Data Compression, Uncertainty Quantification, and Prediction Using Low-Rank Approximation

by

Shaghayegh Zamani Ashtiani

M.S. Aerospace Engineering, Sharif University of Technology, 2017

B.S. Aerospace Engineering, Azad University, 2015

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

2024

UNIVERSITY OF PITTSBURGH

SWANSON SCHOOL OF ENGINEERING

This dissertation was presented

by

Shaghayegh Zamani Ashtiani

It was defended on

March 22nd 2024

and approved by

Dr. Anne M. Robertson, Distinguished Professor, William Kepler Whiteford Professor of Engineering

Dr. Masoud Barati, Assistant Professor, Department of Electrical and Computer Engineering

Dr. Kaveh Laksari, Assistant Professor, Department of Mechanical Engineering, University of California Riverside

Dissertation Director: Dr. Hessam Babaee, Associate Professor, Department of Mechanical Engineering and Materials Sciences

Data Compression, Uncertainty Quantification, and Prediction Using Low-Rank Approximation

Shaghayegh Zamani Ashtiani, PhD

University of Pittsburgh, 2024

Dimension reduction techniques are valuable for both data-rich and data-poor problems. For applications involving massive high-dimensional data, dimension reduction can be utilized for data compression and data-driven discovery. In the data-poor regime, low-rank subspaces enable field reconstruction with only a few sparse measurements. Moreover, reduced-order modeling effectively propagates parametric uncertainty in high-dimensional partial differential equations. This work aims to develop dimension reduction techniques based on spatiotemporal subspaces for applications across the data-availability spectrum as well as performing uncertainty quantification in high-dimensional dynamical systems.

First, we develop a low-rank approximation that compresses the size of transient simulation data in real time, which helps with storage and input/output limitations. These limitations also restrict data analysis and visualization in large-scale simulations. To address this issue, we present an in-situ dimension reduction technique that decomposes the streaming data into a set of time-dependent bases and a core tensor in real time. This method is adaptive and controls the compression error through the addition or removal of modes.

We then develop dimension-reduction methodologies for prediction in data-poor regimes. While it is possible to predict blood flow using machine learning models, clinical measurements, such as Transcranial Doppler ultrasound, may be insufficient or too low-resolution for the training process. Therefore, developing a computational model that provides predictions based on sparse data is crucial. To this end, we present a physics-informed regression framework based on Gaussian process regression to predict blood flow properties using very few sparse measurements.

Lastly, we extend the application of low-rank approximation to uncertainty quantifica-

tion in blood flow simulations. In clinical settings, measurements are often intrusive and inherently uncertain. Numerical simulations could aid in developing non-invasive assessments. However, physiological variability introduces uncertainties into simulation parameters, necessitating a large number of computationally expensive simulations. To address these challenges, we explore implementing a low-rank approximation approach that reduces computational costs while maintaining accuracy.

Table of Contents

List of Tables

List of Figures

Preface

I am deeply grateful to my parents for their unwavering support throughout my higher education journey; their encouragement was indispensable. Similarly, I owe a great deal to my advisor, Professor Hessam Babaee, whose insights into the innovative use of linear algebra in computational fluid dynamics significantly broadened my academic perspective. I also extend my thanks to my committee members, Professor Robertson, Professor Barati, and Professor Laksari, for their invaluable feedback and guidance. Additionally, I am thankful for Professor Yurko's feedback and Professor Givi's advice during my PhD studies. Furthermore, I am blessed to have shared this experience with remarkable colleagues: Prerna, Mike, Alireza, Hossein, Grishma, Sara, Saman, and Saeed, whose camaraderie made this challenging journey enjoyable. Finally, I express my gratitude to Kevin for being a constant source of support and patience through all the ups and downs.

1.0 Introduction

There is significant potential for dimension reduction techniques in applications with a large amount of data, or in applications with very few data points. In this work, dimension reduction techniques is developed for both scenarios. In cases where massive data is available (data-rich), dimension reduction can be used for data compression, visualization, and analysis. This situation arises in exascale computations where memory is limited compared to the generated data. In problems with limited data (data-poor regimes), dimension reduction facilitates field reconstruction with just a few measurements. Predicting blood flow velocity in a network of vessels with only a few clinical measurements available is an example of such data-poor problems. Moreover, reduced-order modeling effectively propagates parametric uncertainty in high-dimensional partial differential equations. Therefore, we also extend the application of low-rank approximation to uncertainty quantification.

1.1 Compression of Transient Simulation Data

The ability to perform large-scale simulations has grown explosively in the past few decades and it will only continue to grow in the near future. On the other hand, our ability to effectively analyze or in some applications even store the data that these simulations generate is lagging behind — impeding many scientific discoveries [1, 2, 3]. For extreme-scale simulations the sheer size of the data imposes input/output (I/O) constraints that impede storing temporally-resolved simulation data [4]. An example is the direct numerical simulation (DNS) of turbulent combustion, in which the creation of an ignition kernel – an intermittent phenomenon – occurs on the order of 10 simulation time steps, while typically every 400th time step is stored to maintain the I/O overhead at a reasonable level [5]. Similarly, climate simulations use time steps of minutes but model outputs are typically

written at daily to monthly intervals, limiting the ability to understand extreme weather events and projections of their future changes. The I/O and network limitations motivate for a paradigm shift from *postprocess-centric* to *concurrent* data analysis, wherein the raw data is analysed as it is generated in an *in situ* or *in-transit* framework and given the size of the data and the demand for real-time analysis, only scalable algorithms with low computational complexity have a chance of being feasible.

Being able to store time-dependent simulation data with higher temporal resolution is critically important for many purposes. One of the most common applications is checkpointing data for simulation restarts, where the simulation may need to restart from the last snapshot [6]. This is particularly important for parallel simulations with a large number of computational nodes with long runtimes, where node failure can interrupt the simulation or queue limits may be exceeded. Frequent checkpointing lowers the cost of loss in computational resources. The other applications are for data analysis, including visualization and extracting coherent structures from the simulation.

Data compression techniques can be broadly divided to *lossy* and *lossless* classes. In the lossless compression, the compressed data has no error when compared with the decompressed data. Examples of lossless compression techniques are FPZIP [7] and ACE [8]. All of these techniques require a significant amount of memory and they are not suitable for large-scale simulations where only on-the-fly compression techniques are practical. On the other hand, lossy compression techniques can achieve significant reduction in data size by allowing controllable error in the reconstructed data. Lossy compression techniques can be further divided into methods that achieve compression by extracting and exploiting spatiotemporal correlations in the data and the methods that do not extract correlation. Examples of techniques that achieve compression by exploiting correlations in the data are methods based on low-rank matrix and tensor decompositions, for example QR decomposition [9], interpolated decomposition [10], singular value decomposition (SVD) [11, 12] or tensor decomposition based on higher order SVD (HOSVD) [13, 14]. Deep learning techniques that are based on autoencoder-decoder also achieve compression by extracting nonlinear correlations in the

data [15, 16, 17]. Examples of the lossy compression techniques that do not exploit correlated structure in the data are multiprecision truncation methods [18], mesh reduction techniques that map the data to a coarse spatiotemporal mesh. See Ref. [16] for an overview of these techniques.

The focus of this work is on correlation-exploiting compression techniques as these lossy methods can achieve significantly more compression compared with other methods. The performance of these techniques can be assessed by two criteria: (i) the compression ratio for a given reconstruction error, and (ii) computational cost of performing the compression. The compression ratio is the ratio of the storage size of the full-dimensional data to that of the compressed data. Methods that can achieve higher compression ratio for a given accuracy or achieve higher accuracy for the same compression ratio perform better on the first metric. Computational cost of performing the compression of some techniques can be exceedingly large. For example, SVD-based reductions require solving large-scale eigenvalue problems and therefore, the computational cost of SVD-based compression can be orders of magnitudes larger than advancing the simulation for one time step – although this depends on the size of the data. This has motivated the randomized SVD that requires a single-pass over data [12], thereby reducing the computational cost significantly. To reduce the computational cost, deep-learning-based compression techniques can be trained on canonical problems and then the trained network can be used in real-time for fast compression [16]. However, it is not clear whether the pretrained model will perform well on any unseen data. This issue of extrapolating to unseen data does not exist in SVD-based techniques, as SVD reduction does not require any training, however, SVD-based techniques often result in lower compression ratios as they can only exploit linear correlations in data as opposed to autoencoder-decoder techniques that extract nonlinear correlations.

Recently, new reduced-order modeling techniques have been introduced in which the low-rank structures are extracted directly from the model – bypassing the need to generate data. In these techniques, a reduced-order model (ROM) is obtained by projecting the full-order model onto a time-dependent basis (TDB). The first applications of TDB have been

in the quantum mechanics and quantum chemistry communities. See [19] for a review. We refer to these techniques as *model-driven* ROM, as opposed to *data-driven* techniques since in the model-driven formulation partial or ordinary differential equations are obtained for the evolution of TDB. Model-driven ROMs have also shown great performance in solving high-dimensional stochastic PDEs [20, 21, 22, 23, 24, 25], reduced-order modeling of time-dependent linear systems [26, 27, 28] as well as combustion [29, 30]. TDB has also been applied in tensor dimension reduction [31, 32, 33].

ROMs based on TDB have been successful in the model-driven applications, in which structures are extracted directly from the model. In this study, we present a data-driven analog of the TDB-based compression. In particular, we present an adaptive and scalable *in situ* data compression that decomposes the streaming multidimensional data using TDBs. This technique extracts multidimensional correlations from high-dimensional streaming data. The presented method is lossy and it is adaptive, i.e., modes are added and removed to maintain the error below a prescribed threshold. Moreover, the method utilizes the time derivative of the instantaneous data and as a result it only requires access to a few snapshots of the data to compute the instantaneous time-derivative as opposed to the entire temporal history of the data. The compression is achieved without having to solve large-scale eigenvalue or nonconvex optimization problems. The cost of extracting the data-driven TDB grows linearly with the data size, making it suitable for large-scale simulations where only on-the-fly compression techniques are feasible. We also present our formulation in a versatile form, where the multidimensional data can be decomposed into arbitrarily chosen lower-dimensional bases. In this new formulation, we recover the dynamically bi-orthonormal (DBO) decomposition [25], and equivalently, the dynamically orthogonal (DO) and bi-orthogonal (BO) [20, 21] decompositions as special cases.

1.2 Blood Flow Prediction in Data-Poor Regimes

Accurate blood flow reconstruction in the vasculature is vital for many clinical applications. For instance, precise blood flow analysis in cerebral vessels helps in diagnosing and monitoring conditions such as cerebral vasospasm, Moyamoya disease, and brain tumors. Accurate flow reconstruction in the aorta aids in evaluating aortic aneurysms, dissections, and coarctations, enabling timely interventions. Additionally, detailed blood flow assessment is crucial for diagnosing and managing conditions like coronary artery disease, heart failure, and peripheral artery disease [34]. These applications underscore the importance of accurate blood flow reconstruction, highlighting its extensive impact across various domains of cardiovascular medicine.

Although advanced computational methods, such as image-based models, can provide detailed predictions of spatiotemporal events, integrating them into clinical practice is still challenging. This limitation primarily arises from two key factors: modeling uncertainties, including the lack of knowledge of the subject-specific boundary conditions, and the substantial computational time required for their execution. In these models, computational fluid dynamics (CFD) techniques are implemented on the extracted geometry from imaging data such as computed tomography (CT) or magnetic resonance imaging (MRI) [35, 36]. There exists a large body of literature on this type of blood flow modeling (for example see [37, 38, 39, 40]) as well as one-dimensional (1D) CFD [41, 42, 43, 44]. Among image-based models Ref. [45] is the first FDA-approved CFD simulation for coronary stenosis assessment.

Building regression models for blood flow properties in the vasculature is challenging due to insufficient clinical data. This is exemplified by techniques such as Doppler ultrasonography, which, although widely used, often suffer from poor spatial resolution, and generally does not provide measurements for every vessel. To better illustrate the challenges of constructing regression models for blood flow, consider a naive linear regression model in the form of $u(x,t) = \sum_{i=1}^{n_x} \sum_{j=1}^{n_t} w_{i,j} \psi_i(x) \chi_j(t)$, where $\psi_i(x)$ and $\chi_j(t)$ represent 1D spatial and temporal basis functions, respectively. With $n_x, n_t \sim \mathcal{O}(50-100)$ basis functions required in

each vessel, training such models necessitates over $\mathcal{O}(10^3 - 10^4)$ measurements in each vessel to compute the weight coefficients ($w_{i,j}$), far exceeding what is typically available in clinical settings.

The underlying issue with linear regression models is that the basis functions are fixed and chosen before analyzing the data. Deep neural networks (DNNs) can be interpreted as *adaptive* basis function approximations, where the basis functions are learned from the data. For example, a feed-forward neural network regression model with x and t as input units can be written as $u(x,t) = \sum_{i=1}^{r} w_i \phi_i(x,t;\theta)$, where w_i are the weights of the last layer, θ represents the vector of neural network parameters, and $\phi_i(x,t;\theta)$ can be viewed as adaptive basis functions that are learned from the data, with $r \ll n_x n_t$. Consequently, the total number of parameters in a DNN that need to be inferred is significantly smaller than in a naive linear regression model, which, to some extent, relaxes the amount of data required to train such models. However, the available data is still grossly insufficient for training such models. For example, the DNN cannot be trained for vessels for which no measurements are available, and for vessels with few measurements, this can result in overfit models.

To address the issue of training DNN in data-poor regimes, physics-informed neural networks (PINNs) have been utilized [46], in which the fluid dynamics conservation laws alongside clinical measurements [47, 48, 49, 50] are used to train the DNN. Training PINNs for 1D blood flow models in the vasculature requires enforcing the conservation of mass and momentum in each vessel, as well as enforcing these constraints at vessel junctions. Specifically, a DNN is considered for each vessel. As a result, PINNs can overcome the issue of data scarcity by enforcing physical laws as regularizers to avoid model overfitting and estimate blood properties in vessels with no measurements.

The computational cost of training PINNs, especially for large vasculature networks, can be significant. For instance, training the PINNs to model brain hemodynamics requires approximately 40 hours using a single NVIDIA Tesla T4 GPU card, starting from a random network initialization. In clinical settings, particularly for stroke, there is an urgent need for near-real-time models capable of rapidly processing and analyzing complex medical data.

This enables healthcare professionals to make swift, informed decisions about patient care, leading to improved outcomes and reduced mortality rates. Accurate and timely modeling can help identify stroke subtypes, predict treatment responses, and optimize personalized treatment strategies, ultimately enhancing patient recovery and quality of life [51]. Another challenge in training PINNs for blood flow is the lack of direct area measurements for vessel cross-sectional area or pressure, which, as shown in [48], necessitates first training another DNN to relate velocity to pressure before training the PINN, which further adds to the training cost.

Bayesian regression techniques offer some favorable features that can be very important when building regression models in data-poor regimes. Bayesian regression models provide posterior probability density distributions for quantities of interest as opposed to deterministic models that yield point estimates. The probabilistic predictions encode the epistemic uncertainties due to insufficient data. These uncertainty estimates can also be used for targeted data acquisition. Gaussian processes (GPs) are powerful Bayesian regression models that have an analytical workflow and as a result, they can be trained quickly, especially in data-poor regimes, where the size of training data is small [52]. Multi-fidelity models based on GPs have also proven useful for building regression models in data-poor regimes [53, 54, 55, 56, 57, 58, 59, 60]. The multi-fidelity models enable the fusion of a small number of high-fidelity data points, for example, clinical measurements, with a large number of low-fidelity measurements for example obtained from CFD simulations. Since the developments of this work are related to GPs, we highlight two major challenges below in applying standard GP for blood flow regression in data-poor regimes:

1. **Choice of kernel:** The choice of the kernel can significantly impact the accuracy of GP predictions, especially in data-poor regimes. Typically square exponential kernel or Matern kernel are used for GP. However, training the GP using these kernels requires a significant amount of data. Determining a suitable kernel for blood flow prediction is not clear. Also, finding kernel hyperparameters requires solving a non-convex optimization problem. However, when the training size is very low, solving the optimization problem

leads to a poor choice of hyperparameters and it can result in overfitting.

2. **GP for a vasculature network:** Although GPs can be used to learn functions in a Euclidean input space (x), i.e., input spaces that are subsets of $\mathbb{R}^d$, there is no standard procedure for applying GPs to a vasculature network, which can be mathematically represented as a graph. The underlying difficulty arises because the notion of distance between two points in Euclidean space should not be directly applied to the network domain. For example, two vessels that are very close to each other in space can have completely different velocities. A brute-force application of GP to vasculature would require one GP per vessel, with the input space for each vessel being Euclidean. This approach would ignore the vessel-to-vessel correlations. In such cases, it is not possible to train a GP for vessels without data. Additionally, this approach can result in overfitting for vessels with very few measurements. Moreover, it is not clear how the conservation of mass and momentum can be enforced at the junctions. Another potential solution is to embed the 1D vasculature network in a three-dimensional (3D) space. However, this approach has its own challenges. Vessels that are physically close to each other may exhibit completely different flow properties. This makes it difficult to find a kernel capable of learning smooth flow behavior along a vessel, where x varies, but also of learning nearby vessels with distinctly different flows.

Solutions have been proposed to address the aforementioned challenges. Related to the choice of kernels different techniques have been proposed in which the kernel is learned from data; see for example [61, 62, 63, 64, 65], in which the kernel is parameterized as a neural network. In [66], kernel flow was introduced in which kernels are learned from data. Multi-fidelity GP is also employed [67, 68, 69] for problems that do not have enough high-fidelity data. Multi-fidelity GP compensates for the lack of high-fidelity data by incorporating low-fidelity data. There is a lack of studies utilizing a global GP for a vasculature network. However, GP has been applied to vasculature-related problems such as uncertainty quantification of 3D in-stent restenosis [70], uncertainty quantification in a 1D fluid-dynamics model of the pulmonary circulation [71], and predicting abdominal aortic aneurysm growth [72].

In this work, we introduce a methodology that utilizes Gaussian Process regression to reconstruct blood flow within a vasculature network, relying on a relatively small number of measurements. Specifically, we develop a spatiotemporal kernel that encompasses the entire vasculature, encoding the correlations both within and between vessels. The offline computational cost of constructing the kernel involves performing $\mathcal{O}(100)$ 1D simulations, which can be run concurrently and take only minutes. The online cost of performing inference is negligible. We demonstrate that the presented methodology can reconstruct blood flow velocity in a vasculature network consisting of $\mathcal{O}(10)$ vessels with only time series measurements at one or two points.

1.3 Reduced Order Modeling of Stochastic Blood Flow Simulations

Cerebral Blood Flow (CBF) plays a crucial role in evaluating brain health and functionality. Its significance extends to diagnosing and understanding a wide spectrum of diseases [73], including Parkinson [74], Alzheimer [75], and cerebrovascular conditions [76, 77, 78], among others. CBF can be quantified using various imaging techniques, notably Transcranial Doppler (TCD) ultrasonography [79, 80] and Magnetic Resonance Imaging (MRI), including 3D Phase Contrast MRI (3D PC-MRI) [81, 82], each offering unique insights into brain hemodynamics (see Ref. [73] for more). However, these techniques have uncertainties in measurements; for example, one limitation of transcranial Doppler ultrasonography is its inability to evaluate vessels located higher in the brain, and it only measures the peak velocity within a vessel [83]. 3D PC-MRI image quality can be adversely affected in high-resolution acquisitions due to a decline in the signal-to-noise ratio as resolution increases which lead to more frequent segmentation errors, uncertainty in determining the direction of blood flow, and obstacles in accurately quantifying flow [84, 85]. In this scenario, Computational Fluid Dynamics (CFD) models could offer insights into cerebral hemodynamics by simulating blood flow and accounting for uncertainties in simulation parameters.

Similarly, the assessment of coronary artery health is vital in detecting and managing coronary artery disease (CAD). The most reliable method for determining the functional impact of CAD is the invasive fractional flow reserve (FFR), a measure obtained by comparing the pressure after a coronary blockage to the aortic pressure during maximal blood flow, achieved by administering adenosine [86]. This approach, supported by extensive clinical trials, helps distinguish between patients who need medical treatment and those who require stent revascularization [87]. Despite being the benchmark for identifying ischemia-causing lesions, FFR's invasiveness and its negative results in about half of the tested patients limit its application [88]. In similar cases, numerical simulations could aid in the development of non-invasive assessments.

Integrating advanced computational methods, such as image-based and one-dimensional (1D) blood flow models, into clinical practice presents both opportunities and challenges. These computational models, including those utilizing CFD techniques, offer detailed spatiotemporal event predictions by leveraging imaging data from computed tomography (CT) or MRI (for example see [37, 38, 39]). Image-based CFD models, in particular, have a rich literature base and have seen applications in areas like coronary stenosis assessment, with notable examples like the FDA-approved simulation referenced in [45]. 1D blood flow models [41, 42, 43, 44] have been gaining momentum due to their fast simulation, simple implementation, and ability to accurately capture the most significant features of pulse wave propagation in the systemic arterial system. Deriving such models is achieved by simplifying the full Navier-Stokes (NS) system under the assumptions of axial symmetry, negligible longitudinal displacements, absence of external body forces, and dominance of the axial velocity component. Because of their rapid simulation capabilities, ease of implementation, and ability to capture essential flow dynamics, one-dimensional simulations provide a robust tool for estimating flow properties and studying problems that cannot be addressed *in vivo*.

There is a body of research employing CFD to explore uncertainty quantification (UQ) in various areas such as cerebral circulation [89, 90], coronary arteries [91], and arterial networks [92]. Despite the insights it offers, the extensive time and computational demands of UQ

render its use in time-sensitive clinical environments and large-scale simulations impractical. Implementation of low-rank approximation on UQ problems is a potential solution to such problems and is investigated in literatures [93, 94, 95, 96]. Low-rank approximations utilizing time-dependent bases (TDBs) offer a promising way to decrease the computational demands of solving stochastic problems [97, 98]. This method involves identifying low-rank structures within the matrices using TDBs and then constructing a reduced-order model (ROM) by projecting the original, full-order model (FOM) onto these bases. However, these models suffer from stability due to extremely small singular values from the low-rank approximation. Retaining these small singular values is crucial for achieving accurate approximations, yet it leads to the creation of ill-conditioned matrices needing inversion in various forms of TDB ROM evolution equations [97, 21, 99, 100]. Recent development in TDB ROM suggest using CUR factorization (TDB-CUR) for low-rank matrices to avoid numerical instability [101]. In the foundation of TDB-CUR, techniques like interpolation and hyper-reduction have been developed to enhance the speed of nonlinear reduced-order models (ROMs) and finite-element models in vector differential equations. Moreover, this approach is easily integrable into existing frameworks and maintains stable time-integration.

We explore the application of the TDB-CUR method to the UQ in vascular networks, focusing on complex systems such as Circle of Willis (CoW). In such network, physiological variability introduces uncertainties, particularly in boundary conditions, leading to simulations with high-dimensional parameter spaces. To address this challenge, we apply the TDB-CUR method to reduce the dimensionality of these complex problems, ensuring the preservation of critical flow characteristics. The vascular network solver, onto which we implement the TDB-CUR method, employs a spectral/hp element approach as outlined by Ref. [102]. This implementation aims to efficiently handle the inherent uncertainties while capturing the essential dynamics of blood flow within these intricate vascular systems.

2.0 Compression of Transient Simulation Data

Large-scale simulations of time-dependent problems generate a massive amount of data and with the explosive increase in computational resources the size of the data generated by these simulations has increased significantly. This has imposed severe limitations on the amount of data that can be stored and has elevated the issue of input/output (I/O) into one of the major bottlenecks of high-performance computing. In this work, we present an *in situ* compression technique to reduce the size of the data storage by orders of magnitude. This methodology is based on time-dependent subspaces and it extracts low-rank structures from multidimensional streaming data by decomposing the data into a set of time-dependent bases and a core tensor. We derive closed-form evolution equations for the core tensor as well as the time-dependent bases. The presented methodology does not require the data history and the computational cost of its extractions scales linearly with the size of data, making it suitable for large-scale streaming datasets. To control the compression error, we present an adaptive strategy to add/remove modes to maintain the reconstruction error below a given threshold. We present four demonstration cases: i) analytical example, ii) incompressible unsteady reactive flow, iii) stochastic turbulent reactive flow, and iv) three-dimensional turbulent channel flow.

2.1 Methodology

2.1.1 Notations and Definitions

We consider that streaming data are generated by a generic d–dimensional nonlinear time-dependent PDE expressed by:

$$\frac{\partial v(\mathbf{x},t)}{\partial t} = \mathcal{M}(v,\mathbf{x},t), \quad \mathbf{x} \in \Omega, \quad t > 0, \tag{2.1.1}$$

where $\mathbf{x} = \{x_1, x_2, \ldots, x_d\}$ are the d-dimensional independent variables, which include *differential* dimensions with respect to which differentiation appears in the PDE as well as *parametric/random* dimensions to which the solution of the PDE has parametric dependence. The examples of the parametric space are design space or random parametric space, where different samples of parameters can be run concurrently. In Eq. (2.1.1), $\mathcal{M}$ represents the model that in general includes linear and nonlinear differential operators augmented with appropriate boundary/initial conditions. In the most generic form, we consider a disjoint decomposition of $\mathbf{x}$ into p groups of variables: $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p\}$, with $2 \leq p \leq d$. The dimension of each space $\mathbf{x}_n$ is denoted by d_n. Therefore, $d = d_1 + d_2 + \cdots + d_p$ and the special case of $p = d$ implies decomposing the d-dimensional space to d one-dimensional spaces.

In the presented methodology, we work with data, which is represented in the discrete form with vectors, matrices and tensors. However, we present our formulation using multidimensional functions, i.e., in the continuous form, since we believe this notation is easier to understand. We show how the continuous formulation can easily be turned to a discrete formulation.

We introduce an L^2 inner-product and its induced norm for the multidimensional space as in the following

$$\Big\langle u(\mathbf{x}), v(\mathbf{x}) \Big\rangle_{\mathbf{x}} := \int_{\mathbf{x}_p} \cdots \int_{\mathbf{x}_1} u(\mathbf{x}) v(\mathbf{x}) \rho(\mathbf{x}_1) \ldots \rho(\mathbf{x}_p) d\mathbf{x}_1 \ldots d\mathbf{x}_p \quad \text{and} \quad \Big\| u \Big\|_{\mathbf{x}} := \Big\langle u, u \Big\rangle_{\mathbf{x}}^{\frac{1}{2}}, \tag{2.1.2}$$

where $\rho(\mathbf{x}_i)$ is the nonnegative density weight in each space. Similarly, an inner product for

$\mathbf{x}_n$ and its induced L^2 norm can be defined as:

$$\Big\langle u(\mathbf{x}_n), v(\mathbf{x}_n) \Big\rangle_{\mathbf{x}_n} = \int_{\mathbf{x}_n} u(\mathbf{x}_n) v(\mathbf{x}_n) \rho(\mathbf{x}_n) d\mathbf{x}_n, \quad \text{and} \quad \|u(\mathbf{x})\|_{\mathbf{x}_n} = \langle u(\mathbf{x}), u(\mathbf{x}) \rangle_{\mathbf{x}_n}^{\frac{1}{2}}. \tag{2.1.3}$$

We also introduce the following notation for the inner product with respect to all dimensions except $\mathbf{x}_j$ as follows:

$$\Big\langle u(\mathbf{x},t), v(\mathbf{x},t) \Big\rangle_{\mathbf{x} \setminus \mathbf{x}_j} = \int_{\mathbf{x}_p} \cdots \int_{\mathbf{x}_{j+1}} \int_{\mathbf{x}_{j-1}} \cdots \int_{\mathbf{x}_1} u(\mathbf{x},t) v(\mathbf{x},t) \rho(\mathbf{x}_1) \ldots \rho(\mathbf{x}_{j-1}) \rho(\mathbf{x}_{j+1}) \ldots \rho(\mathbf{x}_p) d\mathbf{x}_1 \ldots d\mathbf{x}_{j-1} d\mathbf{x}_{j+1} \ldots d\mathbf{x}_p. \tag{2.1.4}$$

We also denote a set of orthonormal TDB in space $\mathbf{x}_n$ by:

$$\mathbf{U}^{(n)}(\mathbf{x}_n, t) = \Big[u_1^{(n)}(\mathbf{x}_n, t) | u_2^{(n)}(\mathbf{x}_n, t) | \ldots | u_{r_n}^{(n)}(\mathbf{x}_n, t) \Big],$$

where the superscript (n) shows the index of the dimension group and r_n is the dimension of the subspace spanned by $\mathbf{U}^{(n)}(\mathbf{x}_n, t)$. We also use the tensor notation that was presented in the review article [103]. For a t-order tensor $\underline{\mathbf{T}} \in \mathbb{R}^{r_1 \times r_2 \times \cdots \times r_p}$, the nth or n-mode unfolding of a tensor to a matrix is denoted by: $\underline{\mathbf{T}}_{(n)}(t) \in \mathbb{R}^{r_n \times r_{n+1} \ldots r_p r_1 r_2 \ldots r_{n-1}}$. The n-mode product of $\underline{\mathbf{T}} \in \mathbb{R}^{r_1 \times r_2 \times \ldots r_p}$ and a matrix $\mathbf{A} \in \mathbb{R}^{N_n \times r_n}$ is defined as $\underline{\mathbf{T}} \times_n \mathbf{A} \in \mathbb{R}^{r_1 \times r_2 \cdots \times r_{n-1} \times N_n \times r_{n+1} \times \cdots \times r_p}$. We also denote the time derivative with $\partial(\sim)/\partial t \equiv \dot{(\sim)}$.

2.1.2 Reduction via Time-dependent Bases

We decompose the multidimensional streaming data into a time-dependent core tensor and a set of time-dependent orthonormal TDB as in the following:

$$v(\mathbf{x},t) = \sum_{i_p=1}^{r_p} \cdots \sum_{i_2=1}^{r_2} \sum_{i_1=1}^{r_1} \underline{\mathbf{T}}_{i_1 i_2 \ldots i_p}(t) u_{i_1}^{(1)}(\mathbf{x}_1, t) u_{i_2}^{(2)}(\mathbf{x}_2, t) \ldots u_{i_p}^{(p)}(\mathbf{x}_p, t) + e(\mathbf{x}, t), \tag{2.1.5}$$

where $\mathbf{U}^{(n)}(\mathbf{x}_n, t) = \Big[u_1^{(n)}(\mathbf{x}_n, t) | u_2^{(n)}(\mathbf{x}_n, t) | \ldots | u_{r_n}^{(n)}(\mathbf{x}_n, t) \Big]$ are a set of orthonormal modes, $\underline{\mathbf{T}}(t)$ is the time-dependent core tensor, and $e(\mathbf{x}, t)$ is the low-rank approximation error. In the above description, $v(\mathbf{x}, t)$ represents the streaming data, which in the discrete form is

represented by a d-dimensional time-dependent tensor. In the following, we derive closed-form evolution equations for the core tensor and the TDB. For the sake of simplicity, we derive the equations for a special case with three bases, where $p = 3$. We also present the formulation for the most generic case. The TDB is instantaneously orthonormal and therefore:

$$\left\langle u_i^{(n)}(\mathbf{x}_n, t), u_{i'}^{(n)}(\mathbf{x}_n, t) \right\rangle_{\mathbf{x}_n} = \delta_{ii'}, \qquad i, i' = 1, 2, \ldots, r_n, \qquad n = 1, 2, \ldots, p. \tag{2.1.6}$$

Taking time derivative of the orthonormality condition results in:

$$\frac{d}{dt}\left\langle u_i^{(n)}(\mathbf{x}_n, t), u_j^{(n)}(\mathbf{x}_n, t) \right\rangle_{\mathbf{x}_n} = \left\langle \dot{u}_i^{(n)}(\mathbf{x}_n, t), u_j^{(n)}(\mathbf{x}_n, t) \right\rangle_{\mathbf{x}_n} + \left\langle u_i^{(n)}(\mathbf{x}_n, t), \dot{u}_j^{(n)}(\mathbf{x}_n, t) \right\rangle_{\mathbf{x}_n} = 0. \tag{2.1.7}$$

Let $\phi_{ij}^{(n)}(t) = \left\langle \dot{u}_i^{(n)}(\mathbf{x}_n, t), u_j^{(n)}(\mathbf{x}_n, t) \right\rangle_{\mathbf{x}_n}$, where $\phi_{ij}^{(n)}(t) \in \mathbb{R}^{r_n \times r_n}$. From Eq. (2.1.7), it is clear that $\phi_{ij}^{(n)}(t)$ is a skew-symmetric matrix $\phi_{ij}^{(n)}(t) = -\phi_{ij}^{(n)^T}(t)$. Based on the above definitions, we derive the evolution equations for the bases and the core tensor. To this end, we take a time derivative of the TDB decomposition given by Eq. (2.1.5). This follows:

$$\dot{v} = \dot{\underline{\mathbf{T}}}_{i_1 i_2 i_3} u_{i_1}^{(1)} u_{i_2}^{(2)} u_{i_3}^{(3)} + \underline{\mathbf{T}}_{i_1 i_2 i_3} \dot{u}_{i_1}^{(1)} u_{i_2}^{(2)} u_{i_3}^{(3)} + \underline{\mathbf{T}}_{i_1 i_2 i_3} u_{i_1}^{(1)} \dot{u}_{i_2}^{(2)} u_{i_3}^{(3)} + \underline{\mathbf{T}}_{i_1 i_2 i_3} u_{i_1}^{(1)} u_{i_2}^{(2)} \dot{u}_{i_3}^{(3)}. \tag{2.1.8}$$

By taking the inner product $\left\langle \bullet, u_{i_1'}^{(1)} u_{i_2'}^{(2)} u_{i_3'}^{(3)} \right\rangle_{\mathbf{x}}$ of Eq. (2.1.8) and using orthonormality conditions, the evolution of the core tensor is obtained as follows:

$$\dot{\underline{\mathbf{T}}}_{i_1' i_2' i_3'} = \left\langle \dot{v}, u_{i_1'}^{(1)} u_{i_2'}^{(2)} u_{i_3'}^{(3)} \right\rangle_{\mathbf{x}} - \underline{\mathbf{T}}_{i_1 i_2' i_3'} \phi_{i_1' i_1}^{(1)} - \underline{\mathbf{T}}_{i_1' i_2 i_3'} \phi_{i_2' i_2}^{(2)} - \underline{\mathbf{T}}_{i_1' i_2' i_3} \phi_{i_3' i_3}^{(3)}. \tag{2.1.9}$$

To derive the evolution equation for the bases, we start with deriving an expression for $\dot{u}_{i_1}^{(1)}$ by taking the inner product $\left\langle \bullet, u_{i_2'}^{(2)} u_{i_3'}^{(3)} \right\rangle_{\mathbf{x} \setminus \mathbf{x}_1}$ of Eq. (2.1.8) as follows:

$$\left\langle \dot{v}, u_{i_2'}^{(2)} u_{i_3'}^{(3)} \right\rangle_{\mathbf{x} \setminus \mathbf{x}_1} = \dot{\underline{\mathbf{T}}}_{i_1 i_2' i_3'} u_{i_1}^{(1)} + \underline{\mathbf{T}}_{i_1 i_2' i_3'} \dot{u}_{i_1}^{(1)} + \underline{\mathbf{T}}_{i_1 i_2 i_3'} u_{i_1}^{(1)} \phi_{i_2' i_2}^{(2)} + \underline{\mathbf{T}}_{i_1 i_2' i_3} u_{i_1}^{(1)} \phi_{i_3' i_3}^{(3)}. \tag{2.1.10}$$

By rearranging the indexes for $\dot{\underline{\mathbf{T}}}_{i_1' i_2' i_3'}$ from Eq. (2.1.9):

$$\dot{\underline{\mathbf{T}}}_{i_1 i_2' i_3'} = \left\langle \dot{v}, u_{i_1}^{(1)} u_{i_2'}^{(2)} u_{i_3'}^{(3)} \right\rangle_{\mathbf{x}} - \underline{\mathbf{T}}_{j_1 i_2' i_3'} \phi_{i_1' j_1}^{(1)} - \underline{\mathbf{T}}_{i_1' i_2 i_3'} \phi_{i_2' i_2}^{(2)} - \underline{\mathbf{T}}_{i_1' i_2' i_3} \phi_{i_3' i_3}^{(3)}.$$

We can substitute it into Eq. (2.1.10) and derive the evolution equation for the first bases:

$$\underline{\mathbf{T}}_{i_1 i'_2 i'_3} \dot{u}^{(1)}_{i_1} = \left\langle \dot{v}, u^{(2)}_{i'_2} u^{(3)}_{i'_3} \right\rangle_{\mathbf{x} \setminus \mathbf{x}_1} - u^{(1)}_{i_1} [\left\langle \dot{v}, u^{(1)}_{i_1} u^{(2)}_{i'_2} u^{(3)}_{i'_3} \right\rangle_{\mathbf{x}} - \underline{\mathbf{T}}_{j_1 i'_2 i'_3} \phi^{(1)}_{i_1 j_1}]. \tag{2.1.11}$$

We denote the orthogonal projection onto the complement space spanned by $\mathbf{U}^{(1)}$ as in the following:

$$\prod_{\mathbf{U}^{(1)}} \left\langle \dot{v}, u^{(2)}_{i'_2} u^{(3)}_{i'_3} \right\rangle_{\mathbf{x} \setminus \mathbf{x}_1} = \left\langle \dot{v}, u^{(2)}_{i'_2} u^{(3)}_{i'_3} \right\rangle_{\mathbf{x} \setminus \mathbf{x}_1} - \left\langle \left\langle \dot{v}, u^{(2)}_{i'_2} u^{(3)}_{i'_3} \right\rangle_{\mathbf{x} \setminus \mathbf{x}_1}, u^{(1)}_{i'_1} \right\rangle u^{(1)}_{i_1}.$$

Incorporating the above definition into Eq. (2.1.11) and rearranging the equation results in:

$$\underline{\mathbf{T}}_{i_1 i'_2 i'_3} \dot{u}^{(1)}_{i_1} = \prod_{\mathbf{U}^{(1)}} \left\langle \dot{v}, u^{(2)}_{i'_2} u^{(3)}_{i'_3} \right\rangle_{\mathbf{x} \setminus \mathbf{x}_1} + \underline{\mathbf{T}}_{i_1 i'_2 i'_3} \phi^{(1)}_{i'_1 i_1} u^{(1)}_{i_1}.$$

The above equation can be an underdetermined system with respect to unknowns $\dot{u}^{(1)}_{i_1}$ if $r_1 > r_2 r_3$ and it could be overdetermined otherwise. In order to address this issue, we find the least-square solution for $\dot{u}^{(1)}_{i_1}$. This can be accomplished by first multiplying both sides of the above equation by $\underline{\mathbf{T}}^T_{(1)}$ and then computing the inverse of the resulting matrix. This amounts to computing the pseudoinverse of the unfolded core tensor, which is denoted by $\mathbf{T}^{(1)^\dagger}$. The resulting equation is:

$$\dot{u}^{(1)}_{i_1} = \prod_{\mathbf{U}^{(1)}} \left\langle \dot{v}, u^{(2)}_{i'_2} u^{(3)}_{i'_3} \right\rangle_{\mathbf{x} \setminus \mathbf{x}_1} \mathbf{T}^{(1)^\dagger}_{i_1, i'_2 i'_3} + u^{(1)}_{i'_1} \phi^{(1)}_{i'_1 i_1}, \quad \mathbf{T}^{(1)^\dagger} = \underline{\mathbf{T}}^T_{(1)} (\underline{\mathbf{T}}_{(1)} \underline{\mathbf{T}}^T_{(1)})^{-1}, \quad \mathbf{T}^{(1)^\dagger}_{i_1, i'_2 i'_3} = \mathbf{T}^{(1)^\dagger}_{i_1, i'_2 + (i'_3 - 1) r_2}.$$

In a similar manner, we can derive $\dot{u}^{(2)}_{i_2}$ and $\dot{u}^{(3)}_{i_3}$:

$$\dot{u}^{(2)}_{i_2} = \prod_{\mathbf{U}^{(2)}} \left\langle \dot{v}, u^{(1)}_{i'_1} u^{(3)}_{i'_3} \right\rangle_{\mathbf{x} \setminus \mathbf{x}_2} \mathbf{T}^{(2)^\dagger}_{i_2, i'_1 i'_3} + u^{(2)}_{i'_2} \phi^{(2)}_{i'_2 i_2}, \quad \mathbf{T}^{(2)^\dagger} = \underline{\mathbf{T}}^T_{(2)} (\underline{\mathbf{T}}_{(2)} \underline{\mathbf{T}}^T_{(2)})^{-1}, \quad \mathbf{T}^{(2)^\dagger}_{i_2, i'_1 i'_3} = \mathbf{T}^{(2)^\dagger}_{i_2, i'_1 + (i'_3 - 1) r_1},$$

$$\dot{u}^{(3)}_{i_3} = \prod_{\mathbf{U}^{(3)}} \left\langle \dot{v}, u^{(1)}_{i'_1} u^{(2)}_{i'_2} \right\rangle_{\mathbf{x} \setminus \mathbf{x}_3} \mathbf{T}^{(3)^\dagger}_{i_3, i'_1 i'_2} + u^{(3)}_{i'_3} \phi^{(3)}_{i'_3 i_3}, \quad \mathbf{T}^{(3)^\dagger} = \underline{\mathbf{T}}^T_{(3)} (\underline{\mathbf{T}}_{(3)} \underline{\mathbf{T}}^T_{(3)})^{-1}, \quad \mathbf{T}^{(3)^\dagger}_{i_3, i'_1 i'_2} = \mathbf{T}^{(3)^\dagger}_{i_3, i'_1 + (i'_2 - 1) r_1}.$$

Similarly, we can derive the evolution equations for the general case where $2 \leq p \leq d$. To this end, we take the time derivative of Eq. (2.1.5) as follow:

$$\begin{aligned} \dot{v} = {} & \dot{\underline{\mathbf{T}}}_{i_1 i_2 \dots i_p} u^{(1)}_{i_1} u^{(2)}_{i_2} \dots u^{(p)}_{i_p} + \underline{\mathbf{T}}_{i_1 i_2 \dots i_p} \dot{u}^{(1)}_{i_1} u^{(2)}_{i_2} \dots u^{(p)}_{i_p} + \underline{\mathbf{T}}_{i_1 i_2 \dots i_p} u^{(1)}_{i_1} \dot{u}^{(2)}_{i_2} \dots u^{(p)}_{i_p} + \dots \\ & + \underline{\mathbf{T}}_{i_1 i_2 \dots i_p} u^{(1)}_{i_1} u^{(2)}_{i_2} \dots \dot{u}^{(p)}_{i_p}. \end{aligned} \tag{2.1.12}$$

By taking the inner product $\left\langle \bullet, u^{(1)}_{i'_1} u^{(2)}_{i'_2} \dots u^{(p)}_{i'_p} \right\rangle_{\mathbf{x}}$ of both sides of the Eq. (2.1.12) and using orthonormality conditions, the evolution of core tensor is obtained as follows:

$$\dot{\underline{\mathbf{T}}}_{i'_1 i'_2 \dots i'_p} = \left\langle \dot{v}, u^{(1)}_{i'_1} u^{(2)}_{i'_2} \dots u^{(p)}_{i'_p} \right\rangle_{\mathbf{x}} - \underline{\mathbf{T}}_{i_1 i'_2 \dots i'_p} \phi^{(1)}_{i'_1 i_1} - \underline{\mathbf{T}}_{i'_1 i_2 \dots i'_p} \phi^{(2)}_{i'_2 i_2} - \dots - \underline{\mathbf{T}}_{i'_1 i'_2 \dots i_p} \phi^{(p)}_{i'_p i_p}. \tag{2.1.13}$$

To derive the evolution equation $\dot{u}_{i_j}(\mathbf{x}_j, t)$, we project both sides of Eq. (2.1.12) onto all bases except $u_{i_j}(\mathbf{x}_j, t)$. This can be accomplished by taking the inner product

$$\left\langle \bullet, u^{(1)}_{i'_1} u^{(2)}_{i'_2} \dots u^{(j-1)}_{i'_{j-1}} u^{(j+1)}_{i'_{j+1}} \dots u^{(p)}_{i'_p} \right\rangle_{\mathbf{x} \setminus \mathbf{x}_j}$$

of Eq. (2.1.12). Similar to the simplified derivation for $p = 3$, we substitute the evolution of the core tensor from Eq. (2.1.13) into our operations. This yields to:

$$\dot{u}^{(j)}_{i_j} = \prod_{\mathbf{U}^{(j)}} \left\langle \dot{v}, u^{(1)}_{i'_1} u^{(2)}_{i'_2} \dots u^{(j-1)}_{i'_{j-1}} u^{(j+1)}_{i'_{j+1}} \dots u^{(p)}_{i'_p} \right\rangle_{\mathbf{x} \setminus \mathbf{x}_j} \mathbf{T}^{(j)^\dagger}_{i_j, i'_1 i'_2 \dots i'_{j-1} i'_{j+1} \dots i'_p} + u^{(j)}_{i_j} \phi^{(j)}_{i'_j i_j}. \tag{2.1.14}$$

Here, any skew symmetric choice for ϕ results in an equivalent TDB approximation. For simplicity, we can choose it to be zero. Therefore, Eq. (2.1.13) and (2.1.14) would become as follow:

$$\dot{\underline{\mathbf{T}}}_{i'_1 i'_2 \dots i'_p} = \left\langle \dot{v}, u^{(1)}_{i'_1} u^{(2)}_{i'_2} \dots u^{(p)}_{i'_p} \right\rangle_{\mathbf{x}}, \tag{2.1.15}$$

$$\dot{u}^{(j)}_{i_j} = \prod_{\mathbf{U}^{(j)}} \left\langle \dot{v}, u^{(1)}_{i'_1} u^{(2)}_{i'_2} \dots u^{(j-1)}_{i'_{j-1}} u^{(j+1)}_{i'_{j+1}} \dots u^{(p)}_{i'_p} \right\rangle_{\mathbf{x} \setminus \mathbf{x}_j} \mathbf{T}^{(j)^\dagger}_{i_j, i'_1 i'_2 \dots i'_{j-1} i'_{j+1} \dots i'_p}. \tag{2.1.16}$$

Not only does the choice of $\phi = 0$ simplify the above evolution equation but also implies a simple interpretation where $\dot{u}^{(j)}_{i_j}$ is always orthogonal to the space spanned by $u^{(j)}_{i_j}$ (dynamically orthogonal condition). This choice is also used for reduced-order modeling in Ref. [24, 25, 28, 30].

Equations (2.1.15) and (2.1.16) constitute the evolution equations for the TDB compression scheme. We note the DBO decomposition [25, 30] is a special case of the above scheme where $p = 2$. It has also been shown in Ref. [25] that DO and BO decompositions are equivalent to DBO, i.e., DO, BO and DBO extract the same subspace from the full-dimensional problem. In that sense, DO and BO are closely related to Eqs. (2.1.15) and (2.1.16) for the special case of $p = 2$. It is possible to derive data-driven evolution in the DO and BO forms.

The data-driven formulation of the DO decomposition is presented in Ref. [24].

Equations (2.1.15) and (2.1.16) must be solved in the space-time discrete form and they must be augmented with appropriate initial conditions.

2.1.3 Time-Dependent Bases in Discrete Space-Time Form

In the previous section, we presented the TDB decomposition in the continuous space-time form. In this section, we show the details of how the above scheme can be applied to streaming data that are discrete in space and time. In the space-discrete form, the multidimensional data ($v(\mathbf{x}, t)$) are represented by a p-order time-dependent tensor denoted by: $\underline{\mathbf{V}}(t) \in \mathbb{R}^{N_1 \times N_2 \times \cdots \times N_p}$. If data are generated by a simulation, $N_1, N_2, \ldots, N_p$ are the number of grid points. Note that the data may be generated by higher-dimensional independent variables, i.e., $d \geq p$. The orthonormal bases in the $\mathbf{x}_n$ dimension can be represented as a time-dependent matrix: $\mathbf{U}^{(n)}(t) = [\mathbf{u}_1^{(n)}(t), \mathbf{u}_2^{(n)}(t), \ldots, \mathbf{u}_{r_n}^{(n)}(t)] \in \mathbb{R}^{N_n \times r_n}$, where $\mathbf{u}_i^{(n)}$ is a discrete (vector) representation of u_i. We show the discrete form of the TDB formulation in Figure 1 where the inner product in the space is approximated with a quadrature rule:

$$\left\langle u_i^{(n)}, u_j^{(n)} \right\rangle_{\mathbf{x}_n} \simeq \mathbf{u}_i^{(n)^T} \mathbf{W}^{(n)} \mathbf{u}_j^{(n)}, \tag{2.1.17}$$

where $\mathbf{W}^{(n)} \in \mathbb{R}^{N_n \times N_n}$ is the diagonal matrix, whose diagonal elements are the quadrature weights. The time derivative of the streaming data ($\dot{\underline{\mathbf{V}}}(t)$) is computed with finite difference and the evolution equation for TDB and the core tensor can be advanced with a standard time-integration scheme. Various time discretization schemes can be used. High-order finite-difference discretizations require keeping the solution from multiple time steps in the memory. In the first demonstration example, we investigate the error introduced by using different temporal schemes for both $\dot{\underline{\mathbf{V}}}(t)$ and the evolution of TDB.

As we discuss in the next section, the TDB decomposition and the instantaneous HOSVD are closely connected. To this end, HOSVD of the full-dimensional data at $t = 0$ is used to initialize the core tensor as well as the TDB.

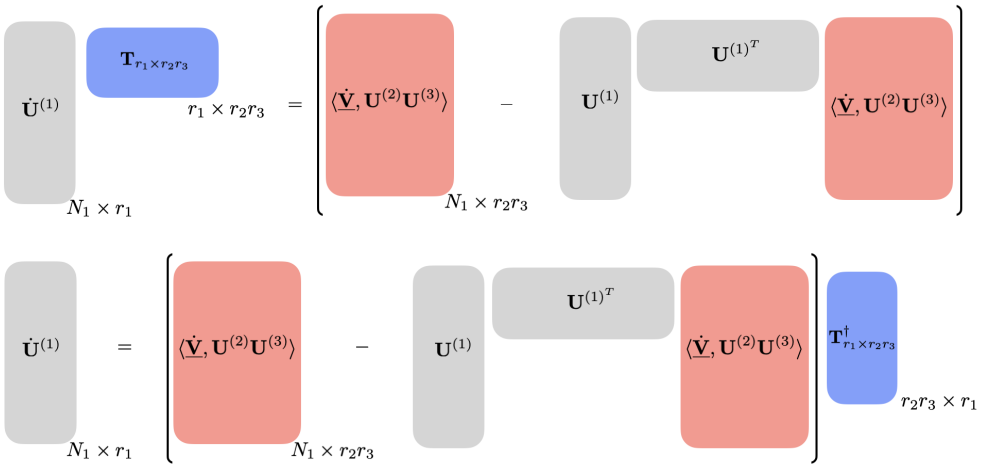


Figure 1: Time-dependent bases in discrete form. For simplicity weights are considered to be identity ($\mathbf{W}^{(1)}_{N_1 \times N_1} = \mathbf{I}$).

2.1.4 Extraction of Coherent Structure from Streaming Data

The TDB decomposition is closely related to the instantaneous HOSVD of the multidimensional data. The HOSVD extracts correlated structures for all p unfoldings of tensor $\underline{\mathbf{V}}(t)$. As we demonstrate in our examples, the HOSVD of the TDB core tensor $\underline{\mathbf{T}}(t)$ follows the leading singular values obtained by HOSVD of the full tensor $\underline{\mathbf{V}}(t)$. To establish the connection between HOSVD and TDB, let

$$\underline{\mathbf{T}}_{(n)}(t) = \mathbf{\Psi}^{(n)}(t) \mathbf{\Sigma}^{(n)}(t) \mathbf{\Theta}^{(n)^T}(t), \tag{2.1.18}$$

be the SVD of the unfolded core tensor, where $\mathbf{\Sigma}^{(n)} = \text{diag}(\sigma_1^{(n)}, \ldots, \sigma_{r_n}^{(n)})$ are the singular values of the n-mode unfolding of the core tensor, $\mathbf{\Psi}^{(n)}(t)$ and $\mathbf{\Theta}^{(n)}(t)$ are the left and right singular vectors of the unfolded core tensor. As we show in our demonstrations, $\mathbf{\Sigma}^{(n)}$ closely follows the r_n leading singular values of the n-mode unfolding of the full-dimensional streaming data ($\underline{\mathbf{V}}_{(n)}(t)$). Moreover, the TDB closely follows the leading left singular vectors of $\underline{\mathbf{V}}_{(n)}(t)$ after rotating TDB along the energetically ranked direction by using $\mathbf{\Psi}^{(n)}(t)$ as

in the following:

$$\tilde{\mathbf{U}}^{(n)}(t) = \mathbf{U}^{(n)}(t)\boldsymbol{\Psi}^{(n)}(t). \tag{2.1.19}$$

In other words, the TDB closely approximate the same subspace spanned by the leading left singular vectors of the the unfolded data. With the rotation given by Eq. (2.1.19), the TDB modes are ranked energetically and they can be compared against the left singular vectors of the the unfolded data. In the case of $p = 2$, where the high-dimensional data are matricized and TDB reduces to DBO, agreements between the TDB subspace and the instantaneously optimal subspace obtained from SVD of the full-dimensional data have already been established. See Ref. [25] for the case of stochastic reduced-order modeling, Ref. [30] for the case of reduced-order modeling of reactive species transport equation. This is also true for DO [104], OTD [105] and BO [106] formulations, which are all equivalent to the DBO formulation. In the case of linear dynamics, the convergence of TDB modes to the dominant subspace obtained by the SVD of the full-dimensional data is theoretically shown [105, 107].

The $\tilde{\mathbf{U}}^{(n)}(t)$ modes obtained from Eq. (2.1.19) captures the dominant structures among the columns of the nth unfolding of the full-dimensional data, and therefore, $\tilde{\mathbf{U}}^{(n)}(t)$ can be interpreted as instantaneous coherent structures in the streaming data.

2.1.5 Error Control and Adaptivity

For highly transient systems, the rank of the systems may change as the system evolves. Therefore, to maintain the error at a desirable level, the multirank $(r_1, r_2, \ldots, r_p)$ must change in time accordingly. There are two types of error in TDB decomposition: (i) temporal discretization error, and (ii) the error of the unresolved subspace, which are a result of neglecting the interactions of the resolved TDB subspace with the unresolved subspace. The lost interactions induce a *memory error* on TDB components that can be properly analyzed in the Mori-Zwanzig framework [108]. Unlike the model-driven TDB, in the data-driven mode, the error can be monitored by computing the Frobenius norm of the difference

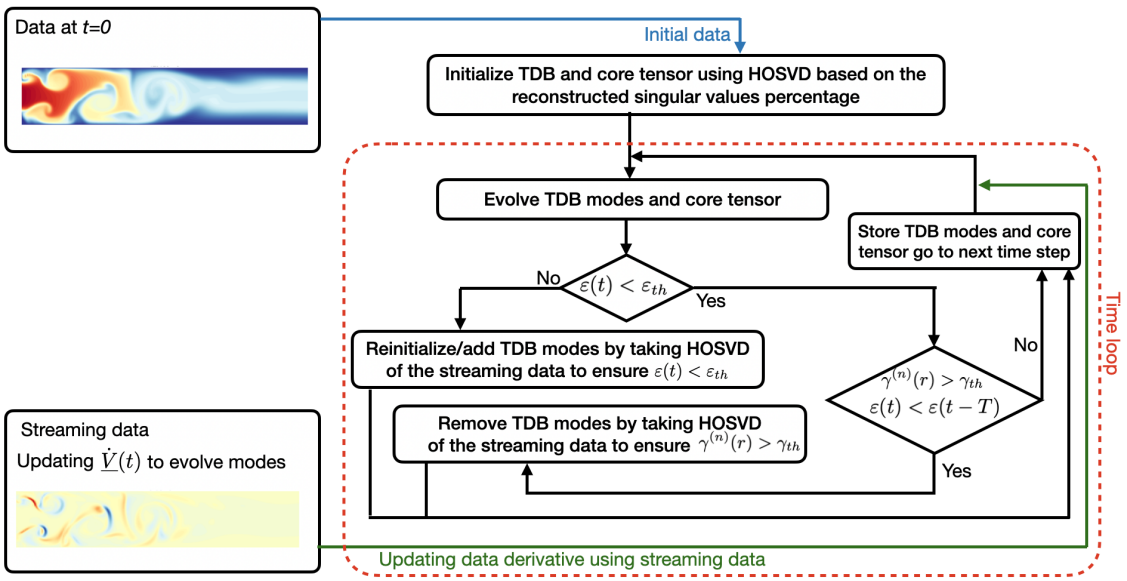


Figure 2: Algorithm flowchart for the compression and error adaptivity.

between the full data and TDB reconstruction, i.e.,

$$\varepsilon(t) = \left\| \underline{\mathbf{V}}(t) - \underline{\mathbf{V}}^{TDB}(t) \right\|_F. \tag{2.1.20}$$

The above Frobenius norm is defined based on the weighted inner product. To develop an adaptive strategy for TDB, we need to define an error criterion for mode addition/removal. Using singular values, we can calculate the percentage of the captured detail as follow:

$$\gamma^{(n)}(r) = \frac{\Sigma_{i=1}^{r_n} \sigma_i^{(n)^2}}{\Sigma_{i=1}^{N_n} \sigma_i^{(n)^2}} \times 100 = \frac{\Sigma_{i=1}^{r_n} \sigma_i^{(n)^2}}{\Sigma_{i=1}^{r_n} \sigma_i^{(n)^2} + \varepsilon^2(t)} \times 100\%. \tag{2.1.21}$$

In order to control the error, we define a maximum allowable error for $\varepsilon(t)$, which is chosen by the practitioner. The adaptive algorithm adds modes if the error exceeds the threshold value and removes modes if the error can be maintained below the threshold with lower ranks based on the defined $\gamma^{(n)}(r)$. In particular, if the error exceeds the maximum error (ε_{th}), the algorithm: (i) reinitializes the core tensor and TDB; (ii) increases the number

of modes if the resulting error from reinitialization stays above the limit; (iii) the algorithm reinitializes and reduces the number of modes if the solution starts to capture unnecessary energy and has a negative average slope for a defined number of iterations. In this process, we can increase/decrease the number of modes with respect to the relation between the defined error and singular values Eq. (2.1.21). For example, assume the maximum error limit is set to be $\varepsilon(t) = \varepsilon_{th}$. First, we initialize the TDB and the core tensor using HOSVD and calculate the required number of modes in each direction based on $\gamma^{(n)}(r) = \gamma_{th}$. During the computation, if $\varepsilon(t) > \varepsilon_{th}$, the algorithm reinitializes the modes. If the resulting error is not less than the prescribed limit, the algorithm increases the number of modes based on γ_{th}. In the subsequent time steps if $\gamma(t) > \gamma_{th}$, the algorithm decreases the number of modes. Note that, rank adjustment reinitializes the modes and the core tensor, which may reduce the error for only a few iterations. This causes frequent rank adjustment with high computational costs. Therefore, in addition to the defined error, the slope of the error for a defined number of iterations ($\varepsilon(t) - \varepsilon(t-T) < 0$ where T is the time duration that is set by the practitioner) must be negative to trigger the rank reduction process (excluding the time steps with HOSVD reinitialization). We show the algorithm of this example in Figure 2.

2.1.6 Scalability and Compression Ratio

We show that the computational complexity of solving the resulting equations linearly scales with the size of the data. Here, for simplicity, we consider ($p = d$), i.e., one-dimensional TDB and $N = N_1 = N_2 = \cdots = N_d$. In this case, the total size of the data is $S = N^d$. We also consider the case where the core tensor has equal n-ranks, i.e., $r = r_1 = r_2 = \cdots = r_d$. The leading costs of evolving TDB equations are:

1. Projection of the time-derivative of the data onto TDB ($\mathbf{U}^{(n)^T}\underline{\dot{\mathbf{V}}}_{(n)}$), which is $\mathcal{O}(rdS)$.
2. Computing the pseudoinverse of the unfolded core tensor ($\mathbf{T}^{(n)^\dagger}$). This requires the computation of $\underline{\mathbf{T}}_{(n)}\underline{\mathbf{T}}_{(n)}^T$, which scales with $\mathcal{O}(r^d)$ and the computation of the inverse of this matrix ($\underline{\mathbf{T}}_1\underline{\mathbf{T}}_1^T)^{-1}$), which scales with $\mathcal{O}(r^3)$.

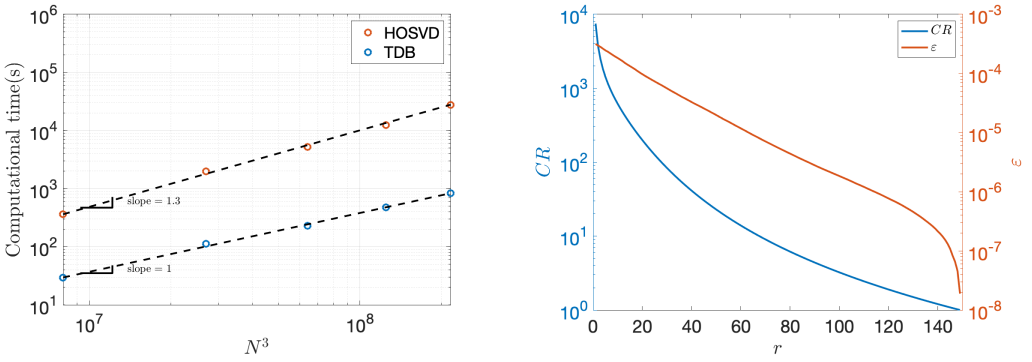
We make the following observations about the cost of evolving TDB:

1. When $r << N$, the computational cost of computing the pseudoinverse of the unfolded core tensor is negligible to the projection of the time-derivative of the data onto TDB, which scales linearly with the total size of the data.
2. The evolution of TDB components does not require computation of any eigenvalue problem as required in SVD-based reductions or solving a nonlinear optimization problem as is required in autoencoder-decoder reductions [109, 16, 110].
3. Many entries of $\dot{\underline{\mathbf{V}}}$ have small values. See Figure 2 for an example of $\dot{\underline{\mathbf{V}}}$. Therefore, the sparse approximation of $\dot{\underline{\mathbf{V}}}$ can significantly reduce overall cost, although this advantage of TDB has not been explored in this work.

The computational cost of solving the TDB equations can be compared against that of computing HOSVD. HOSVD requires computing the SVDs of the unfolded tensor $\underline{\mathbf{V}}_{(n)}$ for $n = 1, 2, \ldots, d$. This requires computation and storage of the correlation matrix $\mathbf{C}^{(n)} = \underline{\mathbf{V}}_{(n)}^T \underline{\mathbf{V}}_{(n)}$, which scales with $\mathcal{O}(dN^{d+1})$ and the eigenvalue computation of this matrix, which scales with N^3. The computational cost of TDB and HOSVD for a block of data generated by the turbulent channel flow simulation (last demonstration) for the case of $N = N_1 = N_2 = N_3$ and the total elapsed time for 500 time-step advancement, are shown in Figure 3(a). This confirms that the computational cost of TDB scales linearly with the total data size $S = N^3$, while the computational cost of HOSVD scales with $S^{4/3}$. This shows that TDB computations are significantly faster than HOSVD and the disparity between the computational cost of TDB and HOSVD only grows as the dimension or number of grid points (or samples) increases.

The compression ratio of the TDB decomposition is computed as the ratio of the storage cost of storing the full-dimensional data to that of solving the TDB components. The TDB requires storing the core tensor and the orthonormal lower-dimensional bases. Therefore, the compression ratio for TDB is:

$$CR = \frac{N_1 N_2 \ldots N_p}{r_1 N_1 + r_2 N_2 + \cdots + r_p N_p + r_1 r_2 \ldots r_p}.$$



(a) (b)

Figure 3: Computational performance for turbulent channel flow: (a) Scalability, (b) Compression ratio and resulting error.

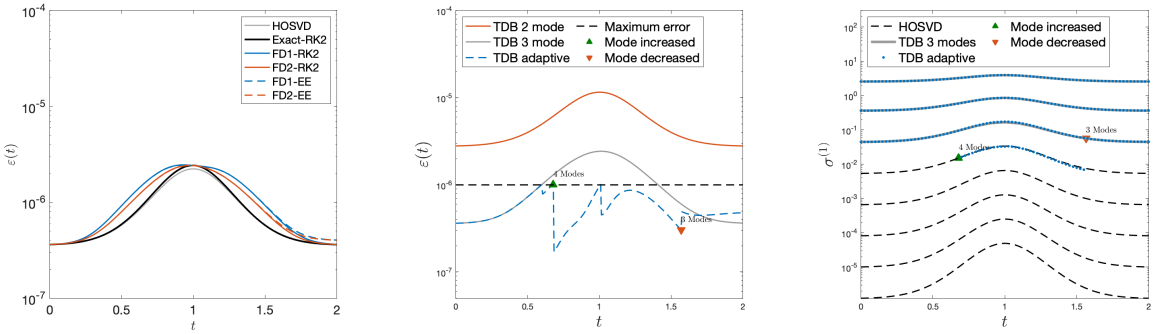
Figure 3(b) shows the compression ratio (CR) for the same problem with the corresponding error resulting from dimension reduction. Since our algorithm is adaptive, it can change the compression ratio by changing the number of modes; therefore, we introduce the weighted compression ratio $\overline{CR}$ as follow:

$$\overline{CR} = \frac{t_m - t_0}{\frac{(t_1-t_0)}{CR_1} + \frac{(t_2-t_1)}{CR_2} \cdots + \frac{(t_m-t_{m-1})}{CR_m}}. \tag{2.1.22}$$

Where, $CR_1, CR_2, \ldots CR_m$ are compression ratios for $(t_1 - t_0), (t_2 - t_1), \ldots, (t_m - t_{m-1})$ time intervals, respectively. This equation allows us to measure the effective compression ratio for all time intervals when the number of modes changes in adaptivity process.

2.2 Demonstration Cases

In this section, we present four demonstration cases for the application of TDB for the compression of streaming data: (i) Runge function; (ii) incompressible unsteady reactive



(a) (b) (c)

Figure 4: Runge function: (a) The effect of numerical and reduction error. (b) Comparing the effect of mode adjustment on error. (c) Singular values comparison.

flow; (iii) stochastic turbulent reactive flow, and (iv) three-dimensional turbulent channel flow.

2.2.1 Runge Function

We first demonstrate the adaptive TDB compression technique with one dimensional modes (i.e., $d = p$) for a time dependent Runge function as follow:

$$f(x_1, x_2, x_3, t) = \frac{1}{a(t)^2 + x_1^2 + x_2^2 + x_3^2}, \tag{2.2.1}$$

where $a(t) = 1 - 0.5\exp(-\alpha(t-1)^2)$ and $\alpha = 0.5$. For this choice of $a(t)$, the rank of the TDB decomposition must increase from $t = 0$ to $t = 1$ and then decrease to ensure that the low-rank approximation error remains below a specified value. The spatial domain is the cube $[-\pi, \pi]^3$ discretized on a uniform grid of size 126^3. The time step $\Delta t = 5 \times 10^{-3}$ is used for evolving the TDB evolution equations.

In this problem, we use the same number of modes in each direction, i.e., $r = r_1 = r_2 = r_3$ due to the isotropy of function f. First, we show the effect of both unresolved and

numerical errors in Figure 4(a). Here, HOSVD error is the weighted Frobenius norm of the difference between reconstructed data from HOSVD and the streaming data, which shows the unresolved error from reduction. The implemented first-order Euler (EE) and second-order Runge-Kutta (RK) add numerical error to the existing unresolved error. The second-order RK scheme has less error and we use this scheme to show adaptivity effect. This figure also shows the error difference between the exact data derivative with its first-order, and central second-order approximation. In order to study the adaptivity procedure for this analytical problem, we used the data derivative instead of finite difference approximations. Figure 4(b) shows the reconstruction error versus time for fixed-rank TDB decompositions for $r = 2$ and $r = 3$ and adaptive TDB initiated with $r = 3$. The error threshold and the captured energy percentage is set by practitioner to be $\varepsilon_{th} = 10^{-6}$ and $\gamma_{th} = 99.999\%$, respectively. It is evident that errors of the fixed-rank approximations $r = 2$ and $r = 3$ exceed the upper limit of the 10^{-6}, while the adaptive TDB maintain the error below the upper limit by increasing the rank to $r = 4$. The number of modes is later reduced to $r = 3$ as the dimensionality of the problem decreases for the given γ_{th}. Figure 4(c) shows the singular values of the unfolded core tensor in the x_1-direction for the fixed-rank and adaptive-rank cases as well as the corresponding HOSVD singular values, which are obtained by taking instantaneous SVD on the unfolded full-dimensional data. This shows that the fixed-rank and adaptive TDB decompositions closely follow the HOSVD.

2.2.2 Incompressible Turbulent Reactive Flow

In the second demonstration, we apply the TDB compression to a turbulent reactive flow. Turbulent reactive flows are multiscale and multivariate, whose high-fidelity numerical simulations result in massive datasets that with the current I/O restrictions of exascale simulations even storing the temporally resolved solution is becoming increasingly challenging let alone probing and analysis of the simulation data. This is particularly the case when a large number of species is involved. To demonstrate the application of TDB, we consider

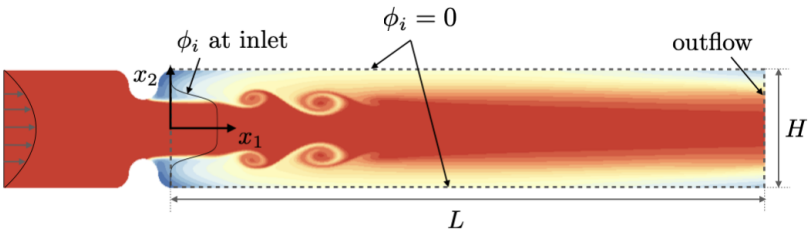


Figure 5: Incompressible turbulent reactive flow schematic.

streaming data generated by a 2D advection-diffusion reaction problem:

$$\frac{\partial \phi_i}{\partial t} + (u \cdot \nabla)\phi_i = \nabla \cdot (\alpha_i \nabla \phi_i) + Q_i^S \tag{2.2.2}$$

where, ϕ_i is the concentration of ith reactant, α_i is the associated diffusion coefficient, Q_i^S denotes the nonlinear source that determines whether ϕ_i is produced or consumed and u is the velocity field of the flow. The reaction mechanisms model the blood coagulation cascade in a Newtonian fluid [111]. The simulation setup chosen in this work is identical to the case considered in reference [30]. The incompressible Navier-Stokes equations at Re=1000 are solved with spectral element using the spectral/hp element code Nektar with an unstructured mesh with 4008 quadrilateral elements and polynomial order 5. For more details on Nektar for incompressible Navier-Stokes equations, see [112]. Equation (2.2.2) is solved for $n_s = 23$ species on the dashed domain using nodal spectral element discretization. We used a uniform quadrilateral grid $N_1 = 251$ elements in the x_1 direction and $N_2 = 76$ in the x_2 direction. The velocity field is interpolated from the unstructured mesh to the uniform mesh at all time instants. Equation (2.2.2) is advanced in time using fourth-order RK with $\Delta t = 5 \times 10^{-4}$. The numerical solution of Eq. (2.2.2) is cast as a streaming third-order tensor of size $N_1 \times N_2 \times N_3$, where $N_3 = n_3 = 23$ is the number of species.

We consider two different TDB compression schemes as follows:

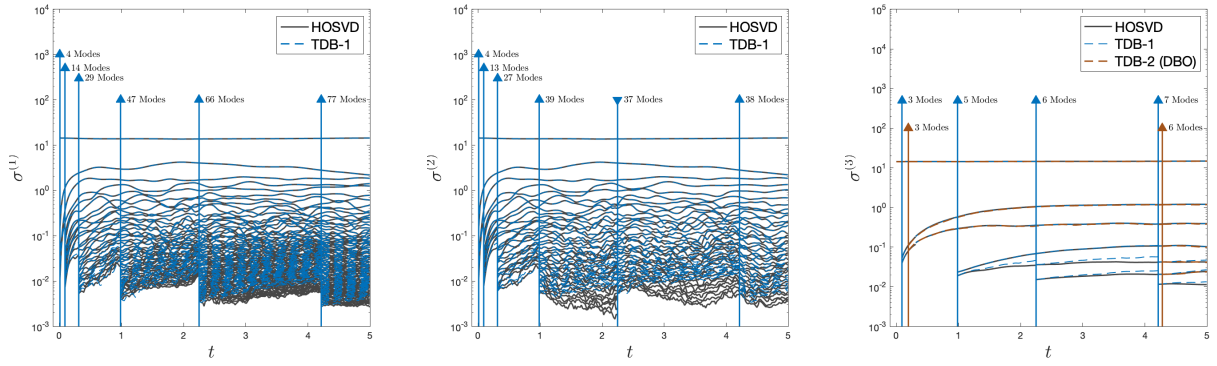
$$\text{TDB-1:} \qquad \phi(x_1, x_2, \eta, t) = \sum_{i_3=1}^{r_3} \sum_{i_2=1}^{r_2} \sum_{i_1=1}^{r_1} \underline{\mathbf{T}}_{i_1 i_2 i_3}(t) u_{i_1}^{(1)}(x_1, t) u_{i_2}^{(2)}(x_2, t) u_{i_3}^{(3)}(\eta, t) \quad (2.2.3)$$

$$\text{TDB-2 (DBO):} \qquad \phi(x_1, x_2, \eta, t) = \sum_{i_2=1}^{r_2} \sum_{i_1=1}^{r_1} \underline{\mathbf{T}}_{i_1 i_2}(t) u_{i_1}^{(1)}(x_1, x_2, t) u_{i_2}^{(2)}(\eta, t), \quad (2.2.4)$$

where the composition space is denoted by η. The inner product in the physical and composition space are as follow:

$$\begin{aligned}
&\text{Physical space (TDB-1):} && \left\langle u_{i_1}^{(1)}(x_1,t), u_{i_1'}^{(1)}(x_1,t) \right\rangle_{x_1} \simeq \mathbf{u}_{i_1}^{(1)^T} \mathbf{W}^{(1)} \mathbf{u}_{i_1}^{(1)} \\
& && \left\langle u_{i_2}^{(2)}(x_2,t), u_{i_2'}^{(2)}(x_2,t) \right\rangle_{x_2} \simeq \mathbf{u}_{i_2}^{(2)^T} \mathbf{W}^{(2)} \mathbf{u}_{i_2}^{(2)} \\
&\text{Physical space (TDB-2):} && \left\langle u_{i_1}^{(1)}(\mathbf{x},t), u_{i_1'}^{(1)}(\mathbf{x},t) \right\rangle_{\mathbf{x}} \simeq \mathbf{u}_{i_1}^{(1)^T} \mathbf{W}^{(1)} \mathbf{u}_{i_1}^{(1)} \\
&\text{Composition space (TDB-1):} && \left\langle u_{i_3}^{(3)}(\eta,t), u_{i_3'}^{(3)}(\eta,t) \right\rangle_{\eta} \simeq \mathbf{u}_{i_3}^{(3)^T} \mathbf{u}_{i_3}^{(3)} \\
&\text{Composition space (TDB-2):} && \left\langle u_{i_2}^{(2)}(\eta,t), u_{i_2'}^{(2)}(\eta,t) \right\rangle_{\eta} \simeq \mathbf{u}_{i_2}^{(2)^T} \mathbf{u}_{i_2}^{(2)}
\end{aligned}$$

In TDB-1, $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$ are quadrature weights obtained from spectral element discretizations of x_1 and x_2 directions, respectively. In TDB-2, $\mathbf{W}^{(1)}$ are the quadrature weights for 2D spectral element discretization i.e., both x_1 and x_2 directions. The inner product weight in the composition space for both schemes is the identity matrix. The above two schemes have different reduction errors and compression ratios. The main difference between the two decompositions is that in Eq. (2.2.3), the correlations between both spatial directions, i.e., x_1, x_2 and the *composition* space, i.e., η are extracted, whereas in TDB-2, the correlation between x_1 and x_2 directions are not extracted. Both schemes are adaptive: modes are added and removed to keep the reconstruction error below $\varepsilon_{th} = 10^{-5}$ and the captured details above the $\gamma_{th} = 99.999\%$. TDB-2 was recently introduced in Ref. [30], where it was referred to as *dynamically bi-orthonormal* (DBO) decomposition since the species are decomposed to two sets of orthonormal modes in the spatial domain and the composition space. In the DBO formulation presented in Ref. [30], full-dimensional Eq. (2.2.2) is not solved; instead



(a) (b) (c)

Figure 6: Compression of the 2D turbulent reactive flow. Singular values of the unfolded tensor in directions: (a) x_1, (b) x_2, and (c) η.

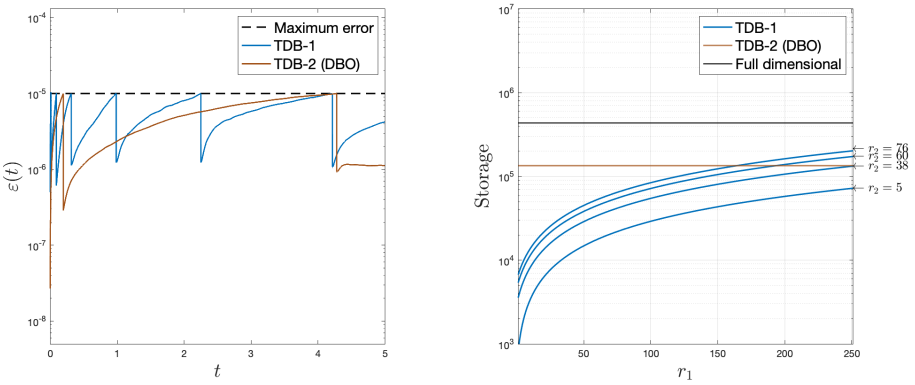
closed-form evolution equations for all three components of TDB-2 are derived, i.e., ODEs for $\underline{\mathbf{T}}_{i_1 i_2}(t)$ and $u^{(2)}_{i_2}(\eta, t)$ and PDEs for $u^{(1)}_{i_1}(x_1, x_2, t)$. We refer to DBO reduction in Ref. [30], as model-driven analogue of the data-driven reduction technique presented in this work, because in the DBO formulation in Ref. [30] data generation is not required. In this section, we compare the model-driven DBO bases and the data-driven DBO bases that are the focus of this work.

TDB-2 shows a slower error growth than TDB-1. To investigate this, in Figures 6(a)-6(c), we show the instantaneous singular values of the unfolded tensor in x_1, x_2 and η directions for TDB-1 and TDB-2 as well as the instantaneous singular values obtained by performing HOSVD. Since in the TDB-2 decomposition, unfolding in x_1 and x_2 directions are not formed, in Figures 6(a)-6(b) only the singular values of TDB-1 and HOSVD can be shown. It is clear that dominant singular values are captured accurately by both schemes. However, the effect of unresolved modes introduces a memory error. This error is driven by the energy of the unresolved modes and it affects the lower-energy modes more intensely

than the higher-energy modes. If this error is left uncontrolled, it will eventually contaminate the higher-energy modes. The reconstruction error evolution for both schemes is shown in Figure 7(a). It is evident that the error grows at a much faster rate in TDB-1 compared to TDB-2. This can be explained by observing that the dimensionality in the spatial domain is often much higher than the dimensionality in the composition space. This means that in TDB-1, $\sigma^{(3)}(t)$ has a much faster decay rate compared to $\sigma^{(1)}(t)$ and $\sigma^{(2)}(t)$ as can be clearly seen in Figures 6(a)-6(c). In TDB-2, there is no reduction error in the x_1 or x_2 directions, and moreover, the turbulent reactive flow shows a very low-dimensional dynamics in the composition space (η), as a result the overall energy of the unresolved modes is much smaller, which in turn leads to a slower error growth rate.

In Figure 8, we compare the reconstructed data for the eighth species at five time instants. The reconstructed data are in good agreement with DNS snapshots, however, the form of error is different. The error is computed as the absolute value of the difference between the DNS and reconstructed TDB data. The resulting error from TDB-1 appears to have much finer structure than that of TDB-2. That is because in TDB-1, the unresolved subspaces in each of the x_1 and x_2 dimensions have very fine structures and they dominate the error, whereas in TDB-2 the error is due to the unresolved subspace in the η direction. The TDB-1 error at $t = 1, 2, 3, 4$ is less than TDB-2, mainly because of frequent HOSVD reinitialization in the adaptivity process. However, the maximum error of TDB-1 is larger than that of TDB-2. For better comparison, the error color bar is set from 0 to 0.5, while the maximum errors for TDB-1 and TDB-2 are 0.14 and 0.4, respectively.

TDB-1 and TDB-2 have different storage costs for the same reconstruction errors. In TDB-2, slower error growth rate comes at the cost of storing r two-dimensional bases with the storage cost of rN_1N_2 as opposed to TDB-1, where the storage cost of the spatial bases is $r_1N_1 + r_2N_2$. The overall storage cost of TDB-1 is: $S_{TDB-1} = r_1N_1 + r_2N_2 + r_3n_s + r_1r_2r_3$ and the storage cost of TDB-2 is: $S_{TDB-2} = rN_1N_2 + rn_s + r^2$. The comparison of the storage cost of TDB-1 and TDB-2 for the same reconstruction error depends on the values of r_1 and r_2. These two storage costs for the same number of modes in the composition space, i.e.,



(a) (b)

Figure 7: Compression of turbulent reactive flow data using TDB-1 and TDB-2 (DBO) schemes: (a) error evolution, (b) storage cost.

$r = r_3 = 7$ are shown in Figure 7(b). If r_1 and r_2 are large enough, the storage cost of TDB-1 exceeds that of TDB-2. In our study, the weighted compression ratio are $\overline{CR}_{TDB-1} = 15.62$ and $\overline{CR}_{TDB-2} = 6.6$ where the total size of data $35GB$ compressed to $2.2GB$ and $5.3GB$ by TDB-1 and TDB-2, respectively.

In the DBO formulation presented in Ref. [30], a coupled set of PDEs for the 2D bases and ODEs for the low-rank matrices are solved without using any data. The goal of the model-driven DBO [30] is to reduce the computational cost and memory requirement of solving species transport equations as well as reducing the I/O load. In this work, our goal is to compress the streaming data generated by the full-dimensional model. In Figure 9, the first and second most dominant modes of model-driven and data-driven DBOs for different instances of times are shown. It is clear that the bases for both model-driven DBO and data-driven DBO are nearly identical.

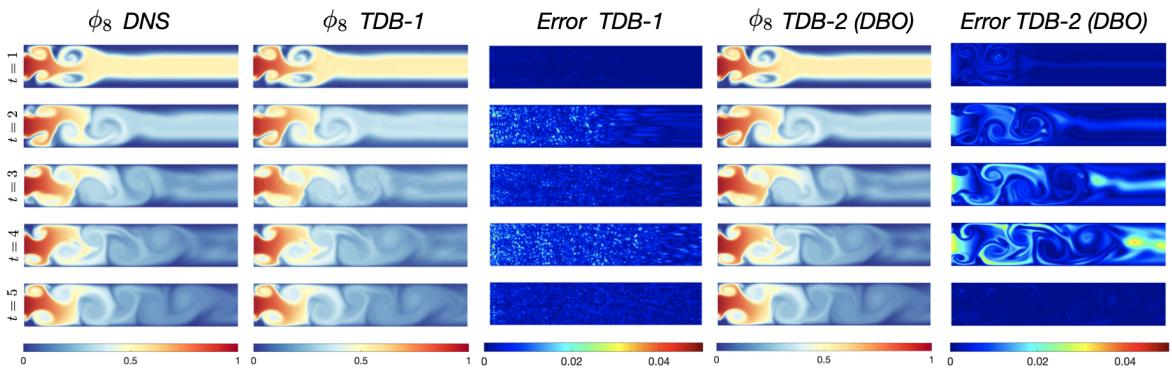


Figure 8: Unsteady reactive flow: Comparison between DNS and TDB reconstructed species concentration.

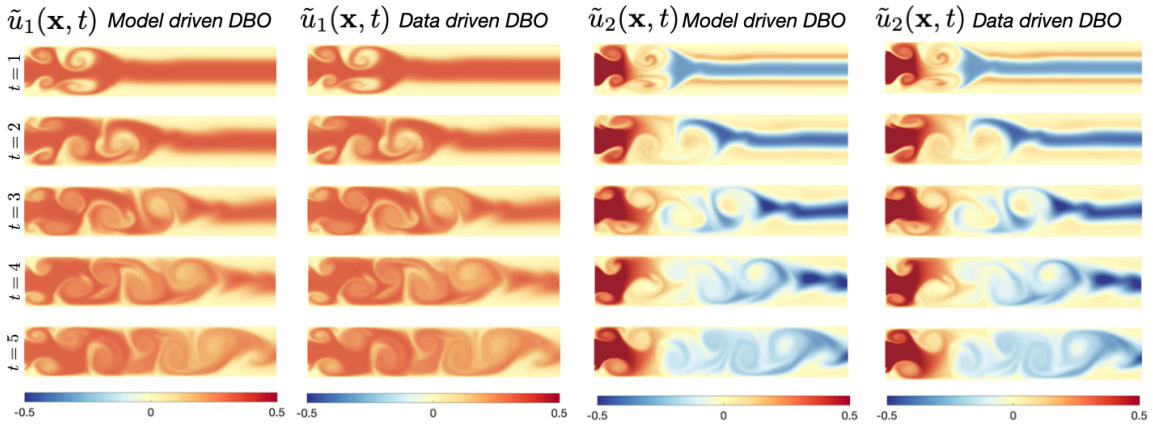


Figure 9: Turbulent reactive flow: Comparison between the first two dominant modes of the model-driven and data-driven DBO.

2.2.3 Stochastic Turbulent Reactive Flow

Quantifying uncertainties of transport and chemistry model parameters has major implications for the field of chemically reactive flows. An uncertainty quantification (UQ) analysis

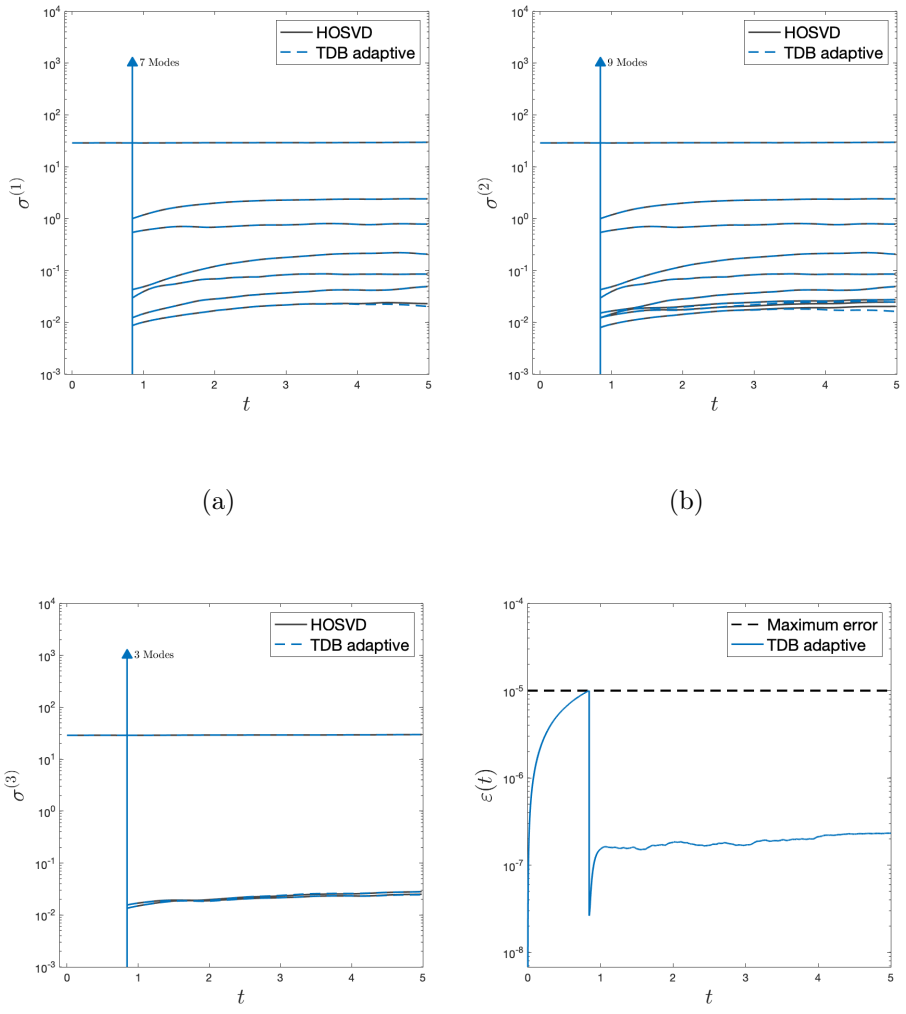


Figure 10: Incompressible turbulent reactive flow uncertainty quantification. Singular values of the unfolded tensor in: (a) x_1, (b) x_2, (c) η, and (d) error evolution.

can significantly reduce the experimental costs by effectively allocating limited resources on reducing the uncertainty of key parameters, and inform mechanism reduction by determining the least important parameters or detecting reaction pathways that are unimportant and can

be eliminated [113, 114, 115]. Any sampling-based technique, e.g., Monte Carlo or probabilistic collocation methods (PCM) for UQ analysis of this problem can generate very large datasets. In this demonstration, we show how TDB can be utilized to extract correlation between different random samples in addition to the multidimensional correlations presented in the previous example, to reduce the storage cost of the data generated. In this section, we study the compression of resulting data from incompressible reactive flow with random diffusion coefficients. We consider the problem of uncertainty diffusion coefficients for two species (α_1 and α_4) in Eq. (2.2.2). We assume both of these two coefficients are independent random variables as in the following:

$$\boldsymbol{\xi} = [\xi_1, \xi_2] = (1 + 0.05\omega)[\alpha_1, \alpha_4],$$

where ω is a uniform random variable $\sim \mathscr{U}[-1, 1]$. The rest of the problem setup remains identical to the problem considered in §3.2. We consider a collocation grid of size $s = 4 \times 4$ in the random space, which requires 16 forward DNS simulations. We consider the following TDB compression scheme:

$$\phi(\mathbf{x}, \eta, \boldsymbol{\xi}, t) = \sum_{i_3=1}^{r_3} \sum_{i_2=1}^{r_2} \sum_{i_1=1}^{r_1} \underline{\mathbf{T}}_{i_1 i_2 i_3}(t) u_{i_1}^{(1)}(x_1, x_2, t) u_{i_2}^{(2)}(\eta, t) u_{i_3}^{(3)}(\xi_1, \xi_2, t). \tag{2.2.5}$$

The rationale for choosing a 2D TDB for the random space rather than two 1D TDBs is that the cost of storing 2D random bases is negligible. On the other hand, choosing two 1D random TDBs would have increased the order of the core tensor from 3 to 4. Another valid choice for TDB is to combine the composition space and the random direction into a 3D space. However, the interpretability of 1D TDB in the composition space is quite appealing. The $u_{i_2}^{(2)}(\eta, t)$'s represent a time-dependent subspace in the composition space. In which the inner product in the random space is defined as follow:

$$\left\langle u_{i_3}^{(3)}(\xi_1, \xi_2, t), u_{i_3'}^{(3)}(\xi_1, \xi_2, t) \right\rangle_{\boldsymbol{\xi}} = \mathbb{E}[u_{i_3}^{(3)}(\xi_1, \xi_2, t)\, u_{i_3'}^{(3)}(\xi_1, \xi_2, t)] \simeq \mathbf{u}_{i_3}^{(3)^T} \mathbf{W}^{(3)} \mathbf{u}_{i_3}^{(3)},$$

where $\mathbf{W}^{(3)}$ is the probabilistic collocation weight. Each random TDB in the discrete form is represented by a vector of size s, i.e., $\mathbf{u}_i^{(3)} \in \mathbb{R}^{s \times 1}$. Different sampling schemes might be

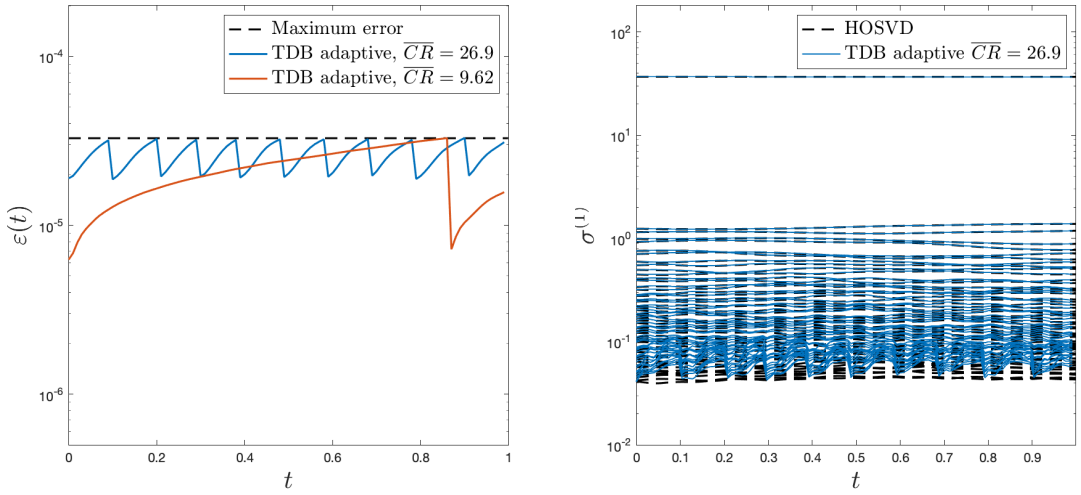
used here. For example, one can use Monte Carlo samples. In that case, the inner product weight would be a diagonal matrix with all diagonal entries equal to $1/s$.

Since there is a high degree of correlation between random samples of species fields, the TDB compression can achieve the high compression ratio $\overline{CR} = 42.75$ compared to the previous cases and compress the size of data from $561.5GB$ to $13.1GB$. To examine this, we show the resulting singular values of the unfolded core tensor and compare them with HOSVD singular values in Figures 10(a)-10(c). Based on these figures, we make the following observations: (i) the dominant singular values are captured accurately; (ii) for this compression scheme, similar to TDB-2 the singular values have a much faster decay rate compared to TDB-1; (iii) the problem dimension does not change after $t = 1$, and therefore, the number of modes remains the same; (iv) the random space has the lowest dimensionality, which means the generated data are highly correlated with respect to the random diffusion coefficients. Figure 10(d) shows the reconstruction error evolution in which it exceeds the maximum limit around $t = 1$ due to the increase in dimensionality. Using the reinitialization and mode adjustment in each direction with respect to the defined $\gamma_{th} = 99.999\%$ the error decreases. Since after $t = 1$ the dimensionality does not change, the error remains the same.

2.2.4 Three-dimensional Turbulent Channel Flow

In the last demonstration case, we use TDB to compress the data obtained by the direct numerical simulation (DNS) of turbulent channel flow. The data are generated by the finite difference solver in Ref. [116]. The Reynolds number based on the friction velocity is $Re_\tau = 180$. The length, width and height of the channel are π, 2π, and 2, respectively. The number of grid points in all dimensions is 150 with uniform distribution in streamwise and spanwise directions. The grid is clustered near the channel wall in the wall-normal direction. We apply TDB to the streamwise velocity component as compression of other field variables result in the same qualitative observations. We consider one-dimensional TDBs as follows:

$$v(x_1, x_2, x_3, t) = \sum_{i_3=1}^{r_3} \sum_{i_2=1}^{r_2} \sum_{i_1=1}^{r_1} \underline{\mathbf{T}}_{i_1 i_2 i_3}(t) u_{i_1}^{(1)}(x_1, t) u_{i_2}^{(2)}(x_2, t) u_{i_3}^{(3)}(x_3, t). \tag{2.2.6}$$



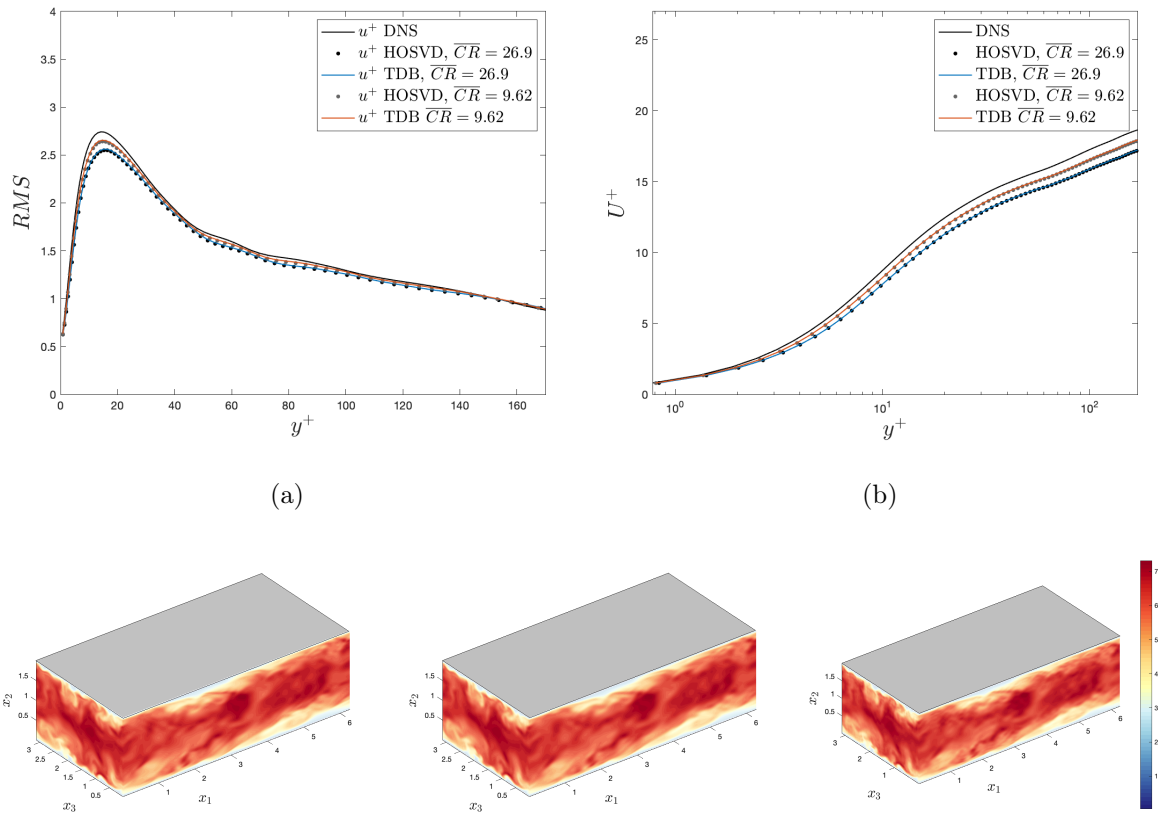
(a) (b)

Figure 11: Turbulent channel flow: (a) Error evolution. (b) Singular values of the unfolded tensor in x_1 direction.

The TDB equations are evolved with the time step of $\Delta t = 10^{-3}$, which is the same as the Δt used in the DNS time integration. We consider two cases with different error thresholds. Case-I with the $\gamma_{th} = 99.9999\%$ and Case-II with the $\gamma_{th} = 99.999\%$. For both cases the error threshold is set to be $\varepsilon_{th} = 3.3 \times 10^{-5}$. In Case-I, the initial value of the multirank is $(r_1, r_2, r_3) = (68, 81, 50)$ and in Case-II the initial multirank is $(r_1, r_2, r_3) = (54, 57, 36)$. The error evolution for both case are shown in Figure 11(a). The compression ratios for Case-I and Case-II are $\overline{CR} = 9.62$ and $\overline{CR} = 26.90$, respectively. For this problem, performing HOSVD at different time instants result in the same multirank and that is because the flow is statistically steady state, and therefore, the rank of the systems does not change in time. However, in the TDB formulation, the error still grows because of the effect of the unresolved subspace. This error is controlled by the adaptive scheme. The error evolution for both cases are shown in Figure 11(a). In Case-II, since the unresolved space has larger

energy, the error grows with faster rate compared to Case-I. As a result, in Case-II, 9 HOSVD initializations are performed as opposed to one initialization in Case-I. Figure 11(b) shows the singular values of the unfolded TDB core tensor along the x_1 direction against that of the full-dimensional data obtained by HOSVD. There is a large gap between the first singular value and the rest of the singular values. We also observe that the low-energy modes values are more contaminated by the error induced by the unresolved subspace.

Figures 12(a) and 12(b) compare the RMS and mean velocity at $t = 1$ from TDB reconstructed calculations with Ref. [116] and HOSVD reconstruction. In these figures, the TDB and HOSVD reconstructed RMS and the mean streamvise velocity are in good agreement, however; due to unresolved modes the reconstructed data has discrepancy compared to the DNS results. We can observe the higher compression ratio is causing more discrepancy due to more unresolved modes. Figures 12(c) to 12(e) compare the TDB and HOSVD reconstructed data with the streamed data for the case with high compression ratio at $t = 1$ in 3D format. The fact that we need a large number of modes to achieve a reasonable agreement with the DNS suggests that the linear TDB subspace has difficulty in approximating the turbulent state, which lies on a nonlinear manifold.



(a) (b)

(c) DNS (d) HOSVD (e) TDB

Figure 12: Turbulent channel flow: (a) RMS, (b) mean streamvise velocity comparison between DNS and the same-rank HOSVD and TDB reductions. Comparison of DNS and the same-rank HOSVD and TDB reductions with $\overline{CR}$ = 26.9: (c) DNS , (d) reconstructed TDB, and (e) reconstructed HOSVD.

3.0 Blood Flow Prediction in Data-Poor Regimes

Blood flow reconstruction in the vasculature is important for many clinical applications. However, in clinical settings, the available data are often quite limited. For instance, Transcranial Doppler ultrasound (TCD) is a noninvasive clinical tool that is commonly used in the clinical settings to measure blood velocity waveform at several locations on brain's vasculature. This amount of data is grossly insufficient for training machine learning surrogate models, such as deep neural networks or Gaussian process regression. In this work, we propose a Gaussian process regression approach based on physics-informed kernels, enabling near-real-time reconstruction of blood flow in data-poor regimes. We introduce a novel methodology to reconstruct the kernel within the vascular network, which is a non-Euclidean space. The proposed kernel encodes both spatiotemporal and vessel-to-vessel correlations, thus enabling blood flow reconstruction in vessels that lack direct measurements. We demonstrate that any prediction made with the proposed kernel satisfies the conservation of mass principle. The kernel is constructed by running stochastic one-dimensional blood flow simulations, where the stochasticity captures the epistemic uncertainties, such as lack of knowledge about boundary conditions and uncertainties in vasculature geometries. We demonstrate the performance of the model on three test cases, namely, i) a simple Y-shaped bifurcation, ii) abdominal aorta, and iii) the Circle of Willis in the brain.

3.1 Methodology

3.1.1 Gaussian Process Regression

Since our work is based on the GP, we briefly review GP and we also introduce the notation that will be used in our methodology. Let $\mathbf{y} \coloneqq [y_1, y_2, \ldots, y_N]^T \in \mathbb{R}^N$ denote the observed data corresponding to inputs $\mathbf{x} \coloneqq [x_1, x_2, \ldots, x_N]^T \in \mathbb{R}^N$, where N is the number of

training points. The objective is to train a surrogate function $f(x)$ using the available data to perform predictions at any x. Using the GP framework, function $f(x)$ is approximated as follows [52]:

$$y = f(x) + \varepsilon, \quad f(\mathbf{x}) \sim \mathcal{GP}\left(0, k\left(x, x'; \boldsymbol{\theta}\right)\right), \tag{3.1.1}$$

where ε represents the noise with a zero-mean normal distribution $\varepsilon \sim \mathcal{N}(0, \sigma_n^2)$ where σ_n is the standard deviation, $k(x, x'; \boldsymbol{\theta})$ is the kernel function, and $\boldsymbol{\theta}$ is vector of hyperparameters.

Applying the GP model given by Eq. 3.1.1 to a discrete data points given by $(\mathbf{x}, \mathbf{y})$ results in the following normal distribution:

$$\mathbf{y} \sim \mathcal{N}(0, \mathbf{K} + \sigma_n^2 \mathbf{I}), \quad \mathbf{K} = k(\mathbf{x}, \mathbf{x}'; \boldsymbol{\theta}). \tag{3.1.2}$$

where $\mathbf{I} \in \mathbb{R}^{N \times N}$ is the identity matrix. The kernel hyperparameters and the noise variance σ_n^2 can be learned from the data by minimizing the negative logarithm of the marginal likelihood (NLML) given by:

$$NLML(\theta) = \frac{1}{2}\mathbf{y}^T(\mathbf{K} + \sigma_n^2\mathbf{I})^{-1}\mathbf{y} + \frac{1}{2}\ln|\mathbf{K} + \sigma_n^2\mathbf{I}| + \frac{N}{2}\ln(2\pi). \tag{3.1.3}$$

Therefore, using the computed hyperparameters and σ_n, allows prediction for a new input $f(x^*)$ using its conditional distribution as follows:

$$f\left(x^*\right) \mid \mathbf{y} \sim \mathcal{N}\left(k\left(x^*, \mathbf{x}\right)\left(\mathbf{K} + \sigma_n^2\mathbf{I}\right)^{-1}\mathbf{y}, k\left(x^*, x^*\right) - k\left(x^*, \mathbf{x}\right)\left(\mathbf{K} + \sigma_n^2\mathbf{I}\right)^{-1}k\left(\mathbf{x}, x^*\right)\right). \tag{3.1.4}$$

The posterior mean and variance are therefore:

$$\mu(x) = k\left(x, \mathbf{x}\right)\left(\mathbf{K} + \sigma_n^2\mathbf{I}\right)^{-1}\mathbf{y}, \tag{3.1.5a}$$

$$\sigma^2(x) = k\left(x, x\right) - k\left(x, \mathbf{x}\right)\left(\mathbf{K} + \sigma_n^2\mathbf{I}\right)^{-1}k\left(\mathbf{x}, x\right), \tag{3.1.5b}$$

where $\mu(x)$ represents the GP prediction and $\sigma(x)$ is the posterior standard deviation that quantifies uncertainty around predictions.

To address the limitations associated with the choice of kernel for a vasculature network, we propose an algorithm to construct physics-informed kernels using data generated from a

1D flow model. This kernel captures both spatiotemporal and vessel-to-vessel correlations.

3.1.2 A Global Spatiotemporal Model

We aim to build a surrogate model for the blood flow velocity in a vasculature network using only a few measurements. In the following, we adopt a *weight-space view* of linear Bayesian regression. We denote the quantity of interest at vessel (k) with $f^{(k)}(x,t) : \Omega^{(k)} \times [0,T] \to \mathbb{R}, k = 1,2,\ldots,K$ over the inputs (x,t), where $\Omega^{(k)}$ denotes the 1D spatial domain along the centerline of vessel k and T in the time duration and K is the number of vessels. The function f is the target quantity, which is the area-averaged blood flow velocity. Consider the following model for learning $f^{(k)}(x,t)$:

$$f^{(k)}(x,t) = \sum_{i=1}^{r} w_i^{(k)}(t)\phi_i^{(k)}(x), \quad x \in \Omega^{(k)}, \quad t \in [0,T], \quad k = 1,2,\ldots K, \tag{3.1.6}$$

where $\phi_i^{(k)}(x) : \Omega^{(k)} \to \mathbb{R}$ are *local* spatial basis functions that are chosen a priori and $w_i^{(k)}(t)$ are the time-dependent weights that must be learned from measurements. In this context, 'local' refers to basis functions that are defined within each vessel. As $r \to \infty$, the above model approximates a GP with a spatial kernel. In this model, either the weights have to be parameterized separately or they have to be learned at each instant of time. In that case, the model requires training K, GP models at any given time—one GP for each vessel. Consequently, for the vessels with insufficient data, the model's predictions are poor. Also, since the model must be trained at each instant of time, its predictions will be poor at time instants for which we have insufficient data. The latter issue can be circumvented by using a model with *spatiotemporal* basis functions as shown below:

$$f^{(k)}(x,t) = \sum_{i=1}^{r} w_i^{(k)}\phi_i^{(k)}(x,t), \quad x \in \Omega^{(k)}, \quad t \in [0,T], \quad k = 1,2,\ldots K, \tag{3.1.7}$$

where $\phi_i^{(k)}(x,t) : \Omega^{(k)} \times [0,T] \to \mathbb{R}$ are spatiotemporal local basis functions and $[w_1^{(k)}, w_2^{(k)}, \ldots, w_r^{(k)}]$ are the corresponding time-invariant weights. In the model given by Eq. 3.1.7, the spatiotemporal correlations are encoded in the basis function $\phi_i^{(k)}(x,t)$, and the weights can

be learned from disparate spatiotemporal measurements. As a result, training the model described by Eq. 3.1.7 requires a significantly smaller amount of data compared to the model outlined in Eq. 3.1.6. However, this model still utilizes local basis functions, and one model must be trained for each vessel. Therefore, for vessels with insufficient measurements, the model described by Eq. 3.1.7 can result in poor predictions.

In the following, we present a model based on *global* spatiotemporal basis functions that exploits spatiotemporal as well as vessel-to-vessel correlations. The model is described by:

$$f(x,t) = \sum_{i=1}^{r} w_i \phi_i(x,t), \quad x \in \Omega = \bigcup_{k=1}^{K} \Omega^{(k)}, \quad t \in [0,T], \tag{3.1.8}$$

where Ω denotes the global vasculature network, mathematically represented as the union of all vessels. In the above model, $\phi_i(x,t) : \Omega \times [0,T] \to \mathbb{R}$ are global spatiotemporal basis functions. Note that in the above model, the weights w_i are global coefficients; in other words, they are not vessel-dependent.

3.1.3 Uncertainty Modeling via Stochastic Simulations

The performance of the model given by Eq. 3.1.8 critically depends on the choice of the basis functions $\phi_i(x,t)$. The choice of the basis function in the weight-space view is closely related to choosing a kernel in the function-space view. Our approach uses stochastic simulations to construct the kernel, which has been employed in building empirical kernels; see, for example, [64, 65]. Previous developments have concentrated on Euclidean input spaces, i.e., the input space is a subset of $\mathbb{R}^d$. Moreover, these techniques require the explicit storage of the kernel matrix. The present work extends these developments to vasculature networks and addresses the challenge of explicit kernel storage, which is cost-prohibitive in terms of memory requirements for the current application.

In the following, we present a data-driven methodology to build global spatiotemporal basis functions using a stochastic 1D blood flow model. The steps are explained below.

Consider the 1D stochastic blood flow model expressed as:

$$\frac{\partial \mathbf{q}}{\partial t} = N(\mathbf{q}, t; \boldsymbol{\xi}), \tag{3.1.9}$$

augmented with appropriate boundary and initial conditions. In the equation above, $\mathbf{q} = \mathbf{q}(x, t; \boldsymbol{\xi})$ represents the state variable vector $\mathbf{q} = [u, A]$, where $u = u(x, t; \boldsymbol{\xi})$ denotes the area-averaged velocity in each vessel, $A = A(x, t; \boldsymbol{\xi})$ represents the cross-sectional area, and $\boldsymbol{\xi} = [\xi_1, \xi_2, \ldots, \xi_d]$ represents the random parameters. These parameters are defined based on *epistemic uncertainties* — uncertainties stemming from incomplete knowledge. For instance, exact inlet boundary conditions might be unknown in clinical settings. Similarly, the measurement of vessel areas, often derived from CT imaging, is subject to variability based on the image segmentation techniques employed. Another source of uncertainty in modeling 1D blood flow arises from the assumptions made about outflow boundary conditions. All these epistemic uncertainties are encapsulated within $\boldsymbol{\xi}$, with d representing the total number of these uncertain parameters.

Given the solution of the 1D model, the spatiotemporal global correlation operator can be formed as shown below:

$$\mathcal{K}(x, t, x', t') = \mathbb{E}[u(x, t; \boldsymbol{\xi}) u(x', t'; \boldsymbol{\xi})], \quad x, x' \in \Omega, \quad \text{and} \quad t, t' \in [0, T]. \tag{3.1.10}$$

where $\mathbb{E}[\sim]$ is the expectation operator defined as:

$$\mathbb{E}[u(x, t; \boldsymbol{\xi})] = \int_{\mathbb{R}^d} u(x, t; \boldsymbol{\xi}) \rho_u(\boldsymbol{\xi}) d\boldsymbol{\xi}, \tag{3.1.11}$$

where $\rho_u(\boldsymbol{\xi}) : \mathbb{R}^d \to \mathbb{R}$ is the joint probability density function. We consider the eigendecomposition of the above correlation operator as shown below:

$$\int_0^T \int_\Omega \mathcal{K}(x, t, x', t') \phi_i(x', t') dx' dt' = \lambda_i \phi_i(x, t), \quad i = 1, 2, \ldots, \infty, \qquad x \in \Omega, \quad \text{and} \quad t \in [0, T]. \tag{3.1.12}$$

Since $\mathcal{K}$ is a self-adjoint positive operator, its eigenvalues are positive ($\lambda_i = \sigma_i^2 \geq 0$) and its

eigenfunctions are orthonormal with respect to the space-time inner product as shown below:

$$\int_0^T \int_\Omega \phi_i(x,t)\phi_j(x,t)dxdt = \delta_{ij}, \quad i,j = 1,2,\ldots,\infty. \tag{3.1.13}$$

Note that in the above definition, the spatial integral is carried out over the entire vasculature. To make this clear, note that the spatial integral can be written as:

$$\int_\Omega \phi_i(x,t)dx = \sum_{k=1}^{K} \int_{\Omega^{(k)}} \phi_i(x,t)dx. \tag{3.1.14}$$

The correlation operator $\mathcal{K}$ is global and x and x' could belong to different vessels. Thus, $\mathcal{K}$ encodes vessel-to-vessel correlations. We choose $\mathcal{K}$ as the customized kernel function for building the regression model for blood flow properties. In the next section, we present a numerical algorithm to compute the spatiotemporal bases $\phi_i(x,t)$.

3.1.4 Kernel Construction

In this section, we present an efficient data-driven methodology to approximate $\mathcal{K}$. To this end, we express the kernel versus its spectral decomposition as follows:

$$\mathcal{K}(x,t,x',t') = \sum_{i=1}^{\infty} \lambda_i \phi_i(x,t)\phi_i(x',t'), \tag{3.1.15}$$

where the eigenvalues are sorted in a decreasing order, i.e., $\lambda_1 \geq \lambda_2 \geq \ldots$. In the following, we approximate the kernel in a finite-dimensional setting. To compute the eigenfunctions $\phi_i(x,t)$, we discretize the spatiotemporal domain. Let $\mathbf{x}^{(k)} = [x_1^{(k)}, x_2^{(k)}, \ldots, x_n^{(k)}]$ be the vector of discrete points in vessel (k). For simplicity in the notation, we consider the same number of discrete pints in each vessel. However, a different number of points could be considered for each vessel. Therefore, the global discrete spatial vector is simply the union of the local spatial vectors: $\mathbf{x} = \cup_{i=1}^{K} \mathbf{x}^{(k)}$, where $\mathbf{x} \in \mathbb{R}^{nK\times 1}$. The temporal domain is also discretized similarly: $\mathbf{t} = [t_1, t_2, \ldots, t_m]$, where m is the number of time steps. The discrete spatiotemporal domain is obtained via the tensor product of $\mathbf{x}$ and $\mathbf{t}$, with $N = nK \times m$ points. We reshape the resulting grid into a matrix $\mathbf{X} = [\underline{\mathbf{x}}|\underline{\mathbf{t}}]$, $\underline{\mathbf{x}} \in \mathbb{R}^{N\times 1}$, $\underline{\mathbf{t}} \in \mathbb{R}^{N\times 1}$, and

$\mathbf{X} \in \mathbb{R}^{N\times 2}$.

In the discrete form, each basis function $\phi_i(x,t)$ is represented by a vector $\boldsymbol{\phi}_i \in \mathbb{R}^{N\times 1}$, which contains the values of the basis function on the spatiotemporal grid. Similarly, the spatiotemporal solution of the physics-based model can be represented in the discrete form with vectors of size $\mathbf{u}_i \in \mathbb{R}^{N\times 1}$, where $\mathbf{u}_i = u(\underline{\mathbf{x}}, \underline{\mathbf{t}}; \boldsymbol{\xi}^{(i)})$, where $\boldsymbol{\xi}^{(i)}$ is a random realization of $\boldsymbol{\xi}$. To estimate the expectation operator, we use Monte Carlo sampling:

$$\mathbb{E}[u(\underline{\mathbf{x}}, \underline{\mathbf{t}}, \boldsymbol{\xi})] \approx \frac{1}{s}\sum_{i=1}^{s} u(\underline{\mathbf{x}}, \underline{\mathbf{t}}, \boldsymbol{\xi}^{(i)}) = \frac{1}{s}\mathbf{U}\mathbf{e}, \tag{3.1.16}$$

where s is the number of Monte Carlo samples, $\{\boldsymbol{\xi}^{(i)}\}_{i=1}^{s}$ are the Monte Carlo samples of the random vector $\boldsymbol{\xi}$, $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_s] \in \mathbb{R}^{N\times s}$, and $\mathbf{e} = [1, 1, \ldots, 1]^T \in \mathbb{R}^{N\times 1}$. Therefore, the kernel in the discrete is obtained by applying the above estimators for the expectation operator in Eq. 3.1.15, which results in:

$$\mathbf{K} = \frac{1}{s}\mathbf{U}\mathbf{U}^T, \tag{3.1.17}$$

where $\mathbf{K} \in \mathbb{R}^{N\times N}$. Typically, $n \sim \mathcal{O}(10^2)$, $m \sim \mathcal{O}(10^2)$ and $K \sim \mathcal{O}(10)$. As a result, $N \sim \mathcal{O}(10^5)$. Consequently, forming the kernel $\mathbf{K}$ and computing its eigendecomposition, which is $\mathcal{O}(N^3)$ are cost prohibitive. Instead, we use the standard procedure by which the eigenvectors of $\mathbf{K}$ are computed without forming $\mathbf{K}$ explicitly via the Singular Value Decomposition (SVD) of the matrix $\mathbf{U}$. As we show in our demonstration examples, the singular values decay quickly for blood flow simulations and as a result, the above decomposition can be truncated at $r < s$ basis functions

$$\mathbf{U} \approx \boldsymbol{\Phi}\boldsymbol{\Sigma}\mathbf{Y}^T, \tag{3.1.18}$$

where $\boldsymbol{\Phi} = [\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_r] \in \mathbb{R}^{N\times r}$ are the orthonormal left singular vectors, $\boldsymbol{\Sigma} \in \mathbb{R}^{r\times r}$ is the matrix of singular values, and $\mathbf{Y} \in \mathbb{R}^{s\times r}$ is the matrix of right singular vectors. The truncation rank (r) is determined based on the number of singular values required to approximate matrix $\mathbf{K}$ up to a desired accuracy. Using the above low-rank approximation of $\mathbf{U}$, the kernel can

be approximated as:

$$\hat{\mathbf{K}} = \frac{1}{s}\mathbf{\Phi\Sigma Y}^T\mathbf{Y\Sigma\Phi}^T = \mathbf{\Phi\Lambda\Phi}^T, \tag{3.1.19}$$

where $\hat{\mathbf{K}}$ is a rank-r approximation of $\mathbf{K}$. In the above, we use the orthonormality of the right singular vectors, i.e., $\mathbf{Y}^T\mathbf{Y} = \mathbf{I}$ and $\mathbf{\Lambda} = 1/s\mathbf{\Sigma}^2 = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_r)$. One of the key advantages of the above formulation is that kernel $\hat{\mathbf{K}}$ does not have to be formed nor stored explicitly. Instead, the matrix of the basis functions ($\mathbf{\Phi}$) and the matrix of eigenvalues ($\mathbf{\Lambda}$) are stored.

The above kernel is constructed as a discrete set of spatiotemporal points. However, the kernel needs to be evaluated at (x, t) that may not correspond exactly to one of the spatiotemporal points that $\mathbf{K}$ is built for. It is straightforward to construct *continuous* spatiotemporal kernel. To this end, we build an interpolant for the basis vectors $\boldsymbol{\phi}_i$. This can be done by reshaping $\boldsymbol{\phi}_i \in \mathbb{R}^{nm\times 1}$ into a matrix of size $n \times m$, which we denote with $[\boldsymbol{\phi}_i] \in \mathbb{R}^{n\times m}$. Let $\mathbf{\Psi}(x) = [\psi_1(x), \psi_2(x), \ldots, \psi_n(x)]$ be global basis functions that are the union of local basis function in each vessel and $\boldsymbol{\chi}(t) = [\chi_1(t), \chi_2(t), \ldots, \chi_m(t)]$ be the temporal basis functions. These basis functions are Lagrange interpolants such that $\psi_i(x_j^{(k)}) = \delta_{ij}$ and $\chi_i(t_j) = \delta_{ij}$. Both $\mathbf{\Psi}(x)$ and $\boldsymbol{\chi}(t)$ are *quasimatrices*, which are matrices whose columns are continuous and rows are discrete [117]. The continuous basis functions can be expressed as:

$$\hat{\phi}_i(x, t) = \mathbf{\Psi}(x)[\boldsymbol{\phi}_i]\boldsymbol{\chi}(t)^T. \tag{3.1.20}$$

In all the cases considered in this paper, we use piece-wise linear interpolants as the basis spatial and temporal basis functions. Therefore, the kernel in the continuous form is given by:

$$\hat{\mathcal{K}}(x, t, x', t') = \sum_{i=1}^{r} \lambda_i \hat{\phi}_i(x, t)\hat{\phi}_i(x', t'). \tag{3.1.21}$$

Using the above kernel, Equations 3.1.5a and 3.1.5b can be used to perform prediction at new unseen locations and times. To this end, let denote N space, time, and velocity labeled data (observational measurements) with $\mathbf{x} = [x_1, x_2, \ldots, x_N]$, $\mathbf{t} = [t_1, t_2, \ldots, t_N]$, and $\mathbf{u} = [u_1, u_2, \ldots, u_N]$, which means that u_i corresponds to measurement at spatiotemporal

coordinate of (x_i, t_i) for $i = 1, \ldots, N$. Then the GP predictions and posterior uncertainties are:

$$\mu(x,t) = \hat{\mathcal{K}}\left(x,t,\mathbf{x},\mathbf{t}\right)\left(\hat{\mathcal{K}}\left(\mathbf{x},\mathbf{t},\mathbf{x},\mathbf{t}\right) + \sigma_n^2\mathbf{I}\right)^{-1}\mathbf{u}, \tag{3.1.22a}$$

$$\sigma^2(x,t) = \hat{\mathcal{K}}\left(x,t,x,t\right) - \hat{\mathcal{K}}\left(x,t,\mathbf{x},\mathbf{t}\right)\left(\hat{\mathcal{K}}\left(\mathbf{x},\mathbf{t},\mathbf{x},\mathbf{t}\right) + \sigma_n^2\mathbf{I}\right)^{-1}\hat{\mathcal{K}}\left(\mathbf{x},\mathbf{t},x,t\right), \tag{3.1.22b}$$

where $\hat{\mathcal{K}}\left(\mathbf{x},\mathbf{t},\mathbf{x},\mathbf{t}\right) \in \mathbb{R}^{N\times N}$ and $\mathbf{I} \in \mathbb{R}^{N\times N}$ is the identity matrix. The variance of the noise (σ_n^2) can be determined by maximizing the negative likelihood. The presented methodology addresses all of the issues raised in the introduction and offers the following advantages:

1. The constructed kernel $\hat{\mathcal{K}}$ encodes both spatiotemporal and vessel-to-vessel correlations. This capability enables the reconstruction of blood flow velocity across the entire vasculature using a very limited number of measurements. For example, it allows for the estimation of blood flow in vessels that lack direct measurements.
2. Any reconstructed flow automatically satisfies the conservation of mass. To demonstrate this, note that the posterior mean from Eq. 3.1.22a can be expressed as a linear combination of the spatio-temporal basis functions, written as:

$$\mu(x,t) = \hat{\mathcal{K}}\left(x,t,\mathbf{x},\mathbf{t}\right)\mathbf{z} = \sum_{j=1}^{N}\sum_{i=1}^{r}\lambda_i\hat{\phi}_i(x,t)\hat{\phi}_i(x_j,t_j)y_j = \sum_{i=1}^{r} w_i\hat{\phi}_i(x,t), \tag{3.1.23}$$

   where $\mathbf{z} = [z_1, \ldots, z_N]^T = \left(\hat{\mathcal{K}}\left(\mathbf{x},\mathbf{t},\mathbf{x},\mathbf{t}\right) + \sigma_n^2\mathbf{I}\right)^{-1}\mathbf{u}$ and $w_i = \lambda_i \sum_{j=1}^{N}\hat{\phi}_i(x_j,t_j)z_j$. From Eq. 3.1.18, the basis functions are themselves a linear combination of the simulated data, i.e., $\hat{\phi}_i(x,t) = \sum_{j=1}^{s} u_j(x,t)Y_{ji}/\sigma_i$, where $u_j(x,t) = \mathbf{\Psi}(x)[\mathbf{u}_j]\boldsymbol{\chi}(t)^T$. Replacing this expression of the basis function into Eq. 3.1.23, results in:

$$\mu(x,t) = \sum_{j=1}^{s} a_j u_j(x,t), \tag{3.1.24}$$

   where $a_j = \sum_{i=1}^{r} w_i Y_{ji}/\sigma_i$. This shows that the posterior mean can be expressed as a linear combination of all s samples. Given that each sample $u_j(x,t)$ satisfies the conservation of mass, their linear combination also preserves the conservation of mass.
3. Since the number of observations is typically small, the computational cost of calculating the posterior means and uncertainty is negligible. The offline costs consist of running 1D

simulations and performing SVD on the generated data to create the kernel. For instance, on a system equipped with a 3.4 GHz Quad-Core Intel Core i5 processor, running each 1D simulation of an abdominal aorta vasculature network, which consists of 17 vessels, takes 28 seconds. A total of 150 simulations can be run in parallel. Creating the kernel requires 41 seconds. The simulation time for each sample scales roughly linearly with the number of vessels. The demonstration cases presented in this paper involve a relatively small number of vessels (less than 30). However, for vasculature networks with a larger number of vessels, recently developed low-rank approximation methods based on time-dependent bases may be employed to accelerate the simulation time [101].

3.1.5 One-dimensional Modeling

The governing equations for 1D simulations are the simplified version of the Navier-Stokes equations based on the following assumptions: i) The vessel's curvature is small and negligible; hence the given equations are written based on x coordinates on the centerline. ii) Vessels are axisymmetric. iii) The properties of each vessel are constant. These assumptions simplify the 3D incompressible Navier-Stokes equations which are written as:

$$\frac{\partial A}{\partial t} + \frac{\partial (Au)}{\partial x} = 0, \tag{3.1.25}$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \frac{fu}{A}, \text{where} \quad f = -22\mu\pi. \tag{3.1.26}$$

$\rho = 1050 kg/m^3$, μ, $A(x,t)$, $p(x,t)$, $f(x,t)$ are the blood density, blood dynamic viscosity, vessel cross-section area, pressure, and friction force per unit length respectively. Here, A, u, and p are unknown. In order to have a closed system of equation, we use pressure area relation.

$$p = p_{ext} + \beta(\sqrt{A} - \sqrt{A_0})) \tag{3.1.27}$$

with

$$\beta = \frac{\sqrt{\pi} hE}{(1-\nu^2)A_0} \tag{3.1.28}$$

where p_{ext}, h, E, A_0, and ν are external pressure, wall thickness, Young modulus, equilibrium cross-section area, and Poisson ratio $\nu = 0.5$ respectively. To compute 1D blood flow properties, characteristic decomposition is used for the given equations. This derivation is described in detail in Ref. [102] and solved by spectral/hp element spatial discretization and a second-order Lax-Wendroff time-integration scheme. This solver is called Nektar and we use it to solve s different simulations/samples with randomized inlet velocity, cross-section areas, and outflow boundary conditions.

3.1.6 Uncertainty Characterization

To generate the kernel as defined in Eq. 3.1.19, we calculate the blood flow properties using Eqs. 3.1.25, 3.1.26, and 3.1.27 for s number of samples. Below, we detail how the stochastic modeling accounts for the uncertainty in precise inlet and outlet boundary conditions.

The human arterial system consists of many vessels and in many cases, it is either impossible or unnecessary to simulate them all. The well-known three-element Windkessel model (RCR), allows us to simulate a part of the vasculature network and apply the effect of neglected vessels at the terminals. In this model, if we denote the total resistance by $R_t = R_1 + R_2$ and compliance by C, for the outlet:

$$p + R_2 C \frac{dp}{dt} - R_t Q - p_\infty - R_1 R_2 C \frac{dQ}{dt} = 0. \tag{3.1.29}$$

Here, $Q = Au$ is the flux at the outlet, p_∞ is the downstream pressure, and $R_1 = \frac{\rho c_0}{A_0}$ where $c_0 = \sqrt{\frac{\beta}{2\rho\sqrt{A_0}}}$ is the speed of wave propagation. In the Nektar solver, this model is adapted to the characteristic waves at the terminals [118].

We randomize the inlet velocity, outflow boundary conditions (R and C), and cross-section areas. For example Figure 13(b) shows the range of randomized inlet velocity for a Y-shaped vessel shown in Figure 13(a). In this figure, the periodic velocity with the periodic

time T is the summation of n_i component functions, each described as follows:

$$u(t;\boldsymbol{\xi}) = a_0 + \sum_{i=1}^{n_i} a_i \exp\left(-\frac{(t-[t/T]T-b_i)^2}{c_i}\right), \tag{3.1.30}$$

where $[z]$ is the rounded integer of z such that $0 \leq z - [z] < 1$, a_i, b_i, and c_i adjust peaks/valleys, their location, and their sharpness. We randomize the inlet profile by randomizing the parameters $[a_0, a_i, b_i, c_i]$ as shown below:

$$\xi_k = \overline{\xi}_k + \sigma_k \psi, \quad \text{where} \quad \xi_k \equiv [a_0, a_i, b_i, c_i], \quad i = 1, \ldots, n_i. \tag{3.1.31}$$

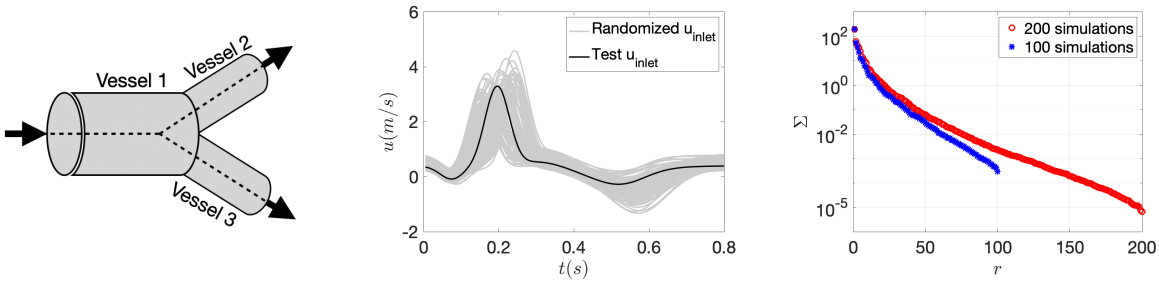
Here $\psi = \mathcal{U}[-0.5, 0.5]$ is a zero-mean uniform random variable with unit variance and σ_k is the standard deviation of variable $\sigma_k\psi$, ξ_k is the kth random variable and $\overline{\xi}_k = \mathbb{E}[\xi_k]$ is the mean of ξ_k. Therefore, the inlet randomization introduces $k = 1, \ldots 3n_i + 1$ random variables. We randomize the outflow parameters (R_t, C) for the vessels that have outflow boundary conditions as follows:

$$\xi_k = \overline{\xi}_k(1 + \sigma_k\psi), \quad \text{where} \quad \xi_k \equiv [R_t, C]. \tag{3.1.32}$$

Therefore, each vessel that has an outflow boundary condition introduces two more random variables that will be appended to the existing random variables. We also randomize the initial area $\xi_k = \{A_{0_i}\}, i = 1, 2, \ldots, K$ for each vessel in a form similar to Eq. 4.1.32. This will also add K (number of vessels) random variables. Using these parameters, we perform 1D simulations and store the resulting data in the matrix $\mathbf{U}$ (Eq. 3.1.18) and follow the algorithm presented in Section 3.1.4 to construct kernel $\hat{\mathcal{K}}$.

3.2 Demonstration Cases

In this section, we demonstrate the performance of the presented methodology in three cases: i) a Y-shaped vessel, where measurements are taken from 1D simulations; ii) the lower part of the abdominal aorta vasculature, where measurements are taken from 3D simulation; and iii) the brain vasculature (Circle of Willis), where measurements are taken from 4D



(a) Y-shaped bifurcation schematic (b) Inlet velocity (c) Singular values

Figure 13: a) Y-shaped vessel schematic. b) Y-shaped vessel random inlet velocities for all samples and the randomly selected sample as the measurement for prediction and validation. c) The dominant singular values remain unchanged across different numbers of simulations.

Flow MRI. In the following examples, we refer to the set of data points in the form of $(\mathbf{x}, \mathbf{t}, \mathbf{u})$, used in Eqs. 3.1.22a-3.1.22b, as *measurement*, and we refer to points at which we perform comparisons as *prediction*. In all cases, the kernel is constructed using stochastic 1D simulations. Once the kernel is constructed, we obtain measurement data, i.e., $(\mathbf{x}, \mathbf{t}, \mathbf{u})$, from various sources, including 1D simulation, 3D simulation, and 4D Flow MRI and we perform predictions using Eqs. 3.1.22a-3.1.22b.

3.2.1 Y-Shaped Vessel

In the first test case, we consider a Y-shaped vessel. The schematic of the problem is shown in Figure 13(a), which is similar to the first example in [47]. The equilibrium area, length, and Windkessel parameters are presented in Table 1. To create the kernel, we generate data using the 1D model (Eqs. 3.1.25-3.1.26), wherein the inlet velocity, cross-sectional areas, resistance, and compliance are randomized using Eqs. 4.1.31 and 4.1.32. For the inlet velocity (Figure 13(b)), $T = 0.8$, $n_i = 4$, and its randomized parameters values are listed in Table 2 along with R_t, C, and A_0. We generate $s = 200$ samples and collect their

simulation results. The results are stored for $n = 100$ equidistant points in each vessel and $m = 160$ equidistant time snapshots to create the kernel based on Eqs. 3.1.18 and 3.1.19. The rank is determined as the smallest r for which $\sum_{i=1}^{r} \sigma_i^2 / \sum_{i=1}^{s} \sigma_i^2 > 0.99$.

To perform prediction, we randomly select a new sample of the vector $\boldsymbol{\xi}$ and we denote this sample with $\boldsymbol{\xi}^*$. We perform the 1D simulation for $\boldsymbol{\xi} = \boldsymbol{\xi}^*$. We use the spatiotemporal result of this sample as the ground truth. As we explain below, we use only a small number of spatiotemporal data points generated by this simulation as the input data to the GP model. In particular, using the kernel created and a few measurements from the selected 1D simulation data, we perform predictions for all three vessels. A 5% Gaussian noise is added to the measurements to intentionally corrupt them with noise, mimicking real data. Figure 13(c) compares the singular values from two different numbers of simulations: $s = 200$ (our case) and $s = 100$. Based on this figure, the leading singular values of the two cases are very close to each other, implying that the number of samples is sufficient.

The GP model generates continuous spatiotemporal velocity predictions throughout the entire network. For comparison, we present the GP prediction results for only two representative points. Specifically, we compare the GP predictions at points 1 and 2, located in the middle of each corresponding vessel, as illustrated in Figure 14(a) and Figure 14(d) for the following two scenarios:

- Case 1 (low spatial resolution and high temporal resolution): The schematic of the spatiotemporal coordinates of the measurement points is illustrated in the x–t coordinate system in Figure 14(a). In particular, we utilize the inlet data from vessel 1 with a temporal resolution of $\Delta t = 0.005s$. The primary aim of this case is to address scenarios involving measurements with very low spatial resolution, in this instance, the extreme case of having only the time series data of one spatial point. According to Figures 14(b) and 14(c), the predicted velocity shows good agreement with the truth, although it exhibits significant uncertainty due to the insufficiency of measurements. It is noteworthy that in this case, despite no measurements being used in vessels 2 and 3, the GP model accurately predicts the velocity in those vessels. This is attributed to the fact that the

kernel is constructed globally which encodes the vessel-to-vessel correlations.

- Case 2 (low spatio-temporal resolution): In this scenario, we analyze a situation with low spatio-temporal resolution measurements, though not as extreme as Case 1 (Figure 14(d)). Specifically, we utilize the inlets of all vessels for measurements with $\Delta t = 0.01s$. As illustrated in Figures 14(e) and 14(f), predictions at points 1 and 2 are very close to the ground truth, and the uncertainties are significantly reduced since more measurements are used in comparison to Case 1.

3.2.2 Abdominal Aorta

In this case, we predict blood flow in the abdominal aorta and iliac branches, where the measurements (ground truth) are derived from 3D simulation. To extract the geometry of the problem (Figure 15(a)), we utilize time of flight (ToF) MRI data from the aortofemoral vasculature of a 67-year-old male subject. This data spans from the thoracic aorta to the femoral artery bifurcation and includes the thoracic, abdominal, renal, and femoral arteries (the model is available at www.vascularmodel.com). The voxel resolution in the right-left, anterior-posterior, and superior-inferior directions is 0.78 mm, 2.00 mm, and 0.78 mm, respectively. 3D simulation is conducted using direct numerical simulation (DNS) of the 3D Navier-Stokes equations, assuming the flow is Newtonian and incompressible, with the spectral/hp element method. Specifically, we employ tetrahedral elements with a polynomial order of $p = 3$. The generated mesh comprises $Ne = 119,720$ elements, and a third stiffly stable time advancement scheme is used with $\Delta t = 0.0001$ s. For more details on the spectral/hp element method, see [119].

To generate the kernel, we solve the 1D model for s simulation samples. To this end, we extract the vessels' centerlines and use them to create the 1D map of the abdominal aorta (shown in Figure 15(b)). Here, the inlet velocity is adapted from [120] for 3D simulation. To define this velocity profile in the form of Eq. 3.1.30 we set $T = 1$ and $n_i = 4$ and randomized parameter are listed in Table 3. The specifications of the problem are detailed in Table 4,

where the numbers of the vessels correspond with those in Figure 15(b). In this problem outflow and cross-section areas are not randomized. Since we use the incompressible form of Navier-stocks equations, the cross-section area does not change in time; therefore, in 1D simulation, β is set to a large value to maintain a $dA/dt \approx 0$. The base cross-sectional area, $A_0(x)$, is derived by fitting polynomials to MRI cross-sectional data and is the same for all 1D samples. Figure 15(c) shows the velocity inlet and its randomized profiles. We simulate $s = 150$ samples and create the kernel with $r = 41$ for which $\sum_{i=1}^{r} \sigma_i^2 / \sum_{i=1}^{s} \sigma_i^2 > 0.99$.

We demonstrate the accuracy of the GP model predictions across two scenarios: Case 1, where the measurements originate from 3D simulation, and Case 2, where the measurements are derived from 1D simulation. This specific setup aims to highlight the GP model's flexibility in adapting to various sources of input measurements. The locations of the measurements and predictions are depicted in Figure 15(a). In this figure, the measurements represent the cross-sectional-averaged velocity at the inlets of the vessels, and the predictions are made at the midpoint of selected vessels. The chosen model of the abdominal vasculature includes 17 vessels, yet only time-series measurements at two spatial locations are utilized for all GP predictions in this scenario. This setup serves as an example of blood flow reconstruction in a data-poor regime.

In Case 1, we utilize average velocity data from 3D simulation as measurements ($\Delta t = 0.001s$), whereas in Case 2, the measurements are derived from 1D simulation ($\Delta t = 0.005s$). In Figure 16, the predictions of the GP model for Case 1 and the ground truth, i.e., the cross-section-averaged velocity obtained from the 3D simulation, are shown. The GP model accurately predicts velocity in different vessels across various velocity ranges, despite no measurements being provided near point 3 and 4. This demonstrates that although the kernel is constructed with 1D simulations, it can effectively predict the results of 3D simulations, utilizing 3D time series data provided at only two locations.

In Case 2, the errors are significantly smaller compared to Case 1. This outcome is not surprising because, in Case 2, the ground truth is based on the 1D simulation results, and the kernel is constructed using 1D results as well. A comparison of the GP predictions

between Cases 1 and 2, for example, Figure 16(d) versus Figure 17(d), reveals differences in maximum velocities. Nonetheless, the GP model, based on the same physics-informed kernel, can accurately predict the velocity for both cases.

3.2.3 Circle of Willis Vasculature

We predict blood flow velocity in the Circle of Willis (CoW) vasculature based on MRI measurements provided in [48]. In this reference, MRI data includes *in vivo* blood flow velocity data across the CoW of a healthy male volunteer (age = 30 years, weight = 94 kg, height = 185 cm), which are collected as follows. First, we performed a ToF angiography scan, which is routinely performed in the clinic, is used to build the geometrical features of the CoW. In addition, we obtained a 4D Flow MRI scan, which provides 3D velocity time series maps in the entire CoW vasculature. The spatial resolution of 4D Flow MRI is $1.5 \times 1.26 \times 2\ mm^3$, and the temporal resolution is 42 msec over 14 cardiac phases. All the MRI scans were performed on a 3T Siemens Skyra. The 3D-ToF scan, representing the CoW architecture, is depicted in Figure 18(a). The vessel centerlines and branching pattern are shown in Figure 18(c).

In the processing of 4D Flow MRI images, we average the velocity data over a cross-sectional area to obtain 1D equivalent values. This is achieved by integrating the velocity components (x, y, and z) across each voxel in the imaging plane, and dividing by the total number of voxels in that plane. This spatial averaging procedure yields a representative velocity profile for the entire cross-section, effectively reducing the dimensionality of the data and enabling straightforward analysis and visualization of blood flow patterns. This analysis effectively produces similar values as TCD measurements at each specific cross section.

According to [48], the arterial geometry of the CoW is acquired by measuring 200 slices of 0.25 (mm) thickness with a 3D-TOF MRI sequence. The in-plane resolution was 0.4 (mm) ×0.4 (mm), the repetition time (TR) and the echo time (TE) were 18 (ms) and 3.57 (ms), respectively, and the flip angle was 15°. In the same scanning session, a prospectively ECG-gated 4D Flow MRI scan was performed using a 3.0 Tesla MRI scanner (Skyra, Siemens

Healthcare, Erlangen, Germany) and a 32-channel head coil. Scanning parameters were as follows: flip angle 7, repetition time/echo time 44.56/2.78 (ms), bandwidth 445 (Hz/pixel), velocity encoding (VENC) 90 cm/s, voxel size $1.5 \times 1.26 \times 2$ mm^3, temporal resolution 41.36 (ms) over 14 cardiac phases.

Before extracting velocity from 4D Flow MRI phase difference images, we perform image pre-processing including noise-masking, eddy-currents correction as well as phase unwrapping following the method explained in [121]. Noise masking was performed by thresholding of the signal intensity in the magnitude data to exclude regions with low signal intensity. Eddy current correction consisted of three steps: i) separation of static regions from blood flow, ii) fit a plane with least-squares method to the static regions from the last time frame (late diastole), iii) Subtraction of the fitted plane to the MRI data in every time frame. Finally, we correct the velocity aliasing effect, *i.e,* if adjacent pixels velocities in temporal or slice direction differ by more than VENC.

As described in [48], the 3D and 1D geometry of the vasculature is extracted from the 3D-TOF slices using SimVascular [37]. Where, the threshold for the basic segmentation was set to one-fourth of the maximum signal intensity value obtained in the measurements, which showed the best results for segmenting arterial structures. Everything above this value was considered to be an artery. Additional segmentation was performed manually by adding or removing voxels from the geometry based on anatomical knowledge.

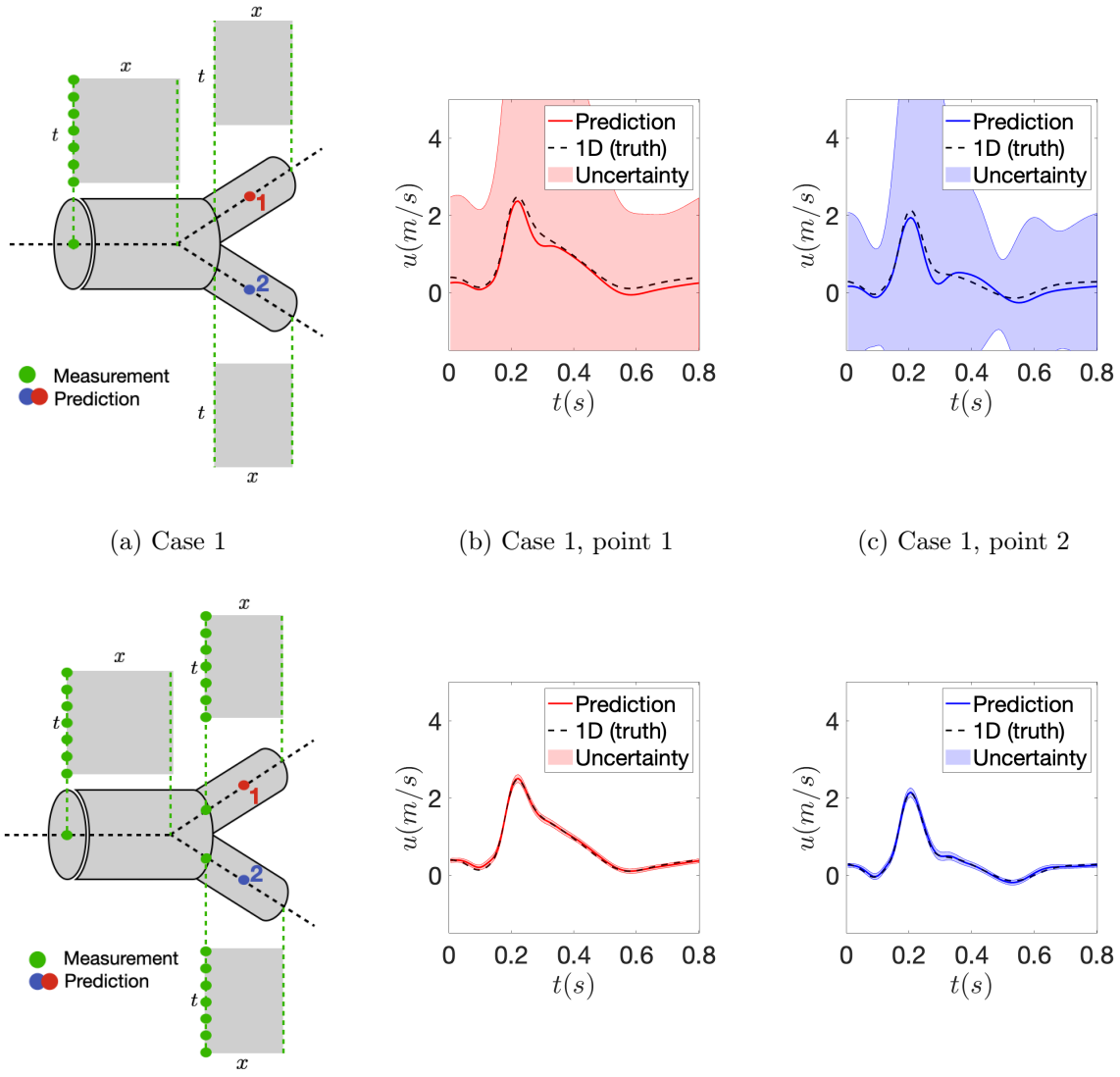
Here, we use 1D geometry and properties used in [48] to perform 1D simulations. This problem has four inlets where the direction of the blood flow is shown in Figure 18(c). The cardiac cycle is $T = 0.579s$ and each inlet is defined using Eq. 3.1.30 and randomized as listed in Table 8. The inlets are located at the same positions as measurement points 1, 2, 3, and 4.

We also randomize R_t and C using Eq. 4.1.32 where $\sigma_{R_t} = \sigma_C = 0.05$. $A_0(x)$ is determined by fitting polynomials to cross-sectional data obtained from MRI scans and we randomize the area using $\sigma_A = 0.5$. Using these parameters, we generate $s = 150$, 1D samples to create the data matrix (Eq. 3.1.18). We use $r = 36$ which captures more than 95% variance of the

data.

We examine predictions for two scenarios: In Case 1, we utilize the collected 4D Flow MRI data (average velocity) as measurements; while in Case 2, we use 1D simulation data taken at the same locations as measurements. Measurements locations are shown in Figure 18(c) where measurements points 1-4 are located at the inlets and measurement point 5 is at the middle of the vessel. Figure 19 compares the prediction with average velocity from 4D Flow MRI (labeled MRI for brevity) in Case 1. We also compare the GP predictions with 1D simulations for Case 2 as shown in Figure 20. In Case 2, where the measurement points are taken from 1D simulation, the accuracy is very high. However, in Case 1, where the ground truth is 4D Flow MRI, we observe around 20% inaccuracy. After closely inspecting the 4D Flow MRI data, it was realized that the 4D Flow MRI measurements violate the conservation of mass up to 27%. On the other hand, the GP predictions by construction satisfy the conservation of mass.

The posterior uncertainty in both Cases 1 and 2 is large (not shown in the figures) due to the lack of having very small amount of data compared to the size of the problem. In Figure 21, we show the convergence of the spatiotemporal averaged mean and standard deviation of the GP mean versus the number of measurement points in time and space. In the figures, the mean axis indicates the average mean across all prediction points, while the uncertainty percentage axis shows the average standard deviation to mean ratio for all prediction points. These results show that as more data become available the GP mean converges and the uncertainty reduces.



(a) Case 1 (b) Case 1, point 1 (c) Case 1, point 2

(d) Case 2 (e) Case 2, point 1 (f) Case 2, point 2

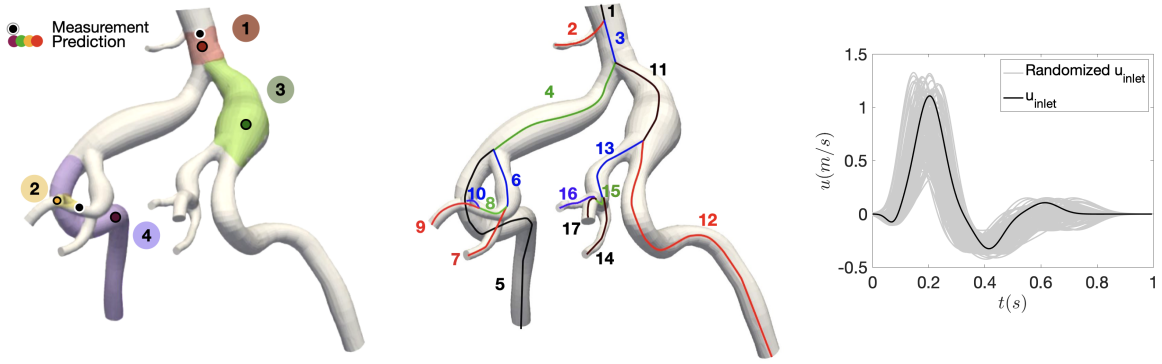
Figure 14: Schematic of the measurement positions. a) Case 1 with high temporal and low spatial resolution. The methodology provides predictions for all spatiotemporal locations, and we have chosen two specific prediction points for result comparison. b and c) Compare the predicted velocity at prediction points for Case 1 which has the most uncertainty due to the lack of nearby measurements. d) Case 2 with low spatiotemporal resolution. e and f) Compare the predicted velocity at prediction points for Case 2 which is improved compared with Case 1.

Table 1: Y-shaped vessel 1D simulation specification.

Vessel	Length(m)	A_0 (m^2)	Compliance($10^{-10}\frac{m^3}{Pa}$)	Total resistance($10^{10}\frac{Pa.s}{m^3}$)	$\beta(\frac{Pa}{m})$
1	0.1703	A_{0_1} = 1.36e-5	-	-	6.97e7
2	0.007	A_{0_2} =1.81e-6	C_1 =0.3428	R_{t_1} =1.19	5.42e8
3	0.00667	A_{0_3} =1.36e-5	C_2 =0.6661	R_{t_2} =0.2702	6.96e7

Table 2: Randomized parameters for the Y-shaped vessel network.

k	a_0	a_i	b_i	c_i	R_{t_i}	C_i	A_{0_i}
$\bar{\xi}_k$	0.5	[-0.5,3,-1,-0.1]	[0.08,0.2,0.4,0.6]	[2e-3,5e-3,1.5e-2,0.01]	[R_{t_1},R_{t_2}]	[C_1,C_2]	[A_{0_1},A_{0_2},A_{0_3}]
σ_k	0.5	[0,0.9,0.5,0.9]	[0.02,0.1,0.15,0.3]	[0,1e-3,1e-3,0]	[0.05,0.05]	[0.05,0.05]	[0.5,0.5,0.5]

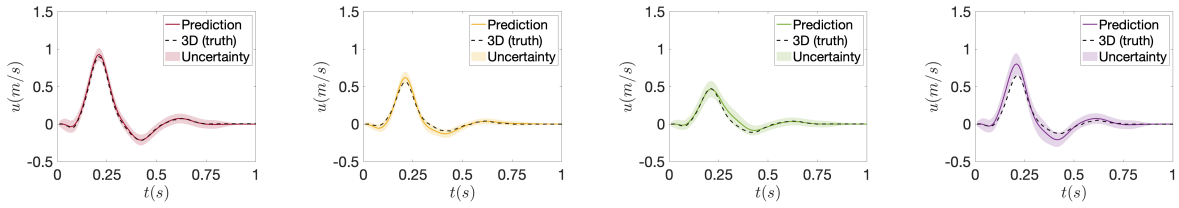


(a) Schematic (b) 1D map (c) Inlet velocity

Figure 15: a) Abdominal aorta schematic with the location of the velocity measurements and prediction. b) 1D map of the abdominal aorta, extracted from the vessel's centerline. c) Random inlet velocities and the equivalent inlet velocity for 3D simulation.

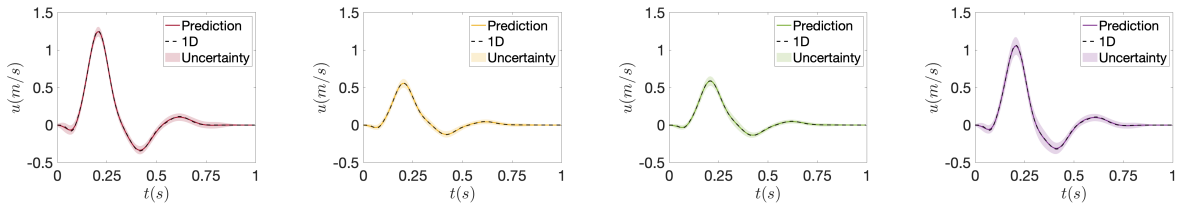
Table 3: Randomized parameters for abdominal aorta.

k	a_0	a_i	b_i	c_i	R_{t_i}	C_i	A_{0_i}
$\bar{\xi}_k$	0	[-0.1,0.9,-0.3,0.01]	[0.08,0.2,0.4,0.6]	[2e-3,5e-3,1.5e-2,0.01]	$[R_{t_1},...,R_{t_9}]$	$[C_1,...,C_9]$	$[A_{0_1},...,A_{0_{17}}]$
σ_k	1e-3	[0,0.9,0.5,0.9]	[0.02,0.1,0.15,0.3]	[0,1e-3,1e-3,0]	[0,...,0]	[0,...,0]	[0,...,0]



(a) Point 1, 3D (b) Point 2, 3D (c) Point 3, 3D (d) Point 4, 3D

Figure 16: Case 1: abdominal aorta prediction using 3D simulation as measurement. Predictions are compared with 3D simulation as the ground truth at a) point 1, b) point 2, c) point 3, and d) point 4. The location of these points and utilized measurements are indicated in Figure 15(a). In these comparisons, despite the kernel relying on 1D simulations, the GP predictions align with 3D simulations across various velocity ranges.

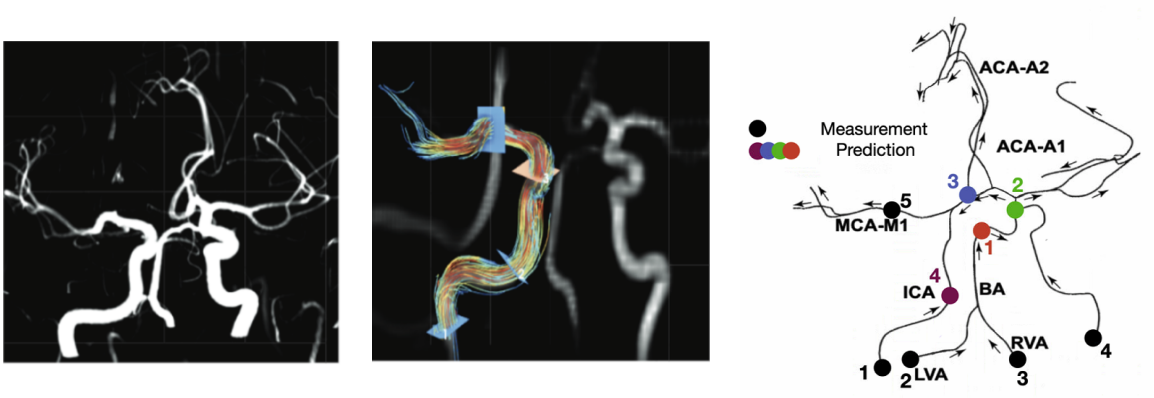


(a) Point 1, 1D (b) Point 2, 1D (c) Point 3, 1D (d) Point 4, 1D

Figure 17: Case 2: abdominal aorta prediction using 1D simulation as measurement. The location of these points and utilized measurements are the same as Case 1 and is indicated in Figure 15(a). Comparisons are made using 1D simulation as the ground truth at a) point 1, b) point 2, c) point 3, and d) point 4. In these comparisons, the GP predictions match with 1D simulations.

Table 4: Abdominal aorta vasculature 1D simulation specification.

Vessel	Length(m)	Compliance($10^{-10}\frac{m^3}{Pa}$)	Total resistance($10^{10}\frac{Pa.s}{m^3}$)	$\beta(\frac{Pa}{m})$
1	0.0346	-	-	1.083e9
2	0.0546	C_1 =0.0102	R_{t_1} =4.789	1.0522e10
3	0.0578	-	-	1.0396e10
4	0.077	-	-	3.2512e9
5	0.1701	C_2 =0.1931	R_{t_2} =0.6989	1.0396e10
6	0.0285	-	-	3.1355e9
7	0.0489	C_3 =0.0936	R_{t_3} =5.35	5.2091e10
8	0.0152	-	-	5.2163e10
9	0.0101	C_4 =0.0194	R_{t_4} =4.1823	1.0332e11
10	0.0217	C_5 =0.0147	R_{t_5} =3.2823	1.0274e10
11	0.0414	-	-	5.1574e9
12	0.1994	C_6 =0.0629	R_{t_6} =0.5974	6.277e9
13	0.0212	-	-	5.4503e9
14	0.0431	C_7 =0.011	R_{t_7} =6.1527	2.8957e10
15	0.0138	-	-	5.063e10
16	0.029	C_8 =0.0439	R_{t_8} =4.5299	3.063e10
17	0.0118	C_9 =0.0128	R_{t_9} =9.5299	5.063e10

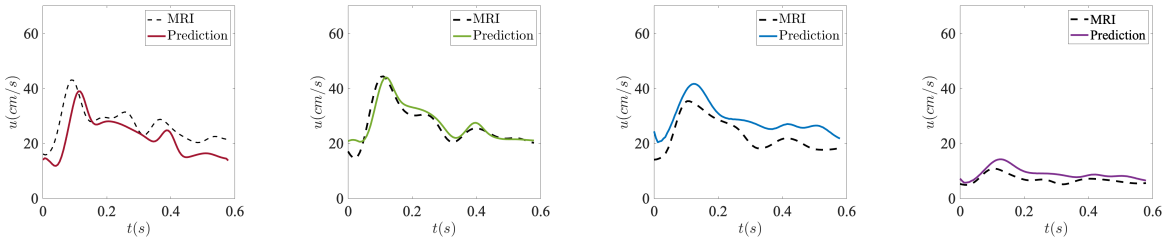


(a) 3D-ToF (b) 4D Flow MRI (c) Schematic and point locations

Figure 18: MRI data: a) 3D ToF MRI scan, representing the CoW architecture. b) 3D blood velocity streamlines at peak systolic phase through the left internal carotid artery acquired from processing the 4D Flow MRI images. c) The location of utilized measurements and predictions on extracted centerlines of the CoW arteries. Left and right internal carotid arteries are marked.

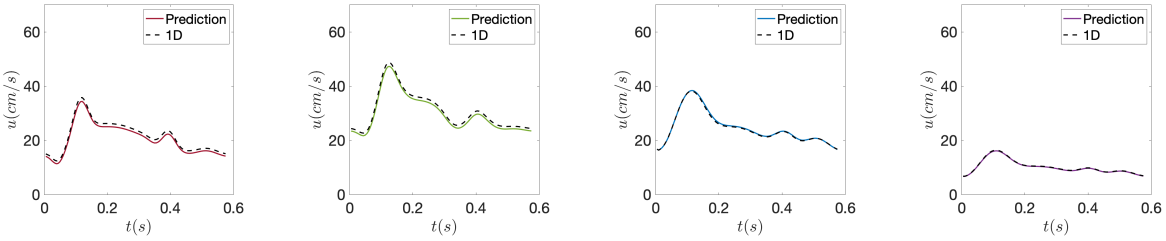
Table 5: Randomized parameters for CoW.

Inlet	j	k	a_0	a_i	b_i	c_i
1	6	$\overline{\xi}_k$	0.06	[0.08,3.6e-2,0.23,0.01,0.02,-8e-3]	[0.1,0.24,0.4,0.05,0.5,0.02]	[4e-3,1.5e-2,9e-3,1e-3,3e-3,1.4e-3]
		σ_k	0.08	0.08	0.05	1e-4
2,3	5	$\overline{\xi}_k$	0.08	[0.09,6.5e-2,0.04,1.2e-2,-0.03]	[0.1,0.2,0.36,0.5,0.03]	[1e-3,0.02,9e-4,2e-3,8e-4]
		σ_k	0.08	0.08	0.05	1e-4
4	5	$\overline{\xi}_k$	0.08	[0.06,0.03,1.4e-2,-0.02,-1.5e-2]	[0.08,0.2,0.4,0.6]	[2e-3,5e-3,1.5e-2,0.01]
		σ_k	0.08	0.08	0.05	1e-4



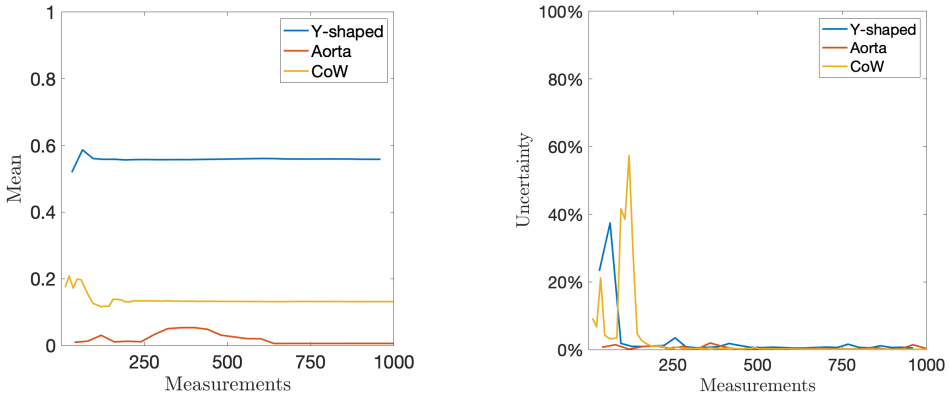
(a) Point 1, MRI (b) Point 2, MRI (c) Point 3, MRI (d) Point 4, MRI

Figure 19: Case 1: Prediction of the brain vasculature using MRI measurements. The location of these points are provided in Figure 18(c). Predictions are compared with 4D Flow MRI data at a) point 1, b) point 2, c) point 3, and d) point 4. The observed discrepancy is due to measurement errors related to mass conservation, whereas the GP model inherently enforces mass conservation in its predictions.



(a) Point 1, 1D (b) Point 2, 1D (c) Point 3, 1D (d) Point 4, 1D

Figure 20: Case 2: Prediction of the brain vasculature using 1D simulation data as measurements. Predictions are compared with 1D simulations at a) point 1, b) point 2, c) point 3, and d) point 4 which their locations are indicated in Figure 18(c). Based on these comparison, GP predictions are in good agreement with 1D simulations across different vessels.



(a) Mean convergence

(b) Uncertainty convergence

Figure 21: a) Average mean convergence vs number of utilized measurements. b) Average uncertainty percentage convergence vs number of utilized measurement.

4.0 Reduced Order Modeling of Stochastic Blood Flow Simulations

Blood flow within the vascular network is subject to various uncertainties, including physiological variability, which significantly impacts the accuracy of hemodynamics assessments. Numerical simulations provide a non-invasive way to quantify these dynamics, where physiological uncertainties often emerge from boundary conditions. To effectively address these uncertainties, a significant number of simulations become necessary, leading to a substantial increase in computational costs as the number of uncertain inputs grows. Therefore, the adoption of low-rank approximation techniques becomes crucial for reducing computational costs. In this work, we apply the time-dependent basis CUR (TDB-CUR) method to one-dimensional vasculature network blood flow simulations based on spectral/hp elements. This approach aims to reduce the computational expense of simulations and enhance the feasibility and efficiency of uncertainty quantification in blood flow simulations.

4.1 Methodology

4.1.1 Governing Equations

Assuming that we have a generic nonlinear stochastic PDE as follows:

$$\frac{\partial v}{\partial t} = \mathcal{F}(v; x, t, \boldsymbol{\xi}) + \mathcal{G}(v; x, t, \boldsymbol{\xi}) \tag{4.1.1}$$

where, $v(x, t; \boldsymbol{\xi})$ represents the parameter of interest at spatial coordinate x and time t, with $\boldsymbol{\xi} \in \mathbb{R}^d$ denoting a set of random parameters. The term $\mathcal{F}(v; x, t, \boldsymbol{\xi})$ refers to a nonlinear spatial differential operator, while $\mathcal{G}(v; x, t, \boldsymbol{\xi})$ represents the source term. We solve the aforementioned equations within a domain $\Omega = (a, b)$, which is partitioned into N_{el} elemental, non-overlapping regions $\Omega_e = (x_{l,e}, x_{u,e})$. Where $x_{u,e} = x_{l,e+1}$ for $e = 1, \ldots, N_{el}$, and $\bigcup_{e=1}^{N_{el}} \bar{\Omega}_e = \bar{\Omega}$. Using the Galerkin projection, we project Eq. 4.1.1 onto an arbitrary basis function ψ.

Therefore, the weak form discretization would be as follows:

$$\left(\frac{\partial v}{\partial t}, \psi\right)_\Omega + \left(\frac{\partial \mathcal{F}}{\partial x}, \psi\right)_\Omega - (\mathcal{G}, \psi)_\Omega = 0, \tag{4.1.2}$$

where, ψ is in Ω and the standard $L^2(\Omega)$ inner product is as follows:

$$(g, h)_\Omega = \int_\Omega gh \mathrm{d}x \tag{4.1.3}$$

In our case, the parameters of interest are the velocity $u(x, t; \boldsymbol{\xi})$ and the cross-sectional area $A(x, t; \boldsymbol{\xi})$. These are computed using the Navier-Stokes (NS) equations under the following simplifying assumptions:

1. The curvature of the vessel is neglected.
2. The vessels are assumed to be axisymmetric.

Based on these assumptions, we simplify the NS equations as follows:

$$\frac{\partial A}{\partial t} + \frac{\partial (Au)}{\partial x} = 0 \tag{4.1.4}$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + \frac{1}{\rho}\frac{\partial p}{\partial x} = \frac{K_R u}{A}, \text{where} \quad K_R = -22\mu\pi \tag{4.1.5}$$

In which $\rho = 1050\,\mathrm{kg/m^3}$, μ, $p(x, t, \boldsymbol{\xi})$, and K_R represent density, the dynamic viscosity of blood, the pressure, and the friction force per unit length, respectively. To establish a complete system of equations, a relationship between pressure and area is utilized.

$$p = p_{ext} + \beta(\sqrt{A} - \sqrt{A_0})) \tag{4.1.6}$$

with

$$\beta = \frac{\sqrt{\pi} hE}{(1 - \nu^2)A_0} \tag{4.1.7}$$

The variables p_{ext}, h, E, A_0, and ν, correspond to the external pressure, the thickness of the vessel wall, the Young's modulus, the area of the vessel's cross-section at equilibrium, and the Poisson's ratio (set at 0.5), respectively. Since the given system is categorized as a hyperbolic system, we use the characteristic method to solve it, which involves decomposing

the system into a set of simpler equations along characteristic curves where the solution is constant. The details are provided in Ref. [102]. We rewrite the Eqs 4.1.4 and 4.1.5 inform of the system of stochastic equation as follows:

$$\frac{\partial \mathbf{v}}{\partial t} + \frac{\partial \mathbf{f}(\mathbf{v})}{\partial x} = \mathbf{g}(\mathbf{v}), \tag{4.1.8}$$

where

$$\mathbf{v} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} = \begin{bmatrix} A \\ u \end{bmatrix}, \qquad \mathbf{f} = \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{bmatrix} = \begin{bmatrix} uA \\ \frac{u^2}{2} + \frac{p}{\rho} \end{bmatrix}, \quad \mathbf{g} = \begin{bmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ -K_R \frac{u}{A} \end{bmatrix}. \tag{4.1.9}$$

Substituting the above format into Eq. 4.1.2 and decomposing the integration into elemental regions:

$$\sum_{e=1}^{N_{\text{el}}} \left[\left(\frac{\partial \mathbf{v}}{\partial t}, \psi \right)_{\Omega_e} + \left(\frac{\partial \mathbf{f}}{\partial x}, \psi \right)_{\Omega_e} - (\mathbf{g}, \psi)_{\Omega_e} \right] = 0 \tag{4.1.10}$$

Performing integration by parts on nonlinear term leads to:

$$\sum_{e=1}^{N_{\text{el}}} \left\{ \left(\frac{\partial \mathbf{v}}{\partial t}, \psi \right)_{\Omega_e} - \left(\mathbf{f}, \frac{\mathrm{d}\psi}{\mathrm{d}x} \right)_{\Omega_e} + \left[\psi \cdot \mathbf{f} \right]_{x_e^l}^{x_e^u} - (\mathbf{g}, \psi)_{\Omega_e} \right\} = 0. \tag{4.1.11}$$

In the above equation, the term $\mathbf{v}$ is chosen from a finite-dimensional subspace of $L^2(\Omega)$, consisting of polynomials of degree p for individual elements. This subset is identified by the superscript δ. Note that $\mathbf{v}^\delta$ may be discontinuities at element boundaries. The propagation of information between elements takes place by upwinding the boundary flux ($\mathbf{f}^u$). Since computing $\left(\mathbf{f}, \frac{\mathrm{d}\psi}{\mathrm{d}x}\right)_{\Omega_e}$ is challenging, integration by part is performed again. Therefore, the above equation becomes as follows:

$$\sum_{e=1}^{N_e} \left\{ \left(\frac{\partial \mathbf{v}^\delta}{\partial t}, \psi^\delta \right)_{\Omega_e} + \left(\frac{\partial \mathbf{f}\left(\mathbf{v}^\delta\right)}{\partial x}, \psi^\delta \right)_{\Omega_e} + \left[\psi^\delta \cdot \left\{ \mathbf{f}^u - \mathbf{f}\left(\mathbf{v}^\delta\right) \right\} \right]_{x_e^l}^{x_e^u} - (\mathbf{g}, \psi)_{\Omega_e} \right\} = 0. \tag{4.1.12}$$

In this expression, $[\psi^\delta \cdot \{\mathbf{f}^u - \mathbf{f}\left(\mathbf{v}^\delta\right)\}]$ represents the difference between the upwinded flux and the local flux, and accounts for the propagation of information between elements. Choosing the Legendre polynomials $L_p(\zeta)$ as basis function the expanded solution for each element

becomes:

$$\mathbf{v}^{\delta}\big|_{\Omega_e}(x_e(\zeta),t;\xi)=\sum_{p=0}^{P}L_p(\zeta)\tilde{\mathbf{v}}_e^p(t;\xi),\text{where}\quad x_e(\zeta)=x_e^l\frac{(1-\zeta)}{2}+x_e^u\frac{(1+\zeta)}{2}. \tag{4.1.13}$$

Where, ζ is in modal space and maps physical space onto a reference element $\Omega_{st}=-1\leq\zeta\leq 1$. Substituting the solution expansion in the Eq. 4.1.12, discretization in modal space would become as follows:

$$\frac{d\tilde{\mathbf{v}}_{i,e}^p}{dt}=-\left(\frac{\partial\mathbf{f}_i}{\partial x},L_p\right)_{\Omega_e}-\frac{1}{J_e}\left[L_p\left[\mathbf{f}_i^u-\mathbf{f}_i\left(\mathbf{v}^{\delta}\right)\right]\right]_{x_e^l}^{x_e^u}+\frac{1}{J_e}\left(\mathbf{g}_i,L_p\right)_{\Omega_e},\quad p=1,\ldots,P,i=1,2. \tag{4.1.14}$$

Where $J_e=\frac{1}{2}(x_e^u-x_e^l)$. After the computation of right-hand side at each quadrature point, we transfer it back to physical space for time integration. Using Eq. 4.1.13 this transformation would become as follows:

$$\frac{d\mathbf{v}_i^{\delta}}{dt}=\sum_{p=0}^{P}L_p(\zeta)\frac{d\tilde{\mathbf{v}}_{i,e}^p}{dt}. \tag{4.1.15}$$

Using the matrix notation for Legendre polynomial with q quadrature points where $\mathbf{L}_p\in\mathbb{R}^{q\times P+1}$, $\mathbf{v}_i^{\delta}\in\mathbb{R}^q$, and $\tilde{\mathbf{v}}_{i,e}\in\mathbb{R}^{P+1}$:

$$\frac{d\mathbf{v}_i^{\delta}}{dt}=\mathbf{L}_p\frac{d\tilde{\mathbf{v}}_{i,e}}{dt},\text{where}\quad \mathbf{L}_p=\begin{bmatrix}L_0(\zeta_1) & L_1(\zeta_1) & \ldots & L_P(\zeta_1)\\ L_0(\zeta_2) & L_1(\zeta_2) & \ldots & L_P(\zeta_2)\\ \vdots & \vdots & \ddots & \vdots\\ L_0(\zeta_q) & L_1(\zeta_q) & \ldots & L_P(\zeta_q).\end{bmatrix} \tag{4.1.16}$$

The above equation describes the computation for a random sample in one element where $\mathbf{v}_i^{\delta}$ is a vector in physical space and $\tilde{\mathbf{v}}_{i,e}$ is a vector in modal space. When extending this notation to s random samples, the corresponding vectors are arranged into a matrix as follows:

$$\frac{d\mathbf{V}_i^{\delta}}{dt}=\mathbf{L}_p\frac{d\tilde{\mathbf{V}}_{i,e}}{dt} \tag{4.1.17}$$

$$\mathbf{V}_i^{\delta}=[\mathbf{v}_i^{\delta}(x_e(\zeta),t,\xi_1),...,\mathbf{v}_i^{\delta}(x_e(\zeta),t,\xi_s)],\quad \mathbf{V}_i^{\delta}\in\mathbb{R}^{q\times s}$$

$$\tilde{\mathbf{V}}_{i,e}=[\tilde{\mathbf{v}}_{i,e}(t,\xi_1),...,\tilde{\mathbf{v}}_{i,e}(t,\xi_s)],\quad \tilde{\mathbf{V}}_{i,e}\in\mathbb{R}^{(P+1)\times s}$$

In the equations mentioned above, $\mathbf{V}_i^\delta$ is a matrix where each column corresponds to a vector of $\mathbf{v}_i^\delta$ which represents a vector for a random sample in physical space, and similarly $\tilde{\mathbf{V}}_{i,e}$ is a matrix where each column corresponds to a vector of $\tilde{\mathbf{v}}_{i,e}$ which represents a random sample in modal space. The computational demand of this equation increases significantly as the number of random samples increase, particularly when the equation is applied to a vast network of vasculature, in which each segment containing at least a few elements. Therefore, to manage the computational complexity effectively, it becomes crucial to employ low-rank approximation techniques within such a system.

We represent the parameters (area for $i = 1$ and velocity for $i = 2$) across all quadrature points, elements, and vessels within a single matrix $\mathbf{V}_i \in \mathbb{R}^{n\times s}$, where $n = qN_{el}d$ denotes the total number of spatial points for d vessels. For instance, Figure 22 illustrates the structure of the velocity matrix, where the columns represent the samples and the rows correspond to the coordinates associated with their respective quadrature point, element, and vessel. In this figure, the subscript of u indicates the element number, and the superscript denotes the vessel number. If we rearrange the parameters in modal space in the same manner and denote the differentiation matrix with $\mathbf{M} = \mathbf{L}_p \frac{d\tilde{\mathbf{V}}_i}{dt}$, the Eq 4.1.17 temporal discretization would become as follows:

$$\mathbf{V}_i^k = \mathbf{V}_i^{k-1} + \Delta t \mathbf{M}_i. \tag{4.1.18}$$

In the above equation, the objective is low rank approximation of $\mathbf{V}_i$ to reduce computational costs.

4.1.2 Low-Rank Approximation

To perform a low-rank approximation of $\mathbf{V}_i$, we need to find a rank-r matrix $\hat{\mathbf{V}}_i$ such that:

$$\mathbf{V}_i = \hat{\mathbf{V}}_i + \mathbf{R}_i, \quad i = 1, 2. \tag{4.1.19}$$

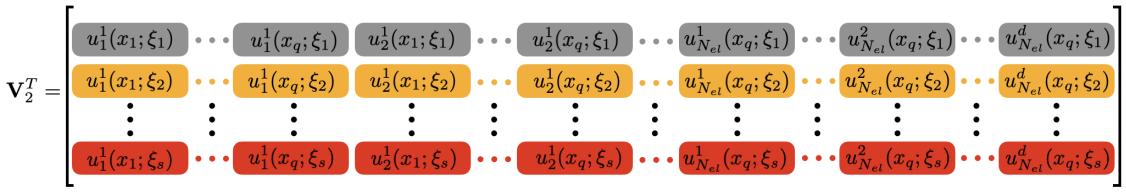


Figure 22: The structure of the velocity matrix is such that each row corresponds to a different sample, and each column corresponds to different coordinates associated with the respective quadrature point, element, and vessel. The structure of the area matrix is similar denoted by $\mathbf{V}_1^T$.

Where $\mathbf{R}_i$ is the low-rank approximation error and the best selection of $\hat{\mathbf{V}}_i$ minimizes Frobenius norm of the residual defined as follows:

$$\varepsilon_i = \left\| \mathbf{V}_i^k - \hat{\mathbf{V}}_i^{\,k} \right\|_F^2 . \tag{4.1.20}$$

To minimize the above expression, the best choice would be the rank-r truncated SVD:

$$\hat{\mathbf{V}}_{i_{best}}^k = \mathrm{SVD}(\mathbf{V}_i^k), \text{where} \quad \mathrm{SVD}\left(\mathbf{V}_i^k\right) = \mathbf{U}_{i_{\text{best}}}^k \, \mathbf{\Sigma}_{i_{\text{best}}}^k \, \mathbf{Y}_{i_{\text{best}}}^{k^T} \tag{4.1.21}$$

In this context, $\mathbf{U}_{i_{\text{best}}}^k \in \mathbb{R}^{n \times r}$ and $\mathbf{Y}_{i_{\text{best}}}^k \in \mathbb{R}^{s \times r}$ are the matrices of left and right singular vectors, respectively, each containing the first r columns, while $\mathbf{\Sigma}_{i_{\text{best}}}^k \in \mathbb{R}^{r \times r}$ contains the corresponding r singular values. Therefore, the best low-rank approximation $\hat{\mathbf{V}}_i^k$ is expressed as follows:

$$\hat{\mathbf{V}}_{i_{\text{best}}}^k = \mathbf{P}^{\llcorner}_{\mathbf{U}_{i_{\text{best}}}^k} \mathbf{V}_i^k \mathbf{P}^{\llcorner}_{\mathbf{Y}_{i_{\text{best}}}^k} = \mathbf{P}^{\llcorner}_{\mathbf{U}_{i_{\text{best}}}^k} \left(\hat{\mathbf{V}}_i^{\,k-1} + \Delta t \mathbf{M}_i \right) \mathbf{P}^{\llcorner}_{\mathbf{Y}_{i_{\text{best}}}^k} , \tag{4.1.22}$$

where $\mathbf{P}^{\llcorner}_{\mathbf{U}_{i_{\text{best}}}^k} = \mathbf{U}_{i_{\text{best}}}^k \mathbf{U}_{i_{\text{best}}}^{k^T}$ and $\mathbf{P}^{\llcorner}_{\mathbf{Y}_{i_{\text{best}}}^k} = \mathbf{Y}_{i_{\text{best}}}^k \mathbf{Y}_{i_{\text{best}}}^{k^T}$ represent the orthogonal projections onto the spaces spanned by the columns and rows of $\hat{\mathbf{V}}_i^k$ respectively. This type of approximation is computationally expensive to be performed per time step and alternatives are to evolve the basis and singular values (dynamical low-rank approximation, DBO). However, these models require the computation of inverse singular value matrices which are numeri-

cally unstable (see Ref. [101]). Therefore, instead of orthogonal projection we use developed oblique projection methodology by Ref. [101] as follows:

$$\hat{\mathbf{V}}_i^k = \mathbf{P}^{\swarrow}_{\mathbf{U}_i^k} \mathbf{V}_i^k \mathbf{P}^{\swarrow}_{\mathbf{Y}_i^k} = \mathbf{P}^{\swarrow}_{\mathbf{U}_i^k} \left(\hat{\mathbf{V}}_i^{k-1} + \Delta t \mathbf{M}_i \right) \mathbf{P}^{\swarrow}_{\mathbf{Y}_i^k}. \tag{4.1.23}$$

Despite using different projections, Eqs. 4.1.22 and 4.1.23 are equivalent. However, while Eq. 4.1.22 requires access to the entire matrix $\hat{\mathbf{V}}_i^{k-1} + \Delta t \mathbf{M}_i$, Eq. 4.1.23 only relies on certain rows and columns. Therefore, we approximate $\hat{\mathbf{V}}_i^k$ using CUR decomposition [122] as follows:

$$\hat{\mathbf{V}}_i^k = \mathrm{CUR}(\mathbf{V}_i^k). \tag{4.1.24}$$

In this scenario, the low-rank approximation residual $\mathbf{R}_i$ becomes zero for strategically selected rows $\mathbf{p}_i \in \mathbb{N}^r$ and columns $\mathbf{s}_i \in \mathbb{N}^r$, where $i = 1, 2$ corresponds to the rows and columns of the area and velocity matrices, respectively. This requires that $\hat{\mathbf{V}}_i^k(\mathbf{p}_i, :) = \mathbf{V}_i^k(\mathbf{p}_i, :)$ and $\hat{\mathbf{V}}_i^k(:, \mathbf{s}_i) = \mathbf{V}_i^k(:, \mathbf{s}_i)$. To identify the optimized rows and samples, we use the Discrete Empirical Interpolation Method (DEIM) [123], which provides near-optimal sampling points for the CUR decomposition algorithm [124]. The DEIM point calculation utilizes a rank-r SVD at time-step k. Since this time step is not computed we utilize the SVD of previous time step instead ($\hat{\mathbf{V}}_i^{k-1} = \mathbf{U}_i^{k-1} \mathbf{\Sigma}^{k-1} \mathbf{Y}_i^{k-1}$). The DEIM computation on the area and velocity matrices does not necessarily yield identical rows and columns. Therefore, since the computation of area and velocity is coupled, we need to perform the computation on the union of rows and columns ($\mathbf{p} = \bigcup_{i=1}^2 \mathbf{p}_i$ and $\mathbf{s} = \bigcup_{i=1}^2 \mathbf{s}_i$). The following steps are similar to procedures in Ref. [101] with minor adjustments for computing $\hat{\mathbf{V}}_i^k$.

(1) Compute optimal rows and columns indices, $\mathbf{p}_i \leftarrow \mathrm{DEIM}(\mathbf{U}_i^{k-1})$, and $\mathbf{s}_i \leftarrow \mathrm{DEIM}(\mathbf{Y}_i^{k-1})$.

(2) Compute $\mathbf{V}^k(\mathbf{p}, :)$ and $\mathbf{V}^k(:, \mathbf{s})$ for one time step according to Eq 4.1.18 where $\mathbf{p} = \bigcup_{i=1}^2 \mathbf{p}_i$ and $\mathbf{s} = \bigcup_{i=1}^2 \mathbf{s}_i$.

(3) Determine the orthonormal matrix $\mathbf{Q}_i \in \mathbb{R}^{n \times r}$ for area ($i = 1$) and velocity ($i = 2$), which spans the range of $\mathbf{V}_i^k(:, \mathbf{s}_i)$, through QR factorization so that $\mathbf{V}_i^k(:, \mathbf{s}_i)$ can be expressed as the product $\mathbf{Q}_i \mathbf{R}_i$, with $\mathbf{R}_i$ being a matrix in $\mathbb{R}^{r \times r}$.

(4) Project each column of $\mathbf{V}_i^k$ onto the orthonormal basis $\mathbf{Q}_i$ at sparse indices $\mathbf{p}_i$:

$$\mathbf{Z}_i = \mathbf{Q}_i(\mathbf{p}_i, :)^{-1}\mathbf{V}_i^k(\mathbf{p}_i, :) \tag{4.1.25}$$

In which, $\mathbf{Z}_i \in \mathbb{R}^{r\times s}$ consists of coefficients for interpolation, enabling the operation $\mathbf{Q}_i\mathbf{Z}_i$ to map $\mathbf{V}_i^k$ onto the orthonormal basis $\mathbf{Q}_i$ at the $\mathbf{p}_i$ points.

(5) Perform the SVD on $\mathbf{Z}_i$, resulting in:

$$\mathbf{Z}_i = \mathbf{U}_{\mathbf{Z}_i}\mathbf{\Sigma}_i^k{\mathbf{Y}_i^k}^T, \tag{4.1.26}$$

where $\mathbf{U}_{\mathbf{Z}_i} \in \mathbb{R}^{r\times r}$, $\mathbf{\Sigma}_i^k \in \mathbb{R}^{r\times r}$, and $\mathbf{Y}_i^k \in \mathbb{R}^{s\times r}$.

(6) Determine $\mathbf{U}_i^k \in \mathbb{R}^{n\times r}$ as the rotation within the subspace by calculating:

$$\mathbf{U}_i^k = \mathbf{Q}_i\mathbf{U}_{\mathbf{Z}_i}. \tag{4.1.27}$$

In the first step, the DEIM algorithm is provided by Ref. [123]. For rank adaptivity, we define a ratio of singular values as follows:

$$\gamma = \frac{\sigma_r^{k-1}}{\|\operatorname{diag}(\mathbf{\Sigma}^{k-1})\|_2}. \tag{4.1.28}$$

If the value exceeds a specified threshold, denoted by ε_{th}, we increase the rank to $r^k = r^{k-1}+1$. Conversely, if it does not exceed the threshold, we reduce the rank accordingly. This adjustment of the rank is aimed at achieving a balance between accuracy and computational efficiency, ensuring that the rank is neither unnecessarily high, which would increase computation time, nor too low, which might compromise the precision of the results.

4.1.3 Boundary Condition

In the utilized 1D model, boundary conditions are applied at junctions, bifurcations, and terminals. A comprehensive description can be found in Ref. [102]. In this model the terminals are modeled using a three-element Windkessel model (RCR), which enables the determination of outflow pressure using the total resistance, $R_T = R_1 + R_2$, and the

compliance, C.

$$p + R_2 C \frac{dp}{dt} = R_T Q + p_\infty + R_1 R_2 C \frac{dQ}{dt}. \tag{4.1.29}$$

Where, $Q = Au$ represents the flow rate, and p_∞ denotes the downstream pressure. The speed of information propagation is denoted by $c_0 = \sqrt{\frac{\beta}{2\rho\sqrt{A_0}}}$, and the first resistance R_1 is calculated as $R_1 = \frac{\rho c_0}{A_0}$. In the Nektar solver, this model is adapted to account for the characteristic waves at the terminals, as described in Ref. [102]

4.1.4 Uncertainty Modeling

In this paper we introduce randomness to both the inlet velocity and the outflow boundary conditions (denoted by R_T and C). The approach to modeling inlet velocity involves summing up components from n_i Gaussian functions, each defined as:

$$u(t;\boldsymbol{\xi}) = a_0 + \sum_{i=1}^{n_i} a_i \exp\left(-\frac{(t - [t/T]T - b_i)^2}{c_i}\right), \tag{4.1.30}$$

where $\lfloor z \rfloor$ represents the largest integer, ensuring $0 \leq z - \lfloor z \rfloor < 1$. , a_i, b_i, and c_i adjust peaks/valleys, their location, and their sharpness. The parameters a_i, b_i, and c_i are responsible for adjusting the magnitude, position, and width of the peaks and valleys, respectively. We introduce variability in the inlet velocity by altering the parameters $[a_0, a_i, b_i, c_i]$ as follows:

$$\xi_k = \overline{\xi}_k + \sigma_k \psi, \quad \text{where} \quad \xi_k \equiv [a_0, a_i, b_i, c_i], \quad i = 1, \ldots, n_i, \tag{4.1.31}$$

In this case, $\psi = \mathcal{U}[-0.5, 0.5]$ denotes a uniform random variable centered at zero, with σ_k representing the standard deviation of the expression $\sigma_k \psi$. The variable ξ_k is identified as the kth random variable, with $\overline{\xi}_k$ being its expected value, denoted by $\mathbb{E}[\xi_k]$. This process of randomizing the inlet leads to the introduction of a series of random variables, from $k = 1$ to $3n_i + 1$. The randomization is extended to the outflow parameters (R_T, C) for vessels subject

Table 6: Pipe, randomized parameters.

k	a_0	a_i	b_i	c_i	$R_T(\frac{Pa.s}{m^3})$	$C(10\frac{m^3}{Pa})$
$\overline{\xi}_k$	0.5	[-0.5,3,-1,-0.1]	[8e-2,0.2,0.4,0.6]	[2e-3,5e-3,1.5e-2,1e-2]	3.262317e9	7.481712e-11
σ_k	0.5	[0,0.9,0.5,0.9]	[0.02,0.1,0.15,0.3]	[0,1e-3,1e-3,0]	0	0

to outflow boundary conditions in a similar fashion.

$$\xi_k = \overline{\xi}_k(1 + \sigma_k\psi), \quad \text{where} \quad \xi_k \equiv [R_T, C]. \tag{4.1.32}$$

Therefore, each vessel that has an outflow boundary condition introduces two more random variables that will be appended to the existing random variables.

4.2 Demonstration Cases

In this section, the TDB-CUR methodology is applied to three scenarios: i) a simple pipe, which acts as a fundamental case to verify our approach; ii) a Y-Bifurcation, introducing an increased complexity with two additional vessels and the splitting of flow; and iii) the Circle of Willis (CoW) vasculature network, which presents a more complex system compared to the other cases.

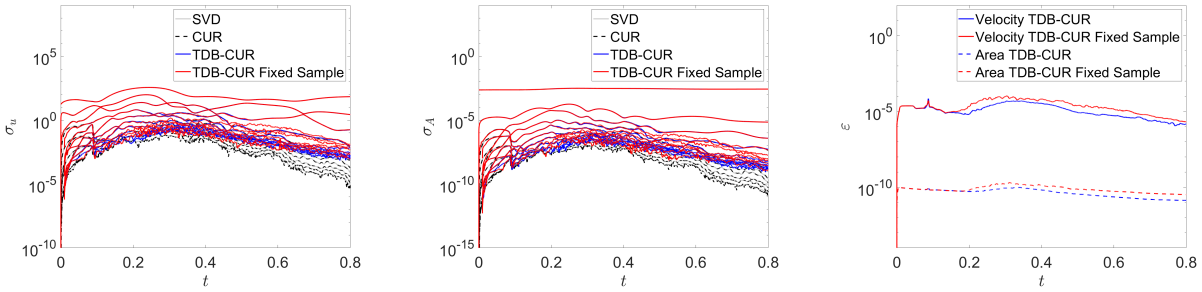
4.2.1 Pipe

In the first case, we examine the performance of TDB-CUR on a pipe that is equivalent to one vessel. We consider a pipe with a length of L = 1 m and a cross-sectional area of A = 1.35e-5m^2, which we discretize into N_{el} = 100 elements with q = 3 quadrature points. In this example, we randomize the inlet velocity using Eq 4.1.30, where T = 0.8s, n_i = 4, and

the randomized parameters are provided in Table 6. To implement TDB-CUR, we utilize $s = 10$ samples, $p = 25$, and a time step of $\Delta t = 1e\text{-}4s$. We examine the performance of this methodology in three cases. In Case 1, we aim to reconstruct the solution for $s = 100$, given that we are restricted to 10 samples (*i.e.* fixed columns), which limits TDB-CUR to strategic row selection. In Case 2, TDB-CUR is permitted to choose samples to minimize the reduction residual for the same samples. Case 3 expands the sample size to $s = 10,000$.

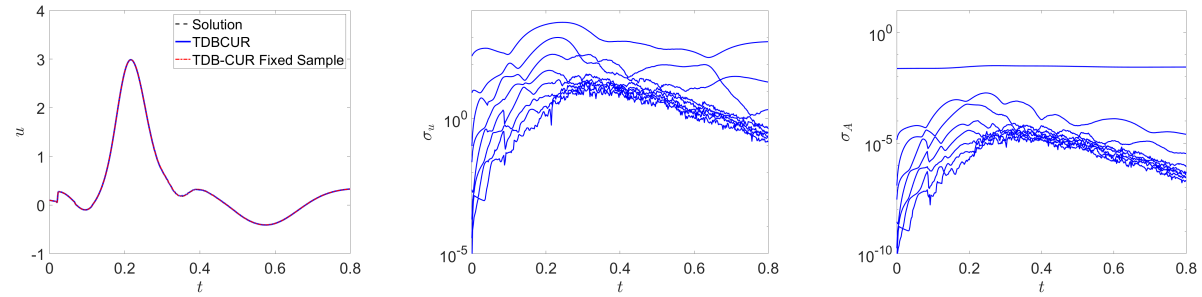
Figures 23(a) and 23(b) validate the evolution of singular values for velocity and cross-sectional area, respectively. In these figures, both CUR and SVD decomposition are performed on the FOM. A comparison of the singular values depicted in the referenced figures demonstrates their alignment with the results from the SVD decomposition. Figure 23(c) examines the Frobenius norm of the reconstruction error, revealing that TDB-CUR, when constrained to a fixed number of columns (equivalent to fixed samples), shows a slightly higher reconstruction error. Figure 24(a) presents the velocity reconstructions at $L/3$ for the 50th sample, which closely match the predictions from the FOM. Figures 26(c) and 24(c) explore the evolution of singular values for velocity and cross-sectional area, respectively, with an expanded sample size of $s = 10,000$. Since, FOM solution is time-consuming, we are unable to perform a comparison for the last case. However, the trend of the singular values appears similar, and this similarity suggests that they are valid.

The analysis of these figures clearly shows that TDB-CUR performs effectively in scenarios involving a single vessel, applicable to both fixed and flexible column configurations. The fixed column approach is particularly advantageous when the available data is restricted to a few samples. It enables the interpolation and reconstruction of a more dataset from these few initial samples, a feature that proves to be critical in situations where data acquisition faces practical limitations. On the other hand, scenarios that allow for flexible sampling has better accuracy. This improvement is due to the ability to select an optimized basis dynamically, which adapts to the underlying data structure and captures its essential features more effectively.



(a) Velocity singular values (b) Area singular values (c) Reconstruction error

Figure 23: Comparing singular values evolution from SVD and CUR decomposition with TDB-CUR with a fixed number of samples versus CUR-selected samples for a) velocity and b) cross-sectional area. c) Comparing the Frobenius norm for the reconstruction of velocity and cross-sectional area.



(a) Velocity reconstruction (b) Velocity singular values (c) Area singular values

Figure 24: a) Comparing TDB-CUR velocity reconstruction. TDB-CUR singular values evolution extended to $10,000$ samples for b) velocity and c) cross-section area.

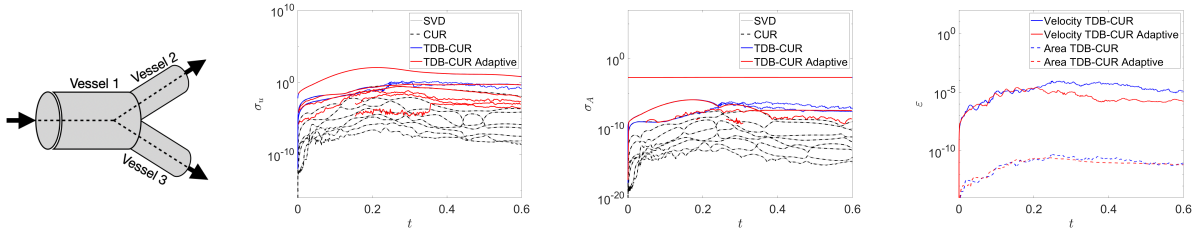
Table 7: Y-Bifurcation, randomized parameters.

k	a_0	a_i	b_i	c_i	$R_{T_i}(\frac{Pa.s}{m^3})$	$C_i(\frac{m^3}{Pa})$
$\bar{\xi}_k$	0	[1,0.2,0.1]	[0.2,0.45,0.5]	[5e-3,1.5e-2,0.01]	[1.189e10,2.701e10]	[3.451e-11,6.667e-11]
σ_k	0	[0.2,0,0]	0	0	0.2	0.2

4.2.2 Y-Bifurcation

In this section, we evaluate the performance of TDB-CUR on a Y-Bifurcation model. The key distinction from the pipe example lies in how the strategic global selection of columns and rows influences computational efficiency and accuracy. The schematic of the Y-Bifurcation is shown in Figure 25(a), where the length and cross-sectional area of each vessel are L = 0.5m and A = 1.3567e-5m^2, respectively. We discretize each vessel into N_{el} = 25 elements with q = 3 quadrature points and randomize the outflow boundary condition and the inlet velocity using Equation 4.1.30 with T = 0.6s and n_i = 3. The randomized parameters are listed in Table 7. We examine three cases: in the first, TDB-CUR maintains a fixed rank with s = 3 and p = 25; in the second, we explore the same setting with adaptive TDB-CUR, applying an error threshold of ε_{th} =1e-7; and in the last case, we perform the simulation with 10,000 samples.

Figures 25(b) and 25(c) compare the evolution of singular values. In these figures, CUR represents the resulting singular values from the FOM, which aligns with the resulting singular values from SVD. As the rank of the problem increases over time, adaptive TDB-CUR shows better accuracy compared with fixed-rank TDB-CUR. This is also evident from Figure 25(d), where the reconstructed data for adaptive TDB-CUR shows less error. This discrepancy appears as small noises in data reconstruction. For instance, Figure 26(a) shows the existence of noise due to the oscillation of the second and third singular values in fixed-rank TDB-CUR, while adaptive TDB-CUR has a smooth reconstruction. This comparison is for one of the random samples at $L/3$ of vessel 2. Figure 26(b) shows the rank adjustment in



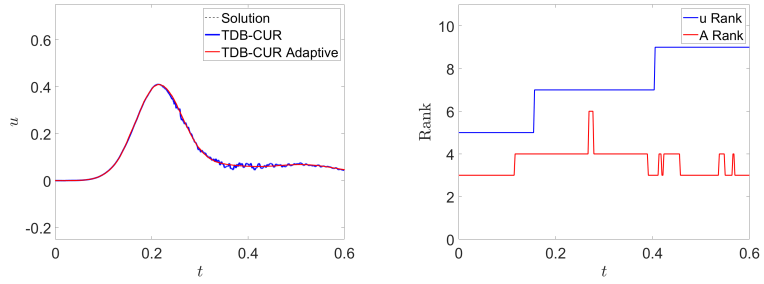
(a) Schematic (b) Velocity singular values (c) Area singular values (d) Reconstruction error

Figure 25: a) Y-Bifurcation schematic and blood flow direction. Comparison of fixed-rank TDB-CUR versus rank-adaptive TDB-CUR in the evolution of singular values for b) velocity and c) cross-sectional area. d) Comparing the Frobenius norm for the reconstruction of velocity and cross-sectional area.

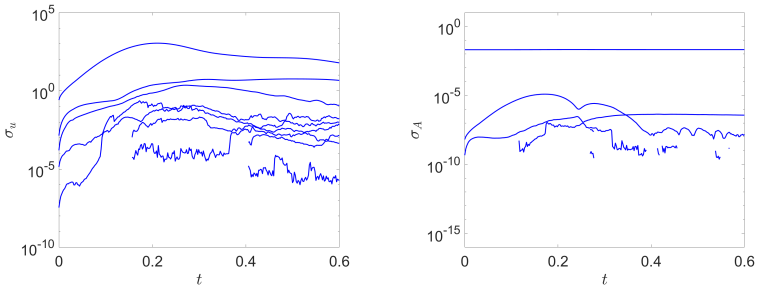
time, this figure indicates that velocity computation requires a higher rank compared with cross-section area computation. Comparing The mentioned cases suggests that the adaptive TDB-CUR has more accuracy due to rank adjustment capability and reduces computational costs by reducing $\mathbf{s}$ required columns, when the rank of the problem is low. Figures 26(c) and 26(d) show the evolution of the singular values computed by adaptive TDB-CUR for 10,000 samples, where the disconnection in lines are due to rank adjustments. The FOM solution in this case is time-consuming, and a comparison is not feasible. However, the similarities between the singular value evolutions suggest that the results are valid.

4.2.3 Circle of Willis

In this section, we examine the performance of the TDB-CUR algorithm applied to the vasculature network of the CoW, which consists of 23 vessels. The layout of this vascular network is adopted from Ref. [48], where the data were obtained from a healthy 30-year-old male volunteer, with a weight of 94 kg and a height of 185 cm. To gather the necessary information, a Time of Flight (ToF) angiography scan was employed. This imaging technique



(a) Velocity reconstruction (b) Rank adaptivity



(c) Velocity singular values (d) Area singular values

Figure 26: a) Velocity reconstruction comparison. b) TDB-CUR rank adjustments. Evolution of c) velocity and d) cross-section area singular values for $10,000$ samples.

is commonly used in clinical settings to delineate the geometric features of the CoW, providing a detailed map of its structure for analysis. The 1D representation of the vasculature was derived from 3D Time of Flight (3D-ToF) imaging slices utilizing the SimVascular [37]. In this process, the initial segmentation threshold was determined to be a quarter of the highest signal intensity observed in the data, a criterion that was found to be most effective for identifying arterial structures. Any signal intensity above this threshold was classified as arterial. To enhance the accuracy of the vascular model, further refinement of the segmentation was conducted manually. This involved the addition or subtraction of voxels within the geometric model, guided by detailed anatomical knowledge to ensure the representation closely matched the actual vascular anatomy. 1D representation of the CoW network, along

with the locations of the inlet velocities, is depicted in Figure 27(a). The inlet velocity is characterized by Eq 4.1.30, and it remains without any random variation. The duration of the cardiac cycle is established at $T = 0.6s$, and the velocity parameters are listed in Table 8.

In the context of this example, the focus is on randomizing only the outflow boundary conditions, which is done by Eq 4.1.32. Here, we set the randomness for R_T and C to be $\sigma_{R_T} = \sigma_C = 0.01$ respectively. Considering the network features nine distinct outflow points, and each outflow is described by two parameters, the total number of dimensions subjected to randomization in this scenario would be eighteen which leads to change in outflow pressure. The specification for 1D simulations are all available in Ref. [48].

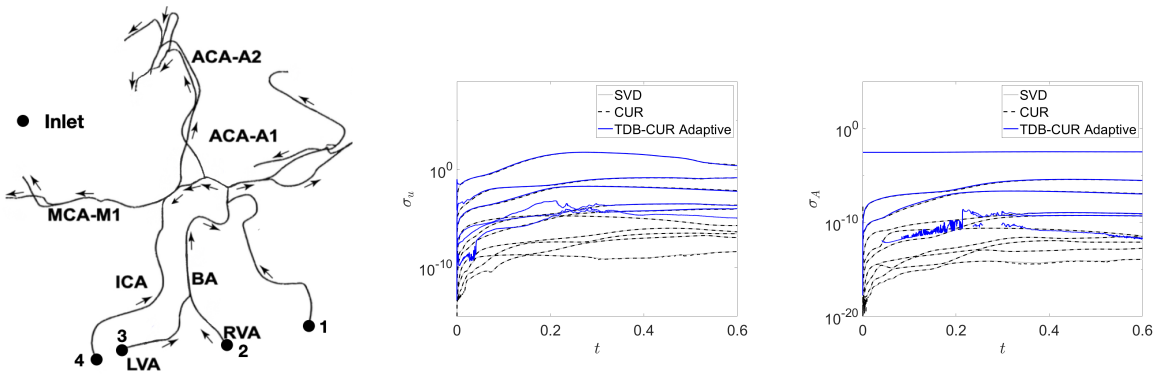
In this example we implement the adaptive TDB-CUR. Given that we have 23 vessels in CoW network, we discretize each vessel by $Nel = 5$ and $q = 3$. Since compared with other examples the overall number of spatial discretization is higher we set the oversampes $p = 35$. The threshold for rank adaptivity and the time step is set to be ε_{th} =1e-7, and Δt =5e-5s respectively. In order to study performance and validation we consider $s = 100$ and then extend the solution to $s = 10,000$.

Figures 27(b) and 27(c) illustrate the evolution of singular values obtained by TDB-CUR method, in comparison with those obtained from SVD and CUR decomposition, which serve as the ground truth. From these figures, it is evident that the singular values derived from the TDB-CUR method are consistent with those from both SVD and CUR decompositions. Despite some minor shifts in the singular values, the implementation of rank adaptivity effectively prevents these deviations from growing excessively and influencing the other singular values adversely. For further comparison, we show the reconstruction error in Figure 28(a) for area and velocity which indicates that TDB-CUR is simulating area and velocity accurately. Figure 28(b) shows an example of velocity reconstruction near inlet 4, which is compared with a 1D simulation. This comparison reveals that the reconstruction closely matches the 1D simulation, but some oscillations are observed in the reconstruction, which are attributed to oscillations in the singular values. Figures 27(b) through 28(b) shows the

Table 8: CoW, inlet parameters.

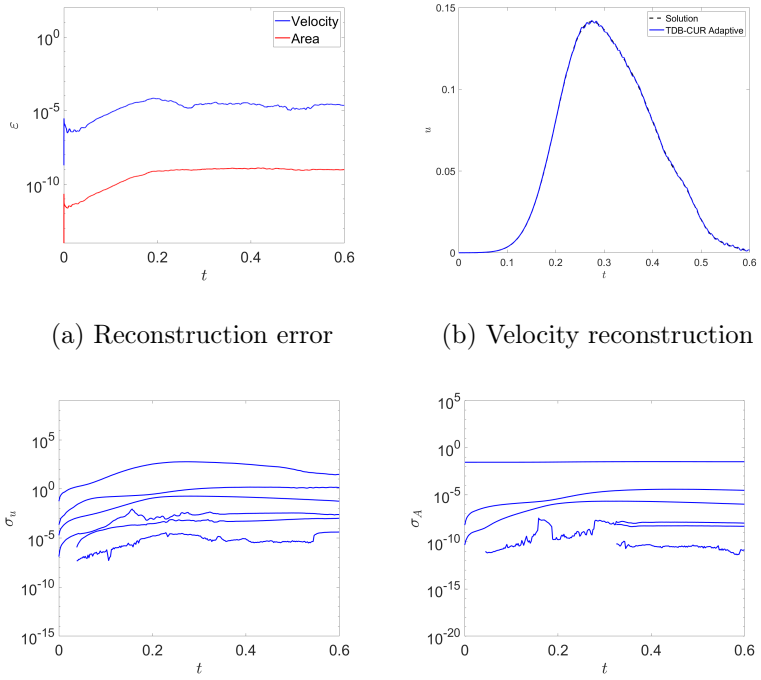
Inlet	j	k	a_0	a_i	b_i	c_i
1	4	$\overline{\xi}_k$	0	[0.15, 1.233e-1,1.97e-2, 5e-3]	[0.25,0.36,0.47,0.54]	[6.5e-3,9e-3,1.5e-3,2.5e-3]
2,3	4	$\overline{\xi}_k$	0	[2.25e-1,1.85e-1,2.95e-2,7.5e-3]	[0.25,0.36,0.47,0.54]	[6.5e-3,9e-3,1.5e-3,2.5e-3]
4	4	$\overline{\xi}_k$	0	[1.125e-1, 9.25e-2,1.475e-2, 3.75e-3]	[0.25,0.36,0.47,0.54]	[6.5e-3,9e-3,1.5e-3,2.5e-3]

performance and validation of the TDB-CUR method for a sample size of $s = 100$. To extend the investigation, we conducted simulations under the same conditions but with a significantly larger sample size, $s = 10,000$. The outcomes of these simulations are depicted in Figures 28(c) and 28(d), which illustrate the evolution of singular values obtained through the TDB-CUR method for this expanded dataset. Due to the computational demands of extracting singular values from the FOM, these figures do not include a comparison to the ground truth. Nonetheless, given the consistency of the problem settings, the singular values are expected to be similar to Figures 27(b) and 27(c), which is indeed the case. This consistency confirms the method's capacity to sustain its efficacy even as the volume of data escalates, highlighting its suitability for large-scale data applications.



(a) Schematic (b) Velocity singular values (c) Area singular values

Figure 27: a) CoW vasculature network schematic and blood flow direction with the location of inlets. b) Velocity and c) cross-sectional area singular value comparison with SVD and CUR decomposition singular values.



(a) Reconstruction error

(b) Velocity reconstruction

(c) Velocity singular values

(d) Area singular values

Figure 28: a) Comparing the Frobenius norm for the reconstruction of velocity and cross-sectional area. b) Velocity reconstruction near inlet 4 and its comparison with FOM. c) Velocity and d) cross-sectional area evolution for $s = 10,000$ samples.

5.0 Conclusions

In Chapter 2 We present an in situ compression method based on TDB, in which the multidimensional streaming data are decomposed into a set of TDB and a time-dependent core tensor. We derived closed-form evolution equations for the TDB and the core tensor. The presented methodology is adaptive and maintains the error bellow the defined threshold ε_{th} by adding/removing ranks. The computational cost of solving TDB computational complexity scales linearly with the data size, making it suitable for large-scale streaming datasets.

We perform this compression method on four cases:

1. Runge Function: Where we demonstrated the adaptive mode addition/removal. We also investigated the effect of different time integration schemes on the compression error of TDB.
2. Incompressible Turbulent Reactive Flow: In this case, we compress the data by two TDB schemes (TDB-1 and TDB-2). TDB-1 has a higher compression ratio ($\overline{CR}$ = 15.62, which compresses $35GB$ to $2.2GB$) since it can decompose the physical space more than TDB-2 using one dimensional basis. TDB-2 scheme has a lower compression ratio ($\overline{CR}$ = 6.6, which compresses $35GB$ to $5.3GB$) because it uses a two dimensional basis for physical space. Since the physical space has a high dimensionality compared to composition space, the error growth rate for TDB-1 is higher and requires frequent reinitialization and mode adjustments.
3. Incompressible turbulent reactive flow with random diffusion Coefficient: This problem is the same as the second case with sixteen samples random diffusion coefficients. The TDB scheme in this problem is similar to the TDB-2 with an additional two dimensional basis for the random space. Since the TDB can exploit more correlation in the streamed data, the compression ratio is higher than the previous cases ($\overline{CR}$ = 42.75, which compresses $561.5GB$ to $13.1GB$).

4. Three-dimensional Turbulent Channel Flow: In this problem, we compare two different compression ratios. We observed that the TDB with higher compression ratio has a slower growth rate compared to the TDB with lower compression ratio. For both cases, the TDB reconstructed results are in good agreement with HOSVD. However, the TDB requires a large number of modes to reconstruct the DNS with a reasonable agreement.

The current methodology only exploits multidimensional correlation in all dimensions except the temporal direction. For the future studies, we will extend this methodology to exploit temporal correlations. We will also pursue building real-time reduced order models for diagnostic and predictive purposes. We introduce a novel methodology that enables the spatiotemporal reconstruction of blood flow velocity within a vasculature network from a relatively small number of observations. This approach is particularly relevant for clinical scenarios such as blood flow reconstruction using TCD ultrasound measurements, which can only capture the blood flow velocity time series at a few points. Our methodology leverages GP regression and introduces a novel approach to construct a spatiotemporal kernel. This kernel is formulated using data generated from 1D blood flow simulations with a set of random parameters. These parameters encapsulate the epistemic uncertainties, such as the lack of precise knowledge about boundary conditions and the geometry of the vessels, for instance, the cross-sectional areas of the blood vessels. One significant feature of the constructed kernel is its encoding of vessel-to-vessel correlations, allowing measurements from one vessel to approximate velocities in others. Also, any prediction using the constructed kernel satisfies the conservation of mass.

In Chapter 3 we demonstrate the performance of low-rank approximation method in data-poor regime through three case studies: (i) a Y-shaped vessel, investigating the impact of measurement placement and their spatial and temporal resolution on accuracy and uncertainty bounds; (ii) abdominal aorta, utilizing a 3D model extracted from MRI scans with its simulation data as measurements. For comparison, we also examine a scenario where the 3D simulation data is replaced by 1D simulation data. In both scenarios, data from only two locations are used to accurately predict blood flow velocity in all 17 vessels; (iii) CoW,

employing 4D Flow MRI as the measurement data and exploring prediction results when 1D simulation data is used instead of MRI measurements.

To put things in perspective, the presented algorithm seeks to find a low-dimensional spatiotemporal function space to approximate velocity, i.e., $u(x,t) = \sum_{i=1}^{r} w_i \phi_i(x,t)$, as opposed to using fixed basis functions, i.e., $u(x,t) = \sum_{i=1}^{n_x} \sum_{j=1}^{n_t} w_{i,j} \psi_i(x) \chi_j(t)$, which requires inferring $\mathcal{O}(n_v n_x n_t)$ parameters, where n_v is the number of vessels. However, $r \ll n_v n_x n_t$, meaning that the presented methodology requires the inference of r parameters, and therefore, it requires a significantly smaller amount of data. PINN also seeks to find r adaptive spatiotemporal basis functions and approximate velocity with $u(x,t) = \sum_{i=1}^{r} w_i \phi_i(x,t;\theta)$. However, a PINN requires solving a nonconvex optimization problem to find θ and the w_i's, and it requires data on pressure, which is not clinically available. As a result, the computational cost of constructing the kernel is significantly lower than that of PINNs. Specifically, the offline (training) cost involves running $\mathcal{O}(100)$ 1D blood flow simulations, each taking only minutes for vasculature with $\mathcal{O}(100)$ vessels. Moreover, these simulations can run concurrently since they are independent of each other. The computation of the kernel, involving an SVD on the data generated by these simulations, costs about $\mathcal{O}(10)$ seconds and is negligible compared to the total cost. The online (inference) cost is minimal because the number of observations is typically very small, allowing for the methodology to be used for real-time flow reconstruction.

Both offline and online computational costs of the presented methodology scale linearly with the size of the problem, i.e., the number of vessels. More broadly, the presented methodology can be similarly developed and utilized for other network-type problems, such as fluid flows in pipe networks, traffic flow, or power grids.

In Chapter 4 we applied the TDB-CUR methodology to 1D simulations of blood flow. Our study evaluates the performance of this method across three examples: a simple pipe, a Y-shaped bifurcation, and the more complex cerebrovascular network CoW. Each example serves to illustrate the method's applicability and effectiveness in modeling blood flow dynamics within varying geometrical configurations and complexities.

In our initial example, we examined the performance of the TDB-CUR method when applied to stochastic blood flow within a single vessel, analogous to a pipe. We introduced significant variability to the inlet velocity, more so than in other cases. This example demonstrated that the TDB-CUR method could accurately simulate blood flow dynamics. We validated its performance using a set of 100 samples and subsequently expanded our analysis to include 10,000 samples. In this example, we explored the implications of constraining the TDB-CUR method to a fixed number of samples. We observed that while limiting the sample size is feasible, it can induce oscillations in the smaller singular values. These oscillations can adversely affect the larger singular values, leading to minor fluctuations and inaccuracies in the reconstructions. This finding suggests that in situations where only a limited number of samples are available, reconstructing the data for a more extensive set of samples may introduce some degree of error. Conversely, allowing for greater flexibility in the selection of samples enhances the accuracy of the reconstructions. This adaptability proves beneficial in situations where the intricate nature of blood flow patterns or the complex geometry of the vessel network demands a more refined strategy for choosing and analyzing samples. In the next example, we extend the use of the TDB-CUR method from a single vessel to a bifurcation, selecting rows and columns on a global scale. This means that while the selected rows and columns might not be the optimal choice for each individual vessel, they are chosen for their overall effectiveness across the entire network. Additionally, this example shows a comparison between the standard TDB-CUR method and its adaptive version. While the standard TDB-CUR method showed accurate results, it also exhibited certain fluctuations. In contrast, the adaptive TDB-CUR exhibited improved accuracy and was free from oscillations, offering a more stable solution. Employing TDB-CUR, particularly its adaptive form, proves to be both computationally efficient and precise. Finally, we applied the adaptive TDB-CUR methodology to the CoW network, consists of 23 vessels. In this comprehensive example, the adaptive TDB-CUR exhibited good accuracy, although it did exhibit some minor oscillations. These results shows the method's capability to accurately model blood flow within complex vascular networks. The presence of minor oscillations suggests areas for

further refinement, potentially enhancing the model's precision and stability in future applications. In all cases, after validating TDB-CUR results with 100 samples, we extend the solution to 10,000 samples. This extension shows method's scalability for a broader range of data. For the future study, we recommend extending this methodology to 3D simulations of blood flow, which could offer deeper insights into complex vascular behaviors and phenomena. Furthermore, integrating TDB-CUR with an emphasis on spatiotemporal factors could not only improve the accuracy of the solution but also offer greater methodological adaptability for analyzing larger random sample spaces.

Bibliography

[1] Jeffrey Slotnick, Abdollah Khodadoust, Juan Alonso, David Darmofal, William Gropp, Elizabeth Lurie, and Dimitri Mavriplis. CFD vision 2030 study: a path to revolutionary computational aerosciences. *National Aeronautics and Space Administration, Langley Research Center*, 2014.

[2] Earl P Duque, Scott T Imlay, Sean Ahern, Chen Guoning, and David L Kao. Nasa cfd vision 2030 visualization and knowledge extraction: Panel summary from aiaa aviation 2015 conference. In *54th AIAA Aerospace Sciences Meeting*, page 1927, 2016.

[3] Jack Dongarra, Jeffrey Hittinger, John Bell, Luis Chacon, Robert Falgout, Michael Heroux, Paul Hovland, Esmond Ng, Clayton Webster, and Stefan Wild. Applied mathematics research for exascale computing. Technical report, Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States), 2014.

[4] Data reduction for science. *Report of the DOE Workshop on Brochure from Advanced Scientific Computing Research Workshop*, 2021.

[5] J. C. Bennett, H. Abbasi, P. Bremer, R. Grout, A. Gyulassy, T. Jin, S. Klasky, H. Kolla, M. Parashar, V. Pascucci, P. Pebay, D. Thompson, H. Yu, F. Zhang, and J. Chen. Combining in-situ and in-transit processing to enable extreme-scale scientific analysis. In *SC '12: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, pages 1–9, 2012.

[6] Seung Woo Son, Zhengzhang Chen, William Hendrix, Ankit Agrawal, Wei-keng Liao, and Alok Choudhary. Data compression for the exascale computing era-survey. *Supercomputing frontiers and innovations*, 1(2):76–88, 2014.

[7] Peter Lindstrom and Martin Isenburg. Fast and efficient compression of floating-point data. *IEEE transactions on visualization and computer graphics*, 12(5):1245–1250, 2006.

[8] Nathaniel Fout and Kwan-Liu Ma. An adaptive prediction-based approach to lossless compression of floating-point volume data. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2295–2304, 2012.

[9] Hongwei Cheng, Zydrunas Gimbutas, Per-Gunnar Martinsson, and Vladimir Rokhlin. On the compression of low rank matrices. *SIAM Journal on Scientific Computing*, 26(4):1389–1404, 2005.

[10] Alec M Dunton, Lluís Jofre, Gianluca Iaccarino, and Alireza Doostan. Pass-efficient methods for compression of high-dimensional turbulent flow data. *Journal of Computational Physics*, 423:109704, 2020.

[11] Charles W Therrien. *Discrete random signals and statistical signal processing*. Prentice Hall PTR, 1992.

[12] Joel A Tropp, Alp Yurtsever, Madeleine Udell, and Volkan Cevher. Practical sketching algorithms for low-rank matrix approximation. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1454–1485, 2017.

[13] H. Kolla, K. Aditya, and J. H. Chen. *Higher Order Tensors for DNS Data Analysis and Compression*, pages 109–134. Springer International Publishing, Cham, 2020.

[14] Guoxu Zhou, Andrzej Cichocki, and Shengli Xie. Decomposition of big tensors with low multilinear rank, 2014.

[15] Kevin T Carlberg, Antony Jameson, Mykel J Kochenderfer, Jeremy Morton, Liqian Peng, and Freddie D Witherden. Recovering missing cfd data for high-order discretizations using deep neural networks and dynamics learning. *Journal of Computational Physics*, 395:105–124, 2019.

[16] Andrew Glaws, Ryan King, and Michael Sprague. Deep learning for in situ data compression of large turbulent flow simulations. *Physical Review Fluids*, 5(11):114602, 2020.

[17] Yang Liu, Yueqing Wang, Liang Deng, Fang Wang, Fang Liu, Yutong Lu, and Sikun Li. A novel in situ compression method for cfd data based on generative adversarial network. *Journal of Visualization*, 22(1):95–108, 2019.

[18] Zhenhuan Gong, Terry Rogers, John Jenkins, Hemanth Kolla, Stephane Ethier, Jackie Chen, Robert Ross, Scott Klasky, and Nagiza F Samatova. Mloc: Multi-level layout optimization framework for compressed scientific data exploration with heterogeneous

access patterns. In *2012 41st International Conference on Parallel Processing*, pages 239–248. IEEE, 2012.

[19] 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.

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

[21] Mulin Cheng, Thomas Y Hou, and Zhiwen Zhang. A dynamically bi-orthogonal method for time-dependent stochastic partial differential equations i: Derivation and algorithms. *Journal of Computational Physics*, 242:843–868, 2013.

[22] 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.

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

[24] 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.

[25] 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.

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

[27] 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.

[28] Michael Donello, Mark Carpenter, and Hessam Babaee. Computing sensitivities in evolutionary systems: A real-time reduced order modeling strategy. *arXiv preprint arXiv:2012.14028*, 2020.

[29] 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.

[30] 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.

[31] Claude Bardos, François Golse, Alex D. Gottlieb, and Norbert J. Mauser. Mean field dynamics of fermions and the time-dependent Hartree–Fock equation. *Journal de Mathématiques Pures et Appliquées*, 82(6):665–683, 2003.

[32] O. Koch and C. Lubich. Dynamical tensor approximation. *SIAM Journal on Matrix Analysis and Applications*, 31(5):2360–2375, 2017/04/02 2010.

[33] A. Dektor and D. Venturi. Dynamically orthogonal tensor methods for high-dimensional nonlinear pdes. *Journal of Computational Physics*, 404:109125, 2020.

[34] KL Ruedinger, S Schafer, MA Speidel, and CM Strother. 4d-dsa: development and current neurovascular applications. *American Journal of Neuroradiology*, 42(2):214–220, 2021.

[35] Aditi Deshpande, Nima Jamilpour, Bin Jiang, Patrik Michel, Ashraf Eskandari, Chelsea Kidwell, Max Wintermark, and Kaveh Laksari. Automatic segmentation, feature extraction and comparison of healthy and stroke cerebral vasculature. *NeuroImage: Clinical*, 30:102573, 2021.

[36] Aditi Deshpande, Jordan Elliott, Nitya Kari, Bin Jiang, Patrik Michel, Nima Toosizadeh, Pouya Tahsili Fahadan, Chelsea Kidwell, Max Wintermark, and Kaveh Laksari. Novel imaging markers for altered cerebrovascular morphology in aging, stroke, and alzheimer's disease. *Journal of Neuroimaging*, 32(5):956–967, 2022.

[37] Adam Updegrove, Nathan M Wilson, Jameson Merkow, Hongzhi Lan, Alison L Marsden, and Shawn C Shadden. Simvascular: an open source pipeline for cardiovascular simulation. *Annals of biomedical engineering*, 45(3):525–541, 2017.

[38] Jaques S Milner, Jennifer A Moore, Brian K Rutt, and David A Steinman. Hemodynamics of human carotid artery bifurcations: computational studies with models reconstructed from magnetic resonance imaging of normal subjects. *Journal of vascular surgery*, 28(1):143–156, 1998.

[39] JA Moore, DAs Steinman, DW Holdsworth, and CR Ethier. Accuracy of computational hemodynamics in complex arterial geometries reconstructed from magnetic resonance imaging. *Annals of biomedical engineering*, 27(1):32–41, 1999.

[40] Michael MacRaild, Ali Sarrami-Foroushani, Toni Lassila, and Alejandro F Frangi. Accelerated simulation methodologies for computational vascular flow modelling. *Journal of the Royal Society Interface*, 21(211):20230565, 2024.

[41] N Stergiopulos, DF Young, and TR Rogge. Computer simulation of arterial flow with applications to arterial and aortic stenoses. *Journal of biomechanics*, 25(12):1477–1488, 1992.

[42] Nico Westerhof, Jan-Willem Lankhaar, and Berend E Westerhof. The arterial windkessel. *Medical & biological engineering & computing*, 47(2):131–141, 2009.

[43] Bo-Wen Lin. Characteristic outflow boundary conditions for simulations of one-dimensional hemodynamics, 2016.

[44] Niema M Pahlevan, Faisal Amlani, M Hossein Gorji, Fazle Hussain, and Morteza Gharib. A physiologically relevant, simple outflow boundary model for truncated vasculature. *Annals of biomedical engineering*, 39(5):1470–1481, 2011.

[45] Charles A Taylor, Timothy A Fonte, and James K Min. Computational fluid dynamics applied to cardiac computed tomography for noninvasive quantification of fractional flow reserve: scientific basis. *Journal of the American College of Cardiology*, 61(22):2233–2241, 2013.

[46] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems

involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[47] Georgios Kissas, Yibo Yang, Eileen Hwuang, Walter R Witschey, John A Detre, and Paris Perdikaris. Machine learning in cardiovascular flows modeling: Predicting arterial blood pressure from non-invasive 4d flow mri data using physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 358:112623, 2020.

[48] Mohammad Sarabian, Hessam Babaee, and Kaveh Laksari. Physics-informed neural networks for brain hemodynamic predictions using medical imaging. *IEEE Transactions on Medical Imaging*, 2022.

[49] Amirhossein Arzani, Jian-Xun Wang, and Roshan M D'Souza. Uncovering near-wall blood flow from sparse data with physics-informed neural networks. *Physics of Fluids*, 33(7):071905, 2021.

[50] Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030, 2020.

[51] Luis García-Terriza, José L Risco-Martín, Gemma Reig Roselló, and José L Ayala. Predictive and diagnosis models of stroke from hemodynamic signal monitoring. *Medical & Biological Engineering & Computing*, 59(6):1325–1337, 2021.

[52] Carl Edward Rasmussen, Christopher KI Williams, et al. *Gaussian processes for machine learning*, volume 1. Springer, 2006.

[53] Paris Perdikaris and George Em Karniadakis. Model inversion via multi-fidelity bayesian optimization: a new paradigm for parameter estimation in haemodynamics, and beyond. *Journal of The Royal Society Interface*, 13(118), 2016.

[54] Alexander Forrester, Andras Sobester, and Andy Keane. *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons, 2008.

[55] Xuhui Meng, Hessam Babaee, and George Em Karniadakis. Multi-fidelity bayesian neural networks: Algorithms and applications. *Journal of Computational Physics*, 438:110361, 2021.

[56] Francisco Sahli Costabal, Kristen Matsuno, Jiang Yao, Paris Perdikaris, and Ellen Kuhl. Machine learning in drug development: Characterizing the effect of 30 drugs on the qt interval using gaussian process regression, sensitivity analysis, and uncertainty quantification. *Computer Methods in Applied Mechanics and Engineering*, 348:313–333, 2019.

[57] H Babaee, C Bastidas, M DeFilippo, C Chryssostomidis, and GE Karniadakis. A multi-fidelity framework and uncertainty quantification for sea surface temperature in the massachusetts and cape cod bays. *Earth and Space Science*, page e458, 2018.

[58] Casey M Fleeter, Gianluca Geraci, Daniele E Schiavazzi, Andrew M Kahn, and Alison L Marsden. Multilevel and multifidelity uncertainty quantification for cardiovascular hemodynamics. *Computer methods in applied mechanics and engineering*, 365:113030, 2020.

[59] H Babaee, P Perdikaris, C Chryssostomidis, and GE Karniadakis. Multi-fidelity modelling of mixed convection based on experimental correlations and numerical simulations. *Journal of Fluid Mechanics*, 809:895–917, 2016.

[60] Maziar Raissi, Hessam Babaee, and George Em Karniadakis. Parametric gaussian process regression for big data. *Computational Mechanics*, 64:409–416, 2019.

[61] Guofei Pang, Liu Yang, and George Em Karniadakis. Neural-net-induced gaussian process regression for function approximation and pde solution. *Journal of Computational Physics*, 384:270–288, 2019.

[62] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[63] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. *arXiv preprint arXiv:1711.00165*, 2017.

[64] Xiu Yang, David Barajas-Solano, Guzel Tartakovsky, and Alexandre M Tartakovsky. Physics-informed cokriging: A gaussian-process-regression-based multifidelity method for data-model convergence. *Journal of Computational Physics*, 395:410–431, 2019.

[65] Xiu Yang, Guzel Tartakovsky, and Alexandre M Tartakovsky. Physics information aided kriging using stochastic simulation models. *SIAM Journal on Scientific Computing*, 43(6):A3862–A3891, 2021.

[66] Houman Owhadi and Gene Ryan Yoo. Kernel flows: From learning kernels from data into the abyss. *Journal of Computational Physics*, 389:22–47, 2019.

[67] Paris Perdikaris, Maziar Raissi, Andreas Damianou, Neil D Lawrence, and George Em Karniadakis. Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2198):20160751, 2017.

[68] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Inferring solutions of differential equations using noisy multi-fidelity data. *Journal of Computational Physics*, 335:736–746, 2017.

[69] Maziar Raissi and George Karniadakis. Deep multi-fidelity gaussian processes. *arXiv preprint arXiv:1604.07484*, 2016.

[70] Dongwei Ye, Pavel Zun, Valeria Krzhizhanovskaya, and Alfons G Hoekstra. Uncertainty quantification of a three-dimensional in-stent restenosis model with surrogate modelling. *Journal of the Royal Society Interface*, 19(187):20210864, 2022.

[71] L Mihaela Paun and Dirk Husmeier. Markov chain monte carlo with gaussian processes for fast parameter estimation and uncertainty quantification in a 1d fluid-dynamics model of the pulmonary circulation. *International journal for numerical methods in biomedical engineering*, 37(2):e3421, 2021.

[72] Huan N Do, Ahsan Ijaz, Hamidreza Gharahi, Byron Zambrano, Jonguen Choi, Whal Lee, and Seungik Baek. Prediction of abdominal aortic aneurysm growth using dynamical gaussian process implicit surface. *IEEE Transactions on Biomedical Engineering*, 66(3):609–622, 2018.

[73] Shin-Lei Peng, Pan Su, Fu-Nien Wang, Yan Cao, Rong Zhang, Hanzhang Lu, and Peiying Liu. Optimization of phase-contrast mri for the quantification of whole-brain cerebral blood flow. *Journal of Magnetic Resonance Imaging*, 42(4):1126–1133, 2015.

[74] Jingrong Zeng, Haixia Song, Peng Liu, Xiaofan Xue, Shanshan Mei, Baolei Xu, Yingqi Xing, Dian Qu, and Erhe Xu. Effect of acute levodopa challenge test on cerebral blood flow in parkinson's disease with the supine-to-standing transcranial doppler test. *Journal of the Neurological Sciences*, 456:122811, 2024.

[75] Ikuo Odano, Fumio Maeyatsu, Tetsuo Hosoya, Mami Asari, Kentaro Oba, and Yasuyuki Taki. Diagnostic approach with z-score mapping to reduce artifacts caused by cerebral atrophy in regional cbf assessment of mild cognitive impairment (mci) and alzheimer's disease by [99mtc]-ecd and spect. *Japanese Journal of Radiology*, pages 1–11, 2024.

[76] Dearbhla M Kelly, Zanfina Ademi, Wolfram Doehner, Gregory YH Lip, Patrick Mark, Kazunori Toyoda, Christopher X Wong, Mark Sarnak, Michael Cheung, Charles A Herzog, et al. Chronic kidney disease and cerebrovascular disease: consensus and guidance from a kdigo controversies conference. *Stroke*, 52(7):e328–e346, 2021.

[77] Michael F Goldberg, Morton F Goldberg, R Cerejo, and AH Tayal. Cerebrovascular disease in covid-19. *American Journal of Neuroradiology*, 41(7):1170–1172, 2020.

[78] Rongrong Bai, Yue Lang, Jie Shao, Yu Deng, Reyisha Refuhati, and Li Cui. The role of nlrp3 inflammasome in cerebrovascular diseases pathology and possible therapeutic targets. *ASN neuro*, 13:17590914211018100, 2021.

[79] Winfred C Wang, Dianne M Gallagher, Charles H Pegelow, Elizabeth C Wright, Elliott P Vichinsky, Miguel R Abboud, Franklin G Moser, and Robert J Adams. Multicenter comparison of magnetic resonance imaging and transcranial doppler ultrasonography in the evaluation of the central nervous system in children with sickle cell disease. *Journal of pediatric hematology/oncology*, 22(4):335–339, 2000.

[80] Fin Stolze Larsen, Karsten Skovgaard Olsen, Bent Adel Hansen, Olaf B Paulson, and Gitte Moos Knudsen. Transcranial doppler is valid for determination of the lower limit of cerebral blood flow autoregulation. *Stroke*, 25(10):1985–1988, 1994.

[81] Chris JG Bakker, Romhild M Hoogeveen, and Max A Viergever. Construction of a protocol for measuring blood flow by two-dimensional phase-contrast mra. *Journal of Magnetic Resonance Imaging: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 9(1):119–127, 1999.

[82] Michael Markl, Frandics P Chan, Marcus T Alley, Kris L Wedding, Mary T Draney, Chris J Elkins, David W Parker, Ryan Wicker, Charles A Taylor, Robert J Herfkens, et al. Time-resolved three-dimensional phase-contrast mri. *Journal of Magnetic Resonance Imaging: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 17(4):499–506, 2003.

[83] Alejandro M Brunser, Eloy Mansilla, Arnold Hoppe, Verónica Olavarría, Emi Sujima, and Pablo M Lavados. The role of tcd in the evaluation of acute stroke. *Journal of Neuroimaging*, 26(4):420–425, 2016.

[84] Roland Bammer, Thomas A Hope, Murat Aksoy, and Marcus T Alley. Time-resolved 3d quantitative flow mri of the major intracranial vessels: initial experience and comparative evaluation at 1.5 t and 3.0 t in combination with parallel imaging. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 57(1):127–140, 2007.

[85] Pim van Ooij, JJM Zwanenburg, F Visser, CB Majoie, E Vanbavel, J Hendrikse, and AJ Nederveen. Quantification and visualization of flow in the circle of willis: Time-resolved three-dimensional phase contrast mri at 7 t compared with 3 t. *Magnetic resonance in medicine*, 69(3):868–876, 2013.

[86] Sethuraman Sankaran, Hyun Jin Kim, Gilwoo Choi, and Charles A Taylor. Uncertainty quantification in coronary blood flow simulations: impact of geometry, boundary conditions and blood viscosity. *Journal of biomechanics*, 49(12):2540–2547, 2016.

[87] Bernard De Bruyne, William F Fearon, Nico HJ Pijls, Emanuele Barbato, Pim Tonino, Zsolt Piroth, Nikola Jagic, Sven Mobius-Winckler, Gilles Rioufol, Nils Witt, et al. Fractional flow reserve–guided pci for stable coronary artery disease. *New England Journal of Medicine*, 371(13):1208–1217, 2014.

[88] Bjarne L Nørgaard, Jonathon Leipsic, Sara Gaur, Sujith Seneviratne, Brian S Ko, Hiroshi Ito, Jesper M Jensen, Laura Mauri, Bernard De Bruyne, Hiram Bezerra, et al. Diagnostic performance of noninvasive fractional flow reserve derived from coronary computed tomography angiography in suspected coronary artery disease: the nxt trial (analysis of coronary blood flow using ct angiography: Next steps). *Journal of the American College of Cardiology*, 63(12):1145–1155, 2014.

[89] Changyoung Yuhn, Marie Oshima, Yan Chen, Motoharu Hayakawa, and Shigeki Yamada. Uncertainty quantification in cerebral circulation simulations focusing on the

collateral flow: Surrogate model approach with machine learning. *PLoS Computational Biology*, 18(7):e1009996, 2022.

[90] Merih I Baharoglu, Clemens M Schirmer, Daniel A Hoit, Bu-Lang Gao, and Adel M Malek. Aneurysm inflow-angle as a discriminant for rupture in sidewall cerebral aneurysms: morphometric and computational fluid dynamic analysis. *Stroke*, 41(7):1423–1430, 2010.

[91] Justin S Tran, Daniele E Schiavazzi, Andrew M Kahn, and Alison L Marsden. Uncertainty quantification of simulated biomechanical stimuli in coronary artery bypass grafts. *Computer methods in applied mechanics and engineering*, 345:402–428, 2019.

[92] Peng Chen, Alfio Quarteroni, and Gianluigi Rozza. Simulation-based uncertainty quantification of human arterial network hemodynamics. *International journal for numerical methods in biomedical engineering*, 29(6):698–721, 2013.

[93] David Galbally, Krzysztof Fidkowski, Karen Willcox, and Omar Ghattas. Non-linear model reduction for uncertainty quantification in large-scale inverse problems. *International journal for numerical methods in engineering*, 81(12):1581–1608, 2010.

[94] Ramakrishna Tipireddy, Panos Stinis, and Alexandre M Tartakovsky. Time-dependent stochastic basis adaptation for uncertainty quantification. *arXiv preprint arXiv:2103.03316*, 2021.

[95] Peng Chen and Christoph Schwab. Model order reduction methods in computational uncertainty quantification. *Handbook of uncertainty quantification*, pages 1–53, 2016.

[96] Benjamin Fröhlich, Dominik Hose, Oliver Dieterich, Michael Hanss, and Peter Eberhard. Uncertainty quantification of large-scale dynamical systems using parametric model order reduction. *Mechanical Systems and Signal Processing*, 171:108855, 2022.

[97] 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.

[98] Mohammad Hossein Naderi and Hessam 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.

[99] Othmar Koch and Christian Lubich. Dynamical low-rank approximation. *SIAM Journal on Matrix Analysis and Applications*, 29(2):434–454, 2007.

[100] Eleonora Musharbash and Fabio Nobile. Dual dynamically orthogonal approximation of incompressible navier stokes equations with random boundary conditions. *Journal of Computational Physics*, 354:135–162, 2018.

[101] M Donello, G Palkar, MH Naderi, DC Del Rey Fernández, and H Babaee. Oblique projection for scalable rank-adaptive reduced-order modelling of nonlinear stochastic partial differential equations with time-dependent bases. *Proceedings of the Royal Society A*, 479(2278):20230320, 2023.

[102] SJ Sherwin, V Franke, J Peiró, and K Parker. One-dimensional modelling of a vascular network in space-time variables. *Journal of engineering mathematics*, 47(3-4):217–250, 2003.

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

[104] 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.

[105] 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), 2016.

[106] 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.

[107] 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.

[108] Alexandre J. Chorin, Ole H. Hald, and Raz Kupferman. Optimal prediction and the mori–zwanzig representation of irreversible processes. *Proceedings of the National Academy of Sciences*, 97(7):2968–2973, 2000.

[109] Tong Liu, Jinzhen Wang, Qing Liu, Shakeel Alibhai, Tao Lu, and Xubin He. High-ratio lossy compression: Exploring the autoencoder to compress scientific data. *IEEE Transactions on Big Data*, 2021.

[110] Dipti Mishra, Satish Kumar Singh, and Rajat Kumar Singh. Wavelet-based deep auto encoder-decoder (wdaed)-based image compression. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(4):1452–1462, 2020.

[111] 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.

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

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

[114] 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.

[115] 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.

[116] Ville Vuorinen and K Keskinen. Dnslab: A gateway to turbulent flow simulation in matlab. *Computer Physics Communications*, 203:278–289, 2016.

[117] L.N. Trefethen. Householder triangularization of a quasimatrix. *IMA Journal of Numerical Analysis*, 30(4):887–897, 2010.

[118] Jordi Alastruey, SM Moore, KH Parker, T David, J Peiró, and SJ Sherwin. Reduced modelling of blood flow in the cerebral circulation: coupling 1-d, 0-d and cerebral auto-regulation models. *International journal for numerical methods in fluids*, 56(8):1061–1067, 2008.

[119] SJ Sherwin and G Em Karniadakis. Spectral/hp element methods for cfd, 1999.

[120] SE Maier, D Meier, P Boesiger, UT Moser, and A Vieli. Human abdominal aorta: comparative measurements of blood flow with mr imaging and multigated doppler us. *Radiology*, 171(2):487–492, 1989.

[121] J Bock, BW Kreher, Jürgen Hennig, and Michael Markl. Optimized pre-processing of time-resolved 2d and 3d phase contrast mri data. In *Proceedings of the 15th Annual meeting of ISMRM, Berlin, Germany*, volume 3138, 2007.

[122] 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.

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

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