t), \mathbf{Y}(t)\}$ and $\{\tilde{\mathbf{U}}(x,t), \tilde{\mathbf{Y}}(t)\}$ represent two reductions that satisfy equations 2.9 and 2.10 with corresponding skew-symmetric matrices $\mathbf{\Phi}(t)$ and $\tilde{\mathbf{\Phi}}(t)$, respectively. If the reductions are equivalent at $t = 0$, i.e. they are initially related by an orthogonal rotation matrix $\mathbf{R}_0 \in \mathbb{R}^{r\times r}$ as $\mathbf{U}(x,0) = \tilde{\mathbf{U}}(x,0)\mathbf{R}_0$ and $\mathbf{Y}(0) = \tilde{\mathbf{Y}}(0)\mathbf{R}_0$, then the two reductions will remain equivalent for $t > 0$ with rotation matrix $\mathbf{R}(t)$ governed by $\dot{\mathbf{R}} = \mathbf{R}\mathbf{\Phi} - \tilde{\mathbf{\Phi}}\mathbf{R}$.*

Proof. We prove the equivalence by using the evolution equation for the $\mathbf{U}, \mathbf{Y}$ and using the matrix differential equation for the rotation matrix $\mathbf{R}$ and recovering the evolution equations for $\tilde{\mathbf{U}}, \tilde{\mathbf{Y}}$. To this end, we substitute $\mathbf{U} = \tilde{\mathbf{U}}\mathbf{R}$ and $\mathbf{Y} = \tilde{\mathbf{Y}}\mathbf{R}$ into the quasimatrix form of equations 2.9 and 2.10. The evolution equation for the orthonormal modes becomes:

$$\begin{aligned}\dot{\mathbf{U}} &= \dot{\tilde{\mathbf{U}}}\mathbf{R} + \tilde{\mathbf{U}}\dot{\mathbf{R}} \\ &= \mathcal{L}(\tilde{\mathbf{U}})\mathbf{R} - \tilde{\mathbf{U}}\mathbf{R}\langle \tilde{\mathbf{U}}\mathbf{R}, \mathcal{L}(\tilde{\mathbf{U}})\mathbf{R}\rangle + [\mathbf{F}'\tilde{\mathbf{Y}}\mathbf{R} - \tilde{\mathbf{U}}\mathbf{R}\langle \tilde{\mathbf{U}}\mathbf{R}, \mathbf{F}'\tilde{\mathbf{Y}}\mathbf{R}\rangle] + \tilde{\mathbf{U}}\mathbf{R}\mathbf{\Phi}.\end{aligned}$$

Substituting $\dot{\mathbf{R}} = \mathbf{R}\mathbf{\Phi} - \tilde{\mathbf{\Phi}}\mathbf{R}$ and solving for $\dot{\tilde{\mathbf{U}}}$ yields

$$\begin{aligned}\dot{\tilde{\mathbf{U}}} = \big[&\mathcal{L}(\tilde{\mathbf{U}})\mathbf{R} - \tilde{\mathbf{U}}\mathbf{R}\langle \tilde{\mathbf{U}}\mathbf{R}, \mathcal{L}(\tilde{\mathbf{U}})\mathbf{R}\rangle + [\mathbf{F}'\tilde{\mathbf{Y}}\mathbf{R} - \tilde{\mathbf{U}}\mathbf{R}\langle \tilde{\mathbf{U}}\mathbf{R}, \mathbf{F}'\tilde{\mathbf{Y}}\mathbf{R}\rangle] \\ &+ \tilde{\mathbf{U}}\mathbf{R}\mathbf{\Phi} - \tilde{\mathbf{U}}[\mathbf{R}\mathbf{\Phi} - \tilde{\mathbf{\Phi}}\mathbf{R}\big]\mathbf{R}^T.\end{aligned}$$

Simplifying the above equation and using $\langle \tilde{\mathbf{U}}\mathbf{R}, \cdot\rangle = \mathbf{R}^T\langle \tilde{\mathbf{U}}, \cdot\rangle$ and $\mathbf{R}^{-1} = \mathbf{R}^T$, since $\mathbf{R}$ is an orthonormal matrix results in:

$$\dot{\tilde{\mathbf{U}}} = \mathcal{L}(\tilde{\mathbf{U}}) - \tilde{\mathbf{U}}\langle \tilde{\mathbf{U}}, \mathcal{L}(\tilde{\mathbf{U}})\rangle + [\mathbf{F}'\tilde{\mathbf{Y}} - \tilde{\mathbf{U}}\langle \tilde{\mathbf{U}}, \mathbf{F}'\tilde{\mathbf{Y}}\rangle]\tilde{\mathbf{C}}^{-1} + \tilde{\mathbf{U}}\tilde{\mathbf{\Phi}},$$

where $\tilde{\mathbf{C}} = \mathbf{R}\mathbf{C}\mathbf{R}^T$ and $\tilde{\mathbf{C}}^{-1} = \mathbf{R}\mathbf{C}^{-1}\mathbf{R}^T$, where $\mathbf{C}$ and $\tilde{\mathbf{C}}$ are similar matrices and thus have the same eigenvalues. Following a similar procedure, the evolution equation for the coefficients becomes:

$$\begin{aligned}\dot{\mathbf{Y}} &= \dot{\tilde{\mathbf{Y}}}\mathbf{R} + \tilde{\mathbf{Y}}\dot{\mathbf{R}} \\ &= \tilde{\mathbf{Y}}\mathbf{R}\langle \mathcal{L}(\tilde{\mathbf{U}})\mathbf{R}, \tilde{\mathbf{U}}\rangle\mathbf{R} + \langle \mathbf{F}', \tilde{\mathbf{U}}\rangle\mathbf{R} + \tilde{\mathbf{Y}}\mathbf{R}\mathbf{\Phi}.\end{aligned}$$

Substituting $\dot{\mathbf{R}} = \mathbf{R}\mathbf{\Phi} - \tilde{\mathbf{\Phi}}\mathbf{R}$ and solving for $\dot{\tilde{\mathbf{Y}}}$ yields

$$\begin{aligned}\dot{\tilde{\mathbf{Y}}} &= \big[\tilde{\mathbf{Y}}\mathbf{R}\mathbf{R}^T\langle \mathcal{L}(\tilde{\mathbf{U}}), \tilde{\mathbf{U}}\rangle\mathbf{R} + \langle \mathbf{F}', \tilde{\mathbf{U}}\rangle\mathbf{R} + \tilde{\mathbf{Y}}\mathbf{R}\mathbf{\Phi} - \tilde{\mathbf{Y}}[\mathbf{R}\mathbf{\Phi} - \tilde{\mathbf{\Phi}}\mathbf{R}]\big]\mathbf{R}^T \\ &= \tilde{\mathbf{Y}}\langle \mathcal{L}(\tilde{\mathbf{U}}), \tilde{\mathbf{U}}\rangle + \langle \mathbf{F}', \tilde{\mathbf{U}}\rangle + \tilde{\mathbf{Y}}\tilde{\mathbf{\Phi}}.\end{aligned}$$

Thus, we have shown that the evolution of $\{\mathbf{U}(x,t), \mathbf{Y}(t)\}$ and $\{\tilde{\mathbf{U}}(x,t), \tilde{\mathbf{Y}}(t)\}$ according to Equations 2.9 and 2.10 are equivalent. □

2.4.3 Exactness of f-OTD

For the case where the full sensitivity quasimatrix is of rank d, the rank d f-OTD equations are exact. To show this, we start by considering an arbitrary perturbation subspace, $\mathbf{V}'(x,t) \in \mathbb{R}^{\infty\times d}$, governed by the quasimatrix form of Equation 2.2:

$$\frac{\partial \mathbf{V}'}{\partial t} = \mathcal{L}(\mathbf{V}') + \mathbf{F}'(x,t), \quad \mathbf{V}'(x,0) = \mathbf{V}'_0(x),$$

where columns of $\mathbf{V}'(x,t)$ are independent, i.e. $\langle \mathbf{v}'_i, \mathbf{v}'_j\rangle = 0$ if $i \neq j$, and the evolution of an orthonormal subspace, $\mathbf{U}(x,t) \in \mathbb{R}^{\infty\times d}$, governed by the quasimatrix form of Equation 2.9:

$$\frac{\partial \mathbf{U}}{\partial t} = \mathcal{L}(\mathbf{U}) - \mathbf{U}\mathbf{L}_r(t) + (\mathbf{F}'\mathbf{Y} - \mathbf{U}\langle \mathbf{U}, \mathbf{F}'\mathbf{Y}\rangle)\,\mathbf{C}^{-1}, \quad \mathbf{U}(x,0) = \mathbf{U}_0(x).$$

The corresponding matrix of sensitivity coefficients are governed by the matrix form of Equation 2.10 as:

$$\frac{d\mathbf{Y}}{dt} = \mathbf{Y}\mathbf{L}_r^T + \langle \mathbf{F}', \mathbf{U}\rangle, \quad \mathbf{Y}(0) = \mathbf{Y}_0,$$

where $\mathbf{L}_r(t) = \langle \mathbf{U}(x,t), \mathcal{L}(\mathbf{U}(x,t))\rangle$ is the $r \times r$ low-rank linear operator. We can show that if the two subspaces are initially equivalent, i.e., $\mathbf{U}_0(x)$ can be mapped to $\mathbf{V}'_0(x)$ via the linear transformation $\mathbf{Y}_0^T$, then $\mathbf{V}'(x,t)$ and $\mathbf{U}(x,t)$ remain equivalent for all time t and are related by the linear transformation $\mathbf{Y}(t)^T$. This leads to the following theorem:

Theorem 2.4.2. *Let* $\mathbf{V}'(x,t) \in \mathbb{R}^{\infty\times d}$ *be an arbitrary subspace evolved by the linear dynamics of Equation 2.2, and* $\mathbf{U}(x,t) \in \mathbb{R}^{\infty\times d}$ *be an orthonormal subspace evolved by Equation 2.9. If initially* $\mathbf{V}'_0(x)$ *and* $\mathbf{U}_0(x)$ *are equivalent, i.e.* $\mathbf{V}'_0(X) = \mathbf{U}_0(x)\mathbf{Y}_0^T$*, then the perturbation subspace can be exactly determined via the linear transformation* $\mathbf{V}'(x,t) = \mathbf{U}(x,t)\mathbf{Y}(t)^T$ *for all time* t*, where* $\mathbf{Y}(t)$ *is governed by Equation 2.10.*

Proof. Start by substituting $\mathbf{V}'(x,t) = \mathbf{U}(x,t)\mathbf{Y}(t)^T$ into the quasimatrix form of Equation 2.2:

$$\dot{\mathbf{U}}\mathbf{Y}^T + \mathbf{U}\dot{\mathbf{Y}}^T = \mathcal{L}(\mathbf{U})\mathbf{Y}^T + \mathbf{F}'.$$

Next we substitute $\dot{\mathbf{Y}}^T$ from Equation 2.10

$$\dot{\mathbf{U}}\mathbf{Y}^T + \mathbf{U}\left(\mathbf{L}_r\mathbf{Y}^T + \langle\mathbf{U},\mathbf{F}'\rangle\right) = \mathcal{L}(\mathbf{U})\mathbf{Y}^T + \mathbf{F}',$$

where we have used $\mathbf{L}_r(t) = \langle\mathbf{U}(x,t), \mathcal{L}(\mathbf{U}(x,t))\rangle$. We multiply by $\mathbf{Y}$ from right and rearrange to get

$$\dot{\mathbf{U}}\mathbf{C} = \mathcal{L}(\mathbf{U})\mathbf{C} - \mathbf{U}\mathbf{L}_r\mathbf{C} + \left(\mathbf{F}'\mathbf{Y} - \mathbf{U}\langle\mathbf{U},\mathbf{F}'\mathbf{Y}\rangle\right),$$

where we have used $\mathbf{C} = \mathbf{Y}^T\mathbf{Y}$. Finally, we multiply by $\mathbf{C}^{-1}$ from right to get

$$\dot{\mathbf{U}} = \mathcal{L}(\mathbf{U}) - \mathbf{U}\mathbf{L}_r + \left(\mathbf{F}'\mathbf{Y} - \mathbf{U}\langle\mathbf{U},\mathbf{F}'\mathbf{Y}\rangle\right)\mathbf{C}^{-1},$$

which is the same as the evolution Equation 2.9 for the orthonormal basis. Here we have shown that the evolution of $\mathbf{V}'(x,t)$ under Equation 2.2 is equivalent to the evolution of $\mathbf{U}(x,t)$ under Equation 2.9. That is, when $r = d$, $\mathbf{Y}(t)^T$ is a linear transformation that exactly maps the orthonormal subspace $\mathbf{U}(x,t)$ to $\mathbf{V}'(x,t)$. $\square$

2.4.4 Approximation error

The approximation error of estimating sensitivities using f-OTD can be expressed as $e(t) = \|\mathbf{V}'(x,t) - \mathbf{U}(x,t)\mathbf{Y}(t)^T\|_F$. This error can be properly analyzed and better understood by considering two types of error: (i) the resolved error, denoted by $e_r(t)$ and (ii) the unresolved error, denoted by $e_u(t)$. The resolved error is the discrepancy between approximating the sensitivities with rank-r f-OTD and the optimal rank-r approximation: $e_r(t) = \|\mathbf{U}(x,t)\mathbf{Y}(t)^T - \tilde{\mathbf{U}}(x,t)\tilde{\mathbf{Y}}(t)^T\|_F$, where $\tilde{\mathbf{U}}(x,t) \in \mathbb{R}^{\infty\times r}$ and $\tilde{\mathbf{Y}}(t) \in \mathbb{R}^{d\times r}$ are the optimal rank-r orthonormal modes and their coefficients, respectively. The unresolved error is the error of the optimal rank-r approximation: $e_u(t) = \|\tilde{\mathbf{U}}(x,t)\tilde{\mathbf{Y}}(t)^T - \mathbf{V}'(x,t)\|_F$, that is a direct result of truncating the $d-r$ least energetic modes. Thus, the optimal rank-r approximation is obtained by minimizing:

$$\mathcal{E}_u(\tilde{\mathbf{U}}(x,t), \tilde{\mathbf{Y}}(t)) = \left\|\tilde{\mathbf{U}}(x,t)\tilde{\mathbf{Y}}(t)^T - \mathbf{V}'(x,t)\right\|_F, \tag{2.11}$$

subject to the orthonormality condition of $\tilde{\mathbf{U}}(x,t)$ modes. The optimal decomposition can be obtained by performing instantaneous SVD of the sensitivity matrix, where $\tilde{\mathbf{U}}(x,t)$ is the matrix of r most dominant left singular vectors of $\mathbf{V}'(x,t)$ and $\tilde{\mathbf{Y}}(t) = \tilde{\mathbf{Z}}(t)\tilde{\mathbf{\Sigma}}(t)$, where $\tilde{\mathbf{Z}}(t) \in \mathbb{R}^{d\times r}$ and $\tilde{\mathbf{\Sigma}}(t) = \text{diag}(\tilde{\sigma_1}(t), \tilde{\sigma_2}(t), \ldots, \tilde{\sigma_r}(t))$ are the matrix of the r most dominant right singular vectors and the matrix of singular values, respectively. It is straightforward to show that: $e_u(t) = (\sum_{i=r+1}^{d} \tilde{\sigma}_i^2(t))^{1/2}$. The error $e_u(t)$ represents the minimum error that any rank-r approximation can achieve, and therefore, it amounts to a lower bound for the f-OTD error: $e(t) \geq e_u(t)$. On the other hand, as with any reduced order model of a time-dependent system, the unresolved subspace induces a *memory error* in the f-OTD approximation. This means that the unresolved error *drives* the resolved error $e_r(t)$, and under appropriate conditions, it has been shown that for similar time-dependent basis low-rank approximations, $e_r(t)$ can be bounded by: $e_r(t) \leq c_1 e^{c_2 t}\int_{t_0}^{t} e_u(s)ds$ [49] for $c_1, c_2 > 0$. The interplay between $e_u(t)$ and $e_r(t)$ can be more rigorously studied within the Mori-Zwanzig formalism [28]. These error estimates can guide an adaptive f-OTD, in which modes are added or removed to maintain the error below some threshold value [9], however these aspects are not in the scope of this paper and are not explored any further here. Since sensitivities can either be very small or very large with errors following the same trend, we compute the relative error percentages as shown here:

$$\% \text{ Error} = \frac{e(t)}{\|\mathbf{V}'(x,t)\|_F} \times 100. \tag{2.12}$$

Similar quantities are computed for $e_u(t)$ and $e_r(t)$.

2.4.5 Mode Ranking

In this section we present a procedure to rank the f-OTD modes and their coefficients according to their significance. To this end, we start by considering the reduced correlation matrix $\mathbf{C}(t)$, which is in general a full matrix. This implies that the sensitivity coefficients are correlated and there exists a linear mapping from the correlated coefficients, $\mathbf{Y}(t)$, to the uncorrelated coefficients, $\hat{\mathbf{Y}}(t)\mathbf{\Sigma}(t)$, where $\hat{\mathbf{Y}}(t)$ are the orthonormal coefficients and $\mathbf{\Sigma}(t) = \text{diag}(\sigma_1(t), \sigma_2(t), \dots, \sigma_r(t))$ is a diagonal matrix of singular values. To find such a mapping, we consider the eigen-decomposition of $\mathbf{C}(t)$ as follows:

$$\mathbf{C}(t)\mathbf{R}(t) = \mathbf{R}(t)\mathbf{\Lambda}(t), \tag{2.13}$$

where $\mathbf{R}(t) \in \mathbb{R}^{r\times r}$ is a matrix whose columns contain the eigenvectors of $\mathbf{C}(t)$ and $\mathbf{\Lambda}(t)$ is a diagonal matrix containing the eigenvalues of $\mathbf{C}(t)$, $\{\lambda_i(t)\}_{i=1}^r$. Since $\mathbf{C}(t)$ is a symmetric positive matrix, the matrix $\mathbf{R}(t)$ is an orthonormal matrix, i.e. $\mathbf{R}(t)^T\mathbf{R}(t) = \mathbf{I}$, and the eigenvalues are all non-negative and can be sorted as: $\lambda_1(t) > \lambda_2(t) > \dots > \lambda_r(t) \geq 0$. It is also straightforward to show that the singular values of the f-OTD low-rank approximation are $\sigma_i(t) = \lambda_i(t)^{1/2}$, for $i = 1, 2, \dots, r$.

The ranked f-OTD components can be defined as:

$$\hat{\mathbf{Y}}(t) = \mathbf{Y}(t)\mathbf{R}(t)\mathbf{\Sigma}^{-1}(t), \qquad \hat{\mathbf{U}}(x,t) = \mathbf{U}(x,t)\mathbf{R}(t),$$

where the columns of $\hat{\mathbf{Y}}(t)$ and $\hat{\mathbf{U}}(x,t)$ are ranked by energy (σ_i^2) in descending order. We shall refer to $\{\hat{\mathbf{Y}}(t), \mathbf{\Sigma}(t), \hat{\mathbf{U}}(x,t)\}$ as the bi-orthonormal form of the reduction. Since the above equations are simply an in-subspace rotation, $\{\hat{\mathbf{Y}}(t)\mathbf{\Sigma}(t), \hat{\mathbf{U}}(x,t)\}$ and $\{\mathbf{Y}(t), \mathbf{U}(x,t)\}$

yield equivalent low-rank approximations of the full-dimensional dynamics. This is easily verified by considering the bi-orthonormal form of the low-rank approximation as $\hat{\mathbf{U}}(x,t)\boldsymbol{\Sigma}(t)\hat{\mathbf{Y}}(t)^T = \mathbf{U}(x,t)\mathbf{Y}(t)^T$, where we have made use of the identity $\mathbf{R}(t)^T\mathbf{R}(t) = \mathbf{I}$. We refer to $\hat{\mathbf{Y}}$ as the *hidden* parametric space as each column of matrix $\hat{\mathbf{Y}}$ can be taken as a new ranked parameter that represents the contribution of all parameters ($\boldsymbol{\alpha}$).

2.5 Demonstration Cases

2.5.1 Rössler System

We first consider a simple demonstration of f-OTD by computing sensitivities of the Rössler system. The Rössler system is governed by:

$$\frac{dv_1}{dt} = -v_2 - v_3, \qquad \frac{dv_2}{dt} = v_1 + \alpha_1 v_2, \qquad \frac{dv_3}{dt} = \alpha_2 + v_3(v_1 - \alpha_3). \tag{2.14}$$

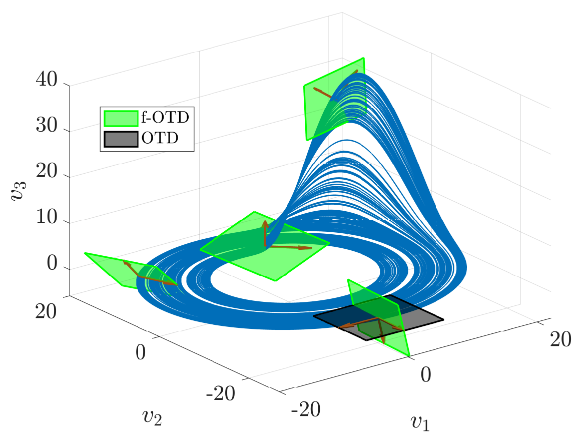
In the above equations, we set $\alpha_1 = \alpha_2 = 0.1$ and $\alpha_3 = 14$, which are common values used to study the chaotic behavior of the attractor. The goal is to calculate the sensitivity of $\mathbf{v}$ with respect to the model parameters $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3)$ as $\partial\mathbf{v}/\partial\boldsymbol{\alpha}$. To this end, we take the derivative of the above system of equations with respect to model parameter α_i to obtain the linear sensitivity equation

$$\frac{d\mathbf{V}'}{dt} = \mathbf{L}\mathbf{V}' + \mathbf{F}', \tag{2.15}$$

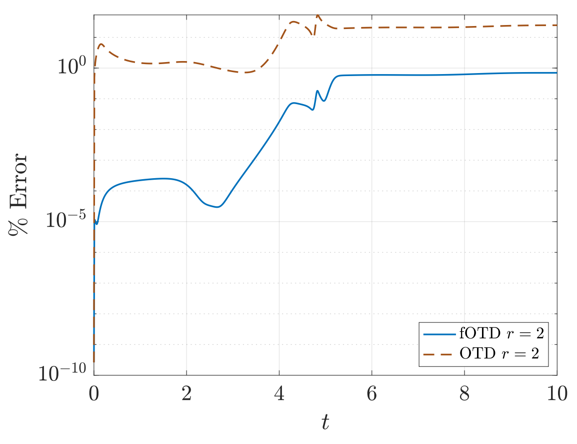
where

$$\mathbf{L} = \begin{bmatrix} 0 & -1 & -1 \\ 1 & \alpha_1 & 0 \\ v_3 & 0 & v_1 - \alpha_3 \end{bmatrix}, \quad \mathbf{V}' = \begin{bmatrix} | & | & | \\ \mathbf{v}'_1 & \mathbf{v}'_2 & \mathbf{v}'_3 \\ | & | & | \end{bmatrix}, \quad \mathbf{F}' = \begin{bmatrix} 0 & 0 & 0 \\ v_2 & 0 & 0 \\ 0 & 1 & -v_3 \end{bmatrix},$$

and $\mathbf{v}'_i$ is the sensitivity of the position with respect to α_i and $\mathbf{L} \in \mathbb{R}^{n\times n}$ and $\mathbf{F}' \in \mathbb{R}^{n\times d}$. We choose a subspace with dimension $r = 2$ for the low-rank approximation of the three-dimensional ($d = 3$) sensitivities ($\mathbf{V}'$). Although it is obvious that OTD modes are not based on parametric sensitivities and they are based on perturbations in the initial condition (IC) in all directions of the phase space, we believe it is instructive to contrast the OTD versus



(a)



(b)

Figure 2: (a) Chaotic Rössler attractor with optimal f-OTD subspace shown in green and OTD subspace shown in black for $r = 2$. Red arrows depict the orthonormal sensitivity vectors that define each subspace. (b) Percent error for $e(t)$ plotted versus time for the f-OTD and OTD subspaces.

f-OTD to better understand f-OTD. To this end, we build two real-time ROMs using OTD modes and f-OTD modes. In the case of OTD, we solve the OTD evolution equation and we project the forced sensitivity Equation 2.15 onto the OTD modes, resulting in:

$$\frac{d\mathbf{U}_{otd}}{dt} = (\mathbf{I} - \mathbf{U}_{otd}\mathbf{U}_{otd}^T)\mathbf{L}\mathbf{U}_{otd} \quad \text{and} \quad \frac{d\mathbf{Y}_{otd}}{dt} = \mathbf{Y}_{otd}\mathbf{U}_{otd}^T\mathbf{L}^T\mathbf{U}_{otd} + \mathbf{F}'^T\mathbf{U}_{otd}.$$

We also solved the f-OTD evolution Equations 2.9 and 2.10 for the finite-dimensional system. Both OTD and f-OTD modes are initialized with the same subspace and the evolution equations are solved for $T_f = 10$ units of time. These subspaces are initialized by first solving the full-dimensional sensitivity, Equation 2.15, for one $\Delta t = 10^{-2}$ and then computing the OTD and f-OTD subspaces as the first two left singular vectors of $\mathbf{V}'(x, t = \Delta t)$. In Figure 2(a), both OTD and f-OTD subspaces are visualized along with the attractor of the Rössler system. The OTD subspace is shown at only one instant for clarity and that point corresponds to the case where the nonlinear dynamics is in the $v_1 - v_2$ plane. At this point, the OTD subspace is oriented such that it nearly coincides with the $v_1 - v_2$ plane. This result is to be expected since the OTD subspace follows the sensitivities associated with the perturbations in the IC and we know that the IC-perturbed solutions will lie on the *same* attractor. On the other hand, the f-OTD subspace is correctly oriented along the most sensitive subspace for perturbations in the model parameters, i.e. $\delta\boldsymbol{\alpha} = (\delta\alpha_1, \delta\alpha_2, \delta\alpha_3)$, which lead to perturbations in the attractor itself. That is, the perturbed solutions lie on *different* attractors which can readily be seen as $\delta\boldsymbol{\alpha}$ results in nonzero δv_3, despite $v_3 \simeq 0$. This results in the f-OTD subspace having a large out-of-plane component in the v_3 direction, which the OTD subspace fails to capture in Figure 2(a). In Figure 2(b), the percent errors of $e(t)$ are shown for OTD and f-OTD, which confirms that f-OTD performs significantly better than OTD. This simple example demonstrates that the OTD basis is not optimal and may be inaccurate for reduced order modeling of the forced sensitivity equation.

2.5.2 Chaotic Kuramoto Sivashinsky Equation

In this example, we evaluate the performance of f-OTD in computing linear sensitivities of a chaotic system with many positive Lyapunov exponents and a high-dimensional parametric space. The intent of this example is not to compute the gradient of a time-averaged quantity for a chaotic system, but rather computing the solution of the sensitivity equation for a chaotic system with much larger unstable directions than the rank of the f-OTD subspace. For computing sensitivities of time-averaged quantities, one can use f-OTD in conjunction with Ruelle's linear response formula [79, 54] to compute ensemble sensitivities. We also refer the reader to references for methods related to long-term sensitivities in chaotic systems [88, 37]. To this end, we consider the sensitivity of the Kuramoto Sivashinsky (KS) equation with respect to a time dependent forcing parameter $\alpha(t)$. The KS equation is a fourth order PDE given by:

$$\frac{\partial \mathbf{v}}{\partial t} + \frac{1}{2}\frac{\partial \mathbf{v}^2}{\partial x} + \frac{\partial^2 \mathbf{v}}{\partial x^2} + \nu\frac{\partial^4 \mathbf{v}}{\partial x^4} = \alpha(t)\sin(2\pi x/L), \quad x \in [0, L], \tag{2.16}$$

where $\mathbf{v} = \mathbf{v}(x, t)$. Approximately 110 positive Lyapunov exponents exist for the parameters used in this study: $\nu = 1$ and $L = 1000$. Here $\alpha(t)$ represents an infinite-dimensional parametric space.

To compute the sensitivities numerically, we consider a discrete representation of $\alpha(t)$ in the interval $t_i \in [0, T_s]$, where $T_s \leq T_f$ is a subset of the full integration time T_f, and t_i is a discrete instance in time. To this end, we consider the value of $\alpha(t)$ at discrete time $t_i = (i-1) \times \Delta t$, where Δt is the time step. This results in a vector, $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \dots, \alpha_d)$, where $\alpha_i = \alpha(t_i)$ and $d = T_s/\Delta t$ is the number of instances in time (i.e. number of parameters). In general, Δt can be chosen independently of the numerical time integration step size, however, for simplicity, we use the same value of Δt for both the parametric discretization and numerical integration of the nonlinear solver and f-OTD equations. In this example, we consider $\Delta t = 10^{-2}$ and $T_s = 10$, which results in $d = 1000$ parameters. Further decrease in Δt did not change our results. This leads to the sensitivity of $\mathbf{v}$ with respect to the value of $\alpha(t)$ at 1000 evenly spaced instances in time. We evolve these sensitivities over the interval $t \in [0, T_f]$ with $T_f = 100$. We also choose $\alpha(t) = 0$ for $t_i \in [0, T_f]$, and therefore, the nonlinear solver $\mathbf{v}(t)$ is the solution of the unforced KS equation.

We consider the time-discrete form of Equation 2.16 and differentiate with respect to design parameter α_i. This leads to an evolution equation for the sensitivity of $\mathbf{v}$ with respect to α_i, in which the linear operator and forcing terms are:

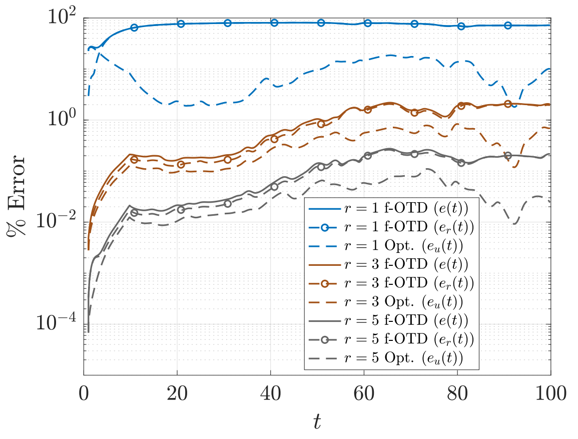
$$\mathcal{L}(\mathbf{v}'_i) = -\left[\frac{\partial(\mathbf{v}\mathbf{v}'_i)}{\partial x} + \frac{\partial^2 \mathbf{v}'_i}{\partial x^2} + \nu \frac{\partial^4 \mathbf{v}'_i}{\partial x^4}\right] \text{ and } \mathbf{f}'_i = \delta(t - t_i)\sin(2\pi x/L), \ i = 1, 2, \dots, d \quad (2.17)$$

where $\delta(t - t_i) = 0$ for $t \neq t_i$ and $\delta(t - t_i) = 1$ for $t = t_i$. Our goal is to solve Equation 2.17 using f-OTD. We discretize the KS equation and the f-OTD equations using $n = 2^{13} = 8192$ Fourier modes and use exponential time-differencing Runge-Kutta fourth-order (ETDRK4) time stepping scheme [45]. We verify our solution by directly solving Equation 2.17 for all 1000 sensitivities. Further decreasing Δt and increasing the number of Fourier modes did not change our results. We also compare the f-OTD error with that of optimal instantaneous same-rank approximation of the full sensitivities, which is obtained by computing the SVD of $\mathbf{V}'(x, t)$ at each time. In Figure 3(a), we compare the reconstruction error of f-OTD ($e(t)$) with the reconstruction error of same-rank SVD ($e_u(t)$). We also show the resolved error $e_r(t)$, which measures the discrepancy between the f-OTD approximation and the optimal same-rank approximation. We compute these errors for $r = 1, 3$ and 5. While the optimal low-rank approximation with a single mode captures approximately 99% of the system energy of the full sensitivity (see Figure 3(b)), the f-OTD approximation performs poorly with only a single mode, i.e., a dramatic reduction for 1000 sensitivities. This is a direct result of the memory effect from the lost interactions with the unresolved modes ($e_r(t)$) that ultimately dominate the error for long term integration. By increasing the number of f-OTD modes, both $e(t)$ and $e_r(t)$ decrease. It is possible to control the error in real-time through an adaptive strategy that adds/removes modes with an appropriate criterion. For example, a candidate criterion could be $p = \sigma_r^2(t)/\sum_{i=1}^{r}\sigma_i^2(t)$, where for $p < p_{th}$ the last mode can be removed and for $p > p_{th}$ a new mode can be added. See [9] for similar strategies for adaptive mode addition and removal.

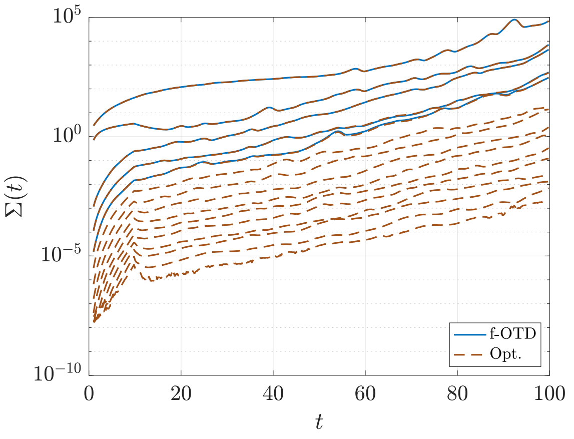
In Figure 3(b), we compare the 15 largest instantaneous singular values of quasimatrix $\mathbf{V}'(x,t)$ with those obtained from f-OTD with rank $r = 5$, which shows that f-OTD closely captures the most dominant subspace. In Figures 4(a) and 4(b) the orthonormalized coefficients of the first two dominant f-OTD modes for the case of $r = 5$ are compared to the right singular vectors from the instantaneous SVD of $\mathbf{V}'(x,t)$. These coefficients represent the hidden parametric space: for example, $\hat{\mathbf{y}}_1$ is a series of weights that represent the contribution of each of the $d = 1000$ sensitivities to the most dominant direction of the full sensitivity matrix, $\hat{\mathbf{u}}_1$. Due to the chaotic nature of this problem, we observe that these coefficients can be highly time-dependent, especially for the lower energy modes; see $\hat{\mathbf{y}}_2$. Nevertheless, we have demonstrated that f-OTD extracts the most dominant subspace and associated coefficients of the sensitivity matrix for a chaotic system with large number of unstable directions and parameters.

2.5.3 Species Transport Equation: Turbulent Reactive Flow

In this example, we show how a single set of f-OTD modes can lead to significant computational gains for computing linear sensitivities in problems with multiple coupled field variables, where each field variable has a different linear operator. We consider a species transport problem, where parameter identification via sensitivity analysis plays an important role in allocating computational and experimental resources to reduce parameter uncertainty. Moreover, the sensitivity analysis is used to create reduced reaction mechanisms for

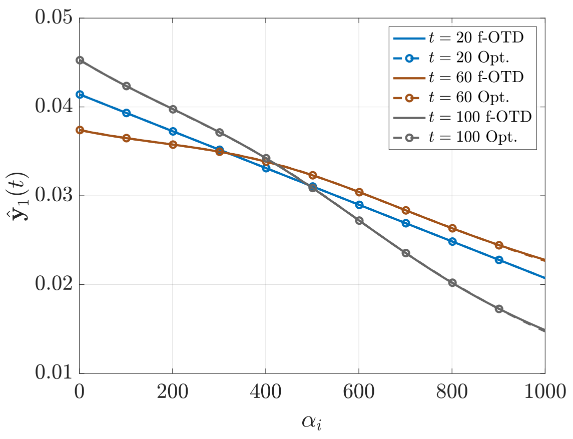


(a)

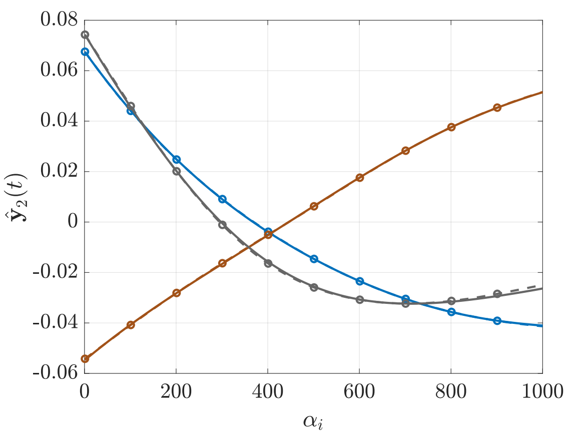


(b)

Figure 3: (a) Comparison of the reconstruction error between f-OTD approximation $(e(t))$ and optimal rank-r approximation $(e_u(t))$ for different reduction sizes. Resolved error, $e_r(t)$, dominates the f-OTD error for long term integration. Error decreases as the number of modes increases. (b) Comparison of singular values between f-OTD and optimal low-rank decomposition for $r = 5$.



(a)



(b)

Figure 4: Kuramoto-Sivashinsky: The first two columns of the orthonormalized design variables matrix shown at different instances in time: (a) $\hat{\mathbf{y}}_1(t)$ (b) $\hat{\mathbf{y}}_2(t)$. The horizontal axis corresponds to the i^{th} design parameter α_i.

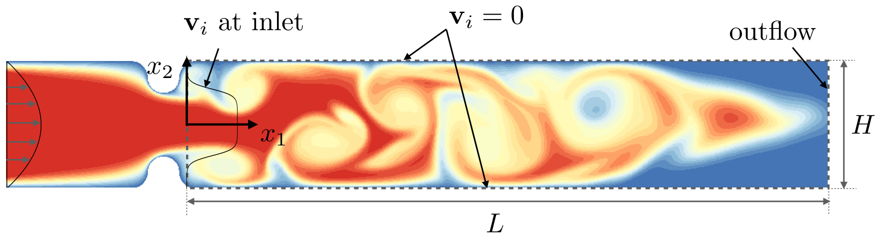


Figure 5: Schematic of the flow visualized with a passive scalar.

complex chemical systems involving a large number of species and reactions. See references [22, 57, 56].

2.5.3.1 Problem Setup

To this end, we consider a 2D incompressible turbulent reactive flow:

$$\frac{\partial \mathbf{v}_i}{\partial t} + (\mathbf{w} \cdot \nabla)\, \mathbf{v}_i = \tilde{\kappa}_{ik} \nabla^2 \mathbf{v}_k + \mathbf{s}_i, \tag{2.18}$$

where $\mathbf{w} = (\mathbf{w}_{x_1}(x_1, x_2, t), \mathbf{w}_{x_2}(x_1, x_2, t))$ is the velocity field from the 2D incompressible Navier-Stokes equations, $\mathbf{v}_i = \mathbf{v}_i(x_1, x_2, t)$ is the concentration of species i, $\tilde{\kappa}_{ik} \in \mathbb{R}^{n_s \times n_s}$ is the diffusion coefficient matrix, and $\mathbf{s}_i = \mathbf{s}_i(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{n_s}; \boldsymbol{\alpha})$ is the non-linear reactive source term. We choose a diagonal diffusion coefficient matrix, where the ith diagonal entry is the diffusion coefficient of the ith species, and n_s is the number of species. For the reactive source term $\mathbf{s}_i$, we consider the biological reactions used in [58]. These terms are listed in Table 1 in Appendix A for reference. A schematic of the flow is shown in Figure 5, where L and H are the channel length and height, respectively. The no-slip boundary condition is enforced at the top and bottom walls while the outflow boundary condition is enforced downstream. At the inlet a parabolic velocity with the average inlet velocity of $\overline{w}$ is prescribed. The Reynolds number based on reference length of half the height $(H/2)$ and the kinematic viscosity ν is $Re = \overline{w}H/2\nu = 1000$. The inlet boundary condition is $\mathbf{v}_i(0, x_2, t) = 1/2\big(\tanh{(x_2 + H/2)/\delta} - \tanh{(x_2 - H/2)/\delta}\big)$ for all species, where $\delta = 0.1$.

The velocity field is governed by two-dimensional incompressible Navier-Stokes equation. We solved the velocity field once as it is independent from the species using spectral/hp elements method with 4008 quadrilateral elements and polynomial order 5. For more details on the spectral element method see for example [44, 8, 12] . We then solve the species transport equations and f-OTD equations in the rectangular domain shown by dashed lines in Figure 5. In the rectangular domain, we used structured spectral elements with 50 elements in x_1 direction and 15 elements in x_2 direction. We used spectral polynomial of order 5 in each direction. The velocity field was interpolated onto this grid. The species transport equation and f-OTD equations, which are presented in the next sections, are integrated forward in time using RK4 with $\Delta t = 5 \times 10^{-4}$.

2.5.3.2 f-OTD Formulation for Tensor

Our goal is to calculate sensitivity of the species concentration with respect to the reaction parameters $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_{n_r})$, where n_r is the number of reaction parameters. To this end, we take the derivative of the above equation with respect to reaction parameter α_j to obtain an evolution equation for the sensitivity:

$$\frac{\partial \tilde{\mathbf{v}}'_{ij}}{\partial t} + (\mathbf{w} \cdot \nabla)\, \tilde{\mathbf{v}}'_{ij} = \tilde{\kappa}_{ik} \nabla^2 \tilde{\mathbf{v}}'_{kj} + \tilde{\mathcal{L}}_{\mathbf{s}_{ik}} \tilde{\mathbf{v}}'_{kj} + \tilde{\mathbf{s}}'_{ij}, \tag{2.19}$$

where $\tilde{\mathbf{v}}'_{ij} = \partial \mathbf{v}_i / \partial \alpha_j \in \mathbb{R}^{\infty \times 1}$ is the sensitivity of the concentration of species $\mathbf{v}_i$ with respect to reaction rate α_j, $\tilde{\mathcal{L}}_{\mathbf{s}_{ik}} = \partial \mathbf{s}_i / \partial \mathbf{v}_k$ is the linearized reactive source term, and $\tilde{\mathbf{s}}'_{ij} = \partial \mathbf{s}_i / \partial \alpha_j$. In the above equation, $\tilde{\mathcal{L}}_{\mathbf{s}_{ik}} \tilde{\mathbf{v}}'_{kj}$ should be interpreted as a matrix-matrix multiplication for any (x_1, x_2) point in the physical space. In this notation, sensitivities are represented by a quasitensor i.e. $\tilde{\mathbf{V}}' = [\mathbf{v}'_{ij}]$ with $i = 1, 2, \ldots, n_s$ and $j = 1, 2, \ldots, n_r$, where $\tilde{\mathbf{V}}' \in \mathbb{R}^{\infty \times n_s \times n_r}$ is the third order quasitensor depicted in the left-hand side of Figure 2.5.3.2. Here $\tilde{\cdot}$ denotes terms associated with the tensor equation. In the discrete representation of $\tilde{\mathbf{V}}'$, the dimension ∞ is replaced with the number of grid points.

Solving for sensitivities involving $\tilde{\mathbf{v}}'_{ij}$ using adjoint would require solving n_s AEs: one adjoint field for each species. See for example [22, 57, 56]. However, it is important to note that these AEs are tied to a specific objective function and do not directly compute $\tilde{\mathbf{v}}'_{ij}$.

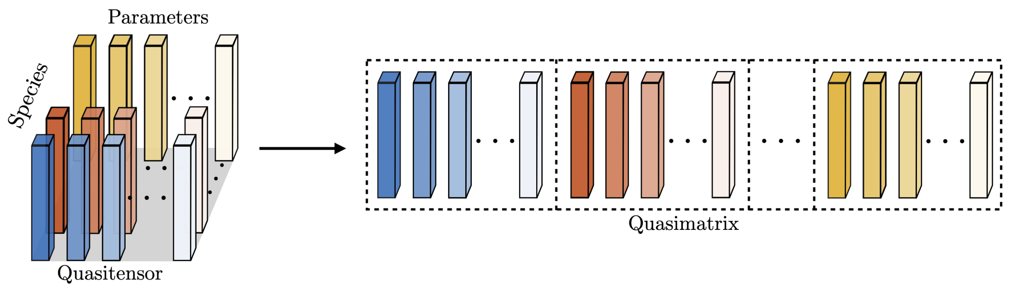
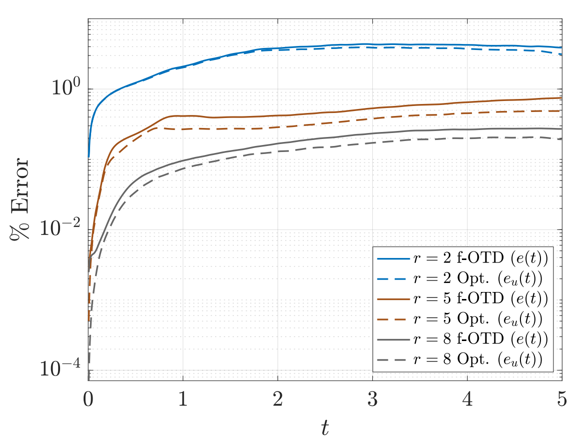


Figure 6: Schematic of the tensor flattening from a 3D quasitensor to a 2D quasimatrix.

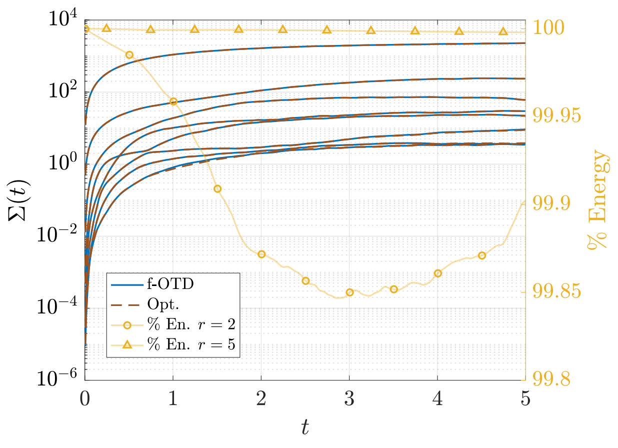
Consequently, each subsequent objective function would require solving another n_s AEs. To directly solve for $\tilde{\mathbf{v}}'_{ij}$ using f-OTD, one could also solve for n_s sets of f-OTD modes, i.e. one set of f-OTD modes for each species. This straightforward approach would only exploit the correlation between sensitivities of each species separately, i.e. correlations between $\mathbf{v}'_{ij}$ for a fixed i, while leaving the correlations between sensitivities of different species unexploited. In this example, we demonstrate how a single set of f-OTD modes can be used to accurately model the entire sensitivity tensor. Therefore, the compression ratio both in terms of memory and computational cost in comparison to the full sensitivity equation is r/d. In comparison to AE, the compression ratio is r/n_s. Also, the f-OTD is a forward system and does not impose any I/O operation. To this end, we flatten the sensitivity tensor, as shown in Figure 2.5.3.2, which results in a quasimatrix of size $\infty \times d$. Here, $d = n_s \times n_r$, where $n_s = 23$ and $n_r = 34$. This leads to a total of $d = 782$ sensitivity equations that we seek to compute. In Appendix B, we show that the flattened sensitivity evolution Equation is:

$$\frac{\partial \mathbf{v}'_m}{\partial t} + (\mathbf{w} \cdot \nabla)\mathbf{v}'_m = \kappa_{mn}\nabla^2\mathbf{v}'_n + \mathcal{L}_{\mathbf{s}_{mn}}\mathbf{v}'_n + \mathbf{s}'_m, \tag{2.20}$$

where $m(i,j) = j + (i-1)n_r$ and $n(i',j') = j' + (i'-1)n_r$, resulting in $m, n = 1, 2, \ldots, d$. Equation 2.19 is a tensor evolution equation, whereas Equation 2.20 is the equivalent matrix evolution equation. The tensor flattening carried out here is similar to the unfolding carried out in the Tucker tensor decomposition [50]. However, unlike Tucker tensor decomposition, we do not consider flattening the tensor in the other two dimensions of species and parameters. Each $\mathbf{y}_k(t)$ is a vector of size $(n_s n_r) \times 1$ and contains coefficients for species and



(a)



(b)

Figure 7: (a) Percent error plotted as a function of time. Error decreases as the number of modes r increases. (b) Singular values plotted as a function of time for $r = 8$.

parameters. Once the sensitivity tensor is flattened to a quasimatrix, we use f-OTD to extract low-rank structure from the quasimatrix. In Equation 2.20, the linear operator changes from one species to the other due to the different diffusion coefficients κ_{mn}. In Appendix B we show how f-OTD evolution equations can be derived for this case, which are different from the previous demonstration cases.

We solve Equations B.2 and B.3 for different f-OTD ranks along with the species transport (Equation 3.29). In Figure 7(a) the f-OTD error ($e(t)$) and optimal low-rank approximation error ($e_u(t)$) are shown using three different ranks of $r = 2, 5$ and 8. Again, we observe that the growth of $e(t)$ surpasses $e_u(t)$ for long term integration as a direct result of the lost interactions with the unresolved modes. However, with only 5-8 modes, we have shown that f-OTD can approximate 782 sensitivities with error on the order of 0.1%. These results can be explained by studying Figure 7(b), where we observe that more than 99% of the system energy is captured by the reduction. The % energy is calculated from the singular values as % En. $= \sum_{i=1}^{r} \sigma_i^2 / \sum_{i=1}^{d} \sigma_i^2 \times 100$, and can be used to get a sense of the dimensionality of the system, when expressed in the time-dependent basis. Since the system is truly low-dimensional in the time-dependent basis, the f-OTD algorithm is

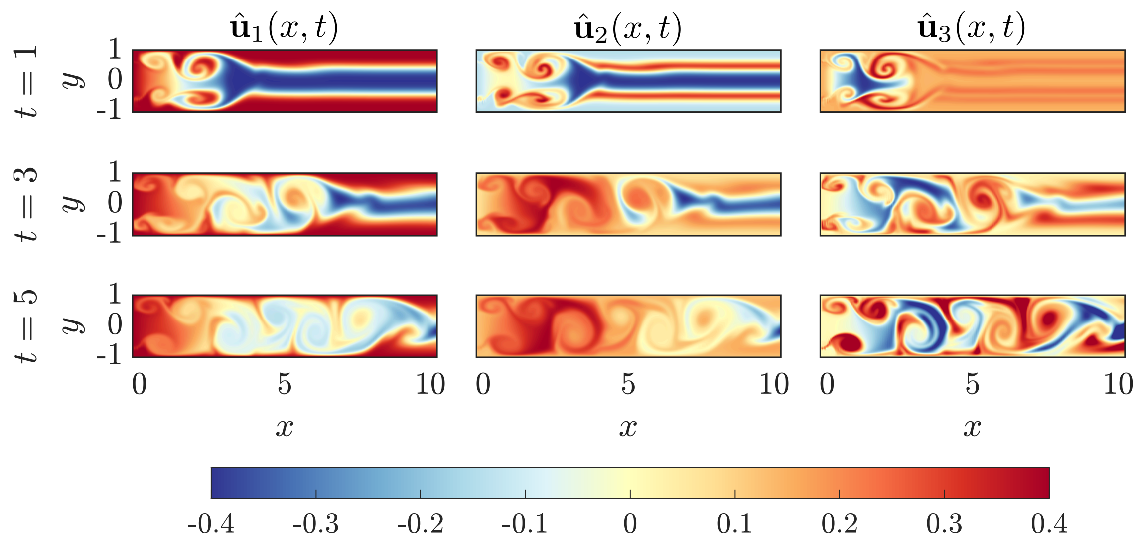


Figure 8: First three orthonormal f-OTD modes shown for $r = 8$. Each row shows the modes at a different instance in time.

able to extract the latent features associated with the most dominant singular values and successfully approximate the full sensitivity tensor with a high degree of accuracy.

In Figure 8, the time-dependent evolution of the three most dominant f-OTD modes are shown. These modes are energetically ranked where low mode numbers correspond to larger (higher energy) structures and high mode numbers correspond to finer (lower energy) structures in the flow. As opposed to static basis, such as POD or DMD, the f-OTD modes evolve with the flow and exploit the instantaneous correlations between sensitivities. While this system is low-dimensional in the time-dependent basis, when expressed in POD or DMD basis, the system is high-dimensional and many modes are needed to capture the complex spatio-temporal evolution of $\mathbf{V}'$. See reference [7] for comparison between time-dependent basis versus POD and DMD and see reference [18] for a recent review of ROM techniques.

To demonstrate the interpretability of the f-OTD decomposition, we show how the hidden parameter space represented by $\hat{\mathbf{Y}}(t)$ can be used to identify the most important reaction parameters. In this context, importance refers to a parameter for which a small change in its value elicits a large change in the response of the system (i.e. highly sensitive). To

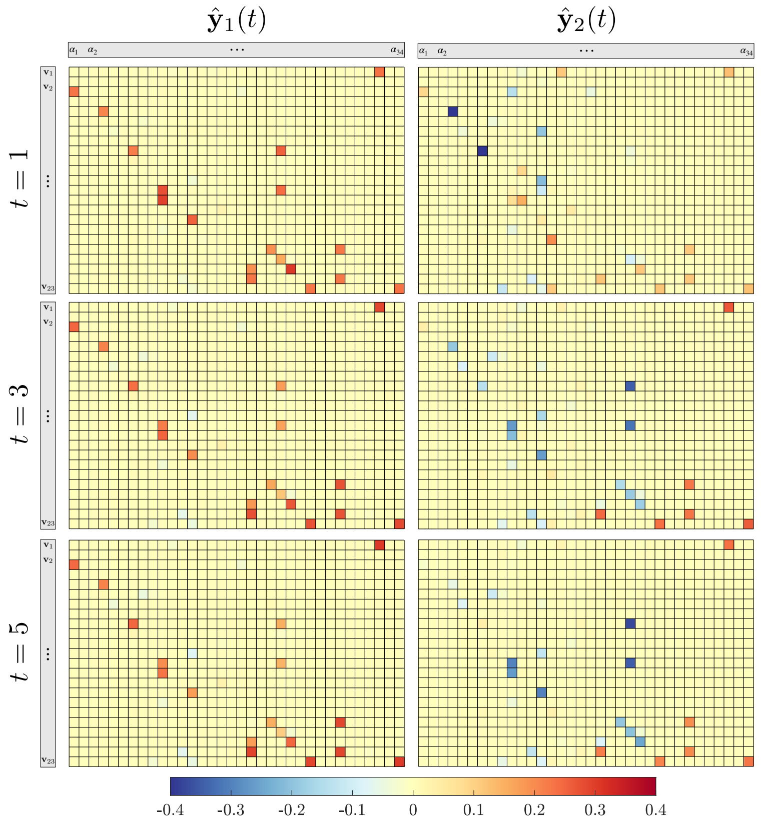


Figure 9: Orthonormalized f-OTD coefficients $\hat{\mathbf{y}}_1(t)$ and $\hat{\mathbf{y}}_2(t)$ visualized as a matrix with rows corresponding to species concentration and columns corresponding to reaction parameters. Color map shows most dominant sensitivities at different time instances.

demonstrate this capability of f-OTD, the first two sensitivity coefficients are visualized as matrices in Figure 9, where each $\hat{\mathbf{y}}_i$ is a $d \times 1$ vector that has been reshaped into an $n_s \times n_r$ matrix. In this form, each $\mathbf{v}'_{ij}$ is visualized using a heat map of the sensitivity coefficients, with rows corresponding to species i and columns corresponding to reaction parameter j. Using this heat map, Figure 9 shows that only a handful of sensitivities are non-zero, while the majority have zero contribution for the entire duration of the simulation.

3.0 Low-Rank Approximation of Nonlinear Sensitivity

As demonstrated in the previous chapter, sensitivities are typically computed with the assumption of *infinitesimal* perturbations to the base state, and are evolved either directly under the action of the linearized system, via the Jacobian, or indirectly by solving an adjoint equation. However, for many practical problems of interest, these perturbations can undergo transient amplification and require the full nonlinear dynamics to accurately describe their evolution, and characterize the short-term behavior of the system.

For example, transition in wall bounded shear flows is a notorious problem that exhibits transient behavior via non-normal growth of infinitesimal perturbations to the base state. A common approach for characterizing these perturbations is known as linear stability analysis, also referred to as modal analysis. This approach involves computing the eigenvalues of the linearized operator to determine if perturbations to a given base flow will grow or decay in time. While linear stability analysis has successfully been used to predict asymptotic stability for canonical flows, e.g., plane Poiseuille flow [41], Poiseuille flow in a circular pipe [80], and plane Couette flow [36], it does not account for short-term energy amplifications that give rise to transient instability. On the other hand, linear nonmodal analysis (LNMA) has effectively been used to characterize short-term energy amplifications of optimal perturbations that can transition the flow to a turbulent state [82]. However, this approach is limited to capturing the initial (linear) growth mechanism, and is not sufficient to describe the nonlinear mechanisms responsible for triggering turbulent flow patterns. More recently, an approach known as nonlinear nonmodal analysis (NLNMA) was developed that considers the full nonlinear system for *finite* perturbations to the base state. Similar to LNMA, NLNMA seeks to find the optimal perturbation to the base state that maximizes energy amplification after a specified period of time. Both LNMA and NLNMA amount to a non-convex optimization problem that requires solving the Navier-Stokes adjoint equation backward in time at each iteration. While this is certainly the preferred approach for steady base flows, the high input output (I/O) overhead required for time dependent base flows could be prohibitively expensive [1]. For additional details on NLNMA, see [46] for a recent review.

In this chapter, we consider the evolution of *finite* parametric perturbations for an arbitrarily time dependent base state. Similar to existing Jacobian-free methods [62, 40, 77], perturbations are evolved via nonlinear equations, however, we do not limit our analysis to the initial value problem. To this end, we compute the nonlinear sensitivity of a parameterized dynamical system as the perturbation to the base state (output) relative to a perturbation in the input parameter. While perturbing each parameter and solving forward evolution equations for the resulting perturbation fields might be manageable for relatively small systems, this approach quickly becomes impractical, or even impossible, as the number of parameters and fields increases.

Motivated in part by the success of f-OTD for effectively computing linear sensitivities, the objective of this chapter is to extend f-OTD to low-rank approximation of finite-time nonlinear sensitivities in cases where the linear system cannot capture important phenomena in the presence of strong nonlinear interactions. While the f-OTD decomposition can effectively be used to approximate nonlinear sensitivities, we adopt a new approach similar to the dynamically/bi-orthonormal(DBO) decomposition that was recently presented in [74]. This new approach for the low-rank approximation of nonlinear sensitivities via TDB is given the name nonlinear f-OTD, which will simply be reffered to as NL-fOTD. While f-OTD and DBO offer mathematically equivalent reductions, the DBO formulation boasts a significant increase in numerical performance, making it the preferred approach for a wide range of systems. For nonlinear sensitivities in a variety of systems with arbitrarily time dependent base state, we demonstrate that (i) low-rank structure exists and (ii) the low-rank structure can be extracted in real-time via closed-form low-rank evolution equations, Jacobian-free.

3.1 Notation and Definitions

Let $u(x,t)$ be a time dependent function defined on the spatial domain $D \subset \mathbb{R}^m$, where $m = 1$, 2, or 3. The spatial coordinate is denoted by $x \in D$ and t is time. We introduce a quasimatrix notation to represent a set of functions in matrix form, and denote the

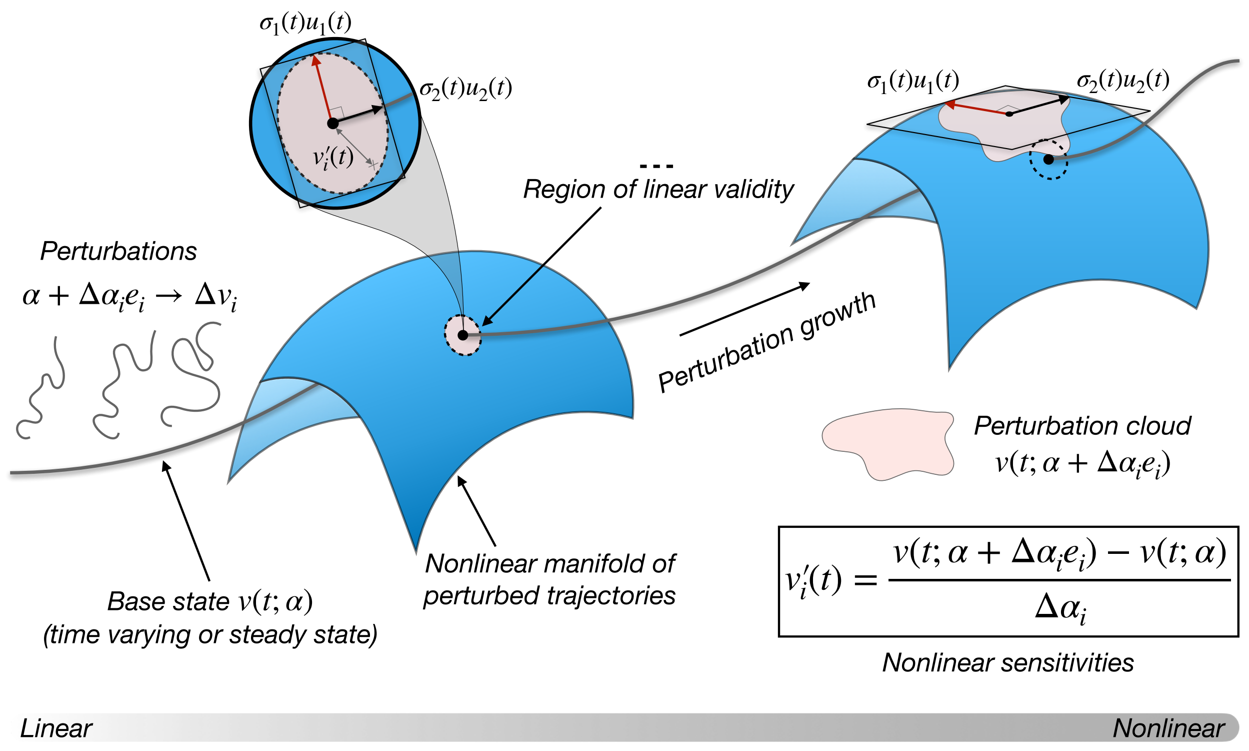

$\sigma_1(t)u_1(t)$
$\sigma_2(t)u_2(t)$
$v_i'(t)$
Region of linear validity
$\sigma_1(t)u_1(t)$
$\sigma_2(t)u_2(t)$
Perturbations
$\alpha + \Delta\alpha_i e_i \rightarrow \Delta v_i$
Perturbation growth
Perturbation cloud
$v(t; \alpha + \Delta\alpha_i e_i)$
Nonlinear manifold of perturbed trajectories
Base state $v(t;\alpha)$ (time varying or steady state)
$v_i'(t) = \dfrac{v(t;\alpha+\Delta\alpha_i e_i) - v(t;\alpha)}{\Delta\alpha_i}$
Nonlinear sensitivities
Linear
Nonlinear

Figure 10: Evolution of finite perturbations. Perturbed states are initially close to the base state and are well approximated by the linear dynamics. In time, perturbations undergo significant growth, causing large deviations from the base state. Therefore, nonlinear interactions must be considered in order to accurately describe the evolution of the perturbations.

quasimatrix $U(x,t) \in \mathbb{R}^{\infty \times r}$ as [15]:

$$U(x,t) = \Big[u_1(x,t) \;\Big|\; u_2(x,t) \;\Big|\; \dots \;\Big|\; u_d(x,t) \Big]_{\infty \times r},$$

where the first dimension is infinite and represents the continuous state space contained by D and the second dimension is discrete. The column-wise inner product of two quasimatrices $U(x,t) \in \mathbb{R}^{\infty \times r}$ and $V(x,t) \in \mathbb{R}^{\infty \times d}$ is defined as

$$S(t) = \langle U(x,t), V(x,t) \rangle,$$

where $S(t) \in \mathbb{R}^{r \times d}$ is a matrix with components

$$S_{ij}(t) = \int_D u_i(x,t) v_j(x,t) dx,$$

where $u_i(x,t)$ and $v_j(x,t)$ are the ith and jth columns of $U(x,t)$ and $V(x,t)$, respectively. The discrete analogue of this operation is the matrix multiplication, $U(t)^T W V(t)$, where $U(t) \in \mathbb{R}^{n \times r}$ and $V(t) \in \mathbb{R}^{n \times d}$ are space discrete with n grid points and $W \in \mathbb{R}^{n \times n}$ is a diagonal weight matrix. For the case of single-column quasimatrices, i.e., functions, the above definition reduces to an inner product between two functions, which induces an L_2 norm:

$$\langle u(x,t), v(x,t) \rangle = \int_D u(x,t) v(x,t) dx, \qquad \|u(x,t)\|_2 = \langle u(x,t), u(x,t) \rangle^{\frac{1}{2}}.$$

The Frobenius norm of a quasimatrix is defined as:

$$\Big\| V(x,t) \Big\|_F = \left(\text{trace} \langle V(x,t), V(x,t) \rangle \right)^{\frac{1}{2}}.$$

3.2 Mathematical Formulation

A general nonlinear system with parametric dependence can be represented as

$$\frac{\partial v(x,t;\alpha)}{\partial t} = \mathcal{M}(v(x,t;\alpha);\alpha), \tag{3.1}$$

where $\mathcal{M}$ is the nonlinear operator and $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_d)^T$ are the parameters. In general, α can be both space and time dependent, i.e. $\alpha(x,t)$, however, for sake of simplicity in the exposition, we take α to be constant. While perturbations can be introduced via a multitude of pathways, we consider independent finite perturbations to parameterized initial conditions and/or governing equations, $\alpha + \Delta\alpha_i e_i$, where $\Delta\alpha_i$ is the perturbation magnitude, $e_i \in \mathbb{R}^d$ is the standard unit vector in the direction of increasing α_i, and the repeated index i does not imply summation. These parametric perturbations result in corresponding perturbations to the base state, $v(x,t;\alpha) + \Delta v_i(x,t)$, where $v(x,t;\alpha)$ is an arbitrarily time dependent base state and $\Delta v_i(x,t) = v(x,t,;\alpha + \Delta\alpha_i e_i) - v(x,t;\alpha)$ is the perturbation to the base state resulting from $\Delta\alpha_i$. The parametric and base state perturbations are used to define the nonlinear sensitivity as

$$v_i'(x,t) = \frac{\Delta v_i(x,t)}{\Delta\alpha_i}. \tag{3.2}$$

In general, these perturbations are finite and v_i' is governed by the nonlinear sensitivity equation (NLSE). Dropping the explicit dependence on x and t, the NLSE is obtained by first considering the Taylor series expansion of Equation 3.1 around the base state $v(\alpha)$:

$$\frac{\partial v(\alpha + \Delta\alpha_i e_i)}{\partial t} = \mathcal{M}(v(\alpha);\alpha) + \frac{\partial \mathcal{M}(v(\alpha);\alpha)}{\partial v}\Delta v_i + \frac{\partial \mathcal{M}(v(\alpha);\alpha)}{\partial \alpha_i}\Delta\alpha_i + \text{h.o.t.}$$

Next, subtract Equation 3.1 from the above equation and divide by $\Delta\alpha_i$ to obtain the NLSE with respect to α_i

$$\frac{\partial v_i'}{\partial t} = \mathcal{L}v_i' + f_i + \text{h.o.t.}, \tag{3.3}$$

where $\mathcal{L} := \dfrac{\partial \mathcal{M}(v(\alpha);\alpha)}{\partial v}$ is the linear operator (i.e. the Jacobian once discretized), $f_i := \dfrac{\partial \mathcal{M}(v(\alpha);\alpha)}{\partial \alpha_i}$ is the forcing, and h.o.t is an aggregate of nonlinear and higher order terms. In the case of sensitivity with respect to initial condition, $f_i = 0$. From observation of Equation 3.3, the linear sensitivity can be recovered by taking $\Delta\alpha_i, v_i' \to 0$, such that h.o.t. $\to 0$.

While one can directly solve Equation 3.3, deriving and implementing this equation is a very intrusive process. In particular, computing the Jacobian, $\mathcal{L}$, can be very time consuming and prone to error, especially for systems with intricate source and boundary terms. Therefore, we seek solutions to Equation 3.3 by leveraging the nonlinear system in Equation 3.1, which allows existing numerical solver(s) for Equation 3.1 to be repurposed to obtain solutions to Equation 3.3. To this end, Equation 3.3 can be written equivalently as

$$\frac{\partial v_i'}{\partial t} = \frac{\mathcal{M}(v(\alpha + \Delta\alpha_i e_i); \alpha + \Delta\alpha_i e_i) - \mathcal{M}(v(\alpha); \alpha)}{\Delta\alpha_i}. \tag{3.4}$$

For the sake of brevity, we will simply denote the right hand side of Equation 3.4 as $\mathcal{M}_i'(v_i'; v(\alpha); \Delta\alpha_i)$, which can be solved non-intrusively by using the pre-existing nonlinear base solver as a black box. This type of approach is not new, and similar procedures have been used for approximating Jacobian-vector products via the nonlinear equations [48, 77].

3.3 Low-Rank NL-fOTD Decomposition

To obtain solutions to Equation 3.4, we utilize a low-rank decomposition similar in form to the DBO decomposition, a method that was developed to solve stochastic PDE's [72]. We present a variational principle, whose first-order optimality conditions lead to evolution equations for (1) $U(x,t)$: low-rank orthonormal spatial basis, (2) $\Sigma(t)$: low-rank correlation matrix, and (3) $Y(t)$: low-rank orthonormal parametric basis. We call this approach NL-fOTD. In this formulation, all three components are time-dependent, which allows for instantaneous correlations between nonlinear sensitivities to be extracted in real-time. Solutions to Equation 3.4 are approximated as

$$V'(x,t) \approx \sum_{j=1}^{r}\sum_{i=1}^{r} u_i(x,t)\Sigma_{ij}(t)y_j^T(t), \tag{3.5}$$

where $V'(x,t) = [v_1'(x,t)|v_2'(x,t)|\ldots|v_d'(x,t)] \in \mathbb{R}^{\infty\times d}$, $\Sigma(t) \in \mathbb{R}^{r\times r}$ is in general a full matrix, $U(x,t) = [u_1(x,t)|u_2(x,t)|\ldots|u_r(x,t)] \in \mathbb{R}^{\infty\times r}$, $Y(t) = [y_1(t)|y_2(t)|\ldots|y_r(t)] \in \mathbb{R}^{d\times r}$, and

r is the rank of the approximation. This low-rank approximation is augmented with the following orthonormality constraints on the spatial and parametric modes:

$$\langle u_i(x,t), u_j(x,t)\rangle = \delta_{ij} \tag{3.6a}$$

$$y_i(t)^T y_j(t) = \delta_{ij}. \tag{3.6b}$$

Together, Equations 3.4, 3.5, and 3.6a–3.6b are used to derive closed-form evolution equations for the NL-fOTD components.

3.3.1 Variational Principle

The NL-fOTD evolution equations are derived from a variational principle that seeks to minimize the residual of the low-rank approximation of Equation 3.4:

$$\mathcal{F}(\dot{U}(x,t), \dot{\Sigma(t)}, \dot{Y}(t)) = \left\| \frac{\partial (U(x,t)\Sigma(t)Y(t)^T)}{\partial t} - \mathcal{M}'\left(U(x,t)\Sigma(t)Y(t)^T; v(\alpha); \Delta\alpha\right) \right\|_F^2, \tag{3.7}$$

where the control parameters are evolution equations for the low-rank components: $\{\dot{U}(x,t), \dot{\Sigma}(t), \dot{Y}(t)\}$, where $\dot{(\sim)} := \mathrm{d}(\sim)/\mathrm{d}t$. The minimization is augmented with equality constraints by taking the time derivative of Equations 3.6a–3.6b:

$$\langle \dot{u}_i(x,t), u_j(x,t)\rangle + \langle u_i(x,t), \dot{u}_j(x,t)\rangle = 0, \tag{3.8a}$$

$$\dot{y}_i(t)^T y_j(t) + y_i(t)^T \dot{y}_j(t) = 0, \tag{3.8b}$$

and we define $\phi_{ij}(t) := \langle u_i(x,t), \dot{u}_j(x,t)\rangle$ and $\theta_{ij}(t) := y_i(t)^T\dot{y}_j(t)$. From the above equations, we see that ϕ_{ij} and θ_{ij} must be skew-symmetric, i.e. $\phi_{ij} = -\phi_{ji}$ and $\theta_{ij} = -\theta_{ji}$. It follows that the unconstrained optimization problem is given by

$$\mathcal{G}(\dot{U}(x,t), \dot{\Sigma(t)}, \dot{Y}(t)) = \left\| \frac{\partial (U(x,t)\Sigma(t)Y(t)^T)}{\partial t} - \mathcal{M}'\left(U(x,t)\Sigma(t)Y(t)^T; v(\alpha); \Delta\alpha\right) \right\|_F^2 \tag{3.9}$$
$$+ \sum_{i,j=1}^{r} \lambda_{ij}(t)\big(\langle u_i(x,t), \dot{u}_j(x,t)\rangle - \phi_{ij}(t)\big) + \sum_{i,j=1}^{r} \gamma_{ij}(t)\big(y_i(t)^T\dot{y}_j(t) - \theta_{ij}(t)\big),$$

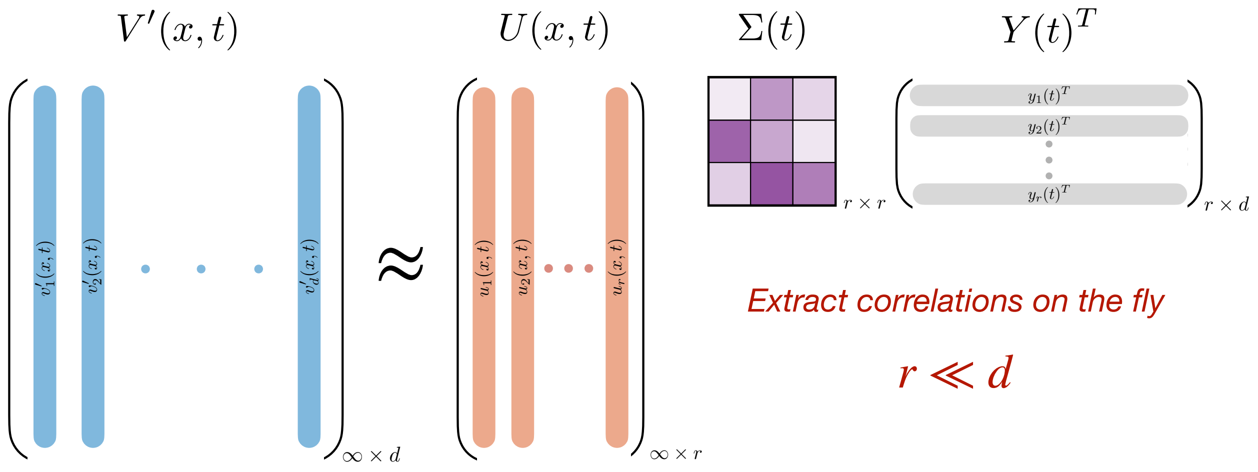


Figure 11: Schematic of the NL-fOTD decomposition. From left to right: full dimensional nonlinear sensitivities, time dependent orthonormal spatial basis, low-rank correlation matrix, and parametric basis. The NL-fOTD decomposition extracts correlations directly from a system's governing equations, on-the-fly, effectively bypassing the need to generate or collect massive amounts of data, which may not even be possible in the first place. The size of the reduction, $r \ll d$, exploits the low-rank structure of the nonlinear sensitivities of interest.

where $\lambda_{ij}(t)$ and $\gamma_{ij}(t)$ are Lagrange multipliers. Following a procedure similar to those presented in [72, 76], the first order optimality conditions of the variational principle leads to closed form evolution equations for the NL-fOTD components:

$$\frac{\partial U}{\partial t} = [\mathcal{M}'Y - U\langle U, \mathcal{M}'\rangle Y]\,\Sigma^{-1}, \tag{3.10a}$$

$$\frac{d\Sigma}{dt} = \langle U, \mathcal{M}'\rangle Y, \tag{3.10b}$$

$$\frac{dY}{dt} = \left[\langle \mathcal{M}', U\rangle - YY^T\langle \mathcal{M}', U\rangle\right]\Sigma^{-T}, \tag{3.10c}$$

where we have taken $\phi_{ij}(t) = \theta_{ij}(t) = 0$, a common choice known as the dynamically orthogonal condition [81], which has now been used for computing sensitivities [11, 31] and stochastic reduced order modeling [7, 72].

3.3.2 Mode Ranking

In their current form, the spatial and parametric NL-fOTD modes are not energetically ranked. To accomplish this, one can compute the singular value decomposition (SVD) of the $\Sigma(t)$ matrix so that

$$\Sigma(t) = R_U(t)\tilde{\Sigma}(t)R_Y^T(t), \tag{3.11}$$

where $\Sigma(t)$ is a diagonal matrix containing the ranked singular values: $\sigma_1(t) > \sigma_2(t) > \cdots > \sigma_r(t)$, and $R_U(t)$ and $R_Y(t)$ are the left and right singular vectors, respectively. It follows that the ranked spatial and parametric NL-fOTD modes can be obtained as:

$$\tilde{U}(x,t) = U(x,t)R_U(t), \tag{3.12a}$$

$$\tilde{Y}(t) = Y(t)R_Y(t), \tag{3.12b}$$

where $R_U(t)$ and $R_V(t)$ are orthogonal rotation matrices that orient the NL-fOTD modes along the most energetic directions of the system. Together, the ranked NL-fOTD components, $\{\tilde{U}(x,t), \tilde{\Sigma}(t), \tilde{Y}(t)\}$, approximate the SVD of the full dimensional system, in its canonical form. In the following sections, we compare our results with the instantaneous SVD of $V'(x,t)$, and refer to this as the "optimal" reduction, since it represents the best rank-r linear approximation in the l_2 sense.

3.3.3 Equivalence of f-OTD and NL-fOTD

It is easy to show that the f-OTD and NL-fOTD decompositions yield a mathematically equivalent reduction in the sense that they span the same r-dimensional subspace. However, the main advantage of NL-fOTD is improved numerical performance in the presence of small singular values. Since an accurate approximation must resolve the system up to a small singular value threshold, the system can become ill-conditioned depending on the choice of decomposition. Evolving the f-OTD decomposition requires inversion of the reduced correlation matrix, $C = Y^T Y$, with condition number $\lambda_{max}/\lambda_{min}$, where λ_{max} and λ_{min} are the maximum and minimum eigenvalues of C, respectively. On the other hand, evolving the NL-fOTD decomposition requires inversion of Σ, which has a condition number of $\sqrt{\lambda_{max}/\lambda_{min}}$. As a result, NL-fOTD exhibits better numerical performance when there is a large disparity between the leading and trailing eigenvalue, as is the case for many practical problems of interest.

Lemma 3.3.1. *Let $\{U_{NL}(x,t), \Sigma_{NL}(t), Y_{NL}(t)\}$ and $\{U_{f\text{-}OTD}(x,t), Y_{f\text{-}OTD}(t)\}$ be equivalent NL-fOTD and f-OTD decompositions with components equated via the linear transformations: $U_{f\text{-}OTD}(t) = U_{NL}(t)R_U$ and $Y_{f\text{-}OTD}(t) = Y_{NL}(t)Q_Y(t)$. Then the following is true: (i) R_U is an orthogonal matrix, (ii) $Q_Y(t) = \Sigma_{NL}^T(t)R_U$, and (iii) $\frac{dR_U}{dt} = 0$.*

Proof. (i) Starting from the equivalence relation of the spatial modes, project both sides of the equation onto U_{NL} to obtain an expression for R_U:

$$U_{f\text{-}OTD} = U_{NL}R_U,$$
$$R_U = \langle U_{NL}, U_{f\text{-}OTD}\rangle. \tag{3.13}$$

Conversely, we can project both sides of the equation onto $U_{f\text{-}OTD}$ to get

$$I = \langle U_{f\text{-}OTD}, U_{NL}\rangle R_U,$$

where I is the $r \times r$ identity matrix. Multiplying the above equation by R_U^{-1} from right yields

$$R_U^{-1} = \langle U_{f\text{-}OTD}, U_{NL}\rangle.$$

From the definition of the inner product of quasi-matrices, it is easy to see that the right hand side of the above equation is the transpose of 3.13. It follows that $R_U^{-1} = R_U^T$, thus R_U is an orthogonal matrix.

(ii) The notion of equivalent decompositions requires that $U_{NL}\Sigma_{NL}Y_{NL}^T = U_{f\text{-}OTD}Y_{f\text{-}OTD}^T$. Using the equivalence relations $U_{f\text{-}OTD} = U_{NL}R_U$ and $Y_{f\text{-}OTD} = Y_{NL}Q_Y$, we obtain the following expression:

$$U_{NL}\Sigma_{NL}Y_{NL}^T = U_{NL}R_U Q_Y^T Y_{NL}^T.$$

Projecting both sides of the above equation onto U_{NL} and multiplying by Y_{NL} from right, we get

$$R_U Q_Y^T = \Sigma_{NL}.$$

Finally, multiply the above equation by R_U^T from left and transpose the resulting equation to obtain

$$Q_Y = \Sigma_{NL}^T R_U, \tag{3.14}$$

where Q_Y is the linear transformation that maps Y_{NL} to $Y_{f\text{-}OTD}$.

(iii) Starting from 3.4 for the evolution of the nonlinear sensitivities,

$$\frac{\partial V'}{\partial t} = \mathcal{M}'(V'), \tag{3.15}$$

we substitute the approximation $U_{f\text{-}OTD}Y_{f\text{-}OTD}^T$ for V' into the above equation and get

$$\frac{\partial U_{f\text{-}OTD}}{\partial t}Y_{f\text{-}OTD}^T + U_{f\text{-}OTD}\frac{dY_{f\text{-}OTD}^T}{dt} = \mathcal{M}'. \tag{3.16}$$

Projecting the above equation onto $U_{f\text{-}OTD}$ yields an evolution equation for $Y_{f\text{-}OTD}^T$

$$\frac{dY_{f\text{-}OTD}^T}{dt} = \langle U_{f\text{-}OTD}, \mathcal{M}'\rangle, \tag{3.17}$$

where the dynamically orthogonal condition causes the first term above to vanish. Next, we multiply 3.16 by $Y_{f\text{-}OTD}$ from right and substitute 3.17 to get:

$$\frac{\partial U_{f\text{-}OTD}}{\partial t}C_{f\text{-}OTD} + U_{f\text{-}OTD}\langle U_{f\text{-}OTD}, \mathcal{M}'\rangle Y_{f\text{-}OTD} = \mathcal{M}'Y_{f\text{-}OTD}, \tag{3.18}$$

where $C_{f\text{-}OTD} = Y_{f\text{-}OTD}^T Y_{f\text{-}OTD}$ is the reduced correlation matrix. Multiplying the above equation by $C_{f\text{-}OTD}^{-1}$ from right and rearranging yields an evolution equation for $U_{f\text{-}OTD}$

$$\frac{\partial U_{f\text{-}OTD}}{\partial t} = \left[\mathcal{M}' Y_{f\text{-}OTD} - U_{f\text{-}OTD}\langle U_{f\text{-}OTD}, \mathcal{M}'\rangle Y_{f\text{-}OTD}\right] C_{f\text{-}OTD}^{-1}. \tag{3.19}$$

Substituting the equivalence relations $U_{f\text{-}OTD} = U_{NL}R_U$ and $Y_{f\text{-}OTD} = Y_{NL}Q_Y$ into the above equation gives

$$\frac{\partial U_{NL}}{\partial t} R_U + U_{NL}\frac{dR_U}{dt} = \left[\mathcal{M}' Y_{NL}Q_Y - U_{NL}\langle U_{NL}, \mathcal{M}'\rangle Y_{NL}Q_Y\right] C_{f\text{-}OTD}^{-1}. \tag{3.20}$$

Projecting both sides of the above equation onto U_{NL} causes the term in square brackets to vanish, resulting in

$$\frac{dR_U}{dt} = 0. \tag{3.21}$$

□

Theorem 3.3.2. *Let the NL-fOTD and f-OTD decompositions be equivalent at $t = 0$ with components equated via the linear transformations: $U_{f\text{-}OTD} = U_{NL}R_U$ and $Y_{f\text{-}OTD} = Y_{NL}Q_Y$. Then for all time $t > 0$, the DBO and f-OTD decompositions remain equivalent.*

Proof. Start by substituting the equivalence relation $Y_{f\text{-}OTD} = Y_{NL}Q_Y$ to obtain $C_{f\text{-}OTD}^{-1} = R_U^T \Sigma_{NL}^{-T}\Sigma_{NL}^{-1} R_U$. Substituting this expression for $C_{f\text{-}OTD}^{-1}$ into 3.20 along with 3.21, results in

$$\frac{\partial U_{NL}}{\partial t} R_U = \left[\mathcal{M}' Y_{NL}Q_Y - U_{NL}\langle U_{NL}, \mathcal{M}'\rangle Y_{NL}Q_Y\right] R_U^T \Sigma_{NL}^{-T}\Sigma_{NL}^{-1} R_U. \tag{3.22}$$

Finally, substitute 3.14 into the above equation and multiply R_U^T from right to get

$$\frac{\partial U_{NL}}{\partial t} = \left[\mathcal{M}' Y_{NL} - U_{NL}\langle U_{NL}, \mathcal{M}'\rangle Y_{NL}\right] \Sigma_{NL}^{-1}, \tag{3.23}$$

where we have made use of $R_U R_U^T = I$. From observation of the above equation, we see this is the same as the evolution equation for the NL-fOTD spatial modes in 3.10a. Following a similar procedure for the parametric modes, substitute the equivalence relations into the transpose of 3.17 to get

$$\frac{dY_{NL}}{dt}\Sigma_{NL}^T + Y_{NL}\frac{d\Sigma_{NL}^T}{dt} = \langle \mathcal{M}', U_{NL}\rangle. \tag{3.24}$$

Multiply the above equation by Y_{NL}^T from left to get

$$\frac{d\Sigma_{NL}^T}{dt} = Y_{NL}^T \langle \mathcal{M}', U_{NL} \rangle \tag{3.25}$$

where we have made use of the dynamically orthogonal condition. Finally, substitute the above equation into 3.24 and multiply by Σ_{NL}^{-T} from right to obtain

$$\frac{dY_{NL}}{dt} = \left[\langle \mathcal{M}', U_{NL} \rangle - Y_{NL} Y_{NL}^T \langle \mathcal{M}', U_{NL} \rangle \right] \Sigma_{NL}^{-T}. \tag{3.26}$$

Again, we observe this is the same equation as 3.10c that governs the evolution of the NL-fOTD parametric modes. Therefore, we have shown that the evolution of the NL-fOTD and f-OTD decompositions are equivalent, and thus, if initially equivalent, will remain equivalent for $t > 0$. $\square$

As we will show in Section 4, this approach, coupled with a sparse sampling algorithm, will enable computationally efficient solutions to Equation 3.4, without the intrusiveness of deriving and implementing Equation 3.3. Furthermore, in Section **??**, we discuss how this approach can easily be extended to approximate linear sensitivities, eliminating the need to derive and implement the linearized equations for complex systems.

3.4 Demonstration Cases

Here we present some preliminary results for computing nonlinear sensitivities using NL-fOTD. Our main objective is to show that if low-rank structure exists, it can be accurately extracted via low-rank evolution equations in real time. Therefore, at this stage, we do not employ any sparse sampling strategies to enable efficient computation of Equations 3.10a–3.10c, and the cost of solving the NL-fOTD equations is roughly equivalent to solving the full order model. In the next chapter, we present the sparse sampling strategy that can be used to reduce the computational cost.

3.4.1 Toy Problem

We start with a simple example to demonstrate the method. We consider the three-dimensional system of parameterized ordinary differential equations that was presented in [71]:

$$\dot{v}_1 = \alpha_4 v_1 - \alpha_3 v_2 - \alpha_1 v_1 v_3 - \alpha_2 v_1 v_2, \tag{3.27a}$$

$$\dot{v}_2 = \alpha_3 v_1 + \alpha_4 v_2 - \alpha_1 v_2 v_3 + \alpha_2 v_1^2, \tag{3.27b}$$

$$\dot{v}_3 = -\alpha_1 v_3 + \alpha_1 (v_1^2 + v_2^2), \tag{3.27c}$$

where $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4) > 0$. It was shown that if $\alpha_2^2 < \alpha_1/\alpha_4$, there is a limit cycle at $v_3 = \alpha_4/\alpha_1$ with period $T = 2\pi/\sqrt{1 - \alpha_2^2\alpha_4/\alpha_1}$ and fundamental frequency $\omega = \sqrt{1 - \alpha_2^2\alpha_4/\alpha_1}$. In our demonstration, we consider the model parameters $\alpha_1 = \alpha_4 = 0.2$, $\alpha_3 = 1$, and $\alpha_2 = 0$ which results in a limit cycle at $v_3 = 1$ with $T = 2\pi$ and $\omega = 1$. Our goal in this example is to independently perturb each α_i by $\Delta\alpha_i$ and compute nonlinear sensitivities using NL-fOTD. This results in a 3×4 matrix of sensitivities, V', with components v'_{ij} that correspond to the sensitivity of v_i with respect to α_j.

From above, it is easy to see that perturbing the model parameters results in perturbations to the location of the limit cycle, period, and fundamental frequency. We initialize the base state on the limit cycle at $(1, 0, 1)^T$ and independently perturb each parameter by constant $\Delta\alpha_i = 0.01$. For comparison, we also compute the linear sensitivities by taking h.o.t. = 0. In Figure 12(a), the optimal nonlinear, NL-fOTD rank $r = 2$, and optimal linear singular values are plotted versus time. We observe the two leading singular values of the linear and nonlinear system are matching while the perturbations are initially small and undergo linear growth. However, as the system is integrated for longer, there is separation as the perturbations grow and nonlinear effects become important. In Figure 12(b), reconstruction errors are reported for the NL-fOTD reduction of the nonlinear system for $r = 2$ along with the error of the optimal nonlinear reconstruction computed via the SVD with $r = 2$. Here, we observe the NL-fOTD reduction performs quite well with maximum error on the order of 1%. However, as with any TDB approach, the error of the NL-fOTD reconstruction is always greater than or equal to the error of the same-rank optimal reconstruction, due to the effect of unresolved modes in the evolution of Equation 3.10. Finally, we consider the effect of varying the magnitude of $\Delta\alpha_i$ on the computed nonlinear sensitivities. We take $\alpha + \Delta\alpha_1 e_1$ for different values of $\Delta\alpha_1$ and observe both the long-term statistics and instantaneous effects. In Figure 12(c), the instantaneous sensitivities are shown over one period (T) for different values of $\Delta\alpha_1$. We observe that even for small $\Delta\alpha_1$, the instantaneous sensitivities can behave nonlinearly (i.e. deviate from the linear sensitivity), even in this simple non-chaotic system. However, in Figure 12(d), we observe that the long-term statistics, i.e. time averaged sensitivities, behave smoothly (linearly) with $\Delta\alpha_1$, despite the nonlinear behavior of the instantaneous sensitivities.

3.4.2 Compressible Flow: Temporally Evolving Jet

In this example, we demonstrate the utility of the nonlinear low-rank approximation for stability analysis of a 2D compressible flow. Specifically, we consider the temporally evolving jet shown in Figure 14, governed by the non-dimensionalized 2D compressible Navier-Stokes equations:

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho v_i}{\partial x_i} = 0, \tag{3.28a}$$

$$\frac{\partial \rho v_i}{\partial t} + \frac{\partial \rho v_i v_j}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j}, \tag{3.28b}$$

$$\frac{\partial E}{\partial t} + \frac{\partial E v_i}{\partial x_i} = -\frac{\partial p v_i}{\partial x_i} + \frac{\partial \tau_{ij} v_i}{\partial x_j} - \frac{\partial q_i}{\partial x_i}, \tag{3.28c}$$

where the viscous and heat fluxes are given by

$$\tau_{ij} = \frac{1}{Re}\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3}\frac{\partial v_k}{\partial x_k}\delta_{ij}\right), \quad q_i = -\frac{1}{EcPe}\frac{\partial T}{\partial x_i}.$$

In the above equations, $E = \rho(e + \frac{1}{2}v_i v_i)$ is the total energy and e is the internal energy. These equations are augmented with periodic boundary conditions and the following initial

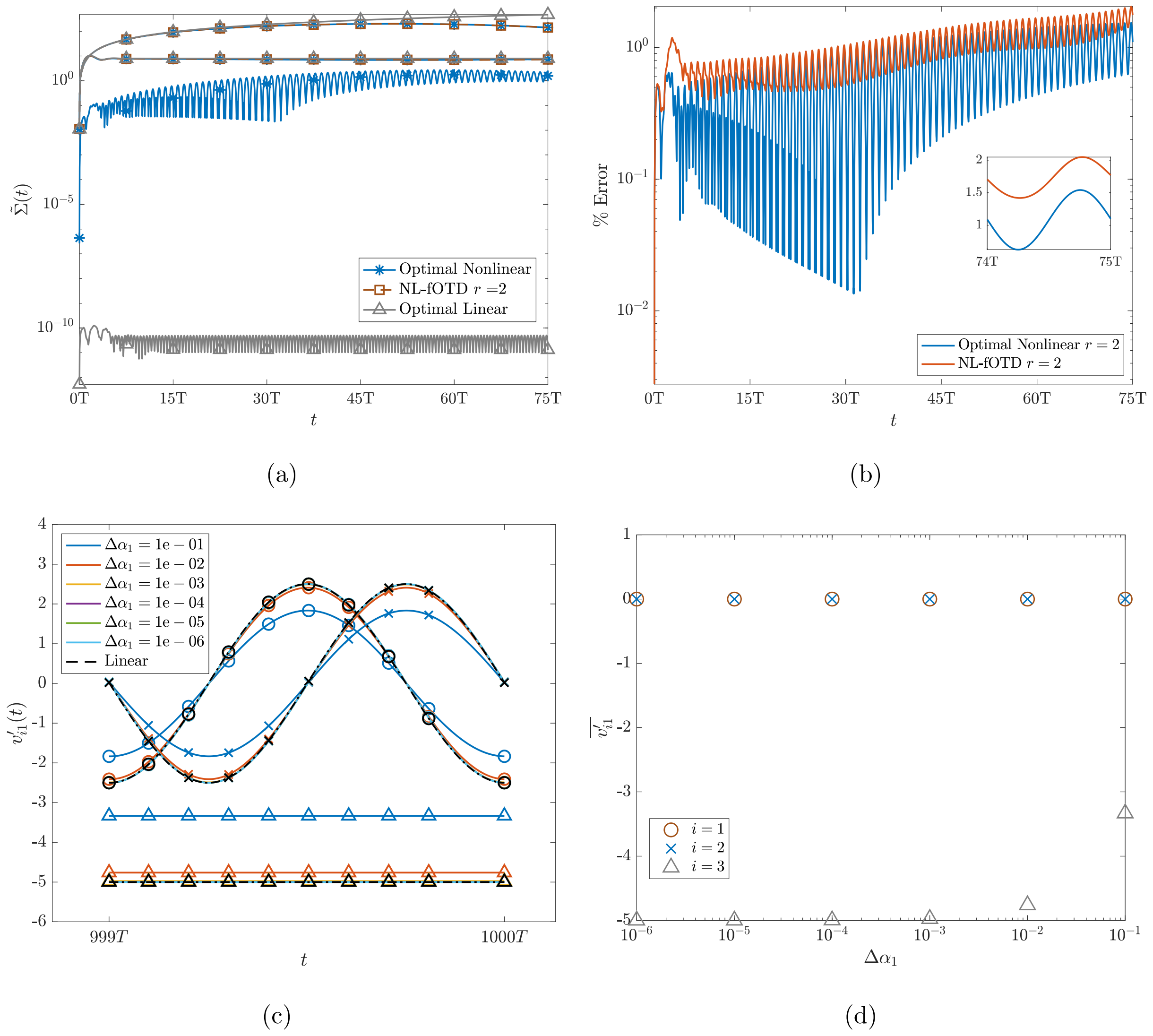


Figure 12: (a) Optimal nonlinear, NL-fOTD rank $r = 2$ approximation, and optimal linear singular values plotted versus time. (b) Error for rank $r = 2$ approximation plotted versus time for optimal nonlinear and NL-fOTD reconstructions. (c) Instantaneous nonlinear sensitivity plotted versus time for different values of $\Delta\alpha_1$. Markers distinguish sensitivities in each direction of the phase space: $i = 1$ ($\bigcirc$), $i = 2$ ($\times$), $i = 3$ ($\triangle$). (d) Time-averaged sensitivities, $\overline{v'_{i1}}$, in the i^{th} direction of the phase space, versus perturbation magnitude of $\Delta\alpha_1$.

conditions:

$$
\begin{aligned}
v_1(x,0) &= \frac{U_{max}}{2}(\tanh((x_2 - x_{2_{min}})/\delta) - \tanh((x_2 - x_{2_{max}})/\delta) - 1) + g(x_2)\sum_{n=1}^{d}\frac{\alpha_n}{A_n}\sin(2\pi n x_1/L_{x_1}),\\
v_2(x,0) &= h(x_2)\sum_{n=1}^{d} n\frac{\alpha_n}{A_n}\cos(2\pi n x_1/L_{x_1}),\\
T(x,0) &= \frac{1}{2} + \frac{1}{4}(\tanh((x_2 - x_{2_{min}})/\delta) - \tanh((x_2 - x_{2_{max}})/\delta) - 1),\\
P(x,0) &= 1,\\
\rho(x,0) &= \gamma Ma^2 P(x,0)/T(x,0),\\
e(x,0) &= \frac{P(x,0)}{(\gamma-1)\rho(x,0)},\\
E(x,0) &= \rho(x,0)(e(x,0) + \frac{1}{2}v_i(x,0)v_i(x,0))
\end{aligned}
$$

where,

$$
\begin{aligned}
g(x_2) &= \frac{2L_{x_1}}{\delta^2}\left[(x_2 - b)\exp(-(x_2-b)^2/\delta^2) + (x_2 - a)\exp(-(x_2-a)^2/\delta^2)\right],\\
h(x_2) &= 2\pi\left[\exp(-(x_2-b)^2/\delta^2) + \exp(-(x_2-a)^2/\delta^2)\right],
\end{aligned}
$$

initially localize the perturbations in the shear layers of the jet. The scalar value A_n is given by $A_n = 40 * \max_{x\in D}\left[(g(x_2)\alpha_n \sin(2\pi n x_1/L_{x_1}))^2 + (h(x_2) n\alpha_n \cos(2\pi n x_1/L_{x_1}))^2\right]^{1/2}$, and the repeated index does not imply summation. We consider the case where $Re = 8,000$ and $Ma = 0.5$. We take $\alpha_n = 0$, with the exception $\alpha_5 = 1$, and these initial conditions give rise to a time-dependent base flow resulting in a train of vortices. Our goal in this example is to compute nonlinear sensitivities using NL-fOTD for independent perturbations to the initial condition via the parameters α_n. We take $\Delta\alpha_n = 1$ and compute the corresponding sensitivities of the conservative states: $\rho'_n, (\rho v_1)'_n, (\rho v_2)'_n$, and $(E)'_n$. The resulting velocity field sensitivities are initialized as

$$
\begin{aligned}
(v_1)'_n(x,0) &= \frac{g(x_2)}{A_n((n-n_0)^2+1)}\sin(2\pi n x_1/L_{x_1}),\\
(v_2)'_n(x,0) &= \frac{n h(x_2)}{A_n((n-n_0)^2+1)}\cos(2\pi n x_1/L_{x_1}),
\end{aligned}
$$

where $n \in \{1, 2, \ldots, d\} \backslash 5$, $d = 50$, $n_0 = 7$, and the repeated index does not imply summation. We consider separate NL-fOTD approximations for the nonlinear sensitivities of each state, e.g., $\{U(x,t), \Sigma(t), Y(t)\}_{\rho'}$, $\{U(x,t), \Sigma(t), Y(t)\}_{(\rho v_1)'}$, etc. Figure 13 shows the singular values versus time for the sensitivities of each conservative state. Initially the perturbations are small, and we observe a good match between the nonlinear and linear systems that are predominantly driven by the linear dynamics. However, for $t > 1$, there is significant deviation between the leading singular values as the perturbation growth has caused the nonlinear effects to dominate. Although we observe deviation in the leading singular values, it is important to note that the lower energy singular values of the nonlinear system still closely follow those of the linear system for $t > 1$. This indicates that a subset of the NL-fOTD components have sufficiently small energy, and are primarily driven by the linear dynamics. To further elucidate the departure between the nonlinear and linear systems, we turn to the spatial and parametric modes. In Figure 14, we observe that the spatial modes of the nonlinear and linear systems are indistinguishable at $t = 1$, i.e., they span the same subspace. However, as the systems evolve, we observe their departure, and it is clear to see that two different subspaces emerge by $t = 5$. At this time, the linear modes are more localized in space with sharp defined structures, while the nonlinear modes are more widespread with soft blurred structures. These differences in structure can be explained by the nonlinear advection terms that drive the sensitivities at different velocities. As a result, the sensitivities collectively occupy a larger portion of the state space, and this is reflected in the shape of the nonlinear spatial modes. It is important to note that these modes are ranked based on energy and so the first mode represents the most energetic response of the system as a result of the IC perturbations. These modes are also time dependent and represent the instantaneously most unstable directions at any time t.

On the other hand, the $Y(t)$ modes are independent directions in the parametric space and represent the optimal linear combination of IC perturbations (from the given set) that elicit the largest growth at some time $t = T$. While the linear system only identifies a single frequency in the IC, the nonlinear system is more broad spectrum, finding a linear combination of IC's that leads to the largest growth. These findings are shown in Figure 15, and they highlight yet another facet in which NL-fOTD can be used to analyze the stability of a seemingly complex system in an interpretable manner.

Finally, we evaluate the performance of the NL-fOTD approximation by computing the reconstruction error for different values of the rank, r. The NL-fOTD reconstruction error is shown in Figure 16 along with the optimal reconstruction error computed via the truncated SVD. As the stable perturbations decay, the intrinsic dimension of the system decreases, which is reflected in the reconstruction error that decreases with time. We observe that increasing the NL-fOTD rank to $r = 10$ is able to capture 50 nonlinear sensitivities with error saturating to $\sim 1\%$.

3.4.3 Turbulent reacting flow

We revisit the species sensitivity problem from Section 2.5.3. We consider a 2D incompressible turbulent reacting flow with species $v_i(x,t)$ governed by:

$$\frac{\partial v_i}{\partial t} + w_k \frac{\partial v_i}{\partial x_k} = \tilde{\kappa}_{ik} \nabla^2 v_k + s_i, \tag{3.29}$$

where $w_i(x,t)$ is a turbulent flow field governed by the 2D incompressible Navier-Stokes equations and $s_i(v_1, v_2, \ldots, v_{n_s}; \alpha)$ is the reactive source term that nonlinearly couples the species transport equations. The goal in this example is to compute the nonlinear sensitivity, v'_{ij} of species v_i with respect to reaction parameter α_j. We proceed by perturbing each reaction parameter by $\Delta\alpha_i = 5\alpha_i$ and computing the resulting nonlinear sensitivities using the NL-fOTD approximation. In Figure 17(a), singular values are plotted for the optimal nonlinear, NL-fOTD with rank $r = 8$ and optimal linear systems. As with each subsequent demonstration, we observe an initial match between the nonlinear and linear singular values that quickly wanes in the presence of increasing nonlinear effects. However, unlike the previous example, Figure 18 shows that the nonlinear and linear spatial modes are strikingly similar to the naked eye, with minor discrepancies becoming detectable at higher mode numbers. This indicates that the linear and nonlinear sensitivities evolve in the same (very similar) subspace, but their evolution within that subspace deviates in time. This result is further elucidated by the parametric modes that describe the evolution of the sensitivities within the low-rank subspace. In Figure 19, the first two columns of $\tilde{Y}(t)$ have been reshaped

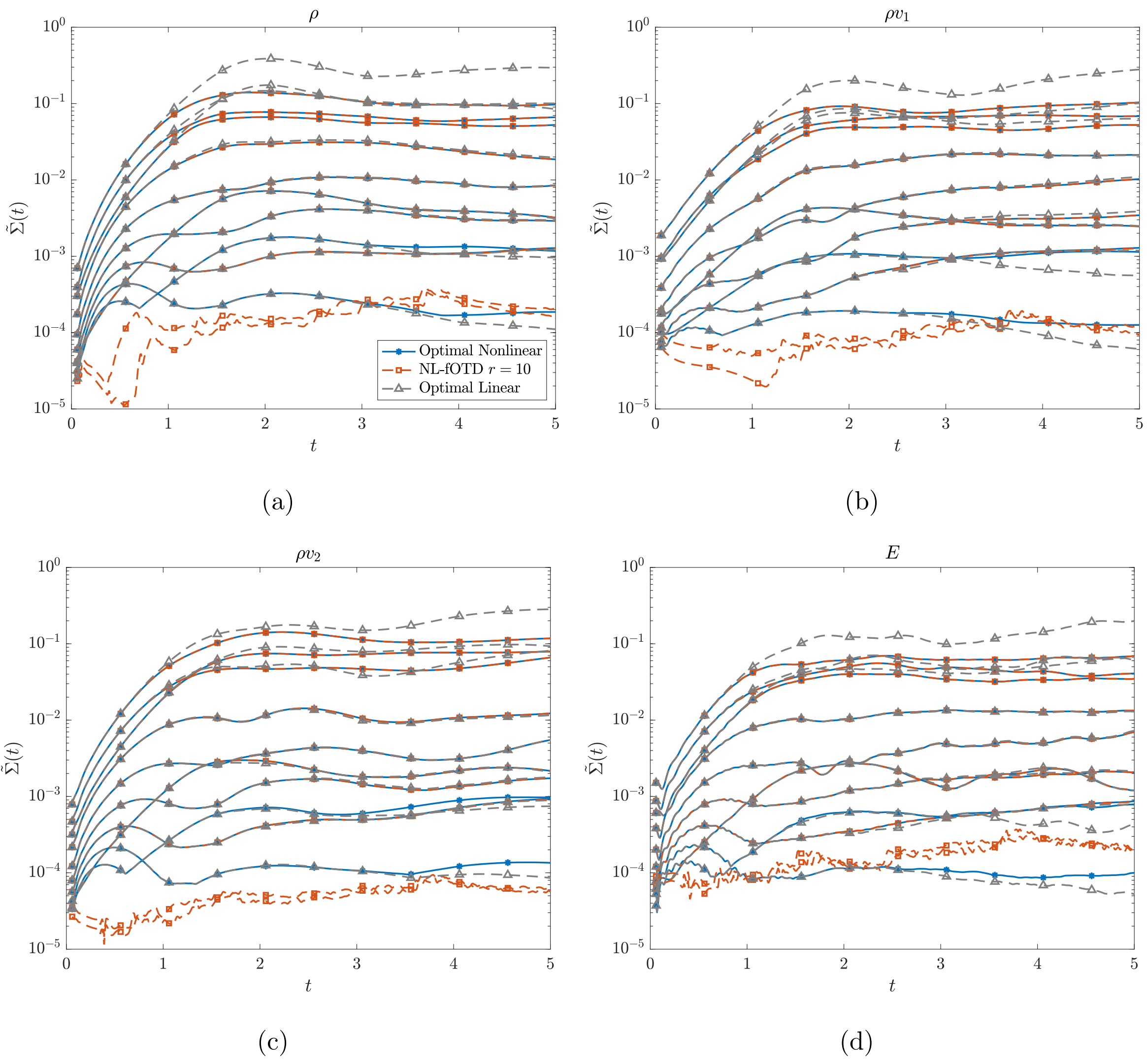


Figure 13: Compressible flow: singular values of the optimal nonlinear, NL-fOTD $r = 10$, and optimal linear sensitivities plotted versus time for (a) ρ (b) ρv_1 (c) ρv_2 (d) E.

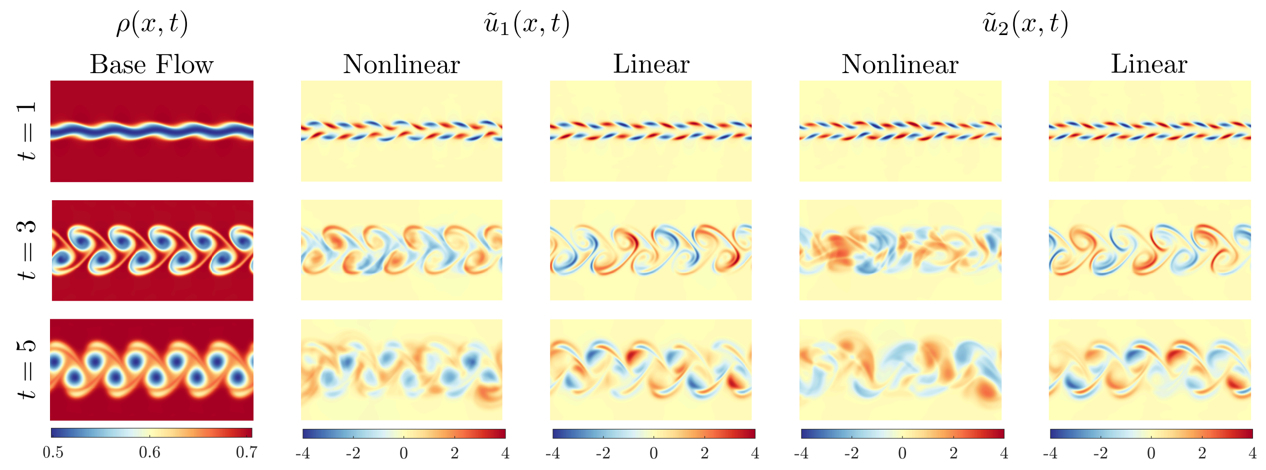


Figure 14: Compressible flow: first and second nonlinear and linear density modes shown at three different instances in time. Perturbation growth causes the nonlinear and linear system to depart in time. Base flow visualized with density in first column, showing development of vortex train.

from $(n_s n_p) \times 1$ column vectors to $n_s \times n_p$ matrices, with rows corresponding to species, and columns corresponding to parameters. Viewing the modes in this configuration provides a systematic and interpretable approach to identifying the most important parameters and species. As it turns out, there are only a handful of parameters that elicit a large change in the response of the system. While the nonlinear and linear system are almost initially identical, there are some minor differences that begin to emerge later in time. Finally, Figure 17(b) shows the reconstruction error of the NL-fOTD approximation for ranks $r = 2$, 5, and 8. With only $r = 5$, NL-fOTD is able to capture 782 nonlinear sensitivities with maximum error of $\sim 1\%$.

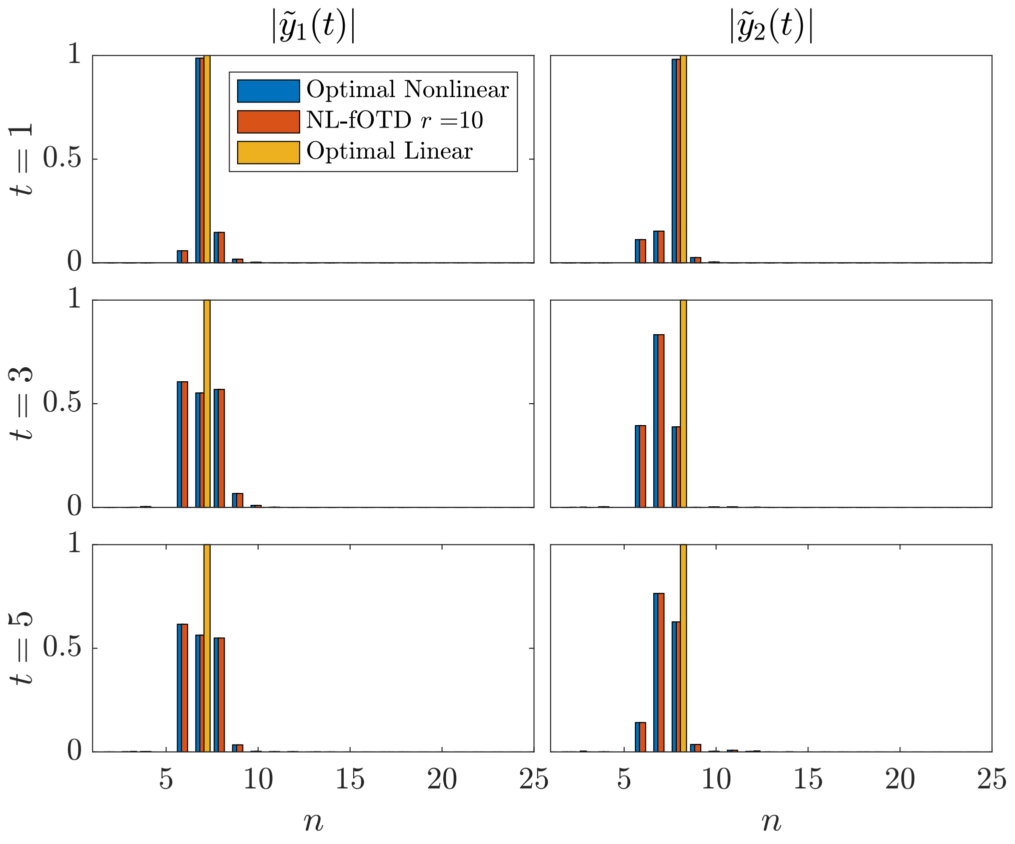
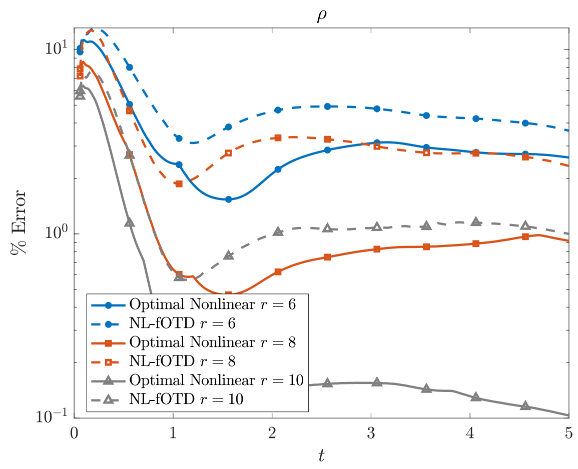
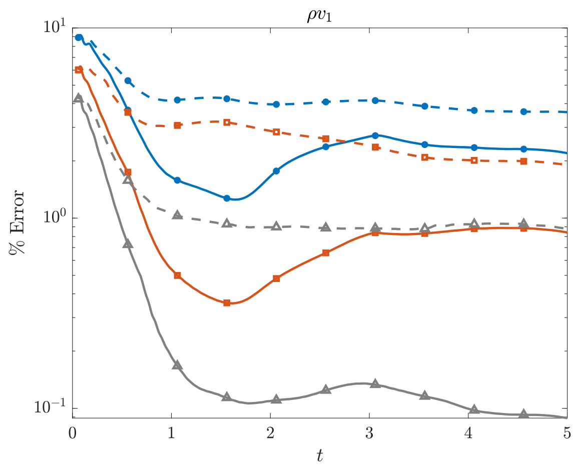


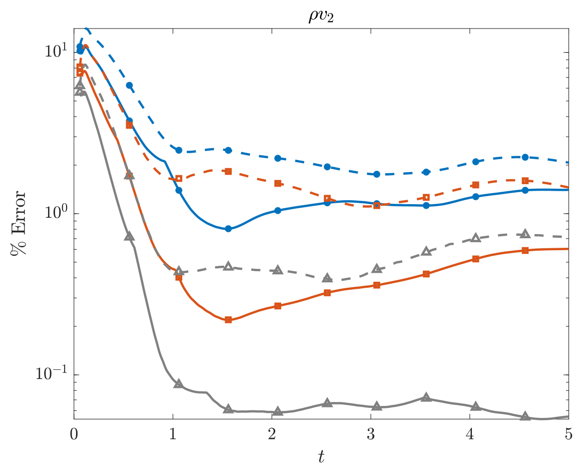
Figure 15: Compressible flow: first two parametric modes shown for optimal nonlinear, NL-fOTD $r = 10$, and optimal linear systems. Horizontal axis centered on $n \in [1, 25]$, with $|\tilde{y}_1(t)|$ and $|\tilde{y}_2(t)| \approx 0$ for $n > 25$.



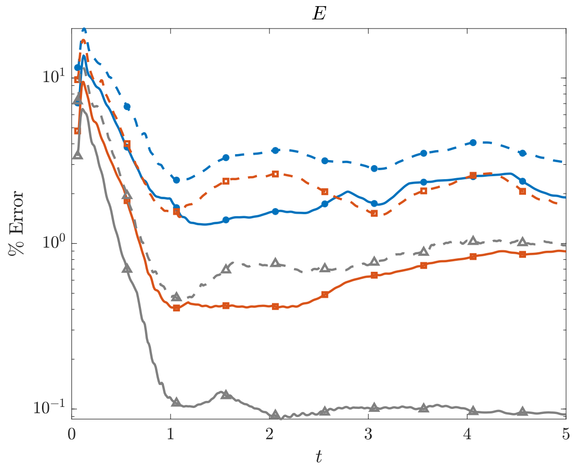
(a)



(b)

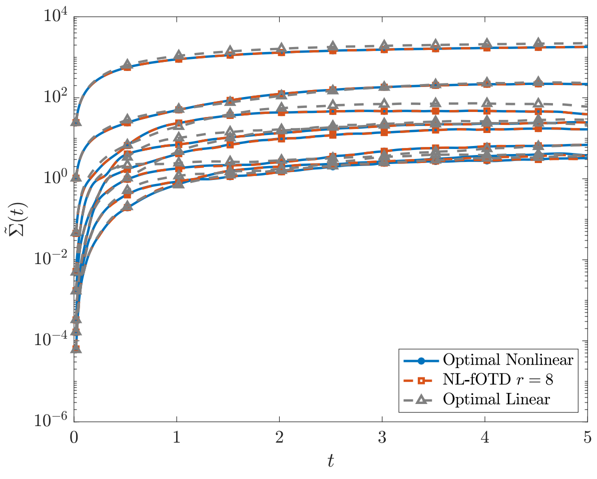


(c)

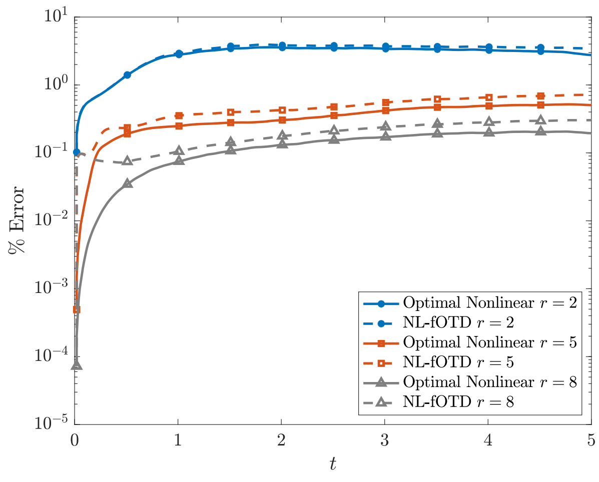


(d)

Figure 16: Compressible flow: reconstruction error plotted versus time for optimal reconstruction and NL-fOTD reconstruction for $r = 6,\ 8,$ and 10.



(a)



(b)

Figure 17: (a) Reacting flow: singular values versus time for optimal nonlinear, NL-fOTD $r = 8$, and optimal linear sensitivities. (b) Reconstruction error versus time for optimal nonlinear and NL-fOTD for $r = 2$, 5, and 8.

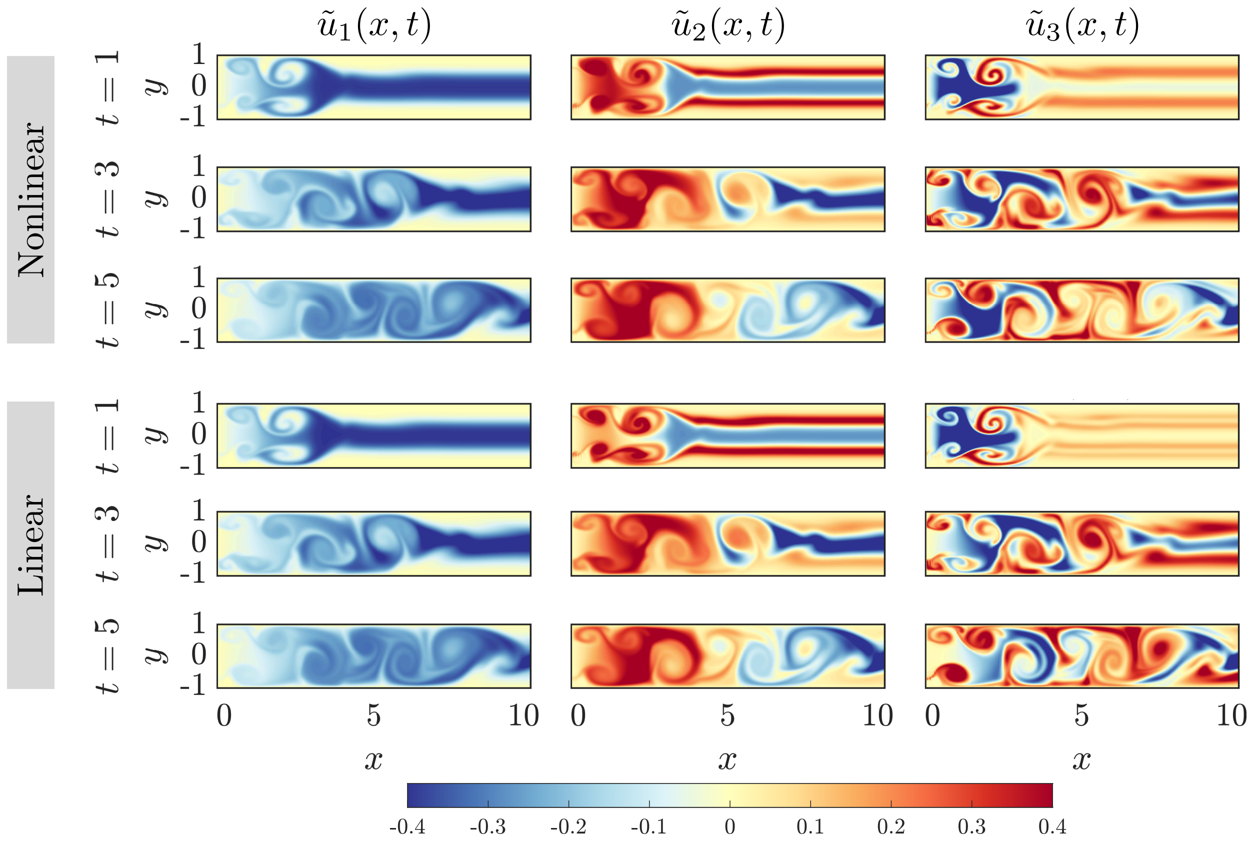


Figure 18: First three spatial modes of the nonlinear and linear sensitivities shown at $t = 1$, 3, and 5.

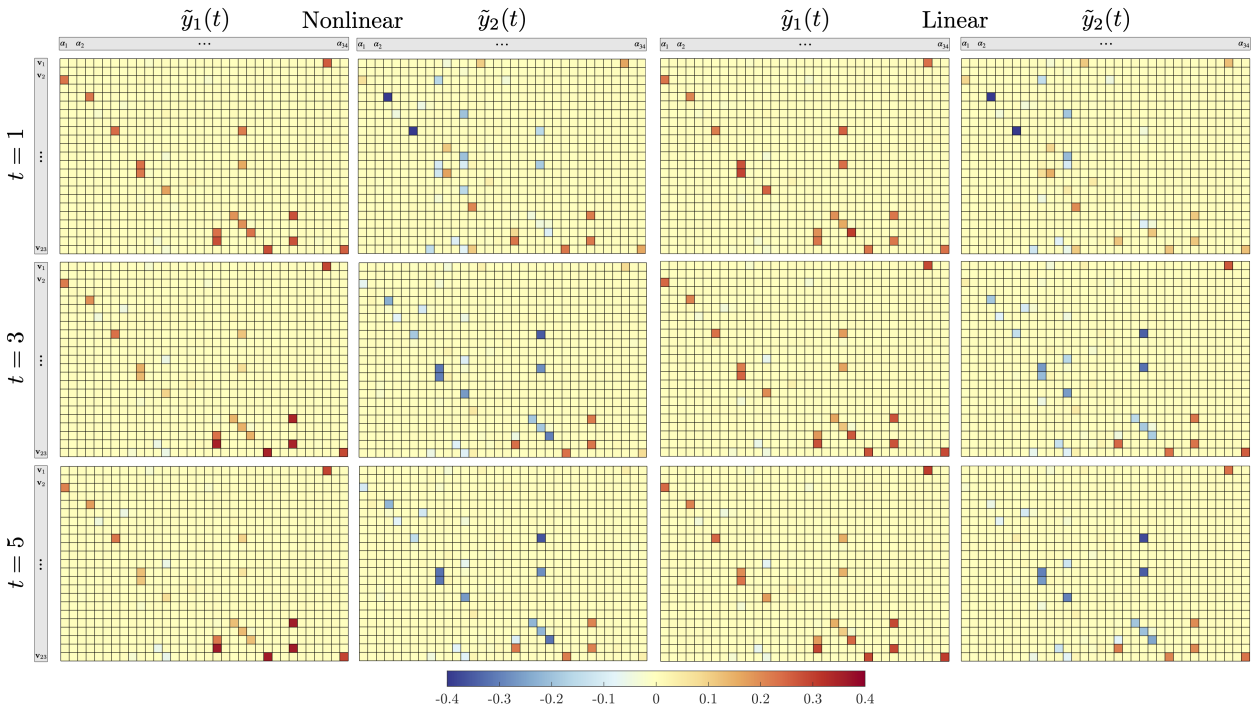


Figure 19: First two parametric modes of the nonlinear and linear sensitivities reshaped as matrices at $t = 1$, 3, and 5. Rows of each matrix correspond to species and columns correspond to parameters. Heat map identifies most dominant sensitivities.

4.0 Sparse Sampling for Nonlinear Model Reduction

In this chapter, we present a TDB ROM methodology based on a CUR factorization of low-rank matrices that addresses challenges C1-C3 from Chapter 1. That is, (1) the computational cost of the methodology scales with $\mathcal{O}(n+s)$ for generic nonlinear SPDEs both in terms of flops and memory costs, (2) it lends itself to simple implementation in existing codes, and (3) the time-integration is robust in the presence of small singular values, and high-order explicit time integration can be used. The main elements of the presented methodology are a time-discrete variational principle for minimization of the residual due to low-rank approximation error, and a CUR factorization based on strategic row and column sampling of the time discrete MDE. While the following work is presented in the context of UQ for stochastic PDEs, the new methodology can be applied to other applications that require solving MDEs.

4.1 Methodology

4.1.1 Setup

Consider the nonlinear stochastic PDE given by:

$$\frac{\partial v}{\partial t} = f(v; x, t, \boldsymbol{\xi}), \tag{4.1}$$

augmented with appropriate initial and boundary conditions. In the above equation, $v = v(x, t; \boldsymbol{\xi})$, x is the spatial coordinate, $\boldsymbol{\xi} \in \mathbb{R}^d$ are the set of random parameters, t is time, and $f(v; x, t, \boldsymbol{\xi})$ includes the nonlinear spatial differential operators. We assume generic nonlinear PDEs, where the nonlinearity of f versus v may be non-polynomial, e.g., exponential, fractional, etc. For the sake of simplicity in the exposition, we consider a collocation/strong-form discretization of Eq. 4.1 in x and $\boldsymbol{\xi}$. Because of the simplicity of the resulting discrete system, this choice facilitates an uncluttered illustration of the main contribution of this paper, which is focused on the efficient low-rank approximation of nonlinear matrix differential equations. However, the presented methodology can also be applied to other types of discretizations, for example, weak form discretizations (finite element, etc). Examples of collocation/strong-form discretizations in the spatial domain are Fourier/polynomial spectral collocation schemes or finite-difference discretizations. Example collocation schemes in the random domain include the probabilistic collocation method (PCM) [89] or any Monte-Carlo-type sampling methods [38, 14, 51]. Applying any of the above schemes to Eq. 4.1 leads to the following *nonlinear matrix differential equation*:

$$\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}t} = \mathcal{F}(t, \mathbf{V}), \quad t \in I = [0, T_f], \tag{4.2}$$

where $I = [0, T_f]$ denotes the time interval, $\mathbf{V}(t) : I \to \mathbb{R}^{n \times s}$ is a matrix with n rows corresponding to collocation points in the spatial domain and s columns corresponding to collocation/sampling points of the parameters $\boldsymbol{\xi}$, and $\mathcal{F}(t, \mathbf{V}) : I \times \mathbb{R}^{n \times s} \to \mathbb{R}^{n \times s}$ is obtained by discretizing $f(v; x, t, \boldsymbol{\xi})$ in x and $\boldsymbol{\xi}$. Eq. 4.2 is augmented with appropriate initial conditions, i.e., $\mathbf{V}(t_0) = \mathbf{V}_0$. We also assume that boundary conditions are already incorporated into Eq. 4.2, which can be accomplished in a number of ways, for example by using weak treatment of the boundary conditions [73].

For the remainder of this paper, we will refer to Eq. 4.2 as the FOM, which will be used as the ground truth for evaluating the performance of the proposed methodology. For the problems targeted in this work, we assume $n > s$ without loss of generality.

Remark. *The columns of MDE given by Eq. 4.2 are independent of each other. However, the rows of Eq. 4.2 are in general nonlinearly dependent, which depends on the spatial discretization used to discretize Eq. 4.1. The presented algorithm in this paper requires sparse spatial discretization, which means that each row is dependent on* p_a *rows, where* $p_a << n$*. The majority of discretization schemes, e.g., finite difference, finite volume, finite element, spectral element, result in sparse row dependence.*

We present our methodology for explicit time-discretization of MDE 4.2. The explicit time integration as well as the sparse row dependence condition means that the computational complexity of solving MDE 4.2 for each column is of $\mathcal{O}(n)$, and therefore, the cost of solving MDE 4.2 for all columns scales with $\mathcal{O}(ns)$.

4.1.2 Time-Continuous Variational Principle

The central idea behind TDB-based low-rank approximation is that the bases evolve optimally to minimize the residual due to low-rank approximation error. The residual is obtained by substituting an SVD-like low-rank approximation into the FOM so that $\mathbf{V}(t)$ is closely approximated by the rank-r matrix

$$\hat{\mathbf{V}}(t) = \mathbf{U}(t)\boldsymbol{\Sigma}(t)\mathbf{Y}(t)^T, \tag{4.3}$$

where $\mathbf{U}(t) \in \mathbb{R}^{n\times r}$ is a time-dependent orthonormal spatial basis for the column space, $\mathbf{Y}(t) \in \mathbb{R}^{s\times r}$ is a time-dependent orthonormal parametric basis for the row space, $\boldsymbol{\Sigma}(t) \in \mathbb{R}^{r\times r}$ is, in general, a full matrix, and $r \ll \min(n, s)$ is the rank of the approximation.

Because this is a low-rank approximation, it cannot satisfy the FOM exactly and there will be a residual equal to:

$$\mathbf{R}(t) = \frac{\mathrm{d}\left(\mathbf{U}\boldsymbol{\Sigma}\mathbf{Y}^T\right)}{\mathrm{d}t} - \mathcal{F}(t, \mathbf{U}\boldsymbol{\Sigma}\mathbf{Y}^T). \tag{4.4}$$

This residual is minimized via the first-order optimality conditions of the variational principle given by

$$\mathcal{J}(\dot{\mathbf{U}}, \dot{\boldsymbol{\Sigma}}, \dot{\mathbf{Y}}) = \left\| \frac{\mathrm{d}\left(\mathbf{U}\boldsymbol{\Sigma}\mathbf{Y}^T\right)}{\mathrm{d}t} - \mathcal{F}(t, \mathbf{U}\boldsymbol{\Sigma}\mathbf{Y}^T) \right\|_F^2, \tag{4.5}$$

subject to orthonormality constraints on $\mathbf{U}$ and $\mathbf{Y}$. Since the above variational principle involves the time-continuous equation (i.e. no temporal discretization is applied), the idea is to minimize the *instantaneous* residual by optimally updating $\mathbf{U}$, $\boldsymbol{\Sigma}$, and $\mathbf{Y}$ in time. Therefore, we refer to this as the *time-continuous* variational principle. As indicated in [49, 76], the optimality conditions of Eq. 4.5 lead to closed-form evolution equations for $\mathbf{U}$, $\boldsymbol{\Sigma}$, and $\mathbf{Y}$:

$$\dot{\boldsymbol{\Sigma}} = \mathbf{U}^T\mathbf{F}\mathbf{Y}, \tag{4.6a}$$

$$\dot{\mathbf{U}} = \left(\mathbf{I} - \mathbf{U}\mathbf{U}^T\right)\mathbf{F}\mathbf{Y}\boldsymbol{\Sigma}^{-1}, \tag{4.6b}$$

$$\dot{\mathbf{Y}} = \left(\mathbf{I} - \mathbf{Y}\mathbf{Y}^T\right)\mathbf{F}^T\mathbf{U}\boldsymbol{\Sigma}^{-T}, \tag{4.6c}$$

where $\mathbf{F} \in \mathbb{R}^{n\times s}$ is a matrix defined as $\mathbf{F} = \mathcal{F}(t, \mathbf{U}\boldsymbol{\Sigma}\mathbf{Y}^T)$, and $\mathbf{I}$ is the identity matrix of appropriate dimensions. The above variational principle is the same as the Dirac–Frenkel time-dependent variational principle in the quantum chemistry literature [16] or the dynamical low-rank approximation (DLRA) [49]. As it was shown in [7], it is possible to derive a similar variational principle for the DO decomposition, $\hat{\mathbf{V}}(t) = \mathbf{U}_{DO}(t)\mathbf{Y}_{DO}^T(t)$, whose optimality conditions are constrained to the orthonormality of the spatial modes, $\mathbf{U}_{DO}^T\mathbf{U}_{DO} = \mathbf{I}$, via the dynamically orthogonal condition, $\dot{\mathbf{U}}_{DO}^T\mathbf{U}_{DO} = \mathbf{0}$. However, for the sake of simplicity and unlike the original DO formulation presented in [81], an evolution equation for the mean field is not derived. Without loss of generality, the low-rank DO evolution equations become

$$\dot{\mathbf{U}}_{DO} = \left(\mathbf{I} - \mathbf{U}_{DO}\mathbf{U}_{DO}^T\right)\mathbf{F}\mathbf{Y}_{DO}\mathbf{C}^{-1}, \tag{4.7a}$$

$$\dot{\mathbf{Y}}_{DO} = \mathbf{F}^T\mathbf{U}_{DO}, \tag{4.7b}$$

where $\mathbf{C} = \mathbf{Y}_{DO}^T\mathbf{Y}_{DO}$ is the low-rank correlation matrix. Note that the low-rank approximation based on DO is *equivalent* to Eq. 4.3, i.e., $\mathbf{U}_{DO}\mathbf{Y}_{DO}^T = \mathbf{U}\boldsymbol{\Sigma}\mathbf{Y}^T$. Similarly, the BO decomposition, $\hat{\mathbf{V}}(t) = \mathbf{U}_{BO}(t)\mathbf{Y}_{BO}^T(t)$, which is subject to BO conditions, $\mathbf{U}_{BO}^T\mathbf{U}_{BO} = \text{diag}(\lambda_1, \dots, \lambda_r)$ and $\mathbf{Y}_{BO}^T\mathbf{Y}_{BO} = \mathbf{I}$, is also identical to DO and Eq. 4.3. As it was shown in [72], one can derive matrix differential equations that transform the factorization $\{\mathbf{U}, \boldsymbol{\Sigma}, \mathbf{Y}\}$ to $\{\mathbf{U}_{DO}, \mathbf{Y}_{DO}\}$ or $\{\mathbf{U}_{BO}, \mathbf{Y}_{BO}\}$. The equivalence of DO and BO formulations was shown in [27]. Using the DO/BO terminology, Eqs. 4.6a-4.6c have both DO and BO conditions, i.e., the dynamically orthogonal conditions for $\mathbf{U}$ and $\mathbf{Y}$: $\dot{\mathbf{U}}^T\mathbf{U} = \mathbf{0}$ and $\dot{\mathbf{Y}}^T\mathbf{Y} = \mathbf{0}$ as well as bi-orthonormality conditions: $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ and $\mathbf{Y}^T\mathbf{Y} = \mathbf{I}$. Despite their equivalence, these three factorizations have different numerical performances in the presence of small singular values. As it was shown in [72], Eqs. 4.6a-4.6c outperform both DO and BO.

Despite the potential of Eqs. 4.6a-4.6c to significantly reduce the computational cost of solving massive matrix differential equations like Eq. 4.2, there are still a number of outstanding challenges for most practical problems of interest. As highlighted in the Introduction, computing $\mathbf{F} = \mathcal{F}(t, \mathbf{U}\boldsymbol{\Sigma}\mathbf{Y}^T)$ requires $\mathcal{O}(ns)$ operations that scale with the size of the FOM. This involves applying the nonlinear map ($\mathcal{F}$) on every column of the matrix $\hat{\mathbf{V}} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{Y}^T$. While it is possible to achieve $\mathcal{O}(n+s)$ for the special cases of homogeneous linear and quadratic nonlinear $\mathcal{F}$, this comes at the expense of a highly intrusive process, that requires a careful term-by-term treatment of the right side of Eqs. 4.6a-4.6c [67, Appendix B]. Furthermore, solving Equations 4.6b and 4.6c become unstable when $\mathbf{\Sigma}$ is singular or near singular. This is particularly problematic because it is often necessary to retain very small singular values in order to have an accurate approximation.

While the low-rank approximation based on TDBs can be cast in different, yet equivalent formulations, we have chosen Eqs. 4.6a-4.6c over DO/BO/DDO decompositions to highlight the underlying challenges. Since, DO/BO/DDO decompositions exhibit all of the above challenges, addressing these challenges in the context of Eqs. 4.6a-4.6c automatically addresses the DO/BO/DDO challenges as well.

4.1.3 Time-Discrete Variational Principle

To address the challenges of low-rank approximations based on TDB using the time-continuous variational principle, we consider a *time-discrete* variational principle for rank-adaptive matrix approximations, which has been recently applied in the context of tensor manifolds in [47, 78]. To this end, consider a generic temporal discretization of Eq. 4.2:

$$\mathbf{V}^k = \mathcal{G}(\mathbf{V}^k, \mathbf{V}^{k-1}, \dots, \mathbf{V}^{k-q}), \tag{4.8}$$

where the superscript k denotes the current time step, q is the number of previous time steps, and $\mathcal{G} : \mathbb{R}^{n\times s\times(q+1)} \to \mathbb{R}^{n\times s}$ is the increment function. In this work, we consider explicit time discretization schemes that are not a function of $\mathbf{V}^k$. For the sake of brevity in notation, we do not show the explicit dependence on time and we denote the increment map with $\mathcal{G}(\mathbf{V}^{k-1})$ for explicit time integration schemes. For example, the first-order explicit map is given by the Euler method: $\mathcal{G}(\mathbf{V}^{k-1}) = \mathbf{V}^{k-1} + \Delta t\mathcal{F}(\mathbf{V}^{k-1})$, where Δt is the step size. Let us consider the rank-r approximation, $\hat{\mathbf{V}}^k$, such that

$$\mathbf{V}^k = \hat{\mathbf{V}}^k + \mathbf{E}^k, \tag{4.9}$$

where $\mathbf{E}^k$ is the low-rank approximation error. Because $\hat{\mathbf{V}}^k$ is a low-rank approximation, it cannot satisfy Eq. (4.8), and there will be a residual equal to

$$\mathbf{R}^k = \hat{\mathbf{V}}^k - \mathcal{G}(\hat{\mathbf{V}}^{k-1}). \tag{4.10}$$

The time-discrete variational principle can be stated as finding the best $\hat{\mathbf{V}}^k \in \mathcal{M}_r$, where $\mathcal{M}_r$ is the manifold of rank-r matrices, such that the Frobenius norm of the following residual is minimized:

$$\mathcal{Z}(\hat{\mathbf{V}}^k) = \left\| \hat{\mathbf{V}}^k - \mathbf{G} \right\|_F^2, \tag{4.11}$$

where $\mathbf{G} \in \mathbb{R}^{n \times s} := \mathcal{G}(\hat{\mathbf{V}}^{k-1})$. The solution of the above residual minimization scheme is the rank-r matrix,

$$\hat{\mathbf{V}}^k_{opt} = \mathrm{SVD}(\mathbf{G}), \tag{4.12}$$

where $\mathrm{SVD}(\mathbf{G})$ is the rank-r truncated SVD of matrix $\mathbf{G}$. An important advantage of Eq. 4.12 over Eqs. 4.6a-4.6c is that the time advancement according to Eq. 4.12 does not become singular in the presence of small singular values. While this solves the issue of ill-conditioning, computing Eq. 4.12 at each iteration of the time stepping scheme is cost prohibitive. This computational cost is due to two sources: (i) computing the nonlinear map $\mathbf{G}$ and (ii) computing the $\mathrm{SVD}(\mathbf{G})$. The cost of (i) alone makes the solution of the time-discrete variational principle as expensive as the FOM, i.e., $\mathcal{O}(ns)$. Besides the flops cost associated with computing $\mathbf{G}$, the memory cost of storing $\mathbf{G}$ is prohibitive for most realistic applications. On the other hand, computing the exact SVD of $\mathbf{G}$ scales with $\min\{\mathcal{O}(n^3), \mathcal{O}(s^3)\}$. While this cost is potentially alleviated by fast algorithms for approximating the SVD, e.g. randomized SVD [43] or incremental QR [85], for general nonlinearities in $\mathcal{G}$, (i) is unavoidable. This ultimately leads to a computational cost that exceeds that of the FOM.

4.1.4 Low-Rank Approximation via Sparse Adaptive Sampling

To overcome these challenges, we present a sparse collocation scheme that enables the computation of $\hat{\mathbf{V}}^k$, via a cost-effective approximation to the rank-r truncated SVD of $\mathbf{G}$. To this end, we consider $\hat{\mathbf{V}}^k = \mathbf{U}^k \mathbf{\Sigma}^k \mathbf{Y}^{k^T}$, where $\mathbf{U}^k \in \mathbb{R}^{n \times r}$ and $\mathbf{Y}^k \in \mathbb{R}^{s \times r}$ are an approximation to the left and right singular vectors of $\mathbf{G}$, and $\mathbf{\Sigma}^k \in \mathbb{R}^{r \times r}$ is a diagonal matrix that contains an approximation to the singular values of $\mathbf{G}$. Our approach is to set the residual to zero at r strategically selected rows and columns of the residual matrix $\mathbf{R}^k$. To this end, we present an algorithm to set $\mathbf{R}^k(\mathbf{p}, :) = \mathbf{0}$ and $\mathbf{R}^k(:, \mathbf{s}) = \mathbf{0}$, where $\mathbf{p} = [p_1, p_2, \dots, p_r] \in \mathbb{N}^r$ and $\mathbf{s} = [s_1, s_2, \dots, s_r] \in \mathbb{N}^r$ are vectors containing the row and column indices at which the residual is set to zero. This simply requires $\hat{\mathbf{V}}^k(\mathbf{p},:) = \mathbf{G}(\mathbf{p},:)$ and $\hat{\mathbf{V}}^k(:,\mathbf{s}) = \mathbf{G}(:,\mathbf{s})$. Here, we have used MATLAB indexing where $\mathbf{A}(\mathbf{p},:)$ selects all columns at the $\mathbf{p}$ rows, and $\mathbf{A}(:,\mathbf{s})$ selects all rows at the $\mathbf{s}$ columns of the matrix $\mathbf{A}$. While there are many possible choices for the indices $\mathbf{p}$ and $\mathbf{s}$, selecting these points should be done in a principled manner, to ensure the residual at all points remains small. To compute these points, we use the discrete empirical interpolation method (DEIM) [25] which has been shown to provide near optimal sampling points for computing CUR matrix decompositions [85]. A similar approach was recently applied in [67] to accelerate the computation of Eqs. 4.6a-4.6c, by only sampling $\mathbf{F}$ at a small number of rows and columns. However, the approach presented in [67] still suffers from the issue of ill-conditioning.

In this work, we present a procedure to compute a cost-effective approximation of the rank-r SVD of $\mathbf{G}$, by interpolating $\mathbf{G}$ at the DEIM-selected rows and columns. To compute the DEIM points, the rank-r SVD (or an approximation) is required [85]. Since we do not have access to the rank-r SVD at the current time step, k, we use the approximation of the SVD from the previous time step, $\hat{\mathbf{V}}^{k-1} = \mathbf{U}^{k-1}\mathbf{\Sigma}^{k-1}\mathbf{Y}^{k-1^T}$, to compute the DEIM points. The algorithm for computing $\hat{\mathbf{V}}^k$ is as follows:

1. Compute the sampling indices, $\mathbf{p} \leftarrow \texttt{DEIM}(\mathbf{U}^{k-1})$, and $\mathbf{s} \leftarrow \texttt{DEIM}(\mathbf{Y}^{k-1})$, in parallel.
2. Compute the nonlinear map at the selected rows and columns, $\mathbf{G}(\mathbf{p},:)$ and $\mathbf{G}(:,\mathbf{s})$, in parallel.
3. Compute $\mathbf{Q} \in \mathbb{R}^{n\times r}$ as the orthonormal basis for the range of $\mathbf{G}(:,\mathbf{s})$ by QR decomposition such that $\mathbf{G}(:,\mathbf{s}) = \mathbf{QR}$, where $\mathbf{R} \in \mathbb{R}^{r\times r}$.
4. Interpolate every column of $\mathbf{G}$ onto the orthonormal basis $\mathbf{Q}$ at sparse indices $\mathbf{p}$:

$$\mathbf{Z} = \mathbf{Q}(\mathbf{p},:)^{-1}\mathbf{G}(\mathbf{p},:), \tag{4.13}$$

   where $\mathbf{Z} \in \mathbb{R}^{r\times s}$ is the matrix of interpolation coefficients such that $\mathbf{QZ}$ interploates $\mathbf{G}$ onto the basis $\mathbf{Q}$ at the interploation points indexed by $\mathbf{p}$.
5. Compute the SVD of $\mathbf{Z}$ so that

$$\mathbf{Z} = \mathbf{U_Z}\mathbf{\Sigma}^k\mathbf{Y}^{k^T}, \tag{4.14}$$

   where $\mathbf{U_Z} \in \mathbb{R}^{r\times r}$, $\mathbf{\Sigma}^k \in \mathbb{R}^{r\times r}$, and $\mathbf{Y}^k \in \mathbb{R}^{s\times r}$.

6. Compute $\mathbf{U}^k \in \mathbb{R}^{n\times r}$ as the in-subspace rotation:

$$\mathbf{U}^k = \mathbf{Q}\mathbf{U}_{\mathbf{Z}}. \tag{4.15}$$

In Step 1 above, the details of the `DEIM` algorithm can be found in [25, Algorithm 1]. A DEIM algorithm based on the QR factorization, a.k.a `QDEIM`, may also be used [32, 64]. Both `DEIM` and `QDEIM` are sparse selection algorithms and they perform comparably in the cases considered in this paper. We explain here how the above algorithm addresses the three challenges mentioned in Chapter 1.

1. **Computational efficiency:** The above procedure returns the updated low-rank approximation $\hat{\mathbf{V}}^k = \mathbf{Q}\mathbf{Z} = \mathbf{U}^k\mathbf{\Sigma}^k\mathbf{Y}^{k^T}$, and only requires sampling $\mathbf{G}$ at r rows and columns. This alone significantly reduces both the required number of flops and memory, compared to computing the entire $\mathbf{G}$. Furthermore, instead of directly computing the SVD of the $n \times s$ matrix $\mathbf{G}$, we only require computing the QR of the $n \times r$ matrix $\mathbf{G}(:,\mathbf{s})$, and the SVD of the $r \times s$ matrix $\mathbf{Z}$. This reduces the computational cost to $\mathcal{O}(s+n)$ for $r \ll s$ and $r \ll n$. Moreover, in most practical applications, computing $\mathbf{G}(:,\mathbf{s})$ is the costliest part of the algorithm, which requires solving s samples of the FOM. However, since these samples are independent of each other, the columns of $\mathbf{G}(:,\mathbf{s})$ can be computed in parallel. Similarly, each row of $\mathbf{G}(\mathbf{p},:)$ can be computed in parallel.
2. **Intrusiveness:** While this significantly reduces the computational burden, perhaps an equally important outcome is the minimally intrusive nature of the above approach. For example, when the columns of $\mathbf{G}$ are independent, e.g. random samples, $\mathbf{G}(:,\mathbf{s})$ can be computed by directly applying Eq. 4.8 to the low-rank approximation, $\mathcal{G}(\mathbf{U}^{k-1}\mathbf{\Sigma}^{k-1}\mathbf{Y}(\mathbf{s},:)^{k-1^T})$. This effectively allows for existing numerical implementations of Eq. 4.8 to be used as a black box for computing $\mathbf{G}(:,\mathbf{s})$. On the other hand, the rows of $\mathbf{G}$ are in general *dependent*, based on a known map for the chosen spatial discretization scheme, e.g. sparse discretizations like finite difference, spectral element, etc. Therefore, computing $\mathbf{G}(\mathbf{p},:)$ does require specific knowledge of the governing equations, namely the discretized differential operators. Based on the discretization scheme, one can determine a set of adjacent points, $\mathbf{p}_a$, that are required for computing the derivatives at the points specified by $\mathbf{p}$. While this introduces an added layer of complexity, this is much

less intrusive than deriving and implementing reduced order operators for each term in the governing equations; which we emphasize again, is only feasible for homogeneous linear or quadratic nonlinear equations. In the present work, that bottleneck is removed, regardless of the type of nonlinearity.

3. **Ill-conditioning:** The presented algorithm is robust in the presence of small or zero singular values. First note that the inversion of the matrix of singular values is not required in the presented algorithm. In fact, the conditioning of the algorithm depends on $\mathbf{Q}(\mathbf{p},:)$ and $\mathbf{Y}(\mathbf{s},:)$, and the DEIM algorithm ensures that these two matrices are well-conditioned. To illustrate this point, let us consider the case of overapproximation where the rank of $\mathbf{G}(:,\mathbf{s})$ is $r_1 < r$. In this case, Eqs. 4.6a-4.6c and Eqs. 4.7a-4.7b cannot be advanced because $\mathbf{\Sigma} \in \mathbb{R}^{r\times r}$ and $\mathbf{C} \in \mathbb{R}^{r\times r}$ will be singular, i.e., rank($\mathbf{\Sigma}$) =rank($\mathbf{C}$) = $r_1 < r$. On the other hand, despite $\mathbf{G}(:,\mathbf{s})$ being rank deficient, $\mathbf{Q}$ will still be a full rank matrix in the presented algorithm. While there is no guarantee that a subset of rows of $\mathbf{Q}$, i.e., $\mathbf{Q}(\mathbf{p},:)$ is well conditioned, the DEIM is a greedy algorithm that is designed to keep $\|\mathbf{Q}(\mathbf{p},:)^{-1}\|$ as small as possible in a near-optimal fashion. In Section 4.1.6, we show that oversampling further improves the condition number of the presented algorithm, and in Theorem 4.1.3, we show that $\|\mathbf{Y}(\mathbf{s},:)^{-1}\|$ plays an equally important role in maintaining a well-conditioned algorithm.

As we will show in the following section, the low-rank approximation computed above is equivalent to a CUR matrix factorization that interpolates $\mathbf{G}$ at the selected rows and columns. Therefore, we refer to the above procedure as the `TDB-CUR` algorithm. The key components of the algorithm are highlighted in Figure 20.

4.1.5 Equivalence to a CUR Decomposition/Oblique Projection

In this section, we show that the low-rank matrix approximation we compute is equivalent to a CUR matrix decomposition. In short, CUR low-rank decompositions explicitly reconstruct a matrix using a small number of actual rows and columns of the original matrix. The key idea is that if a matrix is of low rank, then it is not necessary to use all columns and rows of the matrix to construct a good low-rank approximation. Therefore, the result-

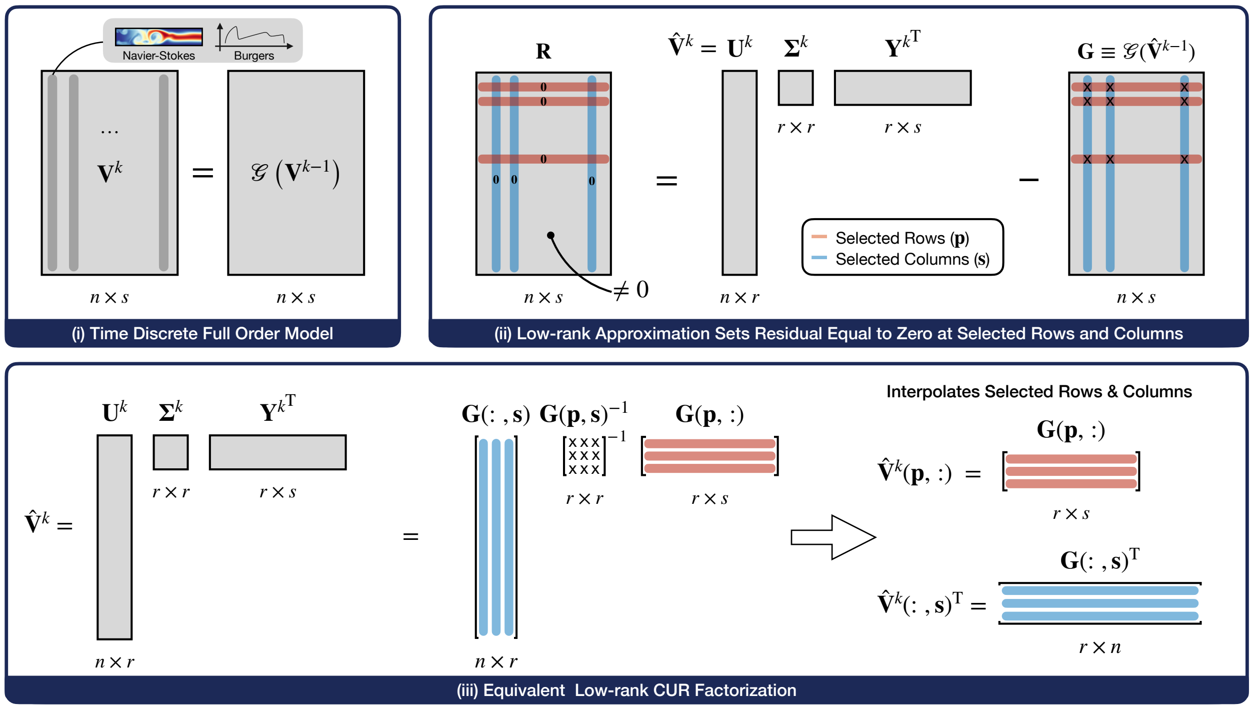

Navier-Stokes
Burgers
$\mathbf{V}^k$
$= \mathscr{G}(\mathbf{V}^{k-1})$
$n \times s$
$n \times s$
(i) Time Discrete Full Order Model
$\mathbf{R}$
$\hat{\mathbf{V}}^k = \mathbf{U}^k \; \boldsymbol{\Sigma}^k \; \mathbf{Y}^{k\mathrm{T}}$
$\mathbf{G} \equiv \mathscr{G}(\hat{\mathbf{V}}^{k-1})$
$r \times r$
$r \times s$
$\neq 0$
$n \times s$
$n \times r$
$n \times s$
Selected Rows (p)
Selected Columns (s)
(ii) Low-rank Approximation Sets Residual Equal to Zero at Selected Rows and Columns
$\hat{\mathbf{V}}^k = \mathbf{U}^k \; \boldsymbol{\Sigma}^k \; \mathbf{Y}^{k\mathrm{T}}$
$n \times r$
$r \times r$
$r \times s$
$= \mathbf{G}(:,\mathbf{s}) \; \mathbf{G}(\mathbf{p},\mathbf{s})^{-1} \; \mathbf{G}(\mathbf{p},:)$
$n \times r$
$r \times r$
$r \times s$
Interpolates Selected Rows & Columns
$\hat{\mathbf{V}}^k(\mathbf{p},:) = \mathbf{G}(\mathbf{p},:)$
$r \times s$
$\hat{\mathbf{V}}^k(:,\mathbf{s})^{\mathrm{T}} = \mathbf{G}(:,\mathbf{s})^{\mathrm{T}}$
$r \times n$
(iii) Equivalent Low-rank CUR Factorization

Figure 20: Schematic of the TDB-CUR methodology. (i) FOM is discretized using an explicit temporal integration scheme, where $\mathcal{G}$ is a nonlinear map of the solution from the previous time step, $\mathbf{V}^{k-1}$, to the current time step, $\mathbf{V}^k$. Columns correspond to independent random samples, e.g. Navier-Stokes or Burgers equation. (ii) The rank-r approximation, $\hat{\mathbf{V}}^k$, is computed such that the residual at the selected rows (red) and columns (blue) is equal to zero. This is accomplished via sparse interpolation of the selected rows and columns. (iii) The resulting $\hat{\mathbf{V}}^k$ is an approximation to the rank-r truncated svd($\mathbf{G}$), and is equivalent to the low-rank CUR factorization that interpolates the selected rows and columns. Although it is equivalent to this CUR factorization, the numerical computation of $\hat{\mathbf{V}}^k$ is different, as it does not require inverting $\mathbf{G}(\mathbf{p}, \mathbf{s})$.

ing decomposition is highly interpretable, making it an attractive option for improved data analysis. To this end, a matrix $\mathbf{A} \in \mathbb{R}^{n \times s}$ that is low-rank, can be approximated accurately by taking a small number of actual rows, $\mathbf{R}$, and columns, $\mathbf{C}$, from the original matrix $\mathbf{A}$.

A small matrix of appropriate dimension, $\mathbf{U}$, is then computed to make $\mathbf{A} \approx \mathbf{CUR}$ a good approximation. Here, the matrices, $\mathbf{C}$, $\mathbf{U}$, and $\mathbf{R}$ are different from the matrices defined in previous sections. For more details on CUR decompositions, we refer the reader to [63].

Different CUR decompositions can be obtained for the same matrix depending on two factors: (i) the selection of columns and rows, and (ii) the method used to compute the matrix $\mathbf{U}$. In a closely related work [85], a DEIM-CUR procedure was introduced in which the columns and rows are selected based on DEIM, and $\mathbf{U}$ is obtained via the *orthogonal projection* of the original matrix onto the selected columns and rows. However, orthogonal projection onto selected columns and rows requires access to *all* entries of $\mathbf{G}$. This would render the DEIM-CUR procedure in [85] ineffective for the present application, since we cannot afford to compute or store the entire matrix $\mathbf{G}$. In the following, we show that (i) the presented algorithm is equivalent to a CUR decomposition (Theorem 4.1.1), and (ii) the matrix $\mathbf{U}$ is obtained via an *oblique projection*, which requires access to only the selected rows and columns of $\mathbf{G}$ (Theorem 4.1.2). In the following, we use the indexing matrices, $\mathbf{P} = \mathbf{I}_n(:,\mathbf{p}) \in \mathbb{R}^{n\times r}$ and $\mathbf{S} = \mathbf{I}_s(:,\mathbf{s}) \in \mathbb{R}^{s\times r}$, where $\mathbf{I}_n$ and $\mathbf{I}_s$ are identity matrices of size $n \times n$ and $s \times s$, respectively. It is easy to verify that $\mathbf{P}^T\mathbf{U} \equiv \mathbf{U}(\mathbf{p},:)$ and $\mathbf{S}^T\mathbf{Y} \equiv \mathbf{Y}(\mathbf{s},:)$. For the sake of brevity, we drop the superscript k in the following.

Theorem 4.1.1. *Let $\hat{\mathbf{V}} = \mathbf{QZ}$ be the low-rank approximation of $\mathbf{G}$ computed according to the TDB-CUR algorithm. Then: (i) $\hat{\mathbf{V}} = \mathbf{QZ}$ is equivalent to the CUR factorization given by $(\mathbf{GS})(\mathbf{P}^T\mathbf{GS})^{-1}(\mathbf{P}^T\mathbf{G})$. (ii) The low-rank approximation is exact at the selected rows and columns, i.e. $\mathbf{P}^T\hat{\mathbf{V}} = \mathbf{P}^T\mathbf{G}$ and $\hat{\mathbf{V}}\mathbf{S} = \mathbf{GS}$.*

Proof.

(i) According to the TDB-CUR algorithm, $\mathbf{Q}$ is a basis for the $\mathrm{Ran}(\mathbf{G}(:,\mathbf{s}))$. Therefore, $\mathbf{G}(:,\mathbf{s}) = \mathbf{GS} = \mathbf{QQ}^T\mathbf{GS}$, and it follows that $\mathbf{P}^T\mathbf{GS} = \mathbf{P}^T\mathbf{QQ}^T\mathbf{GS}$. Substituting this result into the CUR factorization gives

$$(\mathbf{GS})(\mathbf{P}^T\mathbf{GS})^{-1}(\mathbf{P}^T\mathbf{G}) = \mathbf{QQ}^T\mathbf{GS}(\mathbf{P}^T\mathbf{QQ}^T\mathbf{GS})^{-1}\mathbf{P}^T\mathbf{G}.$$

Rearranging the above expression gives the desired result

$$(\mathbf{GS})(\mathbf{P}^T\mathbf{GS})^{-1}(\mathbf{P}^T\mathbf{G}) = \mathbf{QQ}^T\mathbf{GS}(\mathbf{Q}^T\mathbf{GS})^{-1}(\mathbf{P}^T\mathbf{Q})^{-1}\mathbf{P}^T\mathbf{G} = \mathbf{QZ} = \hat{\mathbf{V}},$$

where we have used $\mathbf{Z} = (\mathbf{P}^T\mathbf{Q})^{-1}\mathbf{P}^T\mathbf{G} = \mathbf{Q}(\mathbf{p},:)^{-1}\mathbf{G}(\mathbf{p},:)$, from Eq. 4.13.

(ii) Using the above result, $\hat{\mathbf{V}} = (\mathbf{GS})(\mathbf{P}^T\mathbf{GS})^{-1}(\mathbf{P}^T\mathbf{G})$, we show the selected rows of $\hat{\mathbf{V}}$ are exact, i.e., $\hat{\mathbf{V}}(\mathbf{p},:) = \mathbf{G}(\mathbf{p},:)$:

$$\mathbf{P}^T\hat{\mathbf{V}} = (\mathbf{P}^T\mathbf{GS})(\mathbf{P}^T\mathbf{GS})^{-1}(\mathbf{P}^T\mathbf{G}) = \mathbf{P}^T\mathbf{G}.$$

Similarly for the columns,

$$\hat{\mathbf{V}}\mathbf{S} = (\mathbf{GS})(\mathbf{P}^T\mathbf{GS})^{-1}(\mathbf{P}^T\mathbf{GS}) = \mathbf{GS}.$$

This completes the proof.

□

Now we show that $\hat{\mathbf{V}}$ is an oblique projection of $\mathbf{G}$ onto the selected columns and rows of $\mathbf{G}$. In particular, the oblique projector involved is an *interpolatory projector*. In the following, we define interpolatory projectors. For the sake of brevity, we drop the superscript k in the following.

Definition 4.1.1. *Let $\mathbf{U} \in \mathbb{R}^{n\times r}$ and $\mathbf{Y} \in \mathbb{R}^{s\times r}$ be full rank matrices and let $\mathbf{p}$ and $\mathbf{s}$ be sets of distinct row and column indices, respectively. The interpolatory projectors for $\mathbf{p}$ onto Ran($\mathbf{U}$) and for $\mathbf{s}$ onto Ran($\mathbf{Y}$) are defined as*

$$\mathcal{P} \equiv \mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{P}^T \quad \textit{and} \quad \mathcal{S} \equiv \mathbf{S}(\mathbf{Y}^T\mathbf{S})^{-1}\mathbf{Y}^T, \tag{4.16}$$

provided $(\mathbf{P}^T\mathbf{U})$ and $(\mathbf{Y}^T\mathbf{S})$ are invertible. We refer to $\mathbf{U}$ and $\mathbf{Y}$ as interpolation bases for $\mathcal{P}$ and $\mathcal{S}$, respectively.

For a given matrix $\mathbf{A} \in \mathbb{R}^{n\times s}$, $\mathcal{P}$ operates on the left side of the matrix and $\mathcal{S}$ operates on the right side of the matrix. In general, both $\mathcal{P}$ and $\mathcal{S}$ are oblique projectors, and it is easy to verify that $\mathcal{P}^2 = \mathcal{P}$ and $\mathcal{S}^2 = \mathcal{S}$. Unlike orthogonal projection, the interpolatory projection is guaranteed to match the original matrix at selected rows and columns, i.e.,

$$\mathbf{P}^T\mathcal{P}\mathbf{A} = \mathbf{A}(\mathbf{p},:) \quad \text{and} \quad \mathbf{A}\mathcal{S}\mathbf{S} = \mathbf{A}(:,\mathbf{s}).$$

Theorem 4.1.2. *Let $\hat{\mathbf{V}} = \mathbf{QZ}$ be the low-rank approximation of $\mathbf{G}$ computed according to the TDB-CUR algorithm. Then $\hat{\mathbf{V}} = \mathcal{P}\mathbf{G}\mathcal{S}$ where $\mathcal{P}$ and $\mathcal{S}$ are oblique projectors onto Ran($\mathbf{U}$) and Ran($\mathbf{Y}$), respectively, according to Eq. 4.16.*

Proof. We first show that $\mathcal{P}$ can be represented versus $\mathbf{Q}$ as the interpolation basis. To this end, replacing $\mathbf{U} = \mathbf{QU_Z}$ in the definition of $\mathcal{P}$ results in:

$$\mathcal{P} = \mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{P}^T = \mathbf{QU_Z}(\mathbf{P}^T\mathbf{QU_Z})^{-1}\mathbf{P}^T = \mathbf{QU_Z}\mathbf{U_Z}^{-1}(\mathbf{P}^T\mathbf{Q})^{-1}\mathbf{P}^T = \mathbf{Q}(\mathbf{P}^T\mathbf{Q})^{-1}\mathbf{P}^T.$$

where we have used the fact that $\mathbf{U_Z}$ is a square orthonormal matrix and therefore, $\mathbf{U_Z}\mathbf{U_Z}^{-1} = \mathbf{I}$. Similarly, $\mathcal{S}$ can be represented versus $\mathbf{Z}^T$ as the interpolation basis by replacing $\mathbf{Y}^T = \mathbf{\Sigma}^{-1}\mathbf{U_Z}^{-1}\mathbf{Z}$ in $\mathcal{S}$:

$$\mathcal{S} = \mathbf{S}(\mathbf{Y}^T\mathbf{S})^{-1}\mathbf{Y}^T = \mathbf{S}(\mathbf{\Sigma}^{-1}\mathbf{U_Z}^{-1}\mathbf{ZS})^{-1}\mathbf{\Sigma}^{-1}\mathbf{U_Z}^{-1}\mathbf{Z} = \mathbf{S}(\mathbf{ZS})^{-1}\mathbf{Z}.$$

Using these projection operators we have

$$\mathcal{P}\mathbf{G}\mathcal{S} = \mathbf{Q}(\mathbf{P}^T\mathbf{Q})^{-1}\mathbf{P}^T\mathbf{GS}(\mathbf{ZS})^{-1}\mathbf{Z}. \tag{4.17}$$

Using the results of Theorem 4.1.1, Part (ii), we have: $\mathbf{G}(\mathbf{p},\mathbf{s}) = \hat{\mathbf{V}}(\mathbf{p},\mathbf{s})$. Therefore:

$$\mathbf{P}^T\mathbf{GS} = \mathbf{G}(\mathbf{p},\mathbf{s}) = \hat{\mathbf{V}}(\mathbf{p},\mathbf{s}) = \mathbf{Q}(\mathbf{p},:)\mathbf{Z}(:,\mathbf{s}) = \mathbf{P}^T\mathbf{QZS}.$$

Using this result in Eq. 4.17, yields:

$$\mathcal{P}\mathbf{G}\mathcal{S} = \mathbf{Q}(\mathbf{P}^T\mathbf{Q})^{-1}\mathbf{P}^T\mathbf{QZS}(\mathbf{ZS})^{-1}\mathbf{Z} = \mathbf{QZ} = \hat{\mathbf{V}}.$$

This result completes the proof. $\square$

In the following theorem, we show that the oblique projection error is bounded by an error factor multiplied by the maximum of orthogonal projection errors onto $\mathbf{U}$ or $\mathbf{Y}$. We follow a similar procedure that was used in [85], however, as mentioned above, in [85] the CUR is computed based on orthogonal projections onto the selected columns and rows, whereas in the presented TDB-CUR algorithm, interpolatory projectors are used. In the following, we use the second norm ($\| \sim \| \equiv \| \sim \|_2$).

Theorem 4.1.3. *Let $\mathcal{P}$ and $\mathcal{S}$ be oblique projectors according to Definition 4.1.1 and let $\mathbf{U} \in \mathbb{R}^{n\times r}$ and $\mathbf{Y} \in \mathbb{R}^{s\times r}$ be a set of orthonormal matrices, i.e., $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ and $\mathbf{Y}^T\mathbf{Y} = \mathbf{I}$. Let ϵ_f be the error factor given by: $\epsilon_f = min\{\eta_p(1+\eta_s), \eta_s(1+\eta_p)\}$, where $\eta_p = \|(\mathbf{P}^T\mathbf{U})^{-1}\|$ and $\eta_s = \|(\mathbf{S}^T\mathbf{Y})^{-1}\|$ and $\hat{\sigma}_{r+1} = max\{\|\mathbf{G} - \mathbf{U}\mathbf{U}^T\mathbf{G}\|, \|\mathbf{G} - \mathbf{G}\mathbf{Y}\mathbf{Y}^T\|\}$. Then the error of the oblique projection is bounded by*

$$\|\mathbf{G} - \mathcal{P}\mathbf{G}\mathcal{S}\| \leq \epsilon_f\hat{\sigma}_{r+1}. \tag{4.18}$$

Proof.

The error matrix can be written as:

$$\mathbf{G} - \mathcal{P}\mathbf{G}\mathcal{S} = (\mathbf{I} - \mathcal{P})\mathbf{G} + \mathcal{P}\mathbf{G} - \mathcal{P}\mathbf{G}\mathcal{S} = (\mathbf{I} - \mathcal{P})\mathbf{G} + \mathcal{P}\mathbf{G}(\mathbf{I} - \mathcal{S})$$

where $\mathbf{I}$ is the identity matrix of appropriate size. Also, $\mathcal{P}\mathbf{U} = \mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{P}^T\mathbf{U} = \mathbf{U}$. Therefore, $(\mathbf{I} - \mathcal{P})\mathbf{U} = \mathbf{0}$. Similarly, $\mathbf{Y}^T(\mathbf{I} - \mathcal{S}) = \mathbf{0}$. Therefore,

$$\begin{aligned}
\|\mathbf{G} - \mathcal{P}\mathbf{G}\mathcal{S}\| &\leq \|(\mathbf{I} - \mathcal{P})\mathbf{G}\| + \|\mathcal{P}\mathbf{G}(\mathbf{I} - \mathcal{S})\| \\
&= \|(\mathbf{I} - \mathcal{P})(\mathbf{G} - \mathbf{U}\mathbf{U}^T\mathbf{G})\| + \|\mathcal{P}(\mathbf{G} - \mathbf{G}\mathbf{Y}\mathbf{Y}^T)(\mathbf{I} - \mathcal{S})\| \\
&\leq \big(\|(\mathbf{I} - \mathcal{P})\| + \|\mathcal{P}\|\|(\mathbf{I} - \mathcal{S})\|\big)\hat{\sigma}_{r+1} \\
&= \eta_p(1+\eta_s)\hat{\sigma}_{r+1}.
\end{aligned}$$

In the above inequality, we have made use of the fact that $\|\mathbf{I} - \mathcal{P}\| = \|\mathcal{P}\| = \eta_p$ and $\|\mathbf{I} - \mathcal{S}\| = \|\mathcal{S}\| = \eta_s$ as long as the projectors are neither null nor the identity [86]. In the second line of the above inequality, we have made use of $(\mathbf{I} - \mathcal{P})\mathbf{U} = \mathbf{0}$ and $\mathbf{Y}^T(\mathbf{I} - \mathcal{S}) = \mathbf{0}$. Similarly, it is possible to express the error matrix as:

$$\mathbf{G} - \mathcal{P}\mathbf{G}\mathcal{S} = \mathbf{G}(\mathbf{I} - \mathcal{S}) + \mathbf{G}\mathcal{S} - \mathcal{P}\mathbf{G}\mathcal{S} = \mathbf{G}(\mathbf{I} - \mathcal{S}) + (\mathbf{I} - \mathcal{P})\mathbf{G}\mathcal{S}.$$

Therefore, another error bound can be obtained as

$$\begin{aligned}
\|\mathbf{G} - \mathcal{P}\mathbf{G}\mathcal{S}\| &\leq \|\mathbf{G}(\mathbf{I} - \mathcal{S})\| + \|(\mathbf{I} - \mathcal{P})\mathbf{G}\mathcal{S}\| \\
&= \|(\mathbf{G} - \mathbf{G}\mathbf{Y}\mathbf{Y}^T)(\mathbf{I} - \mathcal{S})\| + \|(\mathbf{I} - \mathcal{P})(\mathbf{G} - \mathbf{U}\mathbf{U}^T\mathbf{G})\mathcal{S}\| \\
&\leq \big(\|(\mathbf{I} - \mathcal{S})\| + \|\mathbf{I} - \mathcal{P}\|\|\mathcal{S}\|\big)\hat{\sigma}_{r+1} \\
&= \eta_s(1+\eta_p)\hat{\sigma}_{r+1}.
\end{aligned}$$

where $\|\mathbf{I} - \mathcal{S}\| = \eta_s$ is used. Combining the above two inequalities yields inequality 4.18. □

In the above error bound, when $\mathbf{U}$ and $\mathbf{Y}$ are the r most dominant exact left and right singular vectors of $\mathbf{G}$, then $\hat{\sigma}_{r+1} = \sigma_{r+1}$, where σ_{r+1} is the $r+1$-th singular value of $\mathbf{G}$, since

$$\|\mathbf{G} - \mathbf{U}\mathbf{U}^T\mathbf{G}\| = \|\mathbf{G} - \mathbf{G}\mathbf{Y}\mathbf{Y}^T\| = \sigma_{r+1}. \tag{4.19}$$

In that case, ϵ_f is the error factor of the CUR decomposition when compared against the optimal rank-r reduction error obtained by SVD. As demonstrated in our numerical experiments, the TDB-CUR algorithm closely approximates the rank-r SVD approximation of $\mathbf{V}$.

4.1.6 Oversampling for Improved Condition Number

The above error analysis shows that the CUR rank-r approximation can be bounded by an error factor ϵ_f times the maximum error obtained from the orthogonal projection of $\mathbf{G}$ onto $\mathbf{U}$ or $\mathbf{Y}$. This analysis reveals that better conditioned $\mathbf{P}^T\mathbf{U}$ and $\mathbf{S}^T\mathbf{Y}$ matrices result in smaller η_p and η_s, which then results in smaller error factor ϵ_f. In the context of DEIM interpolation, it was shown that *oversampling* can improve the condition number of oblique projections [75]. The authors demonstrated that augmenting the original DEIM algorithm with an additional $m = \mathcal{O}(r)$ sampling points can reduce the value of η_p, leading to smaller approximation errors. This procedure of sampling more rows than the number of basis vectors leads to an overdetermined system where an approximate solution can be found via a least-square regression rather than interpolation. Additionally, it was shown in [3] that for matrices with rapidly decaying singular values (as targeted in this work), oversampling improves the accuracy of CUR decompositions.

In the following, we extend the TDB-CUR algorithm for row oversampling. As a direct result of the oversampling procedure, the oblique projection of $\mathbf{G}$ onto the range of the orthonormal basis $\mathbf{Q}$ becomes:

$$\mathbf{Z} = \mathbf{Q}(\mathbf{p},:)^{\dagger}\mathbf{G}(\mathbf{p},:), \quad \text{where} \quad \mathbf{Q}(\mathbf{p},:)^{\dagger} = (\mathbf{Q}(\mathbf{p},:)^T\mathbf{Q}(\mathbf{p},:))^{-1}\mathbf{Q}(\mathbf{p},:)^T, \tag{4.20}$$

and $\mathbf{p} \in \mathbb{N}^{r+m}$ contains the $r + m \ll n$ row indices. Note that the $\mathbf{Q}(\mathbf{p},:)^{\dagger}$ is *not* the pseudo-inverse of $\mathbf{Q}(\mathbf{p},:)$. Therefore, the oblique projection becomes a least squares best-fit solution. Also, increasing the number of oversampling points decreases $\eta_p = \|\mathbf{Q}(\mathbf{p},:)^\dagger\|$ and it follows that for the maximum number of oversampling points, i.e., when all the rows are sampled, the orthogonal projection of every column of $\mathbf{G}$ onto $\text{Ran}(\mathbf{G}(:,\mathbf{s}))$ is recovered, where η_p attains its smallest value, which is $\eta_p = 1$. Similar to the interpolatory projector, the row oversampling can also be formulated as an oblique projector:

$$\mathcal{P} = \mathbf{U}(\mathbf{P}^T\mathbf{U})^\dagger\mathbf{P}^T, \tag{4.21}$$

where $\mathbf{P} \in \mathbb{R}^{n\times(r+m)}$. However, unlike the interpolatory projector, $\mathbf{P}^T\mathcal{P}\mathbf{A} \neq \mathbf{A}(\mathbf{p},:)$.

We refer to the above sampling procedure as `OS-DEIM`, where OS refers to the oversampling algorithm. Since the DEIM only provides sampling points equal to the number of basis vectors, we use the GappyPOD+E algorithm from [75, Algorithm 1] to sample a total of $r+m$ rows. While any sparse selection procedure can be used, the GappyPOD+E was shown to outperform other common choices like random sampling or leverage scores [63]. Finally, it is possible to oversample the columns in an analogous manner to decrease η_s. In all of the examples considered in this paper, we apply row oversampling, but ultimately the decision for row oversampling, column oversampling, or both may be made by requiring that η_p and η_s be smaller than some threshold values.

4.1.7 Rank Adaptivity

In order to control the error while avoiding unnecessary computations, the rank of the TDB must be able to adapt on the fly. We show that it is easy to incorporate mode adaptivity into the TDB-CUR algorithm.

In the case of rank reduction, once the new rank is chosen, such that $r^k < r^{k-1}$, the low-rank matrices are simply truncated to retain only the first r^k components, i.e. $\mathbf{U}(:,1:r^k)$, $\mathbf{\Sigma}(1:r^k,1:r^k)$, and $\mathbf{Y}(:,1:r^k)$. On the other hand, the rank can be increased, such that $r^k > r^{k-1}$, by sampling more columns (r^k) than the number of basis vectors (r^{k-1}), i.e. oversampling. Similar to the procedure used for oversampling the rows, the column indices are determined via the GappyPOD+E algorithm. While this provides a straightforward approach for *how* to adapt the rank, it does not address *when* the rank should be adapted, or *what* that new rank should be.

Informed by the error analysis from the preceding section, we devise a suitable criterion for controlling the error via rank addition and removal. Since it is not possible to know the true error without solving the expensive FOM, we devise a proxy for estimating the low-rank approximation error:

$$\epsilon(t) = \frac{\hat{\sigma}_r(t)}{\left(\sum_{i=1}^{r} \hat{\sigma}_i(t)^2\right)^{1/2}}, \tag{4.22}$$

where $\hat{\sigma}_i$ are the singular values of the low-rank approximation from the previous time step. Assuming the low-rank approximation is near-optimal in its initial condition, we can use the trailing singular value as a proxy for the low-rank approximation error.

To make the error proxy more robust for problems of varying scale and magnitude, we divide by the Frobenius norm of $\hat{\mathbf{V}}$, where it is well-known that $\|\hat{\mathbf{V}}\|_F = (\sum_{i=1}^{r} \hat{\sigma}_i^2)^{1/2}$. Rather than set a hard threshold, we add/remove modes to maintain ϵ within a desired range, $\epsilon_l \leq \epsilon \leq \epsilon_u$, where ϵ_l and ϵ_u are user-specified lower and upper bounds, respectively. If $\epsilon > \epsilon_u$ we increase the rank to $r+1$, and if $\epsilon < \epsilon_l$ we decrease the rank to $r-1$. As a result, this approach avoids the undesirable behavior of repeated mode addition and removal, which is observed by setting a hard threshold. The rank-adaptive TDB-CUR algorithm is detailed in Algorithm 1.

It is important to note that this isn't the only criterion for mode addition and removal, and one can devise a number of strategies based on the problem at hand. However, from our numerical experiments, this approach has proved to be simple and effective, and it does a good job at capturing the trend of the true error. For more details on estimating rank and selection criteria, we refer the reader to [87, Section 2.3].

Algorithm 1 Rank-Adaptive TDB-CUR Algorithm

```
Input: $\tilde{\mathbf{U}} \in \mathbb{R}^{n\times\tilde{r}}$, $\tilde{\mathbf{\Sigma}} \in \mathbb{R}^{\tilde{r}\times\tilde{r}}$, $\tilde{\mathbf{Y}} \in \mathbb{R}^{s\times\tilde{r}}$, $\tilde{r}$, $m$ ($\sim$ indicates previous time step)
Output: $\mathbf{U} \in \mathbb{R}^{n\times r}$, $\mathbf{\Sigma} \in \mathbb{R}^{r\times r}$, $\mathbf{Y} \in \mathbb{R}^{s\times r}$, $r$
1: $\epsilon = \tilde{\mathbf{\Sigma}}(\tilde{r},\tilde{r})/\|\texttt{diag}(\tilde{\mathbf{\Sigma}})\|_2$    ▷ Compute error proxy for adaptive rank criteria.
2: if $\epsilon > \epsilon_u$ then    ▷ Increase rank if $\epsilon$ exceeds the upper threshold, $\epsilon_u$.
3:     $r = \tilde{r} + 1$
4: else if $\epsilon < \epsilon_l$ then    ▷ Decrease rank if $\epsilon$ falls below the lower threshold, $\epsilon_l$.
5:     $r = \tilde{r} - 1$
6:     $\tilde{\mathbf{U}} = \tilde{\mathbf{U}}(:, 1:r)$; $\tilde{\mathbf{\Sigma}} = \tilde{\mathbf{\Sigma}}(1:r, 1:r)$; $\tilde{\mathbf{Y}} = \tilde{\mathbf{Y}}(:, 1:r)$    ▷ Truncate TDB matrices.
7: else    ▷ Keep rank the same.
8:     $r = \tilde{r}$
9: end if
10: $\mathbf{s} \leftarrow \texttt{sparse\_selection}(\tilde{\mathbf{Y}}, r)$ [a]    ▷ Compute $r$ column indices.
11: $\mathbf{p} \leftarrow \texttt{sparse\_selection}(\tilde{\mathbf{U}}, r+m)$    ▷ Compute $r+m$ row indices.
12: $\mathbf{p}_a \leftarrow \texttt{find\_adjacent}(\mathbf{p})$    ▷ Find adjacent points required to compute $\mathbf{G}(\mathbf{p}, :)$.
13: $\hat{\mathbf{V}}(:, \mathbf{s}) = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{Y}}(\mathbf{s}, :)^T$    ▷ Construct low-rank approximation of columns in $\mathbf{s}$.
14: $\mathbf{G}(:, \mathbf{s}) = \mathcal{G}(\hat{\mathbf{V}}(:, \mathbf{s}))$    ▷ Compute nonlinear map at the selected columns.
15: $\hat{\mathbf{V}}([\mathbf{p}, \mathbf{p}_a], :) = \tilde{\mathbf{U}}([\mathbf{p}, \mathbf{p}_a], :)\tilde{\mathbf{\Sigma}}\tilde{\mathbf{Y}}^T$    ▷ Construct low-rank approximation of rows in $[\mathbf{p}, \mathbf{p}_a]$.
16: $\mathbf{G}(\mathbf{p}, :) = \mathcal{G}(\hat{\mathbf{V}}([\mathbf{p}, \mathbf{p}_a], :))$    ▷ Compute nonlinear map at the selected rows.
17: $\mathbf{QR} = \texttt{qr}(\mathbf{G}(:, \mathbf{s}), \texttt{'econ'})$    ▷ Compute the economy QR of $\mathbf{G}(:, \mathbf{s})$.
18: $\mathbf{Z} = \mathbf{Q}(\mathbf{p}, :)^\dagger \mathbf{G}(\mathbf{p}, :)$    ▷ Compute $\mathbf{Z}$ as an oblique projection of $\mathbf{G}$ onto $\mathbf{Q}$.
19: $\mathbf{U_Z}\mathbf{\Sigma}\mathbf{Y}^T = \texttt{svd}(\mathbf{Z}, \texttt{'econ'})$    ▷ Compute the economy SVD of $\mathbf{Z}$.
20: $\mathbf{U} = \mathbf{Q}\mathbf{U_Z}$    ▷ In-subspace rotation of the orthonormal basis, $\mathbf{Q}$.
```

[a] While the present work uses the GappyPOD+E for `sparse_selection`, the user is free to choose their favorite sparse selection algorithm.

4.1.8 Computing $\mathbf{G}(\mathbf{p}, :)$

Up until this point, we have considered $\mathbf{G} = \mathcal{G}(\hat{\mathbf{V}}^{k-1})$ to be an $n \times s$ matrix resulting from an order-p explicit temporal discretization of Eq. 4.8. In Algorithm 1, we showed that sparse row and column measurements, $\mathbf{G}(\mathbf{p},:)$ and $\mathbf{G}(:,\mathbf{s})$, could be used to efficiently compute an approximation to the rank-r SVD of $\mathbf{G}$. While $\mathbf{G}(:,\mathbf{s})$ is straightforward to compute for independent random samples, as discussed in Section 4.1.4, computing $\mathbf{G}(\mathbf{p},:)$ depends on a set of adjacent points, $\mathbf{p}_a$, according to the spatial discretization scheme. See Remark 4.1.1. As a result, for higher-order integration schemes, special care must be taken in the computation of $\mathbf{G}(\mathbf{p},:)$. To demonstrate this, we consider the second-order explicit Runge-Kutta scheme where

$$\mathbf{G} = \hat{\mathbf{V}}^{k-1} + \Delta t \mathcal{F}\left(t^{k-1} + \frac{1}{2}\Delta t,\, \hat{\mathbf{V}}^{k-1} + \frac{1}{2}\Delta t \mathcal{F}\left(t^{k-1},\, \hat{\mathbf{V}}^{k-1}\right)\right).$$

After determining the row indices, $\mathbf{p}$ and $\mathbf{p}_a$, $\mathbf{G}(\mathbf{p},:)$ can be computed as follows:

1. Compute $\hat{\mathbf{V}}^{k-1}([\mathbf{p},\mathbf{p}_a],:) = \mathbf{U}^{k-1}([\mathbf{p},\mathbf{p}_a],:)\mathbf{\Sigma}^{k-1}\mathbf{Y}^{k-1^T}$.
2. Compute the first stage $\mathbf{F}_1 = \mathcal{F}\left(t^{k-1}, \hat{\mathbf{V}}^{k-1}\right)$ at the $\mathbf{p}$ rows as

$$\mathbf{F}_1(\mathbf{p},:) = \mathcal{F}\left(t^{k-1}, \hat{\mathbf{V}}^{k-1}([\mathbf{p},\mathbf{p}_a],:)\right).$$

   Note, if explicit Euler method is used, no additional steps are required to compute $\mathbf{G}(\mathbf{p},:)$. If a higher order scheme is used, proceed with the following steps.
3. The final stage of the second order integration scheme requires taking a half step to evaluate $\mathcal{F}$ at the midpoint:

$$\mathbf{F}_2 = \mathcal{F}\left(t^{k-1} + \frac{1}{2}\Delta t, \hat{\mathbf{V}}^{k-1} + \frac{1}{2}\Delta t \mathbf{F}_1\right).$$

   Here, we require $\mathbf{F}_2(\mathbf{p},:)$, given by

$$\mathbf{F}_2(\mathbf{p},:) = \mathcal{F}\left(t^{k-1} + \frac{1}{2}\Delta t, \hat{\mathbf{V}}^{k-1}([\mathbf{p},\mathbf{p}_a],:) + \frac{1}{2}\Delta t \mathbf{F}_1([\mathbf{p},\mathbf{p}_a],:)\right).$$

   Notice that we now require $\mathbf{F}_1(\mathbf{p}_a,:)$ to evaluate the above expression. While this can be computed according to Step 2, where $\mathbf{p}_a$ will have its own set of adjacent points $\mathbf{p}_{aa}$, this process quickly gets out of hand, especially as more stages are added to the integration scheme. As a result, for higher-order schemes, the efficiency afforded by Algorithm 1 will deteriorate, and the resulting implementation will become increasingly complex. To overcome these challenges, we instead compute the low-rank approximation $\hat{\mathbf{F}}_i \approx \mathbf{F}_i$, using the sparse row and column measurements, $\mathbf{F}_i(\mathbf{p},:)$ and $\mathbf{F}_i(:,\mathbf{s})$, which are already required for computing $\mathbf{G}(\mathbf{p},:)$ and $\mathbf{G}(:,\mathbf{s})$. Here, the subscript denotes the i^{th} stage of the integration scheme. The first step is to compute $\mathbf{U}_{\mathbf{F}_i}$ as an orthonormal basis for the $\text{Ran}(\mathbf{F}_i(:,\mathbf{s}))$, using QR. Next, compute the oblique projection of $\mathbf{F}_i$ onto $\mathbf{U}_{\mathbf{F}_i}$, such that

$$\hat{\mathbf{F}}_i = \mathbf{U}_{\mathbf{F}_i}\mathbf{U}_{\mathbf{F}_i}(\mathbf{p},:)^{\dagger}\mathbf{F}_i(\mathbf{p},:),$$

where $\mathbf{U}_{\mathbf{F}_i}(\mathbf{p},:)^{\dagger} = (\mathbf{U}_{\mathbf{F}_i}(\mathbf{p},:)^T\mathbf{U}_{\mathbf{F}_i}(\mathbf{p},:))^{-1}\mathbf{U}_{\mathbf{F}_i}(\mathbf{p},:)^T$. Using this low-rank approximation, $\mathbf{F}_i(\mathbf{p}_a,:)$ is readily approximated by $\hat{\mathbf{F}}_i(\mathbf{p}_a,:) = \mathbf{U}_{\mathbf{F}_i}(\mathbf{p}_a,:)\mathbf{U}_{\mathbf{F}_i}(\mathbf{p},:)^{\dagger}\mathbf{F}_i(\mathbf{p},:)$. Although we have considered the second-order Runge-Kutta method in the example above, this approach is easily extended to higher-order Runge-Kutta methods as well as linear multistep methods.

4.1.9 A Working Example: Stochastic Burger's Equation

Here we provide a working example to demonstrate the TDB-CUR methodology. We consider the one-dimensional Burgers equation subject to random initial and boundary conditions as follows:

$$\begin{aligned}
&\frac{\partial v}{\partial t} + v\frac{\partial v}{\partial x} = \nu\frac{\partial^2 v}{\partial x^2}, && x \in [0,1], t \in [0,5],\\
&v(x,0;\xi) = \sin(2\pi x)\left[0.5\left(e^{\cos(2\pi x)} - 1.5\right) + \sigma\sum_{i=1}^{d}\sqrt{\lambda_{x_i}}\psi_i(x)\xi_i\right], && x \in [0,1], \xi_i \sim \mathcal{N}(\mu,\sigma^2),\\
&v(0,t;\xi) = -\sin(2\pi t) + \sigma\sum_{i=1}^{d}\lambda_{t_i}\varphi_i(t)\xi_i, && x = 0, \xi_i \sim \mathcal{N}(\mu,\sigma^2),
\end{aligned}$$

where $\nu = 2.5\times 10^{-3}$. The stochastic boundary at $x = 0$ is specified above and the boundary at $x = 1$ is $v(x = 1, t; \xi) = 0$. We use weak treatment of the boundary conditions for both the FOM and TDB [73]. The random space is taken to be $d = 17$ dimensional and ξ_i's are sampled from a normal distribution with mean $\mu = 0$, standard deviation $\sigma = 0.001$, and $s = 256$. In the stochastic boundary specification, we take $\varphi_i(t) = \sin(i\pi t)$ and $\lambda_{t_i} = i^{-2}$. In the stochastic initial condition, λ_{x_i} and $\psi_i(x)$ are the eigenvalues and eigenvectors of the spatial squared-exponential kernel, respectively. The fourth-order explicit Runge-Kutta method is used for time integration with $\Delta t = 2.5 \times 10^{-4}$. For discretization of the spatial domain, we use a second-order finite difference scheme on a uniform grid with $n = 401$. We use relative Frobenius error $\mathcal{E}^k = \|\hat{\mathbf{V}}^k - \mathbf{V}^k\|_F / \|\mathbf{V}^k\|_F$ in our analysis.

We first solve the system using TDB-CUR with fixed rank and compare the results against the DLRA by solving Eqs. 4.6a-4.6c. For TDB-CUR, the rows are oversampled with $m = 5$. No sparse sampling strategy is used for DLRA, and Eqs. 4.6a-4.6c are solved as is. In Figure 21(a), we compare the error of TDB-CUR and DLRA for different values of r. For $r = 6$, TDB-CUR has larger error compared to DLRA. This result is expected since TDB-CUR has an additional source of error from the sparse sampling procedure. However, as the rank is increased to $r = 9$, the conditioning of the DLRA starts to deteriorate and the error of TDB-CUR is actually lower than DLRA. In fact, for $r > 9$, the DLRA is unstable and cannot be integrated beyond the first time step. On the other hand, TDB-CUR remains stable, and the error decays as the rank is increased to a maximum value of $r = 18$. While it is reasonable to expect that the error can be reduced further by increasing the rank to values of $r > 18$, it is important to note that the rank of the initial condition is exactly $r = 18$. Therefore, in order to increase the rank of the system beyond $r = 18$ in a principled manner, we employ the rank adaptive strategy from Section 4.1.7. To this end, we initialize the system with rank $r_0 = 18$, and use an upper threshold of $\epsilon_u = 10^{-8}$ for mode addition. As observed in Figure 21(b), the rank is increased in time to a maximum of 23, leading to a further reduction in the error.

The mean solution is shown in Figure 22 along with the first 10 QDEIM sampling points. We observe that the sampling points are concentrated near the stochastic boundary at $x = 0$ and also at points in the domain where shocks develop. Figure 23 shows the evolution of the first two spatial modes, $\mathbf{u}_1$ (top) and $\mathbf{u}_2$ (bottom), where we observe excellent agreement between the FOM and TDB-CUR. It is important to note that these modes are energetically ranked according to the first and second singular values shown in Figure 24(a). Therefore, we observe that $\mathbf{u}_1$ captures the large scale energy containing structure, while $\mathbf{u}_2$ captures the small scale structure that is highly localized in space.

In Figure 24(a), we show that TDB-CUR accurately captures the leading singular values of the FOM solution, despite the large gap between the first and last resolved singular values. Finally, Figure 24(b) compares the CPU time of the FOM, DLRA, and TDB-CUR

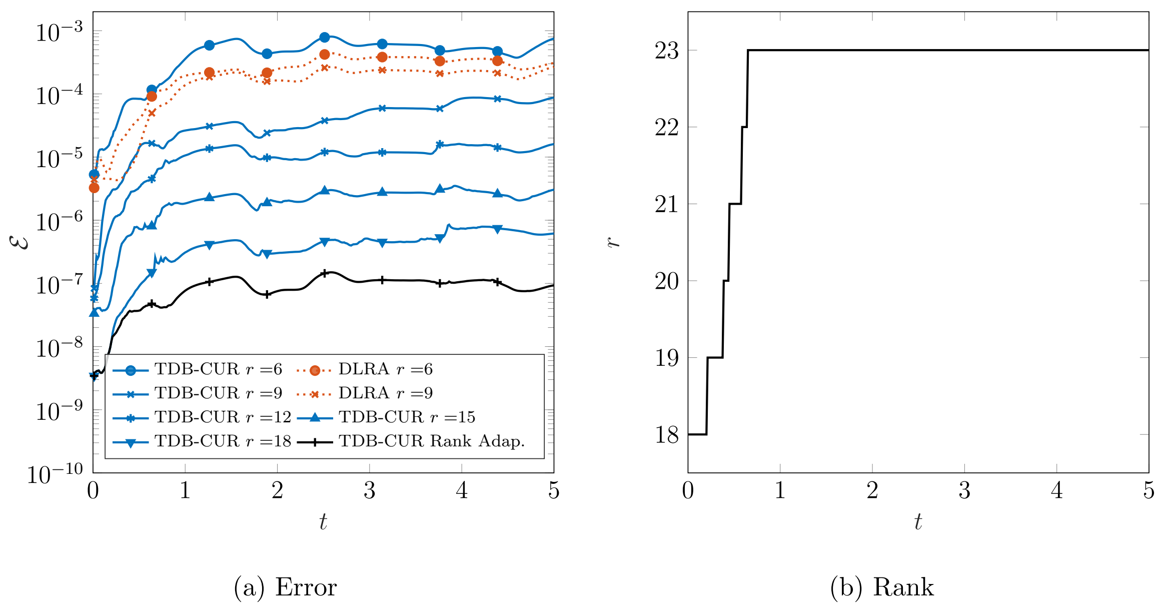


Figure 21: Stochastic Burgers equation: (a) Relative error evolution for fixed and adaptive rank. (b) Rank evolution using upper threshold $\epsilon_u = 10^{-8}$ for mode addition.

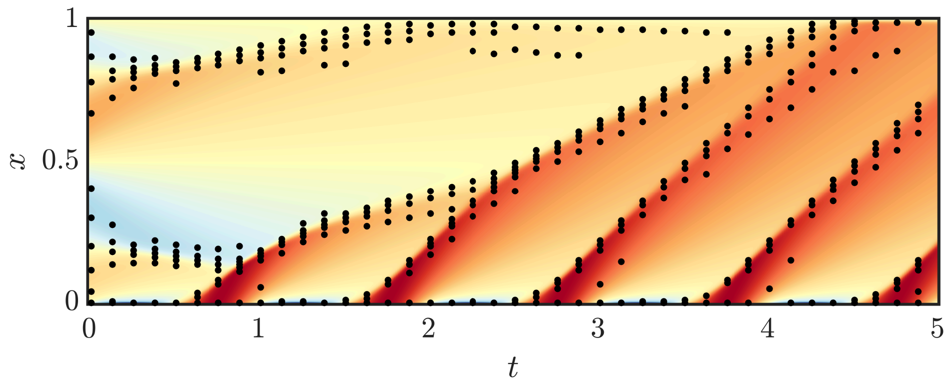


Figure 22: Stochastic Burgers equation: mean solution with the first 10 QDEIM points (black dots).

as the number of rows and columns of the matrix are increased simultaneously. We take $n = s$ and observe that the FOM and DLRA scales quadratically ($\mathcal{O}(ns)$) while TDB-CUR scales linearly ($\mathcal{O}(n + s)$). As the matrix size is increased, the disparity in CPU time

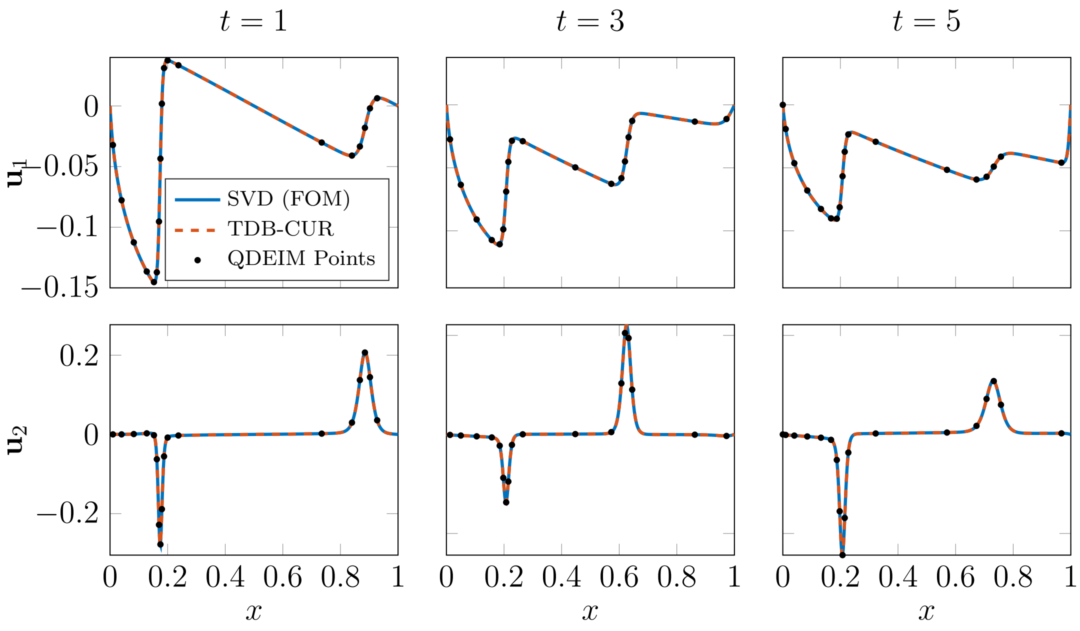
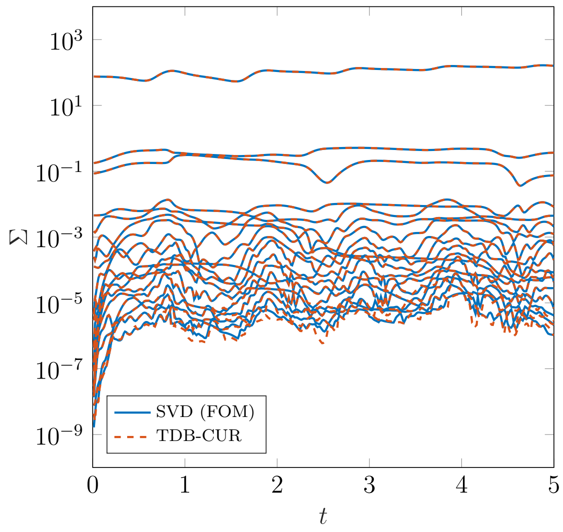


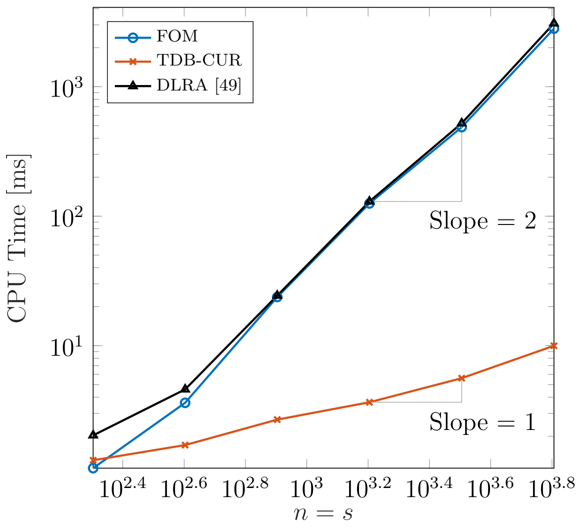
Figure 23: Stochastic Burgers equation: Evolution of first two spatial modes, $\mathbf{u}_1$ and $\mathbf{u}_2$, and QDEIM points. Excellent agreement between the FOM and TDB-CUR is observed.

becomes even more apparent, making the case for solving massive MDEs using TDB-CUR. To further illustrate the reduction in computational cost, we compare the TDB-CUR against the unconventional integrator from [24]. Figure 25 shows the error versus the total time to solution (cost) for $r = 6, 9, 12, 15, 18$. For the TDB-CUR, we observe a rapid decrease in the error as the cost is increased. On the other hand, for the unconventional integrator, not only is the error orders of magnitude larger than TDB-CUR, the time to solution is also significantly higher.

While Figures 21-25 demonstrate the accuracy, efficiency, rank-adaptivity, and favorable numerical performance of the TDB-CUR method, they do not convey the minimally intrusive nature of its implementation. To give a better perspective on the implementation efforts, the MATLAB code for solving the stochastic Burgers equation using the TDB-CUR method is provided in Appendix C (Listings C.1 and C.2). While the code contains lines specific to the TDB-CUR method, after reviewing the entire code, it will become apparent that many of the included lines are already required for solving the FOM Burgers equation. Furthermore, there is no term-by-term implementation required to preserve efficiency and the FOM implementation of the Burgers equation (`function f`) is used to compute the sparse row and column samples. Therefore, given an existing FOM implementation, the code required to implement the TDB-CUR method is minimal. The code blocks required for implementing the method are labeled with `%% TDB-CUR` in the attached code.



(a) Singular Values



(b) CPU Time

Figure 24: Stochastic Burgers equation: (a) First 18 singular values of FOM vs TDB-CUR. (b) CPU time for scaling $n = s$ for FOM and TDB-CUR with fixed $r = 6$.

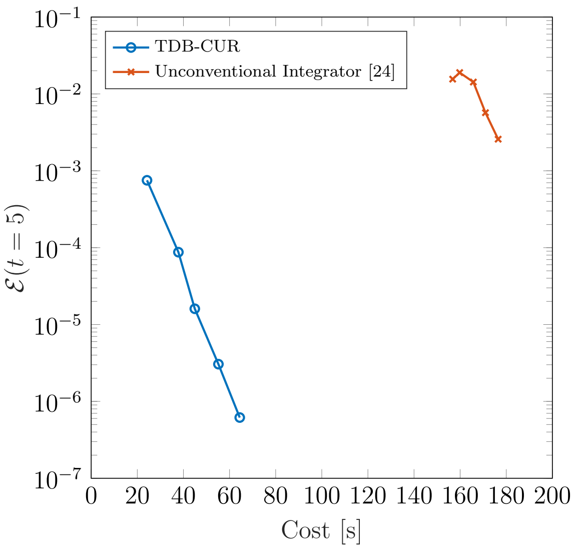


Figure 25: Stochastic Burgers equation: Error at $t = 5$ versus total cost in seconds.

5.0 Conclusions and Outlook

In Chapter 2, we presented the f-OTD method for the computation of sensitivities in evolutionary systems governed by time dependent ordinary/partial differential equations. We demonstrated that the rank of f-OTD for two diverse applications is much smaller than the number of sensitivity fields. In contrast to adjoint based methods, f-OTD is not tied to an objective function and requires solving a system of *forward* equations that does not require any I/O operation. We showed that a single set of f-OTD modes can be formulated to compress the sensitivities of multi-variable PDEs. We demonstrated this capability by computing sensitivities of multiple species with respect to reaction parameters in a turbulent reactive flow. In addition, we contrasted the f-OTD with OTD and demonstrated why OTD is not appropriate for parametric/forced sensitivity analysis. In contrast to traditional ROM approaches, f-OTD extracts the low-rank approximation directly from the sensitivity equations as opposed to a data-driven approach, such as POD or DMD, which requires the full-dimensional sensitivity data. The data-driven techniques have the computational advantage that the modes are computed once and the cost of solving ROM is usually insignificant. However, the low-rank subspace in the data-driven approach is fine tuned to particular operating conditions, whereas the f-OTD subspace is evolved with the dynamics of the system and does not require such fine-tuning. As such, f-OTD is an *on-the-fly* model compression that is achieved by extracting instantaneous correlated structures in the solution.

In Chapter 3, we extended the f-OTD method to compute finite-time nonlinear sensitivities, which we called NL-fOTD. Sensitivities were computed by perturbing parameterized nonlinear systems around a base state, and retaining the higher order, nonlinear terms. We demonstrated the method for computing low-rank approximations of nonlinear sensitivities for (i) 3D ODE (ii) 2D compressible flow and (iii) 2D turbulent reacting flow. In all cases, it was shown that low-rank structure exists, and could be extracted using the NL-fOTD approximation. Solving these systems via the low-rank NL-fOTD components offered an intuitive and interpretable representation of seemingly complex systems, while maintaining good accuracy and numerical performance.

In Chapter 4, we developed a rank-adaptive method to solve nonlinear matrix differential equations (MDEs) that addresses many of the challenges associated with TDB ROMs. To this end, we presented the TDB-CUR algorithm for solving MDEs via low-rank approximation. The algorithm is based on a time-discrete variational principle that leverages sparse sampling to efficiently compute a low-rank matrix approximation at each iteration of the time-stepping scheme. This approach was shown to be accurate, well-conditioned, computationally efficient, and minimally intrusive. Numerical experiments illustrated that the TDB-CUR algorithm provides a near-optimal low-rank approximation to the solution of MDEs, while significantly reducing the computational cost. Moreover, we showed the method is robust in the presence of small singular values, and significantly outperforms DLRA based on the time continuous variational principle. Although not investigated in the present work, the TDB-CUR algorithm is also highly parallelizable, making it an attractive option for high-performance computing tasks.

Despite the meaningful contributions of the present work, further developments are required to enable the routine use of time dependent basis reduced order models for solving a wide variety of science and engineering problems. To this end, future efforts should focus on the following areas:

- **Implicit integration:** While the present work demonstrates the utility of the TDB-CUR method for explicit schemes, many practical problems of interest are stiff, requiring implicit integration to avoid restrictive time step limitations. However, application of off-the-shelf implicit schemes is nontrivial, and new developments will be required to preserve the efficiency of the TDB-CUR method.
- **Nonlinear approximation:** Future work should also investigate the feasibility of using time dependent bases for locally linear and higher-order nonlinear approximation. It is important to remember that even though the basis is evolved with the dynamics of the system, the underlying approximation is still linear. Although we have demonstrated that many systems are low-rank when expressed as a linear expansion in a time dependent basis, this approach is ineffective for high dimensional systems that exhibit a slowly decaying singular value spectrum, e.g. perturbations in advection dominated problems. In order to accurately resolve these systems, a large number of modes are required,

defeating the purpose of the reduced order modeling task. Therefore, to overcome these challenges, nonlinear and locally linear approximations should be investigated as a means to reduce the number of modes required in the time dependent basis.

- **Intrusiveness:** While the presented approach is minimally intrusive and can be applied to systems containing general nonlinearities, future work should aim to make this method fully non-intrusive, allowing the full order model to be leveraged as a black box. This will allow the method to be applied to proprietary solvers while reducing the overall implementation efforts, making this powerful methodology more accessible to researchers and practitioners, alike.

Appendix A Reactive Source Term Specification

Table 1: Reactive source terms with species concentration denoted by $[\cdot]$. Each $\mathbf{s}_i$ is scaled by 10^2 for time scale adjustment with the flow and the parameter values are assigned as follows: $\alpha_1 = 2.54 \times 10^{-2}$, $\alpha_2 = 160$, $\alpha_3 = 3.74 \times 10^{-5}$, $\alpha_4 = 0.449$, $\alpha_5 = 1.12 \times 10^5$, $\alpha_6 = 5.13 \times 10^{-4}$, $\alpha_7 = 2.36 \times 10^{-2}$, $\alpha_8 = 14.6$, $\alpha_9 = 6.24 \times 10^{-2}$, $\alpha_{10} = 140.5$, $\alpha_{11} = 3.93 \times 10^{-4}$, $\alpha_{12} = 2.36 \times 10^{-2}$, $\alpha_{13} = 14.6$, $\alpha_{14} = 5.523$, $\alpha_{15} = 160$, $\alpha_{16} = 8.01 \times 10^{-4}$, $\alpha_{17} = 1.11 \times 10^{-3}$, $\alpha_{18} = 3.105$, $\alpha_{19} = 1060$, $\alpha_{20} = 1.65 \times 10^{-3}$, $\alpha_{21} = 8.177$, $\alpha_{22} = 3160$, $\alpha_{23} = 3.456$, $\alpha_{24} = 2.50 \times 10^5$, $\alpha_{25} = 1.80 \times 10^{-5}$, $\alpha_{26} = 50$, $\alpha_{27} = 3.70 \times 10^{-6}$, $\alpha_{28} = 3.00 \times 10^{-8}$, $\alpha_{29} = 9.01 \times 10^{-2}$, $\alpha_{30} = 3190$, $\alpha_{31} = 1.52 \times 10^{-9}$, $\alpha_{32} = 2.77 \times 10^{-2}$, $\alpha_{33} = 18$, and $\alpha_{34} = 2.22 \times 10^{-4}$.

$$
\begin{aligned}
&\mathbf{s}_1 = (\alpha_1[13][2])/(\alpha_2 + [2]) - \alpha_3[1][15] \\
&\mathbf{s}_2 = -(\alpha_1[13][2]/(\alpha_2 + [2]) \\
&\mathbf{s}_3 = (\alpha_4[9][4]/(\alpha_5 + [4]) - \alpha_6[3] - (\alpha_7[17][3])/(\alpha_8 + [3]) \\
&\mathbf{s}_4 = (\alpha_4[9][4])/(\alpha_5 + [4]) \\
&\mathbf{s}_5 = (\alpha_9[9][6])/(\alpha_{10} + [6]) - \alpha_{11}[5] - (\alpha_{12}[17][5])/(\alpha_{13} + [5]) \\
&\mathbf{s}_6 = -(\alpha_9[9][6]/(\alpha_{10} + [6]) \\
&\mathbf{s}_7 = (\alpha_{14}[24][8])/(\alpha_{15} + [8]) - \alpha_{16}[7][15] - \alpha_{17}[16][7] \\
&\mathbf{s}_8 = -(\alpha_{14}[24][8])/(\alpha_{15} + [8]) \\
&\mathbf{s}_9 = (\alpha_{18}[25][10])/(\alpha_{19} + [10]) - \alpha_{20}[9][15] \\
&\mathbf{s}_{10} = -(\alpha_{18}[25][10])/(\alpha_{19} + [10]) \\
&\mathbf{s}_{11} = (\alpha_{21}[9][12])/(\alpha_{22} + [12]) - (\alpha_{23}[21][11])/(\alpha_{24} + [11]) \\
&\mathbf{s}_{12} = -(\alpha_{21}[9][12])/(\alpha_{22} + [12]) \\
&\mathbf{s}_{13} = (\alpha_{25}[9][14])/(\alpha_{26} + [14]) - \alpha_{27}[13][15] - \alpha_{28}[13][19] \\
&\mathbf{s}_{14} = -(\alpha_{25}[9][14])/(\alpha_{26} + [14]) \\
&\mathbf{s}_{15} = -(\alpha_3[1] + \alpha_{16}[7] + \alpha_{20}[9] + \alpha_{27}[13])[15] \\
&\mathbf{s}_{16} = -\alpha_{17}[16][7] \\
&\mathbf{s}_{17} = (\alpha_{29}[9][18])/(\alpha_{30} + [18]) - \alpha_{31}[17][19] \\
&\mathbf{s}_{18} = -(\alpha_{29}[9][18])/(\alpha_{30} + [18]) \\
&\mathbf{s}_{19} = -\alpha_{31}[17][19] - \alpha_{28}[13][19] \\
&\mathbf{s}_{20} = 0 \\
&\mathbf{s}_{21} = (\alpha_{32}[20][22])/(\alpha_{33} + [22]) - \alpha_{34}[21][23] \\
&\mathbf{s}_{22} = -(\alpha_{32}[20][22])/(\alpha_{33} + [22]) \\
&\mathbf{s}_{23} = -\alpha_{34}[21][23]
\end{aligned}
$$

Appendix B f-OTD Derivation for Tensor Sensitivities

We start by considering the third order quasitensor $\tilde{\mathbf{V}}' = [\tilde{\mathbf{v}}'_{ij}] \in \mathbb{R}^{\infty \times n_s \times n_r}$ that we seek to flatten into a quasimatrix $\mathbf{V}' = [\mathbf{v}'_m] \in \mathbb{R}^{\infty \times d}$. For ease of reference, we rewrite the tensor evolution Equation 2.19 below:

$$\frac{\partial \tilde{\mathbf{v}}'_{ij}}{\partial t} + (\mathbf{u} \cdot \nabla)\, \tilde{\mathbf{v}}'_{ij} = \tilde{\kappa}_{ik} \nabla^2 \tilde{\mathbf{v}}'_{kj} + \tilde{\mathcal{L}}_{\mathbf{s}_{ik}} \tilde{\mathbf{v}}'_{kj} + \tilde{\mathbf{s}}'_{ij},$$

where $i, k = 1, 2, \ldots, n_s$ and $j = 1, 2, \ldots, n_r$. We define the indices $m(i,j) = j + (i-1)n_r$ and $n(i',j') = j' + (i'-1)n_r$, where $i' = 1, 2, \ldots, n_s$ and $j' = 1, 2, \ldots, n_r$. In the above equation, the terms $\tilde{\mathbf{v}}'_{ij}$ and $\tilde{\mathbf{s}}'_{ij}$ are flattened by replacing the index pair ij with the single index m: $\mathbf{v}'_{m(i,j)} = \tilde{\mathbf{v}}'_{ij}$ and $\mathbf{s}'_{m(i,j)} = \tilde{\mathbf{s}}'_{ij}$. Next, we define a new diffusion coefficient matrix $\kappa_{mn} \in \mathbb{R}^{d \times d}$ such that the mth diagonal entry is equal the diffusion coefficient of the ith species. That is, κ_{mn} is independent of parameter index j and remains constant across all sensitivities of a given species i. Finally, the linearized reactive source term is defined as $\mathcal{L}_{\mathbf{s}_{m(i,j)n(i',j')}} = \tilde{\mathcal{L}}_{\mathbf{s}_{ii'}} \delta_{jj'}$, where $\delta_{jj'}$ is the Kronecker delta and n is a dummy index corresponding to $\mathbf{v}'_n$. From this definition, $\delta_{jj'}$ results in non-zero contribution to the summation over n only for sensitivities with respect to parameter $j' = j$. Putting this all together, the above equation can be written as:

$$\frac{\partial \mathbf{v}'_m}{\partial t} + (\mathbf{w} \cdot \nabla)\mathbf{v}'_m = \kappa_{mn} \nabla^2 \mathbf{v}'_n + \mathcal{L}_{\mathbf{s}_{mn}} \mathbf{v}'_n + \mathbf{s}'_m, \tag{B.1}$$

where $\mathcal{L}_{\mathbf{s}_{mn}} \mathbf{v}'_n$ should be interpreted as a matrix-vector multiplication for any (x_1, x_2) point in the physical space. As a result of the parametric dependence of the linear operator, Equations 2.9 and 2.10 do not hold for the tensor flattened equation. Therefore, we must derive new evolution equations for the f-OTD modes and coefficients for tensor flattened quantities. Substituting the approximation $\mathbf{v}'_m = \sum_{i=1}^{r} \mathbf{u}_i Y_{mi}$ into the above equation, it is straightforward to show that the evolution equations for the f-OTD modes and coefficients are:

$$\begin{aligned}\dot{\mathbf{u}}_i = &- \left[(\mathbf{w} \cdot \nabla)\mathbf{u}_i - \mathbf{u}_j \langle \mathbf{u}_j, (\mathbf{w} \cdot \nabla)\mathbf{u}_i \rangle\right] + \left[\nabla^2 \mathbf{u}_k - \mathbf{u}_j \langle \mathbf{u}_j, \nabla^2 \mathbf{u}_k \rangle\right] Y_{nk} \kappa_{mn} Y_{ml} C_{il}^{-1} \\ &+ \left[\mathcal{L}_{\mathbf{s}_{mn}} \mathbf{u}_k - \mathbf{u}_j \langle \mathbf{u}_j, \mathcal{L}_{\mathbf{s}_{mn}} \mathbf{u}_k \rangle\right] Y_{nk} Y_{ml} C_{il}^{-1} + \left[\mathbf{s}'_m - \mathbf{u}_j \langle \mathbf{u}_j, \mathbf{s}'_m \rangle\right] Y_{ml} C_{il}^{-1},\end{aligned} \tag{B.2}$$

and

$$\begin{aligned}\dot{Y}_{mj} = &- \langle \mathbf{u}_j, (\mathbf{w} \cdot \nabla)\mathbf{u}_i \rangle Y_{mi} + \langle \mathbf{u}_j, \nabla^2 \mathbf{u}_i \rangle Y_{ni} \kappa_{mn} \\ &+ \langle \mathbf{u}_j, \mathcal{L}_{\mathbf{s}_{mn}} \mathbf{u}_i \rangle Y_{ni} + \langle \mathbf{u}_j, \mathbf{s}'_m \rangle, \end{aligned} \tag{B.3}$$

where $\mathbf{Y} = [Y_{mi}]$ and the indices $m, n = 1, 2, \ldots, d$ and $i, j, k, l = 1, 2, \ldots, r$.

Appendix C Example MATLAB Code

Listing C.1: Matlab code to solve the stochastic Burgers equation using TDB-CUR with oversampling.

```
close all; clear all; clc
global nu xi d N

rng default
d = 17; sigma = 0.001; % random dimension; standard deviation
Ns = 256; xi = sigma*randn(Ns, d); % # of samples; random parameter samples

dt = 2.5e-4; t0 = 0.0; tf = 5.0; iter_max = round((tf-t0)/dt); save_iter = 50;

nu = 2.5e-3;
N = 401; xmin = 0; xmax = 1; % # of grid points
x = linspace(xmin, xmax, N)'; dx = x(2) - x(1);
e = ones(N,1);
D1 = spdiags([-e, e]/(2*dx), [-1, 1], N, N); D1(1, 1:3) = [-3, 4, -1]/(2*dx); D1(N, N-2:N) = [1, -4, 3]/(2*dx);    % d/dx
D2 = spdiags([e, -2*e, e]/dx^2, -1:1, N, N); D2(1, 1:4) = [2, -5, 4, -1]/dx^2; D2(N, N-3:N) = [-1, 4, -5, 2]/dx^2; % d^2/
     dx^2

lc = 0.6; K = exp(-(x-x').^2/(2*lc^2)); % squared exponential kernel
[Ux,Lx,~] = svd(K); Ux=Ux(:,1:d); Lx=sqrt(diag(Lx(1:d,1:d)))';
% [Ux,Lx,~] = svds(K,d); Lx=sqrt(diag(Lx))';
ub = sin(2*pi*x) .* (0.5*(exp(cos(2*pi*x))-1.5)); % base flow IC
Ux = sin(2*pi*x) .* (Lx.*Ux);                      % IC perturbations

%% TDB-CUR: initial condition
r = 18; % TDB rank; max=18
m = 5;  % # of rows to oversample
U = [ub, Ux];
Y = [ones(Ns,1), xi];
[U ,R1] = qr(U, 0);
[Y, R2] = qr(Y, 0);
[RU,S,RY] = svd(R1*R2');
U = U*RU; Y = Y*RY;
U=U(:,1:r); S=S(1:r,1:r); Y=Y(:,1:r);

t = t0;
umean = U*S*sum(Y)'/Ns; tt = 0.0; Sig_TDB = diag(S)';
for iter=1:iter_max % time integration loop
    if mod(iter,100)==0; disp(iter); end

    %% TDB-CUR: compute row and column indices
    s = gpode(Y, r);     % column indices
    p = gpode(U, r+m); % rows indices
    [~,pa] = find(D2(p,:)); pa = unique(pa);  % adjacent points

    %% TDB-CUR: compute rank-r approximation of G
    u_s = U*S*Y(s,:)'; u_p = U(p,:)*S*Y'; u_pa = U(pa,:)*S*Y';
    [F1_s, F1_p, F1_pa] = f_ss(t, u_s, u_pa, s, p, pa, D1, D2);
    [F2_s, F2_p, F2_pa] = f_ss(t+0.5*dt, u_s+0.5*dt*F1_s, u_pa+0.5*dt*F1_pa, s, p, pa, D1, D2);
    [F3_s, F3_p, F3_pa] = f_ss(t+0.5*dt, u_s+0.5*dt*F2_s, u_pa+0.5*dt*F2_pa, s, p, pa, D1, D2);
    [F4_s, F4_p,   ~   ] = f_ss(t+dt, u_s+dt*F3_s, u_pa+dt*F3_pa, s, p, pa, D1, D2);
    G_s = u_s + dt*(F1_s+2.0*F2_s+2.0*F3_s+F4_s)/6;
    G_p = u_p + dt*(F1_p+2.0*F2_p+2.0*F3_p+F4_p)/6;
    [U,~] = qr(G_s,0);
    Y = (inv(U(p,:)'*U(p,:))*U(p,:)'*G_p)';
    [Y,S,RU] = svd(Y,0);
    U = U*RU;

    t = iter*dt;
    if mod(iter, save_iter)==0
        tt = [tt t]; umean = [umean, U*S*sum(Y)'/Ns]; Sig_TDB = [Sig_TDB; diag(S)'];
    end
end

subplot(1,2,1), surf(x,tt,umean'), shading interp, view([0 0 1]), hold on; % mean solution (x,t)
subplot(1,2,2), semilogy(tt,Sig_TDB,'LineWidth',2,'Color','k')             % singular values vs time
```

```
function dudt = f(t, u, D1, D2)
global nu
dudt = -0.5*D1*u.^2 + nu*D2*u; % Burgers equation
end

function out = gdot(t, xi)
global d
out = -2*pi*cos(2*pi*t) + (pi*cos(pi*(1:d).*t)./(1:d))*xi';
end

%% TDB-CUR: compute Burgers rhs (f) at specified rows and columns
function [F_s, F_p, F_pa]=f_ss(t, u_s, u_pa, s, p, pa, D1, D2)
global xi N
F_s = f(t, u_s, D1, D2);
F_s(1,:) = gdot(t,xi(s,:)); F_s(end,:) = 0;   % apply boudnary conditions
F_p = f(t, u_pa, D1(p,pa), D2(p,pa));
if ismember(1, p);  F_p(1,:) = gdot(t,xi); end % check left boundary  (x=0)
if ismember(N, p); F_p(end,:) = 0; end         % check right boundary (x=1)
[U_F,~] = qr(F_s,0);
Z_F = inv(U_F(p,:)'*U_F(p,:))*U_F(p,:)'*F_p;
F_pa = U_F(pa,:)*Z_F;
end
```

Listing C.2: GappyPOD+E algorithm adapted from [75]

```
%% TDB-CUR: GappyPOD+E Algorithm
function [ p ] = gpode( U, np )
[~,~,p] = qr(U', 'vector'); % QDEIM (or DEIM)
p = p(1:size(U,2));         % take points equal to number of basis
for i=length(p)+1:np
    [~, S, W] = svd(U(p, :), 0);
    g = S(end-1, end-1)^2 - S(end, end)^2;
    Ub = W'*U';
    r = g + sum(Ub.^2, 1);
    r = r-sqrt((g+sum(Ub.^2,1)).^2-4*g*Ub(end, :).^2);
    [~, I] = sort(r, 'descend');
    e = 1;
    while any(I(e) == p)
        e = e + 1;
    end
    p(end + 1) = I(e);
end
p = sort(p);
end
```

Bibliography

[1] Applied mathematics research for exascale computing. *U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research Program*, 2014.

[2] L. Alvergue, H. Babaee, G. Gu, and S. Acharya. Feedback stabilization of a reduced-order model of a jet in crossflow. *AIAA Journal*, 53(9):2472–2481, 2016/02/17 2015.

[3] David Anderson, Simon Du, Michael Mahoney, Christopher Melgaard, Kunming Wu, and Ming Gu. Spectral gap error bounds for improving CUR matrix decomposition and the Nyström method. In *Artificial Intelligence and Statistics*, pages 19–27. PMLR, 2015.

[4] P. Arrué, N. Toosizadeh, H. Babaee, and K. Laksari. Low-rank representation of head impact kinematics: A data-driven emulator. *Frontiers in Bioengineering and Biotechnology*, 8:1049, 2020.

[5] Patricia Astrid, Siep Weiland, Karen Willcox, and Ton Backx. Missing point estimation in models described by proper orthogonal decomposition. *IEEE Transactions on Automatic Control*, 53(10):2237–2251, 2008.

[6] N. Aubry. On the hidden beauty of the proper orthogonal decomposition. 2(5-6):339–352, 1991.

[7] H. Babaee. An observation-driven time-dependent basis for a reduced description of transient stochastic systems. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 475(2231):20190506, 2019.

[8] H. Babaee, S. Acharya, and X. Wan. Optimization of forcing parameters of film cooling effectiveness. *Journal of Turbomachinery*, 136(6):061016–061016, 11 2013.

[9] H. Babaee, M. Choi, T. P. Sapsis, and G. E. Karniadakis. A robust bi-orthogonal/dynamically-orthogonal method using the covariance pseudo-inverse with application to stochastic flow problems. *Journal of Computational Physics*, 344:303–319, 9 2017.

[10] H. Babaee, M. Farazmand, G. Haller, and T. P. Sapsis. Reduced-order description of transient instabilities and computation of finite-time Lyapunov exponents. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(6):063103, 06 2017.

[11] H. Babaee and T. P. Sapsis. A minimization principle for the description of modes associated with finite-time instabilities. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 472(2186):20150779, 2016.

[12] H. Babaee, X. Wan, and S. Acharya. Effect of uncertainty in blowing ratio on film cooling effectiveness. *Journal of Heat Transfer*, 136(3):031701–031701, 11 2013.

[13] Maxime Barrault, Yvon Maday, Ngoc Cuong Nguyen, and Anthony T Patera. An 'empirical interpolation'method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 339(9):667–672, 2004.

[14] Andrea Barth, Christoph Schwab, and Nathaniel Zollinger. Multi-level Monte Carlo finite element method for elliptic PDEs with stochastic coefficients. *Numerische Mathematik*, 119:123–161, 2011.

[15] Z. Battles and L. Trefethen. An extension of matlab to continuous functions and operators. *SIAM Journal on Scientific Computing*, 25(5):1743–1770, 2004.

[16] M. H. Beck, A. Jäckle, G. A. Worth, and H. D. Meyer. The multiconfiguration time-dependent Hartree (MCTDH) method: a highly efficient algorithm for propagating wavepackets. *Physics Reports*, 324(1):1–105, 1 2000.

[17] M. Beneitez, Y. Duguet, P. Schlatter, and D. S. Henningson. Edge manifold as a lagrangian coherent structure in a high-dimensional state space. *Physical Review Research*, 2(3):033258–, 08 2020.

[18] Peter Benner, Serkan Gugercin, and Karen Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM review*, 57(4):483–531, 2015.

[19] Gal Berkooz, Philip Holmes, and John L Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, 25(1):539–575, 1993.

[20] T.R. Bewley, P. Moin, and R. Temam. DNS-based predictive control of turbulence: an optimal benchmark for feedback algorithms. 447:179–225, 2001.

[21] A. Blanchard, S. Mowlavi, and T. Sapsis. Control of linear instabilities by dynamically consistent order reduction on optimally time-dependent modes. *Nonlinear Dynamics*, In press, 2018.

[22] K. Braman, To. A. Oliver, and V. Raman. Adjoint-based sensitivity analysis of flames. *Combustion Theory and Modelling*, 19(1):29–56, 01 2015.

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

[24] Gianluca Ceruti and Christian Lubich. An unconventional robust integrator for dynamical low-rank approximation. *BIT Numerical Mathematics*, pages 1–22, 2021.

[25] Saifon Chaturantabut and Danny C Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 32(5):2737–2764, 2010.

[26] M. Cheng, T. Y. Hou, and Z. Zhang. A dynamically bi-orthogonal method for time-dependent stochastic partial differential equations i: Derivation and algorithms. *Journal of Computational Physics*, 242(0):843 – 868, 2013.

[27] M. Choi, T. P. Sapsis, and G. E. Karniadakis. On the equivalence of dynamically orthogonal and bi-orthogonal methods: Theory and numerical simulations. *Journal of Computational Physics*, 270:1 – 20, 2014.

[28] A. J. Chorin, O. H. Hald, and R. Kupferman. Optimal prediction with memory. *Physica D: Nonlinear Phenomena*, 166(3–4):239 – 257, 2002.

[29] DT Crommelin and AJ Majda. Strategies for model reduction: Comparing different optimal bases. *Journal of the Atmospheric Sciences*, 61(17):2206–2217, 2004.

[30] K. T. Doetsch and K. J. Fidkowski. Combined entropy and output-based adjoint approach for mesh refinement and error estimation. *AIAA Journal*, 57(8):3213–3230, 2020/10/26 2019.

[31] M. Donello, M. H. Carpenter, and H. Babaee. Computing sensitivities in evolutionary systems: A real-time reduced order modeling strategy. *SIAM Journal on Scientific Computing*, pages A128–A149, 2022/01/19 2022.

[32] Z. Drmač and S. Gugercin. A new selection operator for the discrete empirical interpolation method—improved a priori error bound and extensions. *SIAM Journal on Scientific Computing*, 38(2):A631–A648, 2016.

[33] G. Esposito and H. K. Chelliah. Skeletal reaction models based on principal component analysis: Application to ethylene–air ignition, propagation, and extinction phenomena. *Combustion and Flame*, 158(3):477–489, 2011.

[34] Richard Everson and Lawrence Sirovich. Karhunen–loeve procedure for gappy data. *JOSA A*, 12(8):1657–1664, 1995.

[35] M. Farazmand and T. P. Sapsis. Dynamical indicators for the prediction of bursting phenomena in high-dimensional systems. *Phys. Rev. E*, 94:032212, Sep 2016.

[36] AP Gallagher and A McD Mercer. On the behaviour of small disturbances in plane couette flow. *Journal of Fluid Mechanics*, 13(1):91–100, 1962.

[37] A. Garai and S. M. Murman. Stabilization of the adjoint for turbulent flows. *AIAA Journal*, pages 1–13, 2021/03/14 2021.

[38] Michael B Giles. Multilevel Monte Carlo path simulation. *Operations research*, 56(3):607–617, 2008.

[39] Michael B. Giles and Niles A. Pierce. An introduction to the adjoint approach to design. *Flow, Turbulence and Combustion*, 65(3):393–415, 2000.

[40] Francesco Gomez, Raquel Gómez, and Vassilis Theofilis. On three-dimensional global linear instability analysis of flows with standard aerodynamics codes. *Aerospace Science and Technology*, 32(1):223–234, 2014.

[41] Chester E Grosch and Harold Salwen. The stability of steady and time-dependent plane poiseuille flow. *Journal of Fluid Mechanics*, 34(1):177–205, 1968.

[42] M.D. Gunzburger. Sensitivities, adjoints and flow optimization. *International Journal for Numerical Methods in Fluids*, 31(1):53–78, 1999.

[43] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[44] G. E. Karniadakis and S. J. Sherwin. *Spectral/hp element methods for computational fluid dynamics*. Oxford University Press, USA, 2005.

[45] Aly-Khan. Kassam and Lloyd N. Trefethen. Fourth-order time-stepping for stiff pdes. *SIAM Journal on Scientific Computing*, 26(4):1214–1233, 2019/12/23 2005.

[46] RR Kerswell. Nonlinear nonmodal stability theory. *Annual Review of Fluid Mechanics*, 50:319–345, 2018.

[47] Emil Kieri and Bart Vandereycken. Projection methods for dynamical low-rank approximation of high-dimensional problems. *Computational Methods in Applied Mathematics*, 19(1):73–92, 2019.

[48] Dana A Knoll and David E Keyes. Jacobian-free newton–krylov methods: a survey of approaches and applications. *Journal of Computational Physics*, 193(2):357–397, 2004.

[49] O. Koch and C. Lubich. Dynamical low-rank approximation. *SIAM Journal on Matrix Analysis and Applications*, 29(2):434–454, 2017/04/02 2007.

[50] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.

[51] Frances Y Kuo, Christoph Schwab, and Ian H Sloan. Quasi-Monte Carlo finite element methods for a class of elliptic partial differential equations with random coefficients. *SIAM Journal on Numerical Analysis*, 50(6):3351–3374, 2012.

[52] Jonas Kusch, Gianluca Ceruti, Lukas Einkemmer, and Martin Frank. Dynamical low-rank approximation for burgers'equation with uncertainty. *International Journal for Uncertainty Quantification*, 12(5), 2022.

[53] J Nathan Kutz, Steven L Brunton, Bingni W Brunton, and Joshua L Proctor. *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM, 2016.

[54] Eyink G. L., T. W. N. Haine, and D. J. Lea. Ruelle's linear response formula, ensemble adjoint schemes and lévy flights. *Nonlinearity*, 17(5):1867, 2004.

[55] K. Laksari, M. Kurt, H. Babaee, S. Kleiven, and D. Camarillo. Mechanistic insights into human brain impact dynamics through modal analysis. *Physical Review Letters*, 120(13):138101–, 03 2018.

[56] R. Langer, J. Lotz, L. Cai, F. vom Lehn, K. Leppkes, U. Naumann, and H. Pitsch. Adjoint sensitivity analysis of kinetic, thermochemical, and transport data of nitrogen and ammonia chemistry. *Proceedings of the Combustion Institute*, 2020.

[57] M. Lemke, L. Cai, J. Reiss, H. Pitsch, and J. Sesterhenn. Adjoint-based sensitivity analysis of quantities of interest of complex combustion models. *Combustion Theory and Modelling*, 23(1):180–196, 01 2019.

[58] Z. Li, A. Yazdani, A. Tartakovsky, and G. E. Karniadakis. Transport dissipative particle dynamics model for mesoscopic advection-diffusion-reaction problems. *The Journal of Chemical Physics*, 143(1):014101, 2020/07/13 2015.

[59] Yinmin Liu, Hessam Babaee, Peyman Givi, Harsha Chelliah, Daniel Livescu, and Arash Nouri. Skeletal reaction model generation for atmospheric and high pressure combustion of methane. *arXiv preprint arXiv:2201.11038*, 2022.

[60] C. Lubich and I. V. Oseledets. A projector-splitting integrator for dynamical low-rank approximation. *BIT Numerical Mathematics*, 54(1):171–188, 2014.

[61] Paolo Luchini and Alessandro Bottaro. Adjoint equations in stability analysis. *Annual Review of fluid mechanics*, 46:493–517, 2014.

[62] Christoph J Mack and Peter J Schmid. A preconditioned krylov technique for global hydrodynamic stability analysis of large-scale compressible flows. *Journal of Computational Physics*, 229(3):541–560, 2010.

[63] Michael W Mahoney and Petros Drineas. CUR matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106(3):697–702, 2009.

[64] K. Manohar, B. W. Brunton, J. N. Kutz, and S. L. Brunton. Data-driven sparse sensor placement for reconstruction: Demonstrating the benefits of exploiting known patterns. *IEEE Control Systems Magazine*, 38(3):63–86, 2018.

[65] E. Musharbash and F. Nobile. Dual dynamically orthogonal approximation of incompressible Navier Stokes equations with random boundary conditions. *Journal of Computational Physics*, 354:135–162, 2018.

[66] E. Musharbash, F. Nobile, and T. Zhou. Error analysis of the dynamically orthogonal approximation of time dependent random pdes. *SIAM Journal on Scientific Computing*, 37(2):A776–A810, 2018/01/17 2015.

[67] M. H. Naderi and H. Babaee. Adaptive sparse interpolation for accelerating nonlinear stochastic reduced-order modeling with time-dependent bases. *Computer Methods in Applied Mechanics and Engineering*, 405:115813, 2023.

[68] Achim Nonnenmacher and Christian Lubich. Dynamical low-rank approximation: applications and numerical experiments. *Mathematics and Computers in Simulation*, 79(4):1346–1357, 2008.

[69] A. G. Nouri, H. Babaee, P. Givi, H. K. Chelliah, and D. Livescu. Skeletal model reduction with forced optimally time dependent modes. *Combustion and Flame*, page 111684, 2021.

[70] A.G. Nouri, H. Babaee, Givi P., Chelliah H.K., and Livescu D. Skeletal model reduction with forced optimally time dependent modes. *Combustion and Flame*, (111684), 2021.

[71] Alberto Padovan, Samuel E Otto, and Clarence W Rowley. Analysis of amplification mechanisms and cross-frequency interactions in nonlinear flows via the harmonic resolvent. *Journal of Fluid Mechanics*, 900, 2020.

[72] P. Patil and H. Babaee. Real-time reduced-order modeling of stochastic partial differential equations via time-dependent subspaces. *Journal of Computational Physics*, 415:109511, 2020.

[73] P. Patil and H. Babaee. Reduced order modeling with time-dependent bases for PDEs with stochastic boundary conditions, 2021.

[74] Prerna Patil and Hessam Babaee. Real-time reduced-order modeling of stochastic partial differential equations via time-dependent subspaces. *Journal of Computational Physics*, 415:109511, 2020.

[75] Benjamin Peherstorfer, Zlatko Drmac, and Serkan Gugercin. Stability of discrete empirical interpolation and gappy proper orthogonal decomposition with randomized and deterministic sampling points. *SIAM Journal on Scientific Computing*, 42(5):A2837–A2864, 2020.

[76] D. Ramezanian, A. G. Nouri, and H. Babaee. On-the-fly reduced order modeling of passive and reactive species via time-dependent manifolds. *Computer Methods in Applied Mechanics and Engineering*, 382:113882, 2021.

[77] Rajesh Ranjan, S Unnikrishnan, and Datta Gaitonde. A robust approach for stability analysis of complex flows using high-order navier-stokes solvers. *Journal of Computational Physics*, 403:109076, 2020.

[78] Abram Rodgers, Alec Dektor, and Daniele Venturi. Adaptive integration of nonlinear evolution equations on tensor manifolds. *Journal of Scientific Computing*, 92(2):39, 2022.

[79] D. Ruelle. Differentiation of SRB states. *Communications in Mathematical Physics*, 187(1):227–241, 1997.

[80] Harold Salwen, Fredrick W Cotton, and Chester E Grosch. Linear stability of poiseuille flow in a circular pipe. *Journal of Fluid Mechanics*, 98(2):273–284, 1980.

[81] T.P. Sapsis and P.F.J. Lermusiaux. Dynamically orthogonal field equations for continuous stochastic dynamical systems. *Physica D: Nonlinear Phenomena*, 238(23-24):2347–2360, 2009.

[82] Peter J Schmid. Nonmodal stability theory. *Annu. Rev. Fluid Mech.*, 39:129–162, 2007.

[83] P.J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, 2010.

[84] Jeffrey P Slotnick, Abdollah Khodadoust, Juan Alonso, David Darmofal, William Gropp, Elizabeth Lurie, and Dimitri J Mavriplis. CFD vision 2030 study: A path to revolutionary computational aerosciences. Technical report, 2014.

[85] Danny C Sorensen and Mark Embree. A DEIM induced CUR factorization. *SIAM Journal on Scientific Computing*, 38(3):A1454–A1482, 2016.

[86] D. B. Szyld. The many proofs of an identity on the norm of oblique projections. *Numerical Algorithms*, 42(3):309–323, 2006.

[87] Rene Vidal, Yi Ma, and Shankar Sastry. Generalized principal component analysis (GPCA). *IEEE transactions on pattern analysis and machine intelligence*, 27(12):1945–1959, 2005.

[88] Q. Wang, R. Hu, and P. Blonigan. Least squares shadowing sensitivity analysis of chaotic limit cycle oscillations. *Journal of Computational Physics*, 267:210–224, 2014.

[89] D. Xiu and J.S. Hesthaven. High-order collocation methods for differential equations with random inputs. *SIAM Journal on Scientific Computing*, 27(3):1118, 2006.