December 2018

Applications of Dynamic Mode Decomposition and Sparse Reconstruction in the Data-Driven Dynamic Analysis of Physical Systems

Mojtaba F. Fathi
University of Wisconsin-Milwaukee

Applications of Dynamic Mode Decomposition and Sparse Reconstruction in The Data-Driven Dynamic Analysis of Physical Systems

by

Mojtaba F. Fathi

A Dissertation Submitted in

Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

in Engineering

at

The University of Wisconsin–Milwaukee

December 2018

Abstract

Applications of Dynamic Mode Decomposition and Sparse Reconstruction in The Data-Driven Dynamic Analysis of Physical Systems

by
Mojtaba F. Fathi

The University of Wisconsin–Milwaukee, 2018
Under the Supervision of Prof. Roshan M. D'Souza

Recent advancements in data collection methods and equipment have resulted in a huge increase in the amount of data collected by observing various types of physical phenomena. Regardless of the amount of data collected, it is well known for many physical systems, the so-called information rank of the collected data is much lower than the rank of the data itself. This usually means the data may be represented sparsely in terms of a properly-chosen basis. This realization has led to methods for storing large amounts of data through compression by sacrificing negligible data quality. More importantly, with the advent of compressed sensing techniques, using an appropriate representation basis and sampling technique, it is now possible to sample data far below the Shannon-Nyquist limit thus speeding up data acquisition and also reducing the complexity of data-acquisition hardware. In this research, we explore the application of various modern data analysis techniques such as proper orthogonal decomposition (POD), dynamic mode decomposition (DMD), compressed sensing, and Kalman filter and smoother in the data-driven analysis of dynamic systems with many degrees of freedom. This research has resulted in four novel methods. The first method is developed for denoising and spatial resolution enhancement of 4D-Flow MRI data based on POD and sparse reconstruction. The second method combines DMD and compressed sensing and takes discrete cosine transform (DCT) as the representation basis for dynamic denoising and gappy data reconstruction in 2D. The third method is a fast and parameter-free dynamic denoising method which combines a reduced-order model (ROM), a Kalman filter and smoother, and a DMD-based forward model. The fourth method is developed for reconstructing a 2D incompressible flow field by taking sparse measurements from the Fourier

domain. As the reconstruction basis, a custom divergence-free set of basis vectors are derived and implemented.

To

my parents,

my wife,

and my beloved daughter, Ava

Table of Contents

List of Figures

List of Tables

List of Algorithms

Acknowledgment

First of all, I would like to thank my advisor Prof. Roshan M. D'Souza for his guidance over the years. I am thankful for his openness to discussing novel ideas and his willingness to explore novel methods and their applications in real-world problems. I am also thankful to him for his financial support through providing me the opportunity to serve as his teaching assistant for the *Introduction to Control Systems* course seven semesters in a row. Also, I would like to thank my colleagues in the *Complex Systems Simulation Laboratory* at the Mechanical Engineering Department of the University of Wisconsin-Milwaukee. Among my colleagues, I am especially thankful to Dr. Ali Bakhshinejad and Dr. Ahmadreza Baghaie for all collaborations we had together. Finally, I am grateful to my dear wife, Fariba, without whose support I wouldn't be able to accomplish my journey; and to my beloved daughter, Ava, who has brought peace and hope to our family.

Chapter 1

Introduction

Recent advancements in data collection methods and equipment have resulted in a huge increase in the amount of data collected by observing various types of physical phenomena. For example, Particle Image Velocimetry (PIV) experiments for time-resolved analysis of flow results in a rich volumetric three-component flow velocity field which can be in the order of 20+ GB [1]. While the hardware may exist for acquiring high spatial resolution data, the time taken may be impractical. For example, the time-resolved three-dimensional phase contrast magnetic resonance imaging of blood flow in cerebral aneurysms over a single cardiac cycle can typically take about 20 minutes [2] during which no physical motion by the patient is permitted to eliminate motion artifacts.

It is well known that for data resulting from physical processes, the *information rate* of the data may be much lower than the rank of the data itself [3]. This means that given an appropriate representation basis (wavelet basis, for example), the data may be represented using a coefficient vector whose size is much smaller than the acquired data. This realization initially led to the development of various data compression techniques where a process of sparse encoding of the raw data led significant memory saving with negligible loss in fidelity [4]. It has also led to techniques for directly acquiring data in an appropriate sparse basis using techniques which are broadly called *compressed sensing*[3]. Under certain circumstances, this has enabled the sampling of data much below the Shannon-Nyquist limit thus simplifying data acquisition hardware and also reducing the time for data acquisition. In the recent past, the confluence of the increasing amount of data being collected and novel mathematical techniques for analysis of such data has enabled "data-driven dynamical systems" where physical systems can be characterized purely in terms of data. This data

can result from actual observations or through numerical simulation of complex numerical models. Two prominent techniques include *Proper Orthogonal Decomposition* (POD) [5] and *Dynamic Mode Decomposition*. Observation data is typically noisy, and state estimation techniques such as *Kalman Filtering* [6] have been used to mitigate the effect of acquisition noise.

In this research, we explore the use of a combination of techniques from state estimation, compressed sensing, and data-driven dynamic system analysis in various applications including denoising and resolution enhancement of 4D-Flow MRI data by merging observations and simulation, denoising and reconstruction of gappy time-resolved dynamic datasets, purely data-driven Kalman filtering and smoothing of raw 4D-Flow MRI data, and finally, data-driven Kalman filtering and smoothing of sparsely-acquired flow data in frequency space.

The outline of this thesis is as follows: Chapter 2 gives a brief overview of the various techniques used in this dissertation. In Chapter 3, a novel method is introduced for denoising and spatial resolution enhancement of 4D-Flow MRI data. In Chapter 4, a novel framework is proposed by combining DMD and compressed sensing and taking discrete cosine transform (DCT) as the representation basis for dynamic denoising and gappy data reconstruction in 2D. Chapter 5 presents a fast and parameter-free dynamic denoising method which combines a reduced-order model (ROM), a Kalman filter and smoother, and a DMD-based forward model. In Chapter 6, the problem of taking sparse measurements from the Fourier domain to reconstruct the signal in the Cartesian domain is addressed and a solution method is proposed. The method is developed for 2D incompressible flow fields for which the signal representation is done based on a custom divergence-free set of basis vectors. The derivation and implementation method of the custom divergence-free basis is presented in the Appendix.

Chapter 2

Overview

In this chapter, an overview of various techniques used in this research is provided.

2.1 Proper Orthogonal Decomposition (POD)

The POD is a statistical method of obtaining a compact representation of a dataset [7]. As a multivariate method, the POD aims at reducing a given dataset expressed in terms of several independent variables to a much fewer number of uncorrelated variables. The reduction is performed while as much of the variation in the dataset is retained as possible. The POD is also known as the Karhunen-Loève decomposition. It results in an ordered set of orthonormal basis vectors which represent the directions along which most variability in the data is observed [5]. The common approach of finding the POD vectors for a given real-valued dataset is through using a real factorization called the Singular Value Decomposition (SVD) for which various reliable algorithms have been developed.

2.2 Kalman Filter

The Kalman filter was introduced in 1960 as a discrete-time linear filter [6]. It can efficiently estimate the state of a dynamic process by minimizing the mean of the squared error in the presence of system model uncertainty and also, process and measurement noise. Consider a stochastic linear

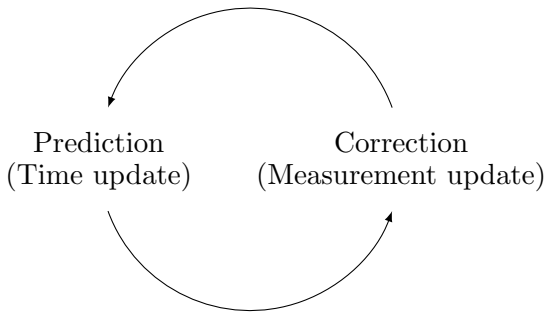
process governed by the following difference equation

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_{k-1} + \mathbf{w}_{k-1} \tag{2.1}$$

where $\mathbf{x}$ is the vector of state variables, $\mathbf{u}$ is the vector of inputs, and $\mathbf{w}$ represents the process noise which is assumed to be a white normally-distributed random variable with the known covariance $\mathbf{Q}$. Suppose the measurements are taken as

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k \tag{2.2}$$

where $\mathbf{z}$ is the vector of measurements and $\mathbf{v}$ is the measurement noise which is also assumed to be a white normally-distributed random variable with the known covariance $\mathbf{R}$. The process and measurement noise are assumed to be independent. The Kalman filter provides a recursive solution to the discrete-data linear filtering problem through a sequence of prediction and correction steps as depicted below



At each step, a prediction of the state variables is made based on the governing difference equation (2.1) which is further corrected according to the measurements equation (2.2).

Since its first introduction, The Kalman filter has been used extensively in various applications. Also, many researchers have attempted to expand its applicability by proposing variants of the Kalman filter, e.g. the extended Kalman filter [8], the unscented Kalman filter [9], and the linear Kalman smoothing [10] to name a few. The linear Kalman filter may be applied to dynamic processes governed or approximated by a linear difference equation, as the name implies.

2.3 Compressed Sensing

Talking about data acquisition, the first thing that comes to mind is Shannon's theorem which states the data acquisition rate must be not less than twice the highest frequency present in the signal [3]. This rate is usually referred to as the Nyquist rate. In contrast to this traditional data acquisition scheme, the theory of compressed sensing states certain signals may be recovered from much fewer samples than Shannon's theorem specifies [3]. To make this happen, the compressed sensing relies on the signal to be sparse when represented in some basis. That means the information rank of the signal is much smaller than its length. As stated earlier, this is usually the case with most natural signals and such signals have a sparse representation in terms of a proper basis [3]. Besides the sparsity of the signal, compressed sensing also relies on the *incoherence* of the measurements. This means if sampling is done in Φ basis and the signal is sparse in Ψ basis, then Φ basis should have a dense representation in Ψ basis. For example, a commonly-used sampling basis is the Dirac delta function (spikes) whereas the Fourier basis is commonly used for signal representation. Spikes have a dense representation in Fourier basis and so, spikes and sinusoids are incoherent. In fact, spikes and sinusoids are maximally incoherent in any dimension. Due to this property, the pair of spikes and sinusoids is a good choice for compressed sensing applications.

In order to solve compressed sensing problems, the *Orthant-Wise Limited-memory Quasi-Newton (OWL-QN)* algorithm was used [11]. The OWL-QN algorithm was developed based on Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) algorithm, which is a well-known quasi-Newtonian method for optimizing the parameters of large-scale log-linear models with l_2-regularization. It can be used for solving the large-scale optimization problems having a very high number of parameters with an l_1-regularized term in a memory-efficient manner. The OWL-QN is well implemented and is proven to converge under certain conditions. The overall procedure of the algorithm is shown in Algorithm 1.

2.4 Dynamic Mode Decomposition (DMD)

Dynamic Mode Decomposition (DMD) was introduced by Schmid and Sesterhenn [12] in 2008 to study the spatial dynamic modes of fluid flow [13, 14]. DMD approximates the nonlinear dynamics underlying a given time-varying dataset in terms of a linear auto-regressive model by extracting

Algorithm 1. The overall procedure of OWL-QN algorithm

Data:
- Initial solution $\mathbf{c}_0$
- Cost function $f(\mathbf{c})$

Result:
- Final solution $\mathbf{c}$

```
c ← c_0
while maximum number of iterations not reached do
    Calculate pseudo-gradient of f(c)
    Find the search direction
    Update c by solving the constrained line search problem
    if convergence achieved then
        return c
    end
    Update the internal variables
end
```

a set of mode shapes and their corresponding eigenvalues, where the mode shapes represent the spatial spread of dominant features and the eigenvalue associated with each mode shape specifies how that feature evolves over time in terms of the frequency of oscillation and the rate of growth or decay.

2.4.1 Expanding the applications of DMD

The widely-used classic implementation of DMD is known as exact DMD [15] and is briefly introduced in section 4.2.1. Rowley et al. [16] and Chen et al. [14] envisioned DMD as an approximation to the modes of Koopman operator, which is an infinite-dimensional linear representation of nonlinear finite-dimensional dynamics. Even though DMD was initially meant to be used for extracting dynamic information from flow fields [13], soon it found new applications in other areas of study as a powerful tool for analyzing the dynamics of nonlinear systems. At first, the input data of DMD was supposed to be time-resolved. That means the data was in the form of a sequence of several snapshots where the time interval between any two consecutive snapshots remained the same. The number of snapshots is usually much smaller than the length of each snapshot. In 2014, Kutz et al. [15] showed that besides time-resolved datasets, DMD may also be applied to paired datasets where there is a mapping between each snapshot of one dataset with the corresponding snapshot from the other dataset. In such case, the snapshots are not required to be time-resolved. Jovanovic

et al. [17] proposed the sparsity-promoting DMD (spDMD) to obtain a sparse representation of the system dynamics by limiting the number of dynamic modes through an l_1-regularization approach. In 2015, the extended DMD (EDMD) was introduced by Williams et al. [18] to approximate the leading eigenvalues, eigenfunctions, and modes of the Koopman operator. The EDMD is a computationally intensive algorithm since it requires the choice of a rich dictionary of basis functions to produce an approximation of the Koopman eigenfunctions. The richer the dictionary is, the more time it takes to compute the inner products which are a key part of EDMD algorithm. In an attempt to overcome this issue, Williams et al. proposed the kernel-based DMD (KDMD) in 2015 [19]. In this approach, rather than choosing the dictionary of the basis functions explicitly, they are defined implicitly by the choice of a kernel function. The kernel function resolves the computational intensity issue of EDMD by finding the inner products of the basis functions without the need to having them defined explicitly.

2.4.2 Compressed sensing and DMD

An initial attempt for incorporating compressed sensing in DMD was made by Guéniat et al. [20], where a subset of an originally-large dataset was taken by non-uniform sampling and was used for finding the temporal coefficients (eigenvalues) through solving an optimization problem. Further, the corresponding modes were found by solving a set of linear equations which involved the fully-sampled dataset. This makes the proposed algorithm (known as NU-DMD) impractical in the case the fully-sampled dataset is not available. Another approach for incorporating compressed sensing in DMD (known as csDMD) was developed by Kutz et al. [21]. In csDMD, the DMD eigenvalues are obtained from a sub-sampled dataset (similar to NU-DMD), which has the advantage of reducing computation time, and then the full DMD mode shapes are reconstructed through using an l_1-minimization scheme based on a chosen set of basis vectors. In contrast to NU-DMD, csDMD does not need the fully-sampled dataset in order to recover the mode shapes.

2.4.3 DMD in presence of noise

By applying DMD on a noisy dataset, many of the dynamic modes identified will be mostly due to noise [22]. As Dawson et al. [22] showed, the effect of sensor noise is seen as a shift in the computed DMD eigenvalues such that they appear to be more stable than they are in reality. In

order to compensate the effect of noise, they proposed three different variants of DMD among which forward-backward DMD (fbDMD) was shown to perform usually better than the other two (Noise-corrected DMD and Total least-squares DMD). The fbDMD also has the advantage of not requiring any prior knowledge about the noise covariance matrix.

Chapter 3

4D-Flow MRI Denoising and Spatial Resolution Enhancement

3.1 Introduction

The 4D-Flow MRI technique (3D+Time)[23, 24] enables non-invasive in vivo volumetric time-resolved three-dimensional blood flow velocity measurements. In addition to basic analysis such as volumetric visualization of blood flow and quantification of volume flow rate, this technique enables advanced analysis such as the estimation of important hemodynamic parameters (wall shear stress (WSS)[25], pressure gradients (PG) [26], and viscous dissipation (VD)[27]). Furthermore, it enables retrospective placement of analysis planes at any location within the acquisition volume for volumetric flow analysis.

However, this technique suffers from issues such as velocity aliasing, phase offsets, random acquisition noise, and low spatial and temporal resolution. (in order of $0.8\,\text{mm} - 1.3\,\text{mm}$ isotropic spatial resolution and $20 - 50\text{ms}$ temporal resolution). Velocity aliasing is caused when the velocity encoding (Venc) range setting is lower than the maximum velocity being imaged. A high Venc setting may ameliorate this problem but will increase acquisition noise [28, 29]. Recently, there have been efforts to simultaneously address velocity aliasing and noise issues using multi-Venc approaches [30, 31]. Phase offset errors are caused by various factors including Maxwell terms [32], gradient field nonlinearities[33], and eddy currents[34]. Techniques have been developed to address

both velocity aliasing [35] and velocity offset errors [36, 37].

Acquisition noise impacts computation of WSS, PG, and VD [27] since these parameters are dependent on gradients of the velocity field. It has been shown that at spatiotemporal resolution available with 4D-Flow, computation of WSS and maximum shear stress (MSS) is not reliable because of averaging effects [38]. Furthermore, studies have demonstrated that $5-6$ voxels across the vessel diameter is required for flow volume quantification [28, 39]. For intra-cranial aneurysms, depending on the location, even sizes of 5 mm (with < 4 voxels across the vessel diameters) can have a high risk of rupture and will need investigation [40]. Theoretically, the spatial resolution of 4D-Flow can be increased using a combination of higher magnetic fields (improving SNR), a smaller field of view (FOV), and accelerated imaging techniques [41, 42]. Even with these enhancements, achieving 0.3 mm isotropic spatial resolution, which is barely sufficient for accurate calculation of WSS and MSS [38], requires rather impractical acquisition times ($\approx$ 20 min) in a clinical setting. Patient-specific computational fluid dynamics (CFD) simulations using boundary conditions from 4D/2D-Flow and geometry from CT/MRI scans address the spatiotemporal resolution issues [43, 44, 45]. However, the assumption on the type of flow, and imprecision in the estimation of boundary conditions means that the resulting simulation can be significantly different from reality [28].

There are two main types of denoising methods: regularization-based methods and projection-based methods. The regularization-based methods use constrained optimization. There are two terms in the objective function. The first minimizes the Euclidean norm between the solution and 4D-Flow data. The second term, called the penalty/regularization term, imposes flow physics on the solution (divergence-free and curl-free conditions) [46, 47]. Projection-based techniques use vector projection of the unprocessed 4D-Flow data on to a divergence-free basis such as wavelets [48, 49, 47], radial basis functions [50], and vector-fields derived using finite differences [51]. These methods address the noise issue and partially account for flow physics (mass balance but not momentum balance). However, the spatiotemporal resolution issues are not addressed.

Recently, methods based on merging CFD with 4D-Flow have attempted to fully account for flow physics and address the spatiotemporal issues. Rispoli et al. [52] used a Tikhonov regularized [53] patient-specific CFD by modifying the semi-implicit method for pressure linked equations-revised (SIMPLER) algorithm [54]. The regularization step used 4D-Flow data to correct the CFD simulation. Our group has developed two methods. The first uses proper orthogonal decomposition

(POD) [55] basis obtained from snapshots of patient-specific CFD and ridge regression to compute the underlying flow-field of the 4D-Flow data at an arbitrary level (depending on the resolution of the CFD mesh) of spatial resolution [56]. This method required Monte Carlo type sampling of boundary conditions and multiple CFD simulations to cover the variance in measurements from unprocessed 4D-Flow data. This can be computationally expensive in cases of multiple inlet/outlet geometries. Another method used an ensemble of patient-specific CFD simulations in a data assimilation framework using an ensemble Kalman filter (EnKF) [57]. For stability of the EnKF, we required up to 80 different CFD solutions in the ensemble, which is again computationally expensive.

In this chapter, a novel method is described which merges patient-specific CFD simulations with 4D-Flow using POD and Least Absolute Shrinkage and Selection Operator (LASSO) regularization [58]. The method has been tested on numerical phantoms derived from two actual intra-cranial aneurysm geometries. Benchmarks against contemporary state-of-the-art techniques (at the low resolution of 4D-Flow acquisition) show significant improvements in error metrics. Furthermore, we show successful recovery of the underlying reference flow field in the high-resolution CFD mesh.

3.2 Materials and Methods

Assuming that a given signal is sparse in a certain representation basis (for example, wavelet/Fourier), the LASSO regression is a method to find the sparse representation of a signal in a given basis while minimizing error. In our work, we use the POD basis vectors derived from a CFD simulation using the method of snapshots [59] where the flow field can be sparsely represented. This allows us to insert the complete flow physics (momentum and mass balance) into the LASSO formulation. Our overall algorithm is as shown in Fig. 3.1. Our algorithm begins by sampling boundary conditions from the 4D-Flow data. These are used to run a patient-specific CFD simulation based on segmented geometry. Snapshots from this CFD simulation are used to generate the POD basis vectors. Subsequently, velocity components from the 4D-Flow data are used to perform the LASSO regularization to obtain the sparse coefficient vector. The resulting coefficient vector is then used to generate the flow velocity profile underlying the 4D-Flow data in the CFD mesh resolution.

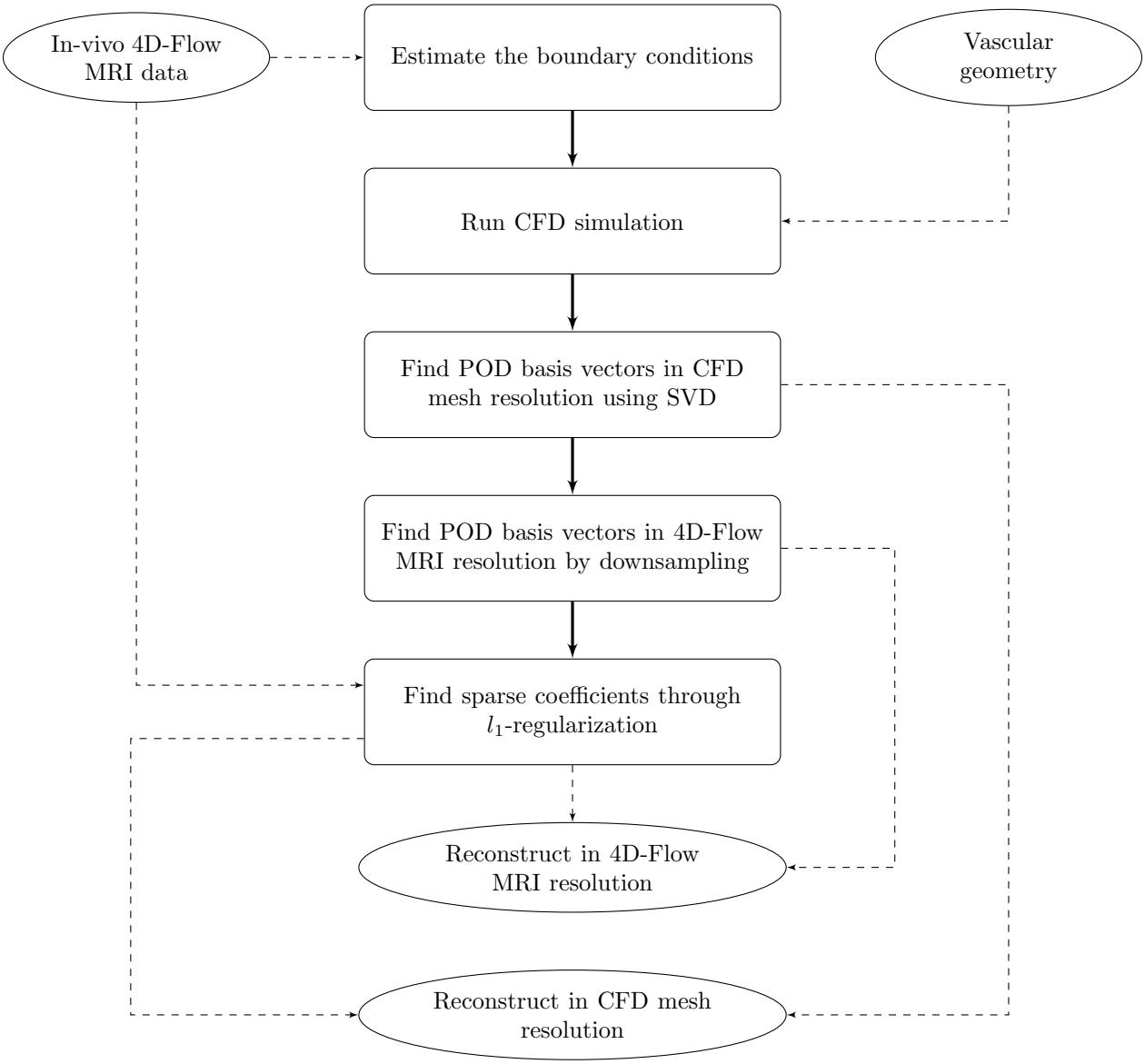


Figure 3.1: The flowchart of the proposed method.

3.2.1 Testing the algorithm using numerical phantom

An actual intra-cranial aneurysm geometry with a realistic pulsatile boundary condition was used to conduct a CFD simulation. Time snapshots of this simulation formed the noiseless CFD reference (*"CFD ref"*) in high resolution. Next, we down-sampled the *"CFD ref"* (*"d-CFD ref"*) into a typical 4D-Flow grid (each grid voxel had a spatial resolution of $1.09 \times 1.09 \times 1.3\text{mm}^3$). Next, the 4D-Flow acquisition process was simulated by adding noise in the *k-space* to the *"d-CFD ref"* based on the algorithm suggested by Johnson and Markl [60]. The calculated noisy and time-resolved datasets (*"d-CFD ref+noise"*) were used as the numerical phantom to test the algorithm, benchmark its performance against other denoising methods in 4D-Flow resolution, and test its ability to reconstruct the *"CFD ref"* in the high-resolution CFD mesh.

It should be noted the *"CFD ref"* is different than the CFD simulation used to calculate the POD modes. The *"CFD ref"* was only used for calculating the error metrics defined in Section 3.2.4 and was never used during the reconstruction process.

3.2.2 POD

The POD technique, similar to principal component analysis, allows the construction of a set of spatial modes or solution bases from time-resolved CFD data. In our method, the CFD simulation is generated from boundary conditions obtained from sampling the 4D-Flow data at the inlets and outlets of the blood vessels of interest. In order to calculate the POD, we used the method of snapshots [59]. Initially, a matrix of CFD solutions was generated as:

$$\mathbf{Z}_H = \begin{bmatrix} \mathbf{s}_1^H & \mathbf{s}_2^H & \cdots & \mathbf{s}_n^H \end{bmatrix} \tag{3.1}$$

where $\mathbf{s}_k^H \in \mathbb{R}^{3N_H \times 1}$ is one snapshot of the simulated flow field calculated using the averaged BC and N_H is the dimension of the high-resolution vectorized flow field (from the CFD mesh). The length of $\mathbf{s}_k^H$ is $3N_H$ because there are three velocity components corresponding to each node of the high-resolution vectorized flow field. Therefore, $\mathbf{Z}_H \in \mathbb{R}^{3N_H \times N_s}$, where N_s is the number of snapshots. Hereafter, the vectors in low- and high-resolution domains will be denoted by L and H subscripts respectively. We can calculate the POD modes of the matrix $\mathbf{Z}_H$ by using singular value

decomposition (SVD) [61]:

$$\mathbf{Z}_H = \mathbf{A}_H \mathbf{\Sigma}_H \mathbf{V}_H^T \tag{3.2}$$

where $\mathbf{A}_H \in \mathbb{R}^{3N_H \times N_s}$ and $\mathbf{V}_H \in \mathbb{R}^{N_s \times N_s}$ are the left and right eigenvectors respectively. The matrix $\mathbf{\Sigma}_H \in \mathbb{R}^{N_s \times N_s}$ is a diagonal matrix containing the singular values. Here, the left eigenvectors $\mathbf{A}_H$ represent the basis vectors that span the space of all possible solutions in the CFD mesh which have BC close to the calculated mean BC. Next, the calculated left eigenvectors were down-sampled into 4D-Flow resolution as:

$$\mathbf{A}_L = \mathbf{D}\mathbf{A}_H \tag{3.3}$$

where $\mathbf{D} \in \mathbb{R}^{3N_L \times 3N_H}$ is the down-sampling matrix. In order to down-sample the modes, $\mathbf{D}$ averages the values belonging to all CFD mesh centers that fall within a 4D-Flow voxel. The calculated matrix $\mathbf{A}_L$ represents the basis vectors that span the solution space in the 4D-Flow resolution.

3.2.3 Reconstructing flow field in the CFD mesh using LASSO regression

Since any flow field is sparse in the POD basis, reconstructed flow field can be written as

$$\mathbf{u}_{rec}^H = \mathbf{A}_H \mathbf{c} \tag{3.4}$$

where $\mathbf{c}$ is a sparse vector of coefficients. The goal is to find this vector. Note that 4D-Flow data is available only in low resolution of the 4D-Flow grid. Therefore, we need to down-sample the reconstructed velocity field in order to compute the error. This is done as follows

$$\mathbf{u}_{rec}^L = \mathbf{D}\mathbf{A}_H \mathbf{c} = \mathbf{A}_L \mathbf{c} \tag{3.5}$$

Finally, we solve a LASSO optimization problem which finds the sparsest solution in the POD basis which is closest to the 4D-Flow data. The objective function is given by

$$\hat{\mathbf{c}} = \underset{c}{\operatorname{argmin}} \left(\|\mathbf{u}_{rec}^L - \mathbf{u}_{4DFlow}^L\|_2^2 + \beta \|\mathbf{c}\|_1 \right) \tag{3.6}$$

The first term penalizes the error between the reconstructed solution and the actual 4D-Flow data, the second term enforces sparsity. The parameter β controls the sparsity of the solution vector $\mathbf{c}$.

The OWL-QN algorithm was used to solve the LASSO regression equation.

3.2.4 Error analysis

The reconstructed flow field in 4D-Flow resolution was compared with different reconstruction methods in the literature using three different metrics [48, 56]. These metrics are: velocity normalized root mean square error (vNRMSE), the speed normalized root mean square error (sNRMSE), direction error (DE), and peak velocity-to-noise ratio (PVNR) which was used to quantify the initial noise level in simulations.

$$\text{vNRMSE} \triangleq \frac{1}{\max_i\left(\left|\mathbf{u}^L_{i,ref}\right|\right)}\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left|\mathbf{u}^L_{i,ref}-\mathbf{u}^L_{i,rec}\right|^2} \tag{3.7}$$

$$\text{sNRMSE} \triangleq \frac{1}{\max_i\left(\left|\mathbf{u}^L_{i,ref}\right|\right)}\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\left|\mathbf{u}^L_{i,ref}\right|-\left|\mathbf{u}^L_{i,rec}\right|\right)^2} \tag{3.8}$$

$$\text{DE} \triangleq \frac{1}{N}\sum_{i=1}^{N}\left(1-\frac{\left|\mathbf{u}^L_{i,ref}\cdot\mathbf{u}^L_{i,rec}\right|}{\left|\mathbf{u}^L_{i,ref}\right|\left|\mathbf{u}^L_{i,rec}\right|}\right) \tag{3.9}$$

$$\text{PVNR} \triangleq 20\log_{10}\frac{1}{\text{vNRMSE}}dB \tag{3.10}$$

where $\mathbf{u}^L_{i,ref}$, and $\mathbf{u}^L_{i,rec}$ are the i^{th} components of the reference flow field (*"d-CFD ref"*) and reconstructed velocity profiles in 4D-Flow resolution, respectively.

3.3 Results

3.3.1 Effect of regularization parameter β on sNRMSE

Figure 3.2 shows the effect of changing the regularization parameter β on the sNRMSE. Small values of β generate denser solutions and push the solution towards minimization of least squares error. Large values of β result in very sparse solutions and also increase sNRMSE. In between, there is an optimal value of β that balances sparsity of the solution with the least squares error. In our numerical phantom tests, $\beta = 5\times10^{-4}$ was optimal.

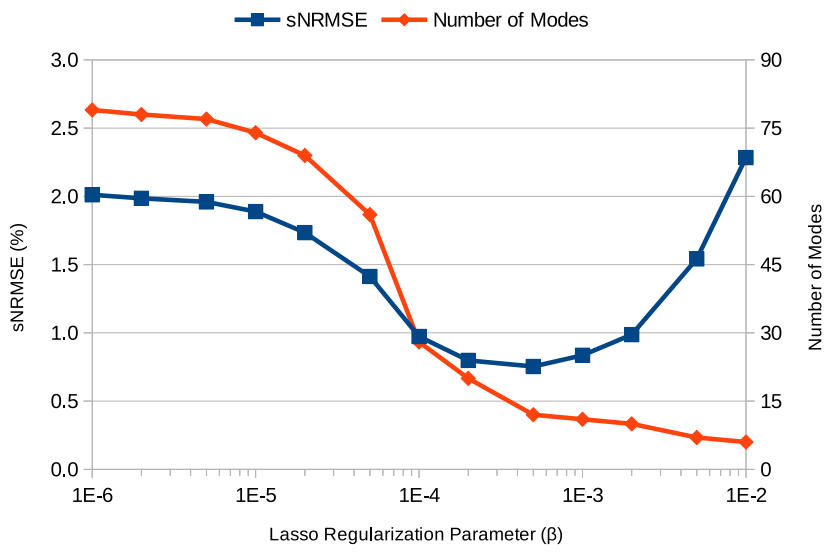


Figure 3.2: The effect of lasso regularization parameter (β) on the number of modes and reconstruction error in high resolution. Reconstruction error is shown as the speed-normalized RMSE (sNRMSE) of sparse representation. The large values of β result in sparsity-dominant solutions while the small values result in over-fitted least-square-dominant solutions. The optimal choice of $\beta = 5 \times 10^{-4}$ results in a good balance between sparsity and reconstruction error (sNRMSE = 0.75% with only 12 modes).

3.3.2 Numerical phantom results

Figure 3.3 represents the results of denoising and spatial resolution enhancement on a numerical phantom generated using an actual aneurysm geometry. The first image from the left (Fig. 3.3(a)) shows the intra-cranial aneurysm geometry. The second image (Fig. 3.3(b)) shows the *"CFD ref"* velocity profile in one slice of the geometry. The third image (Fig. 3.3(c)) shows the *"d-CFD ref"*. The fourth image (Fig. 3.3(d)) shows the *"d-CFD ref+noise"* with noise standard deviation of $\sigma = 0.15 \times |v_{max}|$ in k-space. The fifth image (Fig. 3.3(e)) is the data denoised by using POD and lasso regularization (*"POD-lr"*) in 4D-Flow resolution, and finally, the last image (Fig. 3.3(f)) illustrates the results of spatial resolution enhancement. As can be seen, the results of our method in the mesh resolution are nearly identical to the *"CFD ref"*.

3.3.3 Comparisons with state-of-the-art denoising methods

A benchmark test was run to compare the results from the developed algorithm with available methods using the synthetic data. A series of well-known denoising methods were used for the test,

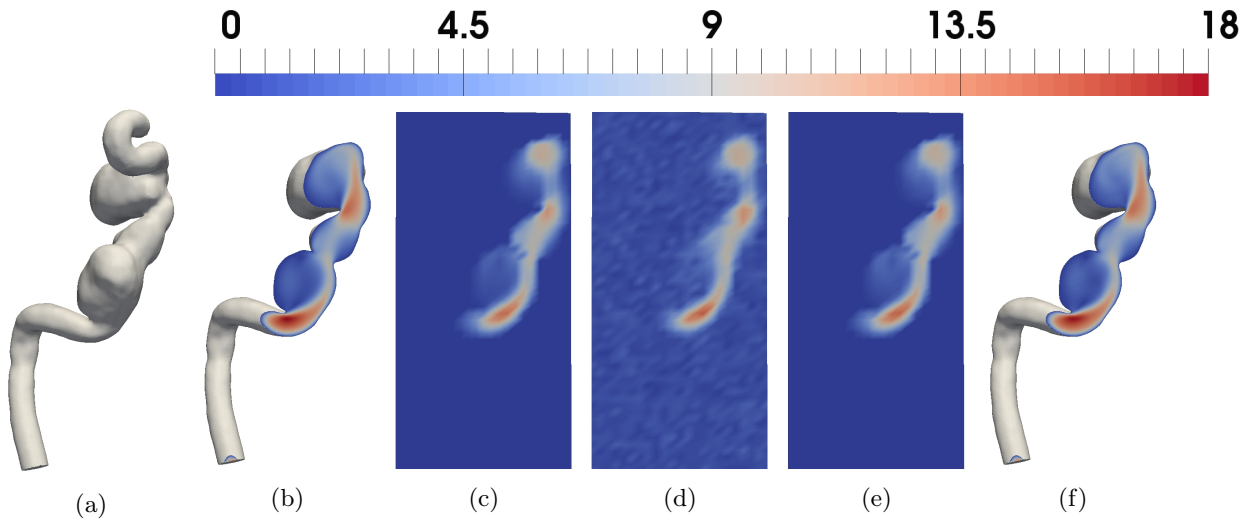


Figure 3.3: Results of the reconstruction process. Column (a) shows the overall geometry, (b) shows the *"CFD ref"*, (c) is the *"d-CFD ref"*, (d) represents *"d-CFD ref+noise"*, (e) illustrates the reconstructed data in 4D-Flow resolution (low resolution), and (f) shows the results of reconstruction using POD and lasso regularization in high resolution (CFD mesh resolution) applied to input data in column d. All velocity profile magnitudes are in cm/s.

such as finite difference method (*"FDM"*), divergence-free wavelets (*"DFW"*), and a total variation method [51, 48, 47]. The kernel size for *"FDM"* was fixed and equal to $3 \times 3 \times 3$. The parameters for *"DFW-sm"* (with automated selection of sub-band dependent threshold using Sure-Shrink and median absolute deviation), *"DFW-sms"* (with Sure-Shrink, median absolute deviation, and partial cycle spinning), and total variation (*"TV"*) methods were used in a such to achieve the minimum value of error metrics.

The error metrics are listed in Table 3.1. Three different noise levels were selected for these tests, 26.32 dB, 24.53 dB, and 20.19 dB PVNR which are corresponding to noise standard deviations of $\sigma = 0.1 \times |v_{max}|$, $\sigma = 0.15 \times |v_{max}|$, and $\sigma = 0.2 \times |v_{max}|$, respectively. All metrics were calculated in 4D-Flow typical resolution as $1.09 \times 1.09 \times 1.3\text{mm}^3$. As represented in Table 3.1, for all three noise levels, *"POD-lr"* results in a much lower error in denoising.

Figure 3.4 shows the visual comparison, (a) shows the *"CFD ref"*, (b) represents *"d-CFD ref"* in 4D-Flow resolution, (c) illustrates *"d-CFD ref+noise"* with noise standard deviation of $\sigma = 0.15 \times |v_{max}|$ in k space, and (d) shows the results of denoising as well as the differences between the denoised data and *"d-CFD ref"* in left and right columns, respectively. *"FDM"* results weren't plotted because as shown in Table 3.1 it almost did not reduce any noise. As it can be observed

Table 3.1: Error metrics of the reconstructed data using several methods with three different noise levels as 10%, 15%, and 20% of the maximum velocity. All values are in percentage.

Noise level	Metric / Method	d-CFD ref +noise	FDM	DFW-sm	DFW-sms	TV	POD-lr
10%	vNRMSE	4.83	4.84	4.53	3.53	2.42	1.20
	sNRMSE	3.87	3.62	3.73	3.07	1.96	0.98
	DE	7.15	7.04	4.55	2.68	1.81	0.36
15%	vNRMSE	5.93	5.76	6.96	6.27	3.72	1.52
	sNRMSE	3.99	3.52	5.82	5.66	3.22	1.28
	DE	11.58	10.96	6.61	4.22	2.59	0.47
20%	vNRMSE	9.78	8.48	6.94	5.41	4.30	2.20
	sNRMSE	8.00	6.77	5.83	4.69	3.44	1.92
	DE	12.70	11.22	7.25	4.84	4.10	0.70

from Table 3.1 and Figure 3.4, *"POD-lr"* gives the best results and others can be listed in order of reconstruction quality as: *"TV"*, *"DFW-sms"*, *"DFW-sm"*, and *"FDM"*.

3.4 Discussion

3.4.1 Comparison with other methods

One drawback of our method as compared to other denoising methods is the fact that it needs an expensive CFD simulation with accurately segmented geometry of the blood vessel. Furthermore, this segmented geometry must be registered with the 4D-Flow dataset. This consumes most of the time in our method. The next most expensive step is the generation of the POD modes with the SVD algorithm. Once the POD modes are generated, reconstructing the velocity profiles in high resolution is fairly quick (10 to 20 sec per time dataset). This means that the user could interactively change the parameter β to generate the best results. In return, unlike other denoising methods, our method is capable of generating high-resolution flow fields to an arbitrarily high level of spatial resolution (depending on the user-defined resolution of the CFD mesh).

3.4.2 Effect of parameter β on speed-normalized error

According to Figure 3.2, there is an optimal value for the lasso regularization parameter β, which is $\beta = 5 \times 10^{-4}$ in this case. The initial value of $\beta = 1 \times 10^{-6}$ uses 79 out of 139 POD modes for

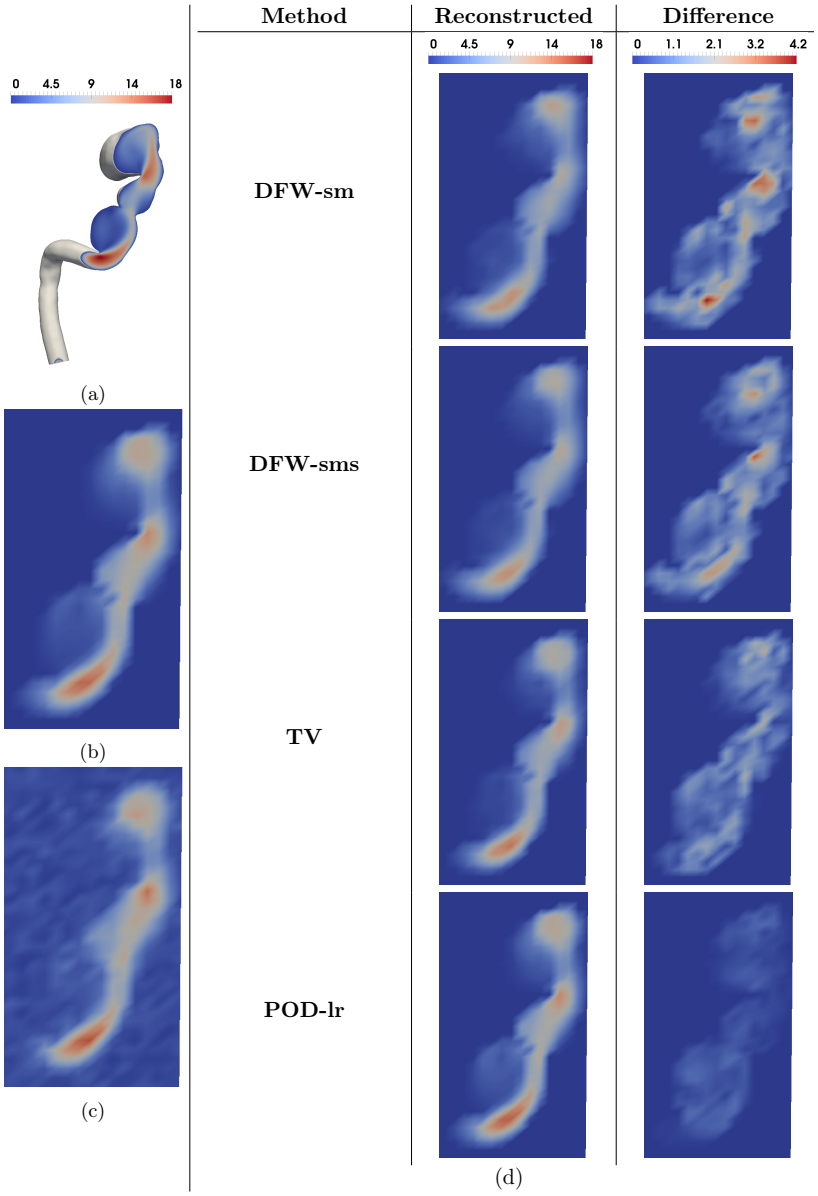


Figure 3.4: Comparison of data reconstruction using different methods. (a) represents the *"CFD ref"*, (b) is the *"d-CFD ref"* in 4D-Flow resolution (each voxel represents $1.09 \times 1.09 \times 1.3\text{mm}^3$ in real space), (c) is the *"d-CFD ref+noise"* which was calculated by adding noise with standard deviation of $\sigma = 0.15 \times |v_{max}|$ to the *"d-CFD ref"* in the k-space. The left column of Table (d) illustrates the reconstructed data using: divergence-free wavelet with Sure-Shrink and median absolute deviation (*"DFW-sm"*), divergence-free wavelet with Sure-Shrink and median absolute deviation and partial cycle spinning (*"DFW-sms"*), total variation (*"TV"*), and POD with lasso regularization (*"POD-lr"*). The right column shows the difference between the reconstruction and *"d-CFD ref"* (image (b)). All images are illustrating velocity profiles in cm/s.

reconstruction resulting in a sNRMSE value of 2.0%. As the β value increases, the number of POD modes and sNRMSE value both drop because higher β corresponds to a sparser reconstruction and so, fewer number of POD modes. The drop in sNRMSE is expected as well since using too many POD modes may result in over-fitting to the noisy data and so, higher error compared to the *"CFD ref"*. Both values drop until $\beta = 5 \times 10^{-4}$ where the lowest reconstruction error is obtained by using only 12 out of 139 POD modes. Afterward, the number of POD modes keeps going down but the reconstruction error increases which means the number of POD modes used for reconstruction is too few.

For some given 4D-Flow data, the proper choice of the LASSO regularization parameter β remains an open question. One possible method of finding the proper value of β is the K-fold cross-validation method [62]. Another method might be taking one snapshot from the reconstructed hi-resolution CFD results, down-sampling that and adding noise to that and then, trying to see what beta value we get the best hi-resolution reconstruction for. The beta value found this way may be taken as a proper selection. We will leave this question to be addressed in further studies. The problem of selecting an appropriate weighting of various regularization parameters exists in other regularization-based denoising methods such as TV as well.

3.5 Conclusion

In this chapter, a novel algorithm was developed that could be used to couple patient-specific CFD simulation with 4D-Flow measurements by use of POD and lasso regularization. The efficacy of the method has been tested using a numerical phantom. This method is capable of reconstructing data to any arbitrary high resolution (depending on the resolution of the CFD mesh) and reconstruct many flow details. Therefore, this method can potentially improve the ability to accurately derive relevant secondary hemodynamic parameters such as wall shear stresses and pressure gradients at a much higher resolution than 4D-Flow.

Chapter 4

Dynamic Denoising and Gappy Data Reconstruction Based on DMD and Discrete Cosine Transform

4.1 Introduction

Dynamic Mode Decomposition (DMD) was a concept that was first introduced by Schmid and Sesterhenn [12] to study the spatial dynamic modes of fluid flow [14]. DMD approximates the nonlinear dynamics underlying a given time-varying dataset in terms of a linear auto-regressive model by extracting a set of mode shapes and their corresponding eigenvalues, where the mode shapes represent the spatial spread of dominant features and the eigenvalue associated with each mode shape specifies how that feature evolves over time in terms of the frequency of oscillation and the rate of growth or decay. Rowley et al. [16] envisioned DMD as an approximation to the modes of Koopman operator, which is an infinite-dimensional linear representation of nonlinear finite-dimensional dynamics [14]. Even though DMD was initially meant to be used for extracting dynamic information from flow fields [13], soon it found new applications in other areas of study as a powerful tool for analyzing the dynamics of nonlinear systems. Kutz et al. [15] [15] expanded the theory of DMD to handle mapping between paired datasets. Jovanovic et al. [17] proposed the sparsity-promoting DMD (spDMD) to obtain a sparse representation of the system dynamics by

limiting the number of dynamic modes through an l_1-regularization approach. In 2015, the extended DMD (EDMD) was introduced by Williams et al. [18] to approximate the leading eigenvalues, eigenfunctions, and modes of the Koopman operator. The EDMD is a computationally intensive algorithm since it requires the choice of a rich dictionary of basis functions in order to produce an approximation of the Koopman eigenfunctions. The richer the dictionary is, the more time it takes to compute the inner products which are a key part of EDMD algorithm. In an attempt to overcome this issue, Williams et al. [19] proposed the kernel-based DMD (KDMD) in 2015. In this approach, rather than choosing the dictionary of the basis functions explicitly, they are defined implicitly by the choice of a kernel function. The kernel function resolves the computational intensity issue of EDMD by finding the inner products of the basis functions without the need to having them defined explicitly.

An initial attempt for incorporating compressed sensing in DMD was made by Guéniat et al. [20], where a subset of an originally-large dataset was taken by non-uniform sampling and was used for finding the temporal coefficients (eigenvalues) through solving an optimization problem. Further, the corresponding modes were found by solving a set of linear equations which involved the fully-sampled dataset. This makes the proposed algorithm (known as NU-DMD) impractical in case the fully-sampled dataset is not available. Another approach for incorporating compressed sensing in DMD (known as csDMD) was developed by [21]. In csDMD, the DMD eigenvalues are obtained from a sub-sampled dataset (similar to NU-DMD), which has the advantage of reducing computation time and then, the full DMD mode shapes are reconstructed through using an l_1-minimization scheme based on a chosen set of basis vectors. In contrast to NU-DMD, csDMD does not need the fully-sampled dataset in order to recover the mode shapes.

One of the initial attempts to deal with the issues involved in recovering a dataset from gappy data is presented in [63]. The proposed method relies on the presence of a set of empirical eigenfunctions which represent an ensemble of like datasets and hence, the fully-sampled dataset is reconstructed based on these empirical eigenfunctions. In case there is no such set available, they described a technique to build one from an ensemble of marred samples. In the case of marred samples, it is assumed there are several marred samples taken from each face, each one taken with a different mask. Also, it is implicitly assumed that for each pixel there is at least one sample available. If there is a pixel which is not included in any marred sample, this method cannot recover it.

Another well-known method for gappy data reconstruction is the Gappy Proper Orthogonal Decomposition (POD) method [64, 65], which was proposed as an extension to POD considering the incomplete datasets. POD captures most of the phenomena in a large amount of high-dimensional dataset while representing it in a low-dimensional space which causes a significant reduction in required computational power [66]. This technique has been used in various problems such as fluid dynamics [66], active control [67], and image reconstruction [56, 68], to name a few. The original POD uses the fully-sampled dataset in order to reconstruct the POD basis functions. Even though Gappy POD aims at reconstructing gappy datasets, it, in fact, relies on the presence of a set of completely-known standard POD basis vectors which we believe makes the whole method inapplicable when there is no such set available. In chapter 3, a POD-based method for denoising and spatial resolution enhancement of 4D-Flow MRI datasets is proposed. This method uses a set of POD basis vectors as the reconstruction basis where the set of POD basis vectors is derived from the results of a computational fluid dynamics (CFD) simulation. Even though this method is shown to outperform the competing state-of-the-art denoising methods, the fact that it is specifically developed for noisy 4D-Flow MRI datasets makes it impractical for the datasets resulting from other types of dynamic systems. None of these methods take into consideration the dynamics of a given dataset.

In the work presented here, an approach similar to csDMD is taken. With csDMD, the aim is to reconstruct the DMD mode shapes based on some given set of basis vectors whereas in our approach, called DMDct hereafter, given the DMD eigenvalues obtained from the sub-sampled dataset, the full dataset is reconstructed through an l_1-minimization scheme. Like csDMD, DMDct relies on the proper choice of the underlying basis functions. In this paper, we are specifically focusing on 2D problems defined over a rectangular grid of equally-spaced nodes. By considering this specific geometry, we can take the one-dimensional discrete cosine transform (DCT) basis vectors and use them for building the two-dimensional basis vectors implicitly, hence requiring less memory.

4.2 Method

The DMDct method is derived for real-valued two-dimensional problems defined over a rectangular mesh of equally-spaced nodes as depicted in Figure 4.1. For each snapshot, only a subset of its

elements is observed which is obtained by applying a pre-defined random sampling mask. The mask is defined as a set of pairs of (i, i') indices, shown as M, for which the samples are taken. All observed elements of each snapshot $\mathbf{S}_k$ are vectorized and represented as a real-valued data vector $\mathbf{s}_k$ of length N_s, where N_s is the number of sampling points. The data vectors $\mathbf{s}_k$ are taken as the input to the DMDct algorithm. First, the $N_s \times m$ matrix $\mathbf{Z}_s = [\mathbf{s}_0 \ \dots \ \mathbf{s}_{m-1}]$ is constructed and the exact DMD method is applied to that to obtain DMD eigenvalues $\boldsymbol{\lambda}$ (Section 2.4). Then, the spatial component of DMD is reconstructed based on the DCT basis vectors by taking random samples from the fully-sampled dataset while maintaining the sparsity of reconstruction coefficient matrices through an l_1-regularization scheme (Section 4.2.2). Finally, each snapshot is reconstructed in full using the calculated reconstruction coefficient matrices and the DCT basis vectors.

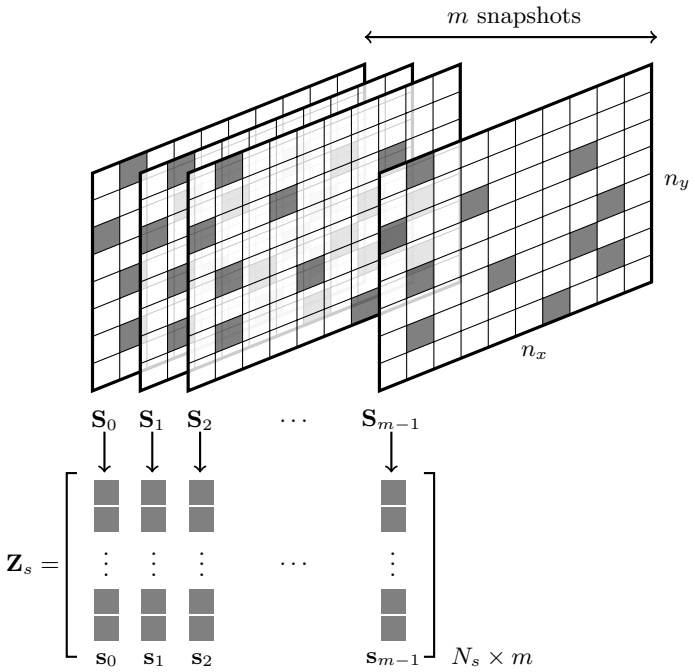


Figure 4.1: Schematic representation of the designated structure of the input data of DMDct algorithm where each snapshot $\mathbf{S}_k$ is an $n_y \times n_x$ matrix of real values. The randomly-sampled points of each snapshot (colored in gray) are vectorized and represented as a real-valued data vector $\mathbf{s}_k$ of length N_s, where N_s is the number of sampling points. The sampling mask remains the same for all snapshots.

4.2.1 Exact DMD

The Exact DMD method [15] is briefly introduced here since DMDct relies on that for finding the eigenvalues and reconstructing the data. Given a sequential set of m data vectors $\mathbf{z}_k$ shown as an $N \times m$ matrix $\mathbf{Z} = [\mathbf{z}_0 \dots \mathbf{z}_{m-1}]$, the exact DMD method gives us the set of r DMD modes $\boldsymbol{\phi}_j$ and their corresponding eigenvalues λ_j (Algorithm 2). The DMD modes and eigenvalues together describe how each vector $\mathbf{z}_{k-1}$ evolves in time and results in the vector $\mathbf{z}_k$. By showing all DMD modes as the $N \times r$ matrix $\boldsymbol{\Phi} = [\boldsymbol{\phi}_1 \dots \boldsymbol{\phi}_r]$ and the corresponding eigenvalues as the $r \times r$ diagonal matrix $\boldsymbol{\Lambda} = \text{diag}(\lambda_1 \dots \lambda_r)$, exact DMD lets us reconstruct the k-th vector as

$$\tilde{\mathbf{z}}_k = \boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^\dagger \mathbf{z}_{k-1} \tag{4.1}$$

where $\tilde{\mathbf{z}}_k$ is the reconstruction of the vector $\mathbf{z}_k$ and $\boldsymbol{\Phi}^\dagger$ is the pseudo-inverse of $\boldsymbol{\Phi}$. When the DMD modes are independent, the pseudo-inverse of $\boldsymbol{\Phi}$ is given as $\boldsymbol{\Phi}^\dagger = (\boldsymbol{\Phi}^*\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^*$ where * denotes the conjugate transpose. In such case, each vector $\mathbf{z}_k$ can be reconstructed based on the first vector ($\mathbf{z}_0$) as

$$\tilde{\mathbf{z}}_k = \boldsymbol{\Phi}\boldsymbol{\Lambda}^k\boldsymbol{\Phi}^\dagger \mathbf{z}_0 \tag{4.2}$$

By showing all reconstructed vectors as the matrix $\tilde{\mathbf{Z}} = [\tilde{\mathbf{z}}_0 \dots \tilde{\mathbf{z}}_m]$, it can be shown that

$$\tilde{\mathbf{Z}} = \boldsymbol{\Phi}\mathbf{D}\mathbf{V}, \ \mathbf{D} \triangleq \text{diag}(\boldsymbol{\Phi}^\dagger \mathbf{z}_0) \tag{4.3}$$

where $\mathbf{V}$ is the $r \times m$ *pseudo-Vandermonde* matrix of the eigenvalues defined as

$$\mathbf{V} = \begin{bmatrix} 1 & \lambda_1 & \lambda_1^2 & \dots & \lambda_1^{m-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \lambda_r & \lambda_r^2 & \dots & \lambda_r^{m-1} \end{bmatrix}_{r \times m} \tag{4.4}$$

4.2.2 Formulation of DMDct

In DMD reconstruction, given as Equation (4.3), we know the matrix product $\boldsymbol{\Phi}\mathbf{D}$ as the spatial component while the matrix $\mathbf{V}$ represents the temporal evolution of the spatial component. Let us

Algorithm 2. The pseudocode of the Exact DMD algorithm.

Data:

- $\mathbf{Z} = [\mathbf{z}_0 \dots \mathbf{z}_{m-1}]$: the $N \times m$ matrix of sequential data vectors
- r: the number of modes to pick

Result:

- $\mathbf{\Phi}$: the matrix of DMD modes
- $\boldsymbol{\lambda}$: the vector of DMD eigenvalues

1 Find the SVD of $\mathbf{X} = [\mathbf{z}_0 \dots \mathbf{z}_{m-2}]$ such that $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$
2 Truncate $\mathbf{U}$ to the first r columns
3 Truncate $\mathbf{\Sigma}$ to the upper-left $r \times r$ matrix
4 Truncate $\mathbf{V}^*$ to the first r rows
5 Define $\tilde{\mathbf{A}} \triangleq \mathbf{U}^*\mathbf{Y}\mathbf{V}\mathbf{\Sigma}^{-1}$ where $\mathbf{Y} = [\mathbf{z}_1 \dots \mathbf{z}_{m-1}]$
6 Find the eigenvalues $\boldsymbol{\lambda}$ and eigenvectors $\mathbf{W}$ of $\tilde{\mathbf{A}}$, i.e., $\tilde{\mathbf{A}}\mathbf{W} = \mathbf{W}\,\mathrm{diag}(\boldsymbol{\lambda})$
7 Compute the DMD modes $\mathbf{\Phi} \triangleq \mathbf{Y}\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{W}$
8 **return** $\mathbf{\Phi}$, $\boldsymbol{\lambda}$

assume there is a set of basis vectors $\mathbf{u}_l$ represented as an $N \times s$ matrix $\mathbf{U} = [\mathbf{u}_1 \dots \mathbf{u}_s]$ based on which the matrix product $\mathbf{\Phi}\mathbf{D}$ can be approximated as

$$\mathbf{\Phi}\mathbf{D} \approx \mathbf{U}\mathbf{C} \tag{4.5}$$

where $\mathbf{C}$ is the $s \times r$ matrix of the unknown complex coefficients. In many cases, the N-dimensional data vectors $\mathbf{z}_k$ and the basis vectors $\mathbf{u}_l$ are real-valued. Based on this assumption and the approximation given above, the reconstructed real-valued data matrix $\hat{\mathbf{Z}}$ is defined as

$$\hat{\mathbf{Z}} \triangleq \Re\left(\mathbf{U}\mathbf{C}\mathbf{V}\right) = \mathbf{U}\Re\left(\mathbf{C}\mathbf{V}\right) \tag{4.6}$$

$$\rightarrow \hat{\mathbf{z}}_k = \mathbf{U}\sum_{j=1}^{r}\left(\alpha_{jk}\mathbf{a}_j - \beta_{jk}\mathbf{b}_j\right) \tag{4.7}$$

where $\mathbf{a}_j$ and $\mathbf{b}_j$ are the respective real and imaginary parts of the j-th column of $\mathbf{C}$ and $\lambda_j^k = \alpha_{jk} + i\beta_{jk}$. For the class of two-dimensional problems addressed here, each snapshot $\mathbf{S}_k$ is an $n_y \times n_x$ matrix of real values for which Equation (4.7) may be rewritten as

$$\hat{\mathbf{S}}_k = \mathbf{U}_y\left(\mathbf{U}_x\sum_{j=1}^{r}\left(\alpha_{jk}\mathbf{A}_j^T - \beta_{jk}\mathbf{B}_j^T\right)\right)^T \tag{4.8}$$

where $\hat{\mathbf{S}}_k$ is the real-valued reconstruction of k-th snapshot, $\mathbf{U}_y$ is the $n_y \times s_y$ matrix of the basis vectors along the columns of $\mathbf{S}_k$, $\mathbf{U}_x$ is the $n_x \times s_x$ matrix of the basis vectors along the rows of $\mathbf{S}_k$, and $\mathbf{A}_j$ and $\mathbf{B}_j$ are the $s_y \times s_x$ matrices of unknown coefficients corresponding to the j-th dynamic mode. The columns of $\mathbf{U}_x$ and $\mathbf{U}_y$ are the basis vectors. For the special case of DCT basis vectors, Equation (4.8) may be rephrased as

$$\hat{\mathbf{S}}_k = \mathcal{D}_y^{-1}\left(\mathcal{D}_x^{-1}\left(\sum_{j=1}^{r}\left(\alpha_{jk}\mathbf{A}_j^T - \beta_{jk}\mathbf{B}_j^T\right)\right)^T\right) \tag{4.9}$$

where the operator $\mathcal{D}_x$ and its inverse $\mathcal{D}_x^{-1}$ are defined as

$$\mathcal{D}_x\left(\mathbf{X}\right) \triangleq \mathbf{U}_x^T\mathbf{X},\ \mathcal{D}_x^{-1}\left(\mathbf{X}\right) \triangleq \mathbf{U}_x\mathbf{X} \tag{4.10}$$

The forward and inverse operators $\mathcal{D}_x$ and $\mathcal{D}_x^{-1}$, respectively, apply the forward and inverse DCT transforms of length s_x to the columns of their arguments. The forward and inverse operators $\mathcal{D}_y$ and $\mathcal{D}_y^{-1}$ are defined similarly. Most numerical analysis packages provide forward and inverse DCT transforms as built-in functions hence eliminating the need to define the matrices $\mathbf{U}_x$ and $\mathbf{U}_y$ explicitly.

Given the sampling mask M, the reconstruction error of the k-th snapshot is defined as

$$\mathbf{E}_k = \left[e_{ii'}^{(k)}\right]_{n_y \times n_x},\ e_{ii'}^{(k)} = \begin{cases} \hat{s}_{ii'}^{(k)} - s_{ii'}^{(k)} & ,(i,i') \in M \\ 0 & ,(i,i') \notin M \end{cases} \tag{4.11}$$

$$E_k \triangleq \|\mathbf{E}_k\|_F^2 = \sum_{\forall (i,i') \in M}\left(\hat{s}_{ii'}^{(k)} - s_{ii'k}^{(k)}\right)^2 \tag{4.12}$$

where $\hat{s}_{ii'}^{(k)}$ and $s_{ii'}^{(k)}$ are the respective (i,i') elements of the matrices $\hat{\mathbf{S}}_k$ and $\mathbf{S}_k$. The unknown matrices $\mathbf{A}_j$ and $\mathbf{B}_j$ are found by solving the l_1-regularization problem

$$\underset{\mathbf{A}_j,\mathbf{B}_j}{\operatorname{argmin}}\left(E + \beta\sum_{j=1}^{r}\left(\|\mathbf{A}_j\|_1 + \|\mathbf{B}_j\|_1\right)\right),\ E \triangleq \frac{1}{2}\sum_{k=0}^{m-1}E_k \tag{4.13}$$

Some l_1-regularization methods rely on derivatives of E with respect to the unknown matrices $\mathbf{A}_j$

and $\mathbf{B}_j$. The derivatives are given as

$$\frac{\partial E}{\partial \mathbf{A}_j} = \mathcal{D}_y\left(\mathcal{D}_x\left(\mathbf{F}_j^T\right)^T\right),\ \mathbf{F}_j = \left[f_{ii'}^{(j)}\right]_{n_y \times n_x},\ f_{ii'}^{(j)} = \sum_{k=0}^{m-1} e_{ii'}^{(k)} \alpha_{jk} \tag{4.14}$$

$$\frac{\partial E}{\partial \mathbf{B}_j} = \mathcal{D}_y\left(\mathcal{D}_x\left(\mathbf{G}_j^T\right)^T\right),\ \mathbf{G}_j = \left[g_{ii'}^{(j)}\right]_{n_y \times n_x},\ g_{ii'}^{(j)} = \sum_{k=0}^{m-1} e_{ii'}^{(k)} \beta_{jk} \tag{4.15}$$

The implementation steps of DMDct are listed as Algorithm 3.

4.3 Results

To compare DMDct against csDMD as well as show its effectiveness in dynamic denoising and reconstruction, three tests were performed. For each case, both csDMD and DMDct were tried with several levels of sparsity and then, the best results were picked for comparison. The root mean square error (RMSE) defined below was used as the comparison metric for all noise-free cases

$$\text{RMSE} \triangleq \frac{1}{\sqrt{n}} \left\|\mathbf{Z}_{rec} - \mathbf{Z}_{ref}\right\|_F \tag{4.16}$$

where $\mathbf{Z}_{rec}$ is the reconstructed dataset, $\mathbf{Z}_{ref}$ is the reference dataset, and n is the total number of elements of the dataset. A lower RMSE value represents a better reconstruction. For the noisy cases, the peak value to noise ratio (PVNR), inspired by PVNR defined in [48] and as defined below was used as the comparison metric

$$\text{PVNR} \triangleq 20 \log_{10} \frac{\max\left|\mathbf{Z}_{ref}\right|}{\text{RMSE}}\ (\text{dB}) \tag{4.17}$$

A higher PVNR value represents a less-noisy reconstruction.

The original implementation of csDMD was partly based on the method of compressive sampling matching pursuit (CoSaMP) [69]. We used the Orthant-Wise Limited-memory Quasi-Newton (OWL-QN) algorithm [11] to solve the l_1-regularization problem; thus, to ensure all the differences between the results of the two methods are due to the methods themselves and not the l_1-regularization algorithms, the csDMD was re-implemented by using OWL-QN rather than CoSaMP.

Algorithm 3. The implementation steps of DMDct.

Data:

- $\mathbf{s}_k$: the m randomly-sampled real-valued data vectors of length N_s (see Figure 4.1)
- M: the sampling mask (set of N_s pairs of indices (i, i') for which the samples are taken)
- r: the number of modes to pick
- s_x, s_y: the respective lengths of DCT transforms along the rows and columns of each snapshot
- β: l_1-regularization parameter
- $\mathbf{A}_j, \mathbf{B}_j$ $(1 \leq j \leq r)$: initial values of the $s_y \times s_x$ unknown matrices of coefficients (optional, can be zero)

Result:

- $\hat{\mathbf{S}}_k$ $(0 \leq k < m)$: the m reconstructed snapshots

Form the sampled data matrix $\mathbf{Z}_s = [\mathbf{s}_0 \ \ldots \ \mathbf{s}_{m-1}]$
Apply Exact DMD to the matrix $\mathbf{Z}_s$ to get the vector of DMD eigenvalues $\boldsymbol{\lambda}$(see Algorithm 2)
Initialize the unknown matrices $\mathbf{A}_j, \mathbf{B}_j$ $(1 \leq j \leq r)$
repeat
  $E \leftarrow 0$
  for $k = 0$ **to** $m - 1$ **do**
    $\mathbf{T} \leftarrow \mathbf{0}$
    for $j = 1$ **to** r **do**
      $\alpha_{jk} \leftarrow \Re\left(\lambda_j^k\right)$, $\beta_{jk} \leftarrow \Im\left(\lambda_j^k\right)$
      $\mathbf{T} \leftarrow \mathbf{T} + \alpha_{jk}\mathbf{A}_j - \beta_{jk}\mathbf{B}_j$
    end
    $\hat{\mathbf{S}}_k \leftarrow \mathcal{D}_y^{-1}\left(\mathcal{D}_x^{-1}\left(\mathbf{T}^T\right)^T\right)$
    $\hat{\mathbf{s}}_k \leftarrow$ vectorized sample of $\hat{\mathbf{S}}_k$ according to the sampling mask M
    $\mathbf{s}_k \leftarrow$ vectorized sample of $\mathbf{S}_k$ according to the sampling mask M
    $E \leftarrow E + \frac{1}{2}\left\|\hat{\mathbf{s}}_k - \mathbf{s}_k\right\|_2^2$
  end
  if *convergence not achieved* **then**
    for $j = 1$ **to** r **do**
      Update $\mathbf{A}_j$ and $\mathbf{B}_j$ according to the chosen l_1-regularization method (use Equations (4.14) and (4.15) to find the Jacobians $\frac{\partial E}{\partial \mathbf{A}_j}$ and $\frac{\partial E}{\partial \mathbf{B}_j}$ if needed)
    end
  end
until *convergence achieved*
return *all* $\hat{\mathbf{S}}_k$ $(0 \leq k < m)$

4.3.1 DMD Mode-Shapes Reconstruction

As the first test, the vorticity of the double-gyre flow (as represented in [21]) was taken and used. The vorticity w is given as

$$\begin{aligned} w = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} =& \pi A \cos\left(\pi f(x,t)\right) \sin\left(\pi y\right) \frac{\partial^2 f}{\partial x^2} - \pi^2 A \sin\left(\pi f(x,t)\right) \sin\left(\pi y\right) \left(\frac{\partial f}{\partial x}\right)^2 \\ & - \pi^2 A \sin\left(\pi f(x,t)\right) \sin\left(\pi y\right) \end{aligned} \tag{4.18}$$

where $A = 0.1$, $\omega = \frac{2\pi}{10}$, $\epsilon = 0.25$, and

$$f(x,t) = \epsilon \sin\left(\omega t\right) x^2 + x - 2\epsilon \sin\left(\omega t\right) x \tag{4.19}$$

The equation was evaluated over the bounded region $[0,2] \times [0,1]$ for 10 s with time intervals of 0.05 s which resulted in 201 snapshots. The region was discretized as a 512×256 mesh. The number of sampling points was 2500 and they were randomly spread over the region. The same sampling mask was used for both methods. Due to the very few numbers of nonzero Fourier coefficients, only 10 DCT basis vectors along each spatial direction were used ($s_x = s_y = 10$). The csDMD method directly resulted in the reconstruction of DMD mode shapes whereas DMDct resulted in the reconstruction of the fully-sampled dataset. After the fully-sampled dataset was reconstructed by DMDct, the exact DMD method was applied to get the DMD mode shapes which were further used for comparison. All DMDs were performed with nine modes. Since the complex eigenvalues come in pairs of conjugate numbers, only those having non-negative imaginary parts are represented here. Note that, similar to an eigenvector, a mode shape may be multiplied by any non-zero scalar without making a difference. Thus, to compare the mode shapes, they should be aligned with each other prior to making any comparison. Given $\boldsymbol{\phi}_i$ and $\boldsymbol{\psi}_i$ are the vectorized mode shapes corresponding to the i-th eigenvalue resulted from DMD and csDMD, respectively, the complex scalar c_i that results in the best alignment of the vector $\boldsymbol{\psi}_i$ with the vector $\boldsymbol{\phi}_i$ is found by solving the following minimization problem

$$c_i = \underset{c}{\operatorname{argmin}} \|\boldsymbol{\phi}_i - c\boldsymbol{\psi}_i\|_2^2 = \frac{\boldsymbol{\phi}_i^T \bar{\boldsymbol{\psi}}_i}{\boldsymbol{\psi}_i^T \bar{\boldsymbol{\psi}}_i} \tag{4.20}$$

Similarly, for the vectorized mode shape $\boldsymbol{\theta}_i$ resulted from DMDct, the alignment factor d_i is found as

$$d_i = \underset{d}{\operatorname{argmin}} \|\boldsymbol{\phi}_i - d\boldsymbol{\theta}_i\|_2^2 = \frac{\boldsymbol{\phi}_i^T \bar{\boldsymbol{\theta}}_i}{\boldsymbol{\theta}_i^T \bar{\boldsymbol{\theta}}_i} \tag{4.21}$$

Thus, the comparison was made between the vectors $\boldsymbol{\phi}_i$ and the corresponding aligned vectors $c_i\boldsymbol{\psi}_i$ and $d_i\boldsymbol{\theta}_i$.

Figure 4.2 shows the real parts of the mode shapes of the first five DMD modes related to the eigenvalues with non-negative imaginary parts and the reconstruction of their mode shapes. The top row shows the mode shapes obtained by applying exact DMD on the fully-sampled dataset, whereas the second and third rows show the aligned csDMD and DMDct reconstructions, respectively. Each column is titled with the corresponding eigenvalue. Both csDMD and DMDct resulted in the reconstruction of the mode shapes with correlation coefficients of approximately 1 which means the reconstructed mode shapes almost identically resembled the references.

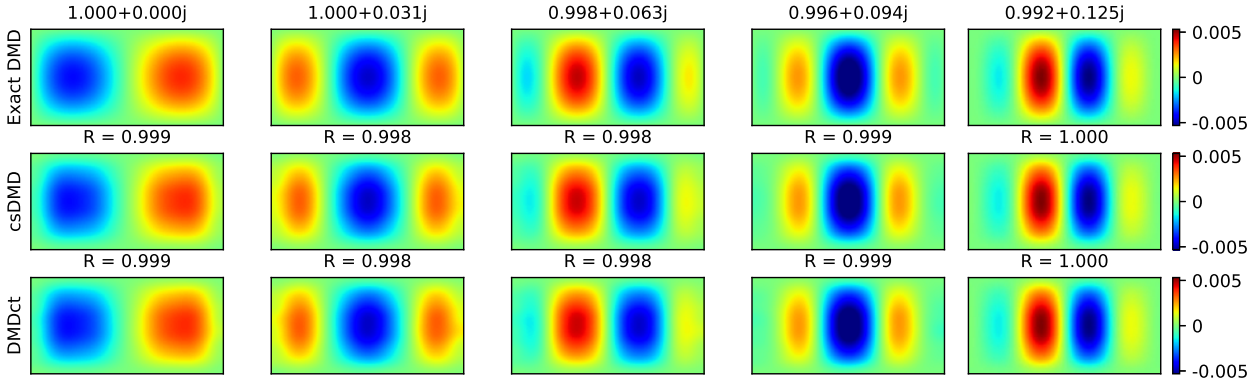


Figure 4.2: The real parts of the mode shapes of the first five DMD modes corresponding to the eigenvalues with non-negative imaginary parts for the double-gyre dataset. The top row shows the mode shapes obtained by applying exact DMD on the fully-sampled dataset. The second and third rows show the aligned csDMD and DMDct reconstructions, respectively. The corresponding eigenvalues are represented above the columns. The Pearson correlation coefficients between the aligned reconstructed mode shapes and those of exact DMD are shown as well.

Five sample snapshots of the fully-sampled dataset reconstruction are shown in Figure 4.3. Both methods resulted in reconstruction RMSE of 0.002. The top row of Figure 4.3 shows the reference snapshots. The samples are shown in the second row. The third and fourth rows show the reconstruction of csDMD and DMDct, respectively.

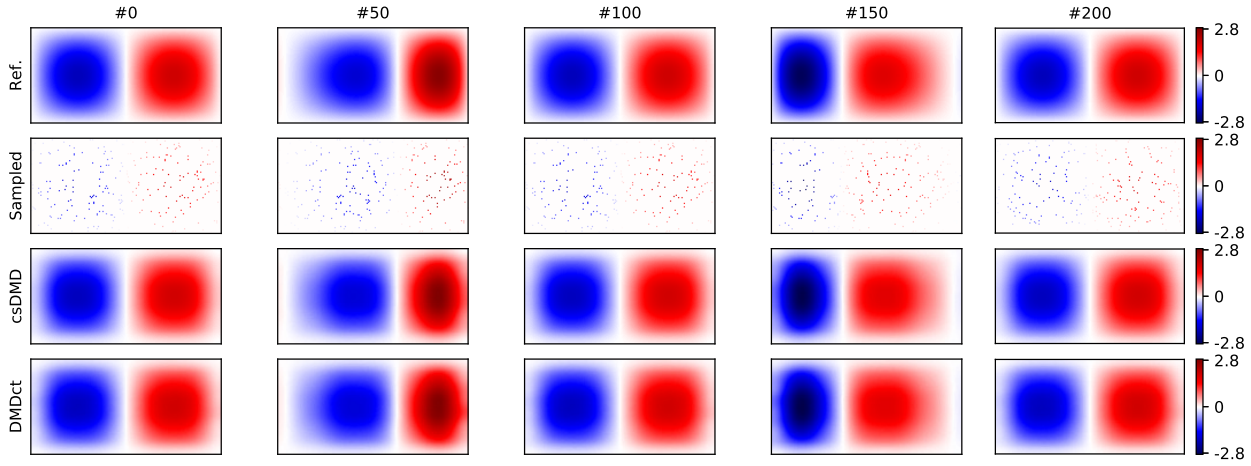


Figure 4.3: Five sample snapshots of the fully-sampled dataset reconstruction for the double-gyre dataset. The top row shows the reference snapshots titled with the snapshot numbers. The second row shows the random samples taken. The third and the fourth rows show csDMD and DMDct reconstructions, respectively.

4.3.2 Dynamic Denoising and Reconstruction

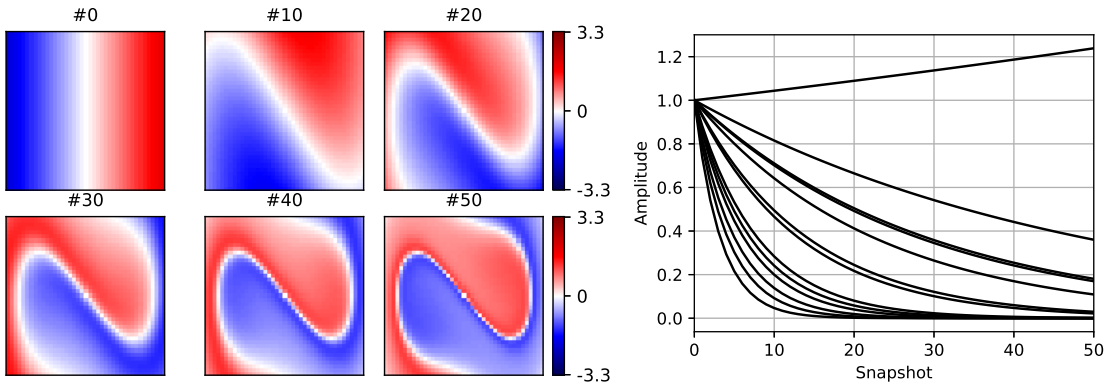
As the second test, the unforced Duffing equation taken from [18] was used to generate the test dataset. The governing differential equation is

$$\ddot{x} = -\delta\dot{x} - x(\gamma + \alpha x^2) \tag{4.22}$$

where $\delta = 0.5$, $\gamma = -1$, and $\alpha = 1$. The equation was solved over the region $x, \dot{x} \in [-2, 2]$, which was discretized as a 41×41 mesh. For each node of the mesh, the corresponding values of x and $\dot{x}$ were taken as the initial conditions and the ODE was solved for 5 s during which the snapshots were taken every 0.1 s resulting in a total of 51 snapshots. Even though the numerical solution resulted in both x and $\dot{x}$ values, only x values were taken and used as the test dataset. Figure 4.4 shows six sample snapshots of the reference dataset.

Two cases are presented here for comparison. The first case does not have a gap, whereas the second case has a rectangular gap. Both cases were evaluated with noise-free and noisy samples. In all cases, 20% of the available data of each snapshot was taken as the measurement samples and were used for reconstruction. For each case, twenty different random sampling masks were tested. For each mask, the sampling locations remained the same over all the snapshots.

The noisy cases were to study the effect of measurement noise and to see how well the two



(**a**) (**b**)

Figure 4.4: The reference Duffing dataset. (**a**) Six sample snapshots are shown. All 51 snapshots were used in calculations. (**b**) The amplitudes of the dynamic mode shapes are shown. Since the complex eigenvalues come in pairs of conjugate numbers, only those with non-negative imaginary parts are presented.

methods could denoise the data. To make the noisy dataset, random Gaussian noise with the standard deviation of 0.25 was added to the reference dataset. The PVNR metric was calculated only for the noisy reconstructions. The same set of basis vectors was used by both methods. For the noise-free samples, the maximum number of basis vectors were used ($s_x = s_y = 41$), whereas, for the noisy samples, a reduced set of basis vectors was incorporated ($s_x = s_y = 20$), hence dropping the high-frequency components from reconstruction. The eigenvalues derived by csDMD were used for DMDct reconstruction as well. The number of DMD modes to use was found through the method of singular value hard thresholding (SVHT) [70]. According to SVHT, the number of DMD modes for the noise-free and noisy samples was taken as 25 and 5, respectively. Figure 4.4b shows the amplitudes of the dynamic mode shapes of the reference Duffing dataset. In the figures depicting the snapshots, the first (#0), the middle (#25), and the last (#50) snapshots of the first sampling mask are presented for comparison.

The csDMD method aims at reconstructing the mode shapes and not the fully-sampled dataset. Since no fully-sampled snapshot is available, it is not possible to reconstruct the whole dataset solely based on Exact DMD framework by simply marching forward/backward in time using Equation (4.2). One possible workaround is to find the optimal amplitudes of DMD modes by minimizing the RMS of reconstruction error as proposed in [17] which leads to

$$\mathbf{b}_{opt} = \left((\mathbf{\Phi}_s^* \mathbf{\Phi}_s) \odot \left(\overline{\mathbf{V}\mathbf{V}^*} \right) \right)^{-1} \overline{\mathrm{diag}(\mathbf{V}\mathbf{Z}^* \mathbf{\Phi}_s)} \tag{4.23}$$

where $\odot$ is the element-wise product of two matrices and $\mathbf{\Phi}_s$ is the matrix of mode shapes as reconstructed by csDMD but only the rows corresponding to the sampled points are kept. Then, the fully-sampled dataset can be reconstructed in full as

$$\tilde{\mathbf{Z}}_{cs} = \mathbf{\Phi}\,\mathrm{diag}(\mathbf{b}_{opt})\mathbf{V} \tag{4.24}$$

Equation (4.24) was used for csDMD reconstruction.

4.3.3 No-Gap Reconstruction

In this case, the reference dataset without any gap was reconstructed by using the two methods. Figures 4.5a and 4.5b respectively show the sample snapshots of the noise-free and noisy reconstructions for the first sampling mask. The noisy dataset had the total PVNR of 19.0 dB and RMSE of 0.250, as depicted in the top row of Figure 4.5b. In Figure 4.5a, the top row shows the reference and the second row shows the sampling mask. The third row shows the sample noise-free snapshots as reconstructed by csDMD method resulting in an RMSE value of 0.291. The bottom row shows the same snapshots as reconstructed by the DMDct method. The RMSE value of DMDct reconstruction is 0.130. In Figure 4.5b, the third row shows the results obtained from the csDMD method by using the noisy samples. This resulted in an RMSE value of 0.182 and PVNR of 21.8 dB. The bottom row shows the results of DMDct reconstruction which resulted in an RMSE value of 0.119 and PVNR of 25.5 dB.

4.3.4 Rectangular Gap Reconstruction

For the second case, a rectangular gap was made in the dataset, as shown in the top rows of Figures 4.6a and 4.6b. The size of the gap was 30×10 with the bottom-left and top-right corners at $(-1.5, 0)$ and $(1.5, 1)$, respectively. The gap covers almost 18% of the area of the region. The first row of Figure 4.6a shows the reference without the gap, which is what both methods were aimed at recovering by filling the gap. The second row shows the noise-free and noisy samples taken by using the first random sampling mask. The third row shows the reconstruction of the csDMD method with the corresponding RMSE values of 0.334 for the noise-free samples and 0.181 for the noisy samples. The bottom row shows the reconstruction of the DMDct method, where the

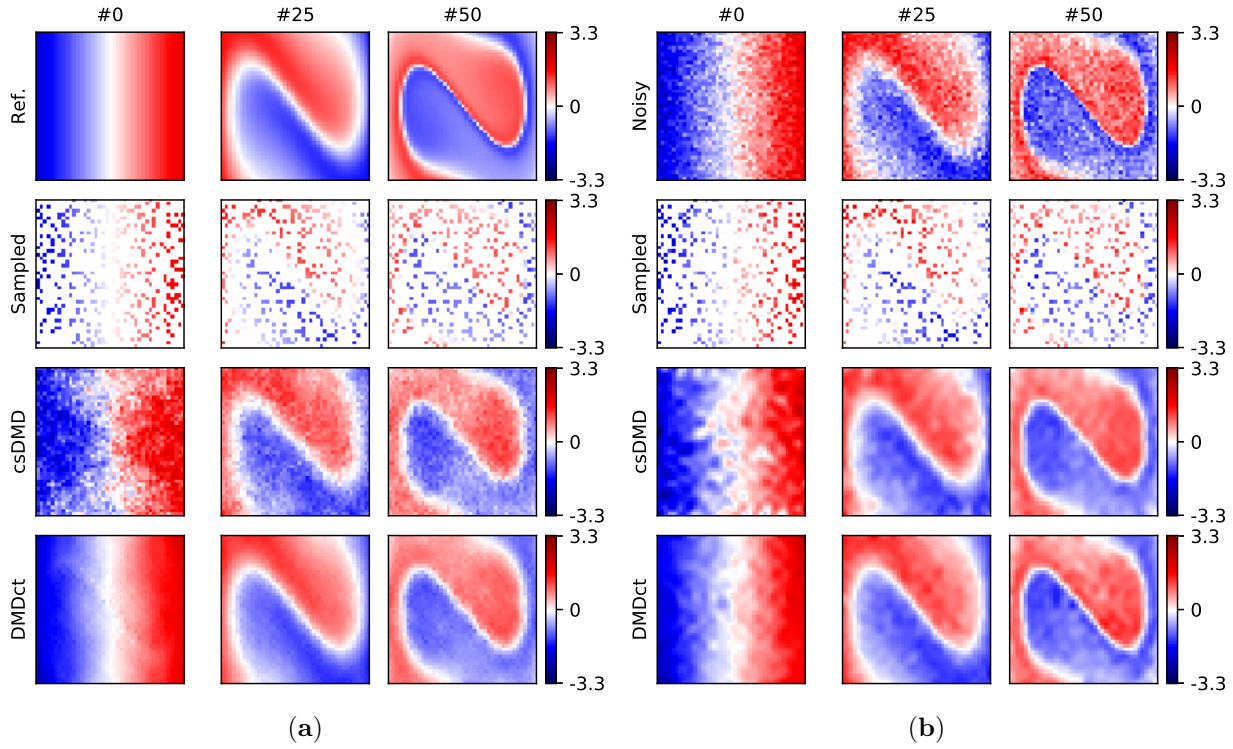


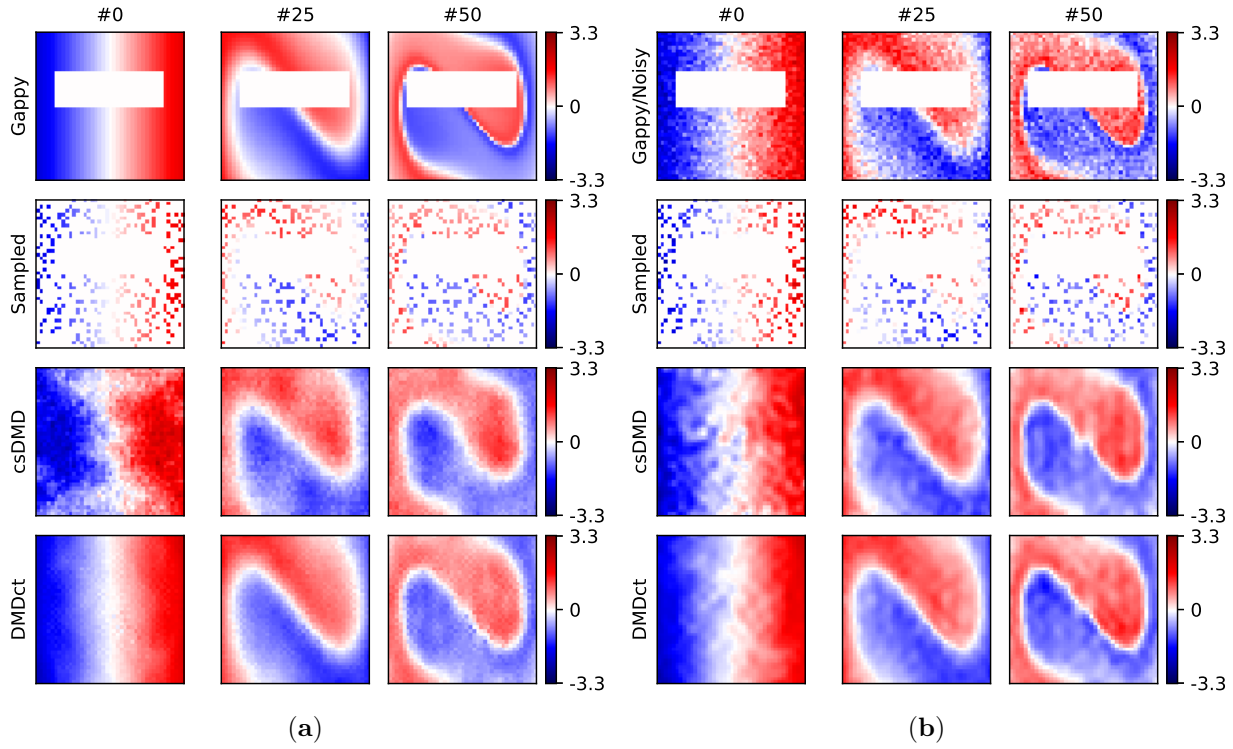
Figure 4.5: Results of Duffing dataset reconstruction using the first random sampling mask with both DMDct and csDMD (no gap). The snapshot numbers are shown at the top of each column. The error metrics are listed in Table 4.1. (**a**) The noise-free case without any gap. (**b**) The noisy case without any gap.

RMSE values were found as 0.154 and 0.138 for the noise-free and noisy samples, respectively. The respective PVNR values of csDMD and DMDct for the noisy case were 21.8 dB and 24.2 dB.

A summary of the error metrics of reconstruction based on the first random sampling mask is presented in Table 4.1 for comparison.

4.3.5 Statistical Analysis

Three-factor analysis of variance was conducted to determine whether the reconstruction error significantly changed with the three factors method, noise, gap, and their interaction. The RMSE was taken as the error metric and the significance level of 0.05 was used. Tukey post hoc analysis was used for desired pairwise comparisons of significant factors. In all four cases, the two methods were found to result in significantly different reconstruction errors (Tukey post hoc test, $p < 0.001$) with the DMDct method having a lower error. The effect of noise on DMDct was insignificant ($p = 0.797$), whereas the error of csDMD for noisy cases was significantly lower than its error for



(**a**) (**b**)

Figure 4.6: Results of Duffing dataset reconstruction using the first random sampling mask with both DMDct and csDMD (rectangular gap). The snapshot numbers are shown at the top of each column. The error metrics are listed in Table 4.1. (**a**) The noise-free case with the rectangular gap (the white hollow). (**b**) The noisy case with the rectangular gap (the white hollow).

the noise-free cases ($p < 0.001$). Both methods resulted in significantly higher errors for the gappy cases ($p < 0.001$). Figure 4.7 shows the mean RMSE values of DMDct and csDMD for the four test cases studied with the error bars showing the standard deviations.

4.3.6 Variation of Parameters

As the third test, a dataset representing the 2D velocity field for the wake behind a cylinder at Reynolds number Re = 100 taken from [71] was used. The size of the mesh grid is 449×199. The dataset consists of 151 snapshots with regular time intervals of 0.2 s. Random Gaussian noise with a known standard deviation was added to both components. Two rectangular gaps were made in the dataset. The size of the first gap was 60×70 with the bottom-left and top-right corners at $(270, 115)$ and $(329, 184)$, respectively. The size of the second gap was 46×46 with the bottom-left corner at $(97, 44)$ and the top-right corner at $(142, 89)$. The aim of this test was to investigate the effect of changing various parameters on the quality of reconstruction. The parameters are noise

Table 4.1: The summary of the error metrics for the first random sampling mask. The numbers given are RMSE values with the PVNR values in dB shown inside parenthesis when applicable. In all cases studied, DMDct resulted in lower reconstruction error than csDMD.

Type of gap	**Region**	**Noise-Free**		**Noisy**		
		csDMD	**DMDct**	**Dataset**	**csDMD**	**DMDct**
None	whole	0.291	**0.130**	0.250 (19.0 dB)	0.182 (21.8 dB)	**0.119** (**25.5 dB**)
Rectangular	inside	0.408	**0.199**	-	0.231	**0.185**
	outside	0.316	**0.142**	-	0.168	**0.126**
	whole	0.334	**0.154**	0.250 (19.1 dB)	0.181 (21.8 dB)	**0.138** (**24.2 dB**)

standard deviation, sampling ratio, number of basis vectors, and number of dynamic modes. The nominal values of the parameters were chosen as noise standard deviation of 0.25, 2% sampling, $s_x = 67, s_y = 30$, and five dynamic modes (according to SVHT). Although both u and v velocity components were used for analysis, only the results corresponding to the u component are presented here. Figure 4.8 shows four sample snapshots of the reference noise-free u velocity components, the reference with noise added, the random sample, reconstructions of csDMD and DMDct, and reconstruction errors for the nominal values of the parameters.

Figure 4.9 shows the effects of the variation of parameters on PVNR values of csDMD and DMDct reconstructions per snapshot. In all sub-figures, the blue and red curves correspond to DMDct and csDMD results, respectively. The solid lines represent the results based on the nominal values. Figure 4.9a shows the effect of changing noise standard deviation. As the noise standard deviation increases, the PVNR values drop but in all snapshots, DMDct results in higher PVNR values than csDMD. The effect of changing the sampling ratio is shown in Figure 4.9b . As expected, increasing the sampling ratio results in higher PVNR values. Figure 4.9c shows the effect of taking different numbers of basis vectors. As the number of basis vectors increases, the PVNR values drop slightly. Finally, the effect of changing the number of dynamic modes is shown in Figure 4.9d. Picking a fewer number of modes than SVHT's result slightly lowers the PVNR values, whereas picking more modes does not make any improvements. The curves corresponding to 5 and 10 modes are almost always overlapping. In all cases studied here, DMDct resulted in higher PVNR values than csDMD in all snapshots.

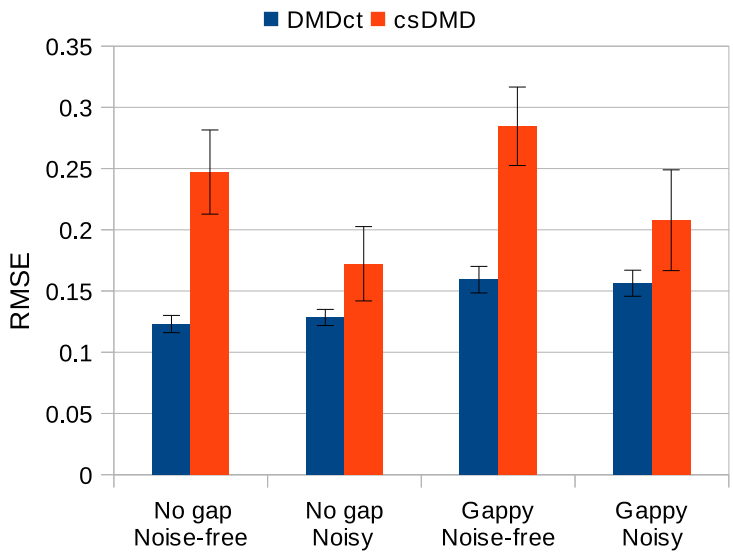


Figure 4.7: The mean RMSE values of the two methods for the four test cases of the Duffing dataset. Each test case consisted of twenty different random sampling masks. The error bars show the standard deviations. A three-factor analysis of variance was conducted to determine whether the reconstruction error significantly changed with the three factors method, noise, gap, and their interaction. Tukey post hoc analysis showed the reconstruction error of DMDct was significantly lower than the error of csDMD in all cases (all $p < 0.001$).

4.4 Discussion

The three tests performed aimed at comparing DMDct vs csDMD in terms of both dynamic mode shape reconstruction and fully-sampled dataset reconstruction based on a sparsely-sampled dataset. While csDMD is developed to reconstruct the mode shapes, DMDct reconstructs the fully-sampled dataset. To use csDMD for fully-sampled dataset reconstruction, the spDMD method was incorporated to find the optimal amplitudes of DMD modes.

The first test showed both methods reconstructed the mode shapes almost identical to the ones resulting from applying exact DMD on the fully-sampled dataset even though a very small set of basis vectors was used. Both methods resulted in RMSE of 0.002 in reconstructing the fully-sampled dataset. These results show neither method outperforms the other in dealing with the test dataset which has a few dynamic modes.

The second test consisted of four cases. In the first case, where there is no gap in the data and the samples are noise-free, csDMD reconstruction shows some glitches, especially in the first snapshot, whereas the DMDct reconstruction has much fewer glitches (Figure 4.6a). The glitches

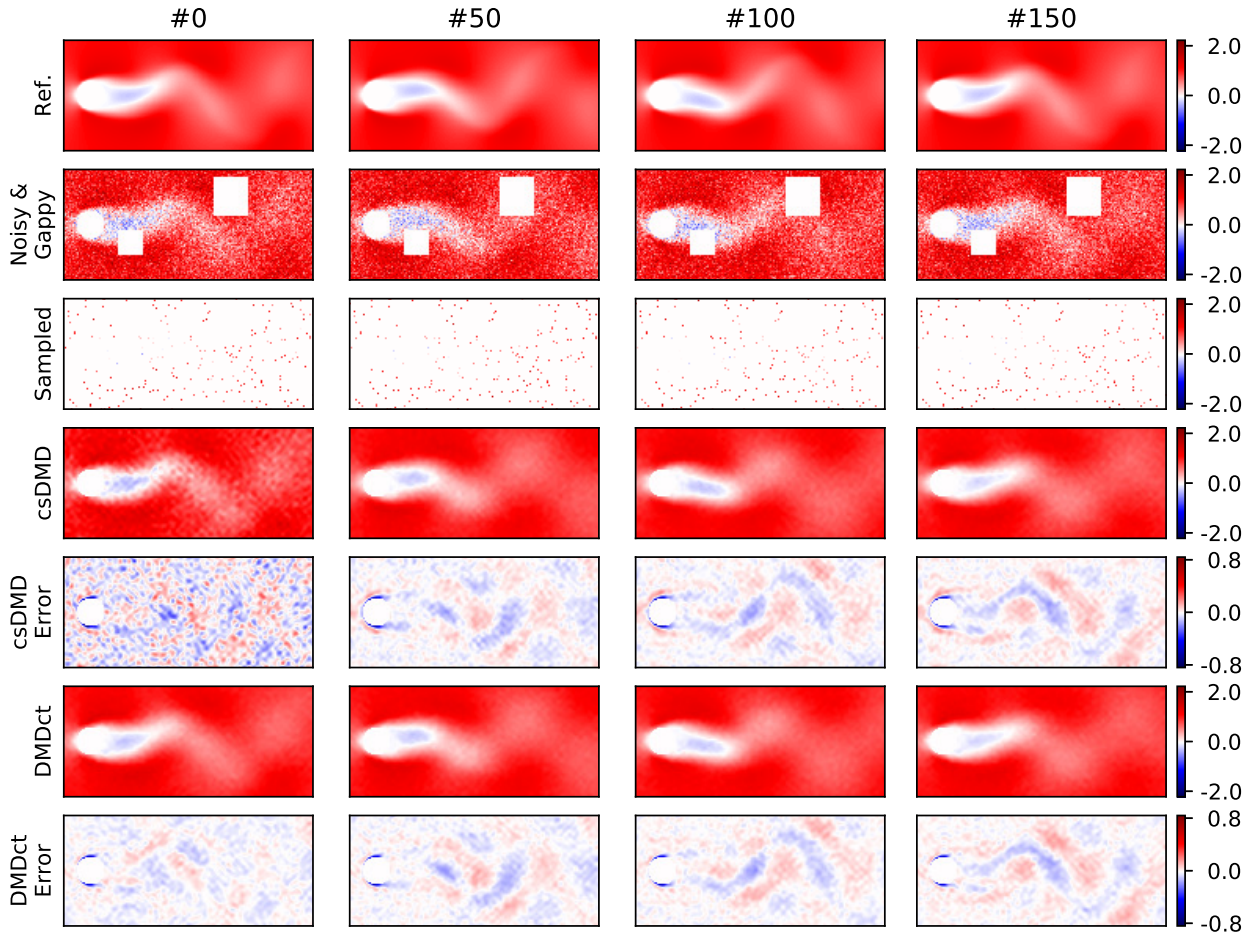


Figure 4.8: Four sample snapshots of reconstructing the noisy u velocity component of the wake behind a cylinder at Reynolds number Re = 100. The circular hollow represents the cylinder. The top row shows the noise-free reference u velocity component. The second row shows the reference with random Gaussian noise having the standard deviation of 0.25 added. The two rectangular gaps are seen as two white rectangular hollows. The third row shows the random samples taken (2% sampling). The fourth and fifth rows show csDMD reconstruction and its error. The two bottom rows show DMDct reconstruction and its error.

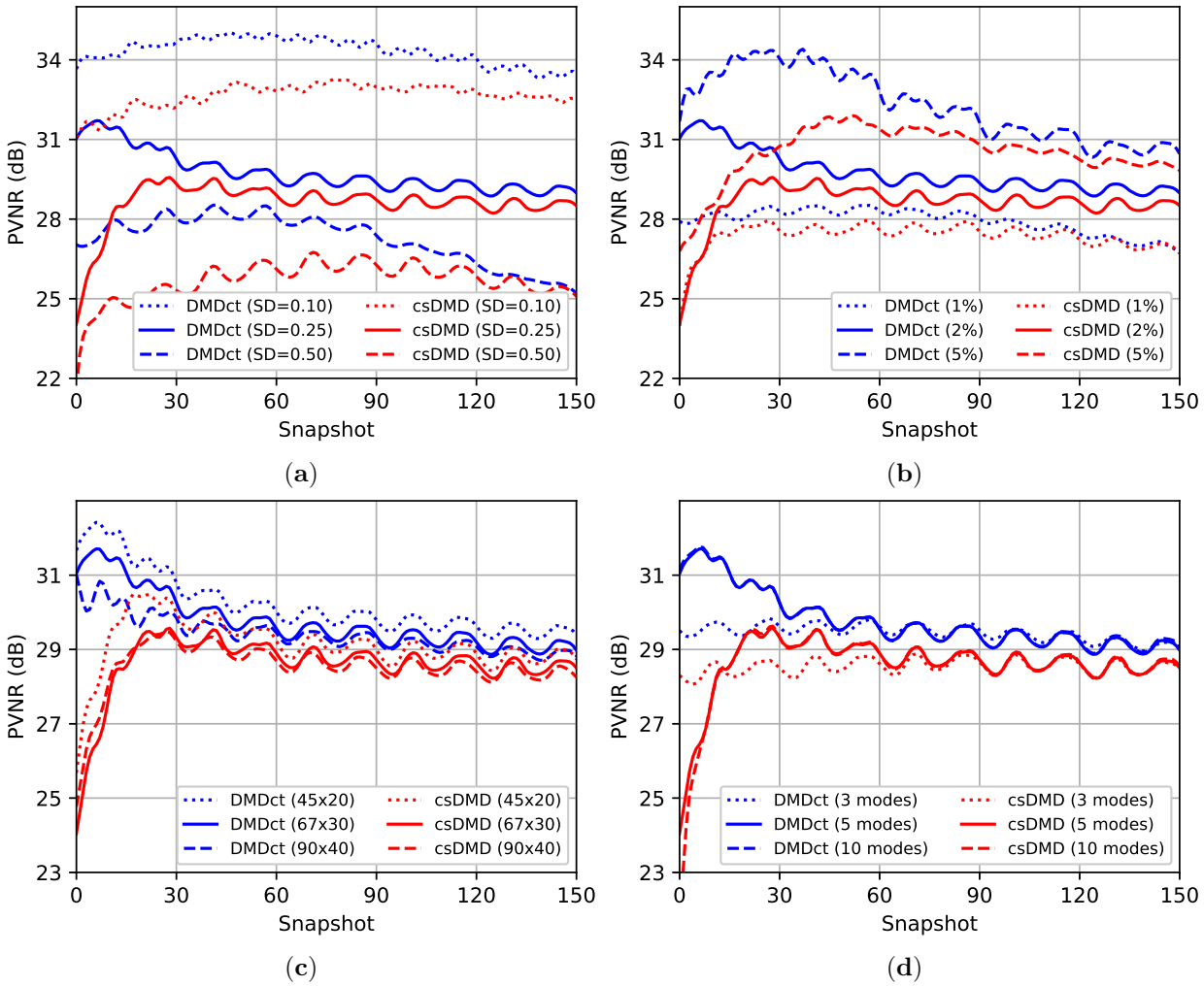


Figure 4.9: Investigating the effect of changing various parameters on per-snapshot PVNR values of DMDct and csDMD reconstructions of u velocity component of the wake behind a cylinder at Reynolds number Re = 100. Blue and red, respectively, represent DMDct and csDMD. The solid lines correspond to the nominal values. (**a**) The effect of variation of noise standard deviation. (**b**) The effect of variation of sampling ratio. (**c**) The effect of changing the number of basis vectors. (**d**) The effect of changing the number of dynamic modes. The lines corresponding to 10 dynamic modes are overlapped by the lines corresponding to 5 modes most of the times.

reduce as the time goes on which is probably due to the high decay rate of the corresponding modes. As depicted in Figure 4.4b, the amplitudes of about half of the modes reduce to 10% or less of their initial values after 20 snapshots which means the corresponding modes die out quickly. In the third and fourth cases, where there is a rectangular gap in the data, DMDct has resulted in less reconstruction error than csDMD in both noisy and noise-free cases. Obviously, the RMSE values are higher compared to those of the no-gap case. Visually comparing, both methods were able to fill the rectangular gap but DMDct seems to have resulted in a smoother and more consistent filling than csDMD. This is also confirmed numerically for the first sampling mask through the RMSE values listed in row "*inside*" of Table 4.1.

As the statistical analysis showed, the RMSE values of DMDct reconstruction are significantly lower than those of csDMD. The post hoc analysis also showed the noise has no significant effect on the error of DMDct. This means DMDct is robust with respect to the noise. The glitches in the noisy reconstruction of csDMD seem to be less than the noise-free case, which is probably due to the smaller number of DMD modes taken (5 vs. 25) and the fewer basis vectors used (20 vs. 41). It is also seen DMDct has resulted in more reconstruction error for the noisy cases than the noise-free cases which is as expected, but the reconstruction errors of csDMD for the noisy cases are less than those of the noise-free cases, which indicates csDMD is more sensitive to the number of mode shapes and basis vectors than DMDct.

As stated earlier, the noisy reconstructions were performed using a fewer number of DMD modes and basis functions than the noise-free ones. Comparing the RMSE values in Figure 4.7 reveals DMDct resulted in smaller changes in the RMSE values compared to csDMD. In addition, the standard deviation of DMDct results is much lower than csDMD's according to the error bars in Figure 4.7. Thus, DMDct is more robust than csDMD.

The third test showed the effect of changing the values of various parameters on the PVNR values of DMDct and csDMD reconstructions. The first parameter to investigate was the standard deviation of the random Gaussian noise. As shown in Figure 4.9a, as the standard deviation increases, the PVNR values drop which is as expected since higher noise standard deviation means a lower signal-to-noise ratio. For the case of high noise (SD = 0.50), csDMD resulted in a very low PVNR value (¡10 dB) in all snapshots (not shown in the figure). This was even lower than the PVNR values of the noisy dataset which means csDMD failed to denoise the data in that case. The

second parameter was the sampling ratio. Figure 4.9b shows higher sampling ratio results in higher PVNR and so, better reconstruction. This is expected as well since higher sampling ratio means more information is provided. In contrast to the first and second cases, the results of changing the number of basis vectors are interesting and unexpected. As shown in Figure 4.9c, the highest PVNR values correspond to the case of the fewest number of basis vectors (45×20). We initially expected to observe an improvement in the results as the number of basis vectors increased which did not happen. The reason is that the number of unknowns is determined by the number of basis vectors, i.e., for the case of 45×20 basis vectors, there is a total of 900 unknowns, whereas, for the case of 90×40 basis vectors, the number of unknowns is 3600. Increasing the number of unknowns affects the performance of the l_1-regularization method and makes it more difficult to find the proper non-zero subset of coefficients. Thus, limiting the number of basis vectors to a reasonable value is the key. The last parameter to study was the number of dynamic modes. The SVHT method suggested five dynamic modes to pick. Picking fewer modes than five resulted in lower PVNR values over the first half of the snapshots, whereas picking more modes did not make any improvement. This shows the number of modes resulted from SVHT is a good choice.

In all cases studied, the PVNR values of csDMD over the first few snapshots were too low whereas DMDct resulted in less deviation of PVNR values than csDMD. In addition, in all cases, DMDct almost always resulted in higher PVNR values than csDMD.

Even though DMDct was developed for the special case of 2D problems defined over a rectangular grid of equally-spaced nodes, the method can be extended to the 3D problems as well. It is also possible to adapt the method to an arbitrary grid of nodes.

In summary, DMDct outperforms csDMD in terms of reconstructing the whole dataset regarding the defined metrics. One disadvantage of DMDct compared to csDMD is the more computation time it needs. This is because there are more data to fit in DMDct than csDMD. Since DMDct aims at reconstructing the whole dataset, the Exact DMD must be employed at the end if the mode shapes are desired. The results of both DMDct and csDMD are sensitive to the value of sparseness coefficient β in Equation (4.13). Here, we ran each algorithm with various β values and then picked the best ones for comparison. For a real case, where the actual solution is unknown, this approach is impractical. The proper choice of sparseness coefficient β remains an open question and will be addressed later.

4.5 Conclusion

In this chapter, a novel approach for dynamic reconstruction of a given dataset based on DMD and a set of basis vectors and by taking a random sub-sample of the fully-sampled dataset was proposed. The proposed approach was compared against csDMD in terms of reconstruction error for three test cases. The results of the tests showed that while the two methods performed similarly on the dataset with a few numbers of dynamic modes, the proposed method outperformed csDMD in terms of both denoising and gap-filling. The third test also showed per-snapshot reconstruction error of DMDct has less variation than csDMD reconstruction.

Chapter 5

Denoising Dynamic Data With a Reduced-order DMD-based Kalman Filter and Smoother

5.1 Introduction

The 4D-Flow MRI (3D+Time) [23, 24] is a non-invasive technique for three-dimensional in vivo measurement of volumetric time-resolved blood flow velocity. However, this technique suffers from various shortcomings such as velocity aliasing, phase offsets, low spatial and temporal resolution, and random acquisition noise. Among these, acquisition noise impacts computation of secondary parameters which depend on gradients of the velocity field. In an attempt to address both acquisition noise and low resolution, a denoising method was proposed in Chapter 3 for 4D-Flow MRI datasets. The proposed method was based on proper orthogonal decomposition (POD) and least absolute shrinkage and selection operator (Lasso) regularization [72] where a set of POD basis vectors derived from the results of a computational fluid dynamics (CFD) simulation was used as the reconstruction basis. It was shown to outperform the competing state-of-the-art denoising methods but it relied on a CFD simulation for generating the set of POD basis vectors which makes it a computationally-intensive method.

The concept of Dynamic Mode Decomposition (DMD) was first introduced by Schmid and

Sesterhenn [12] in 2008. It was initially aimed at studying the spatial dynamic modes of fluid flow [13, 14]. Given a time-varying dataset, DMD can approximate its underlying nonlinear dynamics as a linear autoregressive model. DMD extracts a set of mode shapes and associates each one with an eigenvalue. Each mode shape represents the spatial spread of a dominant feature and the corresponding eigenvalue specifies how that feature evolves over time in terms of the frequency of oscillation and the rate of growth or decay. The DMD was initially meant to be used for extracting the dominant dynamic features from flow fields [13]. Nonetheless, it shortly found applications in other areas as well. At first, the input data of DMD was supposed to be time-resolved. That means the data was in the form of a sequence of several snapshots where the time interval between any two consecutive snapshots remained the same. The number of snapshots is usually much smaller than the length of each snapshot. In 2014, Kutz et al. [15] showed that besides time-resolved datasets, DMD may also be applied to paired datasets where there is a mapping between each snapshot of one dataset with the corresponding snapshot from the other dataset. In such case, the snapshots don't need to be time-resolved. In 2009, Rowley et al. [16] showed the connection between DMD and the Koopman operator. The Koopman operator is an infinite-dimensional linear representation of nonlinear finite-dimensional dynamics and the DMD was shown to approximate its modes [16, 14]. Williams et al. [18] further introduced the extended DMD (EDMD) in 2015 to approximate not only the modes of the Koopman operator but also its leading eigenvalues and eigenfunctions. The extended DMD is a computationally intensive algorithm. As a workaround, the kernel-based DMD (KDMD) was proposed in 2015 [19].

By applying DMD on a noisy dataset, many of the dynamic modes identified will be mostly due to noise [22]. As Dawson et al. [22] showed, the effect of sensor noise is seen as a shift in the computed DMD eigenvalues such that they appear to be more stable than they are in reality. In order to compensate the effect of noise, they proposed three different variants of DMD among which forward-backward DMD (fbDMD) was shown to perform usually better than the other two (Noise-corrected DMD and Total least-squares DMD). The fbDMD also has the advantage of not requiring any prior knowledge about the noise covariance matrix. Another attempt to deal with noise in a partially-sampled 2D dataset was made by combining DMD and a set of discrete cosine transform (DCT) basis vectors [73]. Even though this method was shown to be effective in denoising a partially-sampled 2D dynamic dataset, it is inapplicable to 4D-Flow MRI datasets due to having

three spatial dimensions. In Chapter 4, a method called DMDct was proposed to deal with noise in a partially-sampled 2D dataset. The method was obtained by combining DMD and a set of discrete cosine transform (DCT) basis vectors. Even though DMDct was shown to be effective in denoising a partially-sampled 2D dynamic dataset, it is inapplicable to 4D-Flow MRI datasets due to having three spatial dimensions.

The Kalman filter was introduced in 1960 as a discrete-time linear filter [6]. It can efficiently estimate the state of a dynamic process by minimizing the mean of the squared error in the presence of system model uncertainty and also, process and measurement noise. Since its first introduction, The Kalman filter has been used extensively in various applications. Also, many researchers have attempted to expand its applicability by proposing variants of the Kalman filter, e.g. the extended Kalman filter [8], the unscented Kalman filter [9], and the linear Kalman smoothing [10] to name a few. The linear Kalman filter may be applied to dynamic processes governed or approximated by a linear difference equation, as the name implies. Since the DMD and its variants can derive such approximation for a given dynamic dataset solely based on the data, the idea of integrating DMD and the Kalman filter sounds promising. In 2015, Iungo et al. [74] proposed a new paradigm for prediction of wind turbine wakes by embedding a reduced-order model (ROM) in a Kalman filter where the ROM was derived by using DMD. The reduced order model was achieved by projection onto the subspace spanned by the set of basis vectors generated by proper orthogonal decomposition (POD).

In this chapter, the similar approach of combining the Kalman filter, fbDMD, and a reduced-order model is taken. The fbDMD method provides a linear approximation to the forward model. The Kalman filter is used to reduce the noise in the process and improve the state estimation in the presence of process and measurement noise. The linear approximate forward model derived by fbDMD is used by the Kalman filter for state prediction. Since the size of the state vector is too large (> 1000), the reduced-order model is utilized to lower the size of the state vector and make the Kalman filter implementation feasible. We also propose a new method to find an estimate to the process and measurement noise covariance matrices used by the Kalman filter. Finally, we use a linear Kalman smoother to smooth the data even further. The resulting method (known as KfDMD hereafter) is a fast and parameter-free denoising method which denoises each snapshot by taking advantage of the dominant dynamic features of the whole dataset. Even though the

proposed method is derived having 4D-Flow MRI data in mind, in fact, it may be applied to any dynamic dataset as will be shown in test case 1.

5.2 Method

The flowchart of the proposed method is presented in Figure 5.1. Since the proposed method is parameter-free, the only input is the matrix $\mathbf{Z}$.

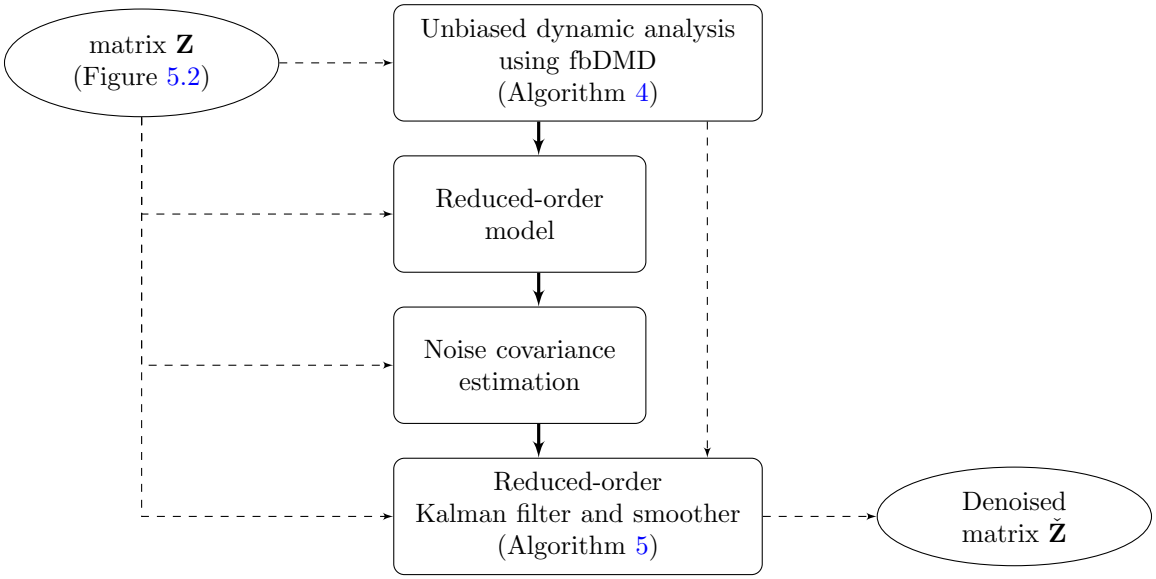


Figure 5.1: The flowchart of the proposed method.

5.2.1 Input data

The input data is in the form of a sequential set of time-resolved data arrays, known as snapshots hereafter, as depicted in Figure 5.2. The number of snapshots is m. Even though the method proposed here can be applied to the data from various kinds of dynamic systems, we will specifically focus on data generated by an in-vivo 4D-Flow MRI scan. In such data, each snapshot is composed of three velocity components of the blood flow passing through the scanned vessel where each velocity component is represented as a 3-dimensional array. The scanned volume is box-shaped within which the scanned points are spread equidistantly as a grid. The vessel takes only a portion of the scanned volume. The scanned points falling outside the vessel are assumed to be representing the surrounding soft tissues. In order to distinguish the points falling inside the vessel from the

points outside, the geometry of the vessel should be specified as well. This is done by defining the geometry mask, which is given as a 3-dimensional boolean array the size of which is the same as that of the snapshots. Each element of the mask is true (false) if the corresponding point in the snapshots falls inside (outside) the vessel. For snapshot k, the points falling inside the vessel are extracted and represented as the data vector $\mathbf{z}_k$ of length N. Since each snapshot is composed of three velocity components, $N = 3N'$ where N' is the number of points falling inside the vessel. The m data vectors $\mathbf{z}_k$ are used to form the $N \times m$ matrix $\mathbf{Z}$ as shown in Figure 5.2.

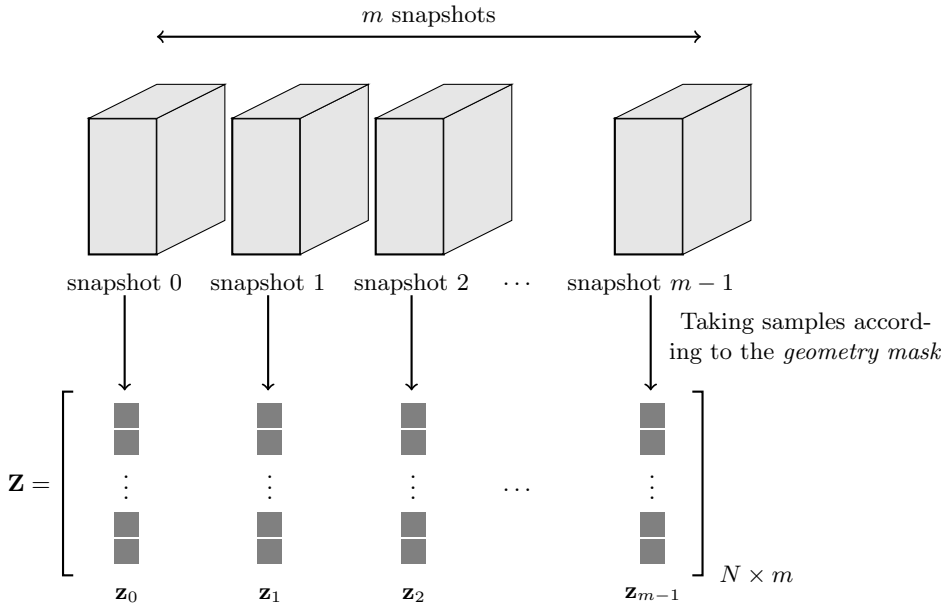


Figure 5.2: Schematic representation of the input data of the proposed method. The points sampled from each snapshot according to the geometry mask are vectorized and represented as a real-valued data vector $\mathbf{z}_k$ of length N.

5.2.2 Unbiased dynamic analysis

If the input data is noisy, applying Exact DMD will result in a biased set of dynamic modes. As stated earlier, the bias is seen as a shift in the eigenvalues so that they appear to be more stable than they are in reality. As a workaround, the fbDMD was used instead of Exact DMD to remove the bias and find the unbiased eigenvalues. The steps of fbDMD are presented in Algorithm 4.

Like most variants of DMD, the desired number of dynamic modes to pick should be specified. Here, we first employed the method of singular value hard thresholding (SVHT) [70] prior to fbDMD

to find the optimal number of dynamic modes. Then, fbDMD was performed to find the unbiased eigenvalues and their corresponding modes. Since fbDMD involves an SVD, the set of POD basis vectors resulting from that was retained and later used as the basis for the reduced-order model hence eliminating the need for running a separate SVD.

Algorithm 4. Forward-backward DMD (fbDMD)

Data:
- $\mathbf{Z} = [\mathbf{z}_0 \dots \mathbf{z}_{m-1}]$: the $N \times m$ matrix of sequential data vectors
- r: the number of modes to pick

Result:
- $\boldsymbol{\Phi}$: the matrix of DMD modes
- $\boldsymbol{\lambda}$: the vector of unbiased DMD eigenvalues
- $\mathbf{U}$: the matrix of POD basis vectors

$\mathbf{X} \leftarrow [\mathbf{z}_0 \dots \mathbf{z}_{m-2}]$
$\mathbf{Y} \leftarrow [\mathbf{z}_1 \dots \mathbf{z}_{m-1}]$
begin the forward path
  Find the SVD of $\mathbf{X}$ such that $\mathbf{X} = \mathbf{U}_x \boldsymbol{\Sigma}_x \mathbf{V}_x^*$
  $\mathbf{U}_{rx} \leftarrow \mathbf{U}_x$ truncated to the first r columns
  $\boldsymbol{\Sigma}_{rx} \leftarrow \boldsymbol{\Sigma}_x$ truncated to the upper-left $r \times r$ matrix
  $\mathbf{V}_{rx}^* \leftarrow \mathbf{V}_x^*$ truncated to the first r rows
  $\tilde{\mathbf{A}}_m \leftarrow \mathbf{U}_{rx}^* \mathbf{Y} \mathbf{V}_{rx} \boldsymbol{\Sigma}_{rx}^{-1}$
end
begin the backward path
  Find the SVD of $\mathbf{Y}$ such that $\mathbf{Y} = \mathbf{U}_y \boldsymbol{\Sigma}_y \mathbf{V}_y^*$
  $\mathbf{U}_{ry} \leftarrow \mathbf{U}_y$ truncated to the first r columns
  $\boldsymbol{\Sigma}_{ry} \leftarrow \boldsymbol{\Sigma}_y$ truncated to the upper-left $r \times r$ matrix
  $\mathbf{V}_{ry}^* \leftarrow \mathbf{V}_y^*$ truncated to the first r rows
  $\tilde{\mathbf{B}}_m \leftarrow \mathbf{U}_{ry}^* \mathbf{X} \mathbf{V}_{ry} \boldsymbol{\Sigma}_{ry}^{-1}$
end
$\tilde{\mathbf{A}} \leftarrow \left(\tilde{\mathbf{A}}_m \tilde{\mathbf{B}}_m^{-1} \right)^{1/2}$ `// an improved estimate`
Find the eigenvalues $\boldsymbol{\lambda}$ and eigenvectors $\mathbf{W}$ of $\tilde{\mathbf{A}}$, i.e. $\tilde{\mathbf{A}}\mathbf{W} = \mathbf{W}\,\mathrm{diag}(\boldsymbol{\lambda})$
Compute the DMD modes $\boldsymbol{\Phi} \triangleq \mathbf{Y} \mathbf{V}_{rx} \boldsymbol{\Sigma}_{rx}^{-1} \mathbf{W}$
return $\boldsymbol{\Phi}$, $\boldsymbol{\lambda}$, $\mathbf{U}_{rx}$ `//` $\mathbf{U}_{rx}$ `is returned as the matrix of POD basis vectors` $\mathbf{U}$

5.2.3 The reduced-order Kalman filter and smoother

Consider an autonomous discrete-time dynamic process governed by the linear stochastic equation

$$\hat{\mathbf{y}}_k = \mathbf{A}\hat{\mathbf{y}}_{k-1} + \mathbf{w}_{k-1}^{(y)} \tag{5.1}$$

and the full state measurements

$$\mathbf{y}_k = \hat{\mathbf{y}}_k + \mathbf{v}_k^{(y)} \tag{5.2}$$

where $\hat{\mathbf{y}} \in \mathbb{R}^n$ is the state vector, $\mathbf{y} \in \mathbb{R}^n$ is the measurement, $\mathbf{w}^{(y)} \in \mathbb{R}^n$ is the white normally-distributed process noise with covariance $\mathbf{Q}_y$, and $\mathbf{v}^{(y)} \in \mathbb{R}^n$ is the white normally-distributed measurement noise with covariance $\mathbf{R}_y$. The Kalman filter provides an estimate to the state vector $\hat{\mathbf{y}}$ given the noisy measurements $\mathbf{x}$ and the noise covariance matrices $\mathbf{Q}_y$ and $\mathbf{R}_y$ through a sequence of *prediction* and *correction* steps. The prediction step is performed as

$$\begin{cases} \mathbf{y}_k^{(p)} = \mathbf{A}\mathbf{y}_{k-1}^{(c)} \\ \mathbf{P}_k^{(p)} = \mathbf{A}\mathbf{P}_{k-1}^{(c)}\mathbf{A}^T + \mathbf{Q}_y \end{cases} \tag{5.3}$$

where $\mathbf{P}$ denotes the estimate error covariance matrix and (p) and (c) superscripts respectively represent the predicted and corrected values. The predicted estimate $\mathbf{y}_k^{(p)}$ is then corrected as

$$\begin{cases} \mathbf{K}_k = \mathbf{P}_k^{(p)} \left(\mathbf{P}_k^{(p)} + \mathbf{R}_y \right)^{-1} \\ \mathbf{y}_k^{(c)} = \mathbf{y}_k^{(p)} + \mathbf{K}_k \left(\mathbf{y}_k - \mathbf{y}_k^{(p)} \right) \\ \mathbf{P}_k^{(c)} = \left(\mathbf{I} - \mathbf{K}_k \right) \mathbf{P}_k^{(p)} \end{cases} \tag{5.4}$$

where $\mathbf{K}$ denotes the Kalman gain.

Due to the matrix inversion operation in equation (5.4), applying the Kalman filter to large-scale problems might not be feasible. As a possible workaround, we project the snapshots onto the subspace spanned by the POD basis vectors and take the projection coefficients as the new state variables. The number of the new state variables will be the same as the number of POD basis vectors. As seen in Algorithm 4, as many POD basis vectors as the DMD modes are taken and so, the number of state variables will equal the rank of DMD.

The approximate forward model (equations (5.1)) is derived through fbDMD. The real-valued fbDMD-based prediction of snapshot k (shown as $\mathbf{z}_k^{(p)}$) is given in terms of the previous snapshot as

$$\mathbf{z}_k^{(p)} = \Re \left(\mathbf{\Phi} \mathbf{\Lambda} \mathbf{\Phi}^\dagger \right) \mathbf{z}_{k-1} \tag{5.5}$$

where $\mathbf{\Phi}$ is the matrix of mode shapes and $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues $\boldsymbol{\lambda}$ (both $\mathbf{\Phi}$ and

$\boldsymbol{\lambda}$ are resulting from fbDMD). By showing the matrix of POD basis vectors as $\mathbf{U}$ (also resulting from fbDMD), the projection $\mathbf{y}_k^{(p)}$ of prediction of snapshot k onto the subspace spanned by the basis vectors (columns of $\mathbf{U}$) is found as

$$\begin{aligned} \mathbf{y}_k^{(p)} &= \mathbf{U}^T \mathbf{z}_k^{(p)} \\ &= \mathbf{U}^T \Re\left(\boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^\dagger\right) \mathbf{z}_{k-1} \end{aligned} \tag{5.6}$$

Suppose the snapshot k and its reconstruction based on the POD basis vectors are approximately equal, i.e.

$$\mathbf{U}\mathbf{y}_k \approx \mathbf{z}_k \tag{5.7}$$

$$\therefore \mathbf{y}_k^{(p)} \approx \mathbf{U}^T \Re\left(\boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^\dagger\right) \mathbf{U}\mathbf{y}_{k-1} = \mathbf{A}\mathbf{y}_{k-1} \tag{5.8}$$

where $\mathbf{A} \triangleq \mathbf{U}^T \Re\left(\boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^\dagger\right) \mathbf{U}$.

Each snapshot $\mathbf{z}_k$ is a full-state noisy measurement, i.e.

$$\mathbf{z}_k = \hat{\mathbf{z}}_k + \mathbf{v}_k^{(z)} \tag{5.9}$$

where $\hat{\mathbf{z}} \in \mathbb{R}^N$ is the state vector and $\mathbf{v}^{(z)} \in \mathbb{R}^N$ is the white normally-distributed measurement noise with covariance $\mathbf{R}_z$. Projecting onto the space of POD basis vectors gives

$$\mathbf{U}^T \mathbf{z}_k = \mathbf{U}^T \hat{\mathbf{z}}_k + \mathbf{U}^T \mathbf{v}_k^{(z)} \tag{5.10}$$

$$\rightarrow \mathbf{y}_k = \hat{\mathbf{y}}_k + \mathbf{U}^T \mathbf{v}_k^{(z)} \tag{5.11}$$

Comparing equations (5.2) and (5.11) implies

$$\mathbf{v}_k^{(y)} = \mathbf{U}^T \mathbf{v}_k^{(z)} \tag{5.12}$$

So, given $\mathbf{v}_k^{(z)}$ is a white normally-distributed variable with covariance $\mathbf{R}_z$, $\mathbf{v}_k^{(y)}$ will also be a white normally-distributed variable with covariance $\mathbf{R}_y$ such that

$$\mathbf{R}_y = \mathbf{U}^T \mathbf{R}_z \mathbf{U} \tag{5.13}$$

Consequently, the correction equations of the Kalman filter (equations (5.4)) become

$$\begin{cases} \mathbf{K}_k = \mathbf{P}_k^{(p)} \left(\mathbf{P}_k^{(p)} + \mathbf{R}_y \right)^{-1} \\ \mathbf{y}_k^{(c)} = \mathbf{y}_k^{(p)} + \mathbf{K}_k \left(\mathbf{U}^T \mathbf{z}_k - \mathbf{y}_k^{(p)} \right) \\ \mathbf{P}_k^{(c)} = \left(\mathbf{I} - \mathbf{K}_k \right) \mathbf{P}_k^{(p)} \end{cases} \tag{5.14}$$

The initial values of the predicted state $\mathbf{y}_0^{(p)}$ and the predicted estimate error covariance $\mathbf{P}_0^{(p)}$ are taken as

$$\begin{cases} \mathbf{y}_0^{(p)} = \mathbf{U}^T \mathbf{z}_0 \\ \mathbf{P}_0^{(p)} = \mathbf{Q} \end{cases} \tag{5.15}$$

After the prediction and correction steps are performed for all snapshots (equations (5.3) and (5.14) respectively), a backward smoother [75] is applied to further smooth the data as given below

$$\begin{cases} \mathbf{K}_k^{(s)} = \mathbf{P}_k^{(c)} \mathbf{A}^T \left(\mathbf{P}_{k+1}^{(p)} \right)^{-1} \\ \mathbf{P}_k^{(s)} = \mathbf{P}_k^{(c)} - \mathbf{K}_k^{(s)} \left(\mathbf{P}_{k+1}^{(p)} - \mathbf{P}_{k+1}^{(s)} \right) \left(\mathbf{K}_k^{(s)} \right)^T \\ \mathbf{y}_k^{(s)} = \mathbf{y}_k^{(c)} - \mathbf{K}_k^{(s)} \left(\mathbf{y}_{k+1}^{(p)} - \mathbf{y}_{k+1}^{(s)} \right) \end{cases} \tag{5.16}$$

where the initial values at $k = m - 1$ are given as

$$\begin{cases} \mathbf{y}_{m-1}^{(s)} = \mathbf{y}_{m-1}^{(c)} \\ \mathbf{P}_{m-1}^{(s)} = \mathbf{P}_{m-1}^{(c)} \end{cases} \tag{5.17}$$

Algorithm 5 summarizes the proposed implementation of the Kalman filter and smoother.

5.2.4 The noise covariance matrices

A key part of the Kalman filter is properly estimating the covariance matrices of the process and measurement noise. Here, a scheme is proposed to find an estimate to those matrices. Let's combine equations (5.1) and (5.2)

$$\begin{aligned} \mathbf{y}_k &= \hat{\mathbf{y}}_k + \mathbf{v}_k^{(y)} \\ &= \mathbf{A} \hat{\mathbf{y}}_{k-1} + \mathbf{w}_{k-1}^{(y)} + \mathbf{v}_k^{(y)} \\ &= \mathbf{A} \left(\mathbf{y}_{k-1} - \mathbf{v}_{k-1}^{(y)} \right) + \mathbf{w}_{k-1}^{(y)} + \mathbf{v}_k^{(y)} \end{aligned} \tag{5.18}$$

Algorithm 5. Kalman filtering and smoothing in the reduced-order space of POD basis vectors

Data:

- $\mathbf{U}$: the $N \times r$ matrix of orthonormal POD basis vectors
- $\mathbf{\Phi}$: the $N \times m$ matrix of DMD modes
- $\boldsymbol{\lambda}$: the vector of r DMD eigenvalues
- $\mathbf{z}_k$: the m vectorized noisy snapshots of length N $(0 \leq k < m)$
- $\mathbf{Q}$: the $r \times r$ matrix of process and measurement noise covariance

Result:

- $\check{\mathbf{x}}_k$: the m denoised snapshots $(0 \leq k < m)$

$\mathbf{A} \leftarrow \mathbf{U}^T \Re\left(\mathbf{\Phi}\,\mathrm{diag}(\boldsymbol{\lambda})\mathbf{\Phi}^\dagger\right)\mathbf{U}$
begin the Kalman filter
  Initialize $\mathbf{y}_0^{(p)}$ and $\mathbf{P}_0^{(p)}$ (equation (5.15))
  Do correction for the first snapshot (equation (5.14))
  for $k \leftarrow 1$ **to** $m-1$ **do**
    Do prediction (equation (5.3))
    Do correction (equation (5.14))
  end
end
begin the backward smoother
  Initialize $\mathbf{y}_{m-1}^{(s)}$ and $\mathbf{P}_{m-1}^{(s)}$ (equation (5.17))
  Reconstruct the last snapshot as $\check{\mathbf{z}}_{m-1} \leftarrow \mathbf{U}\mathbf{y}_{m-1}^{(s)}$
  for $k \leftarrow m-2$ **downto** 0 **do**
    Do smoothing (equation (5.16))
    Reconstruct the k-th snapshot as $\check{\mathbf{z}}_k \leftarrow \mathbf{U}\mathbf{y}_k^{(s)}$
  end
end
return *all* $\check{\mathbf{z}}_k$ $(0 \leq k < m)$

So, the error $\mathbf{e}_k$ between the measured snapshot k and its fbDMD-based prediction is defined as

$$\therefore \mathbf{e}_k \triangleq \mathbf{y}_k - \mathbf{A}\mathbf{y}_{k-1} = -\mathbf{A}\mathbf{v}_{k-1}^{(y)} + \mathbf{w}_{k-1}^{(y)} + \mathbf{v}_k^{(y)} \tag{5.19}$$

Since the three random variables on the right-hand side of equation (5.19) are white, normally-distributed, and independent from each other, the error $\mathbf{e}_k$ will also be a white normally-distributed random variable with the covariance $\mathbf{S}$ given as

$$\mathbf{S} \triangleq \mathrm{E}\left[\mathbf{e}_k \mathbf{e}_k^T\right] = \mathbf{A}\mathbf{R}_y\mathbf{A}^T + \mathbf{Q}_y + \mathbf{R}_y \tag{5.20}$$

The covariance matrix $\mathbf{S}$ may be calculated by finding the error between the projected snapshots and their corresponding projected fbDMD-based predictions and then, employing an Oracle Approximating Shrinkage (OAS) estimator [76] to estimate the covariance of the projected error.

If the measurement noise covariance matrix $\mathbf{R}_z$ is known, the projected covariance matrix $\mathbf{R}_y$ can be calculated from equation (5.13) and so, equation (5.20) can be solved for $\mathbf{Q}_y$ easily. If the matrix $\mathbf{R}_z$ (or $\mathbf{R}_y$) is not known, we may assume $\mathbf{R}_y = \mathbf{Q}_y$ and then, solve equation (5.20) for both as

$$\text{vec}\{\mathbf{R}_y\} = \text{vec}\{\mathbf{Q}_y\} = (\mathbf{A} \otimes \mathbf{A} + 2\mathbf{I})^{-1} \text{vec}\{\mathbf{S}\} \tag{5.21}$$

where $\otimes$ is the Kronecker product of two matrices and the vec operator turns a matrix into a vector by stacking the columns [77].

5.3 Results

The proposed method was evaluated in three different test cases against the total variation method (TV) [47] and divergence-free wavelets with Sure-Shrink, median absolute deviation, and partial cycle spinning (DFW-sms) [48] which are the leading state-of-the-art denoising methods as shown in Chapter 3. For the first test case, a synthetic dataset generated by using the unforced Duffing equation was used. Although our algorithm was developed having 4D-Flow MRI datasets in mind, in fact, it does not care about the source and structure of the input data, as most other DMD-based methods do. As seen in Algorithm 5, the noisy snapshots are taken as a vector input which is nothing but a list of real numbers without knowing how those numbers are related to each other. For this reason, the unforced Duffing equation was taken as the first test case to show the applicability of our method to a general dynamic dataset. For the second test case, a synthetic 4D-Flow MRI dataset generated based on the geometry of an actual aneurysm was used. The synthetic dataset was generated according to the method proposed in Chapter 3. For the third test case, two datasets acquired by in-vivo 4D-Flow MRI scans were used. For the first and second test cases, since the noise-free datasets were available, the root mean square error (RMSE) defined below was used as the comparison metric

$$\text{RMSE} \triangleq \frac{1}{\sqrt{n}} \|\mathbf{Z}_{rec} - \mathbf{Z}_{ref}\|_F \tag{5.22}$$

where $\mathbf{Z}_{rec}$ is the reconstructed dataset, $\mathbf{Z}_{ref}$ is the reference noise-free dataset, and n is the total number of elements of the dataset.

5.3.1 Test case 1: the unforced Duffing equation

The unforced Duffing equation was used to generate the test dataset. The governing differential equation is

$$\ddot{x} = -\delta\dot{x} - x(\gamma + \alpha x^2) \tag{5.23}$$

where $\delta = 0.5$, $\gamma = -1$, and $\alpha = 1$. The region $x, \dot{x} \in [-2, 2]$ was discretized as a 41×41 mesh over which the equation was solved as explained in section 4.3.2 and then, the $\dot{x}$ values were taken as the noise-free dataset consisting of 51 snapshots. Random Gaussian noise with the standard deviation of 0.25 was added to the noise-free dataset to make the noisy dataset as seen in Figure 5.3.

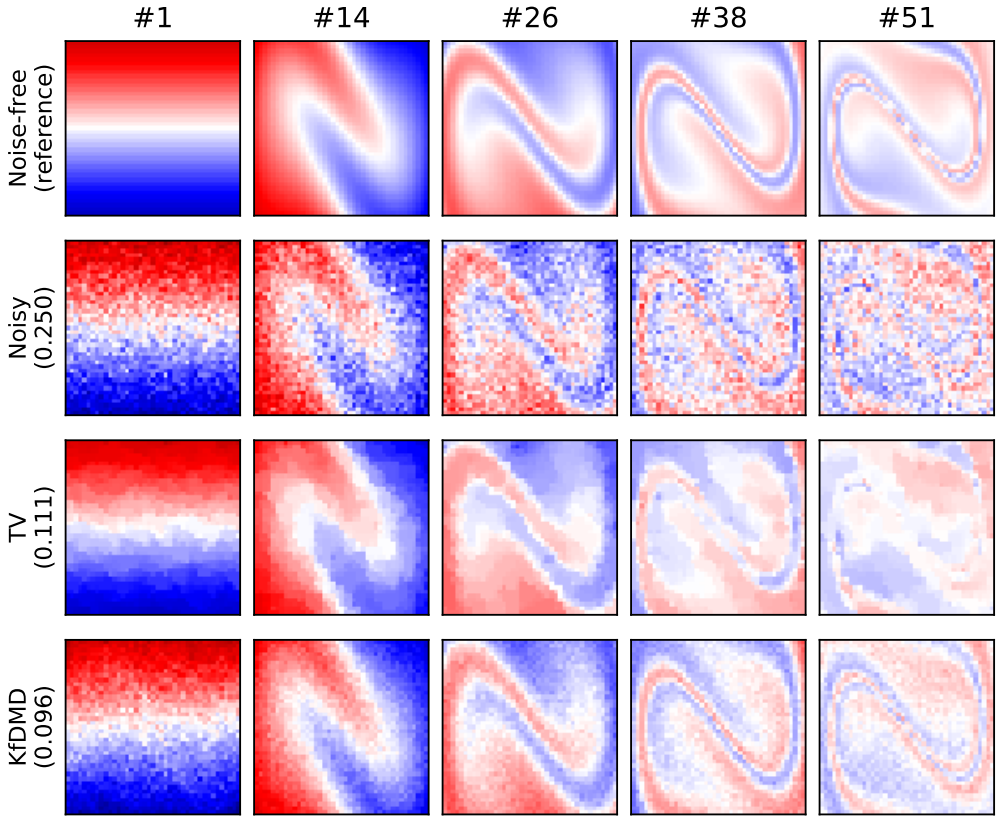


Figure 5.3: The reconstruction results of the unforced Duffing dataset. The numbers at the top of the columns show the corresponding snapshot numbers. The two top rows show the noise-free (reference) and noisy datasets, respectively. The reconstructions of TV and KfDMD methods are shown as indicated. The numbers inside the parenthesis are the overall RMSE values, where the lower is better.

The number of DMD modes for the noisy dataset was taken as 6, according to SVHT. Figure 5.3 shows a few sample snapshots from the reconstructed datasets. The two top rows show the noise-free and noisy datasets, respectively. The reconstructions of the TV and KfDMD are also shown as indicated. Several runs of TV were conducted and the one resulting in the least RMSE was picked for comparison. Figure 5.4 shows the effect of the λ parameter of TV on the RMSE of its reconstruction. As seen in the figure, the error of TV reconstruction is minimum when $\lambda \approx 0.2$. Since KfDMD is a parameter-free method, only one run was conducted to generate the results. The simulation times were 5-6 seconds for TV ($\lambda = 0.2$) and about 0.2 seconds for KfDMD. For the noisy and the two reconstructed datasets, the overall RMSE values are provided as well. Figure 5.5 shows the RMSE values calculated for the noisy and the two reconstructed datasets. The solid and dotted lines represent the per-snapshot and overall RMSE values, respectively.

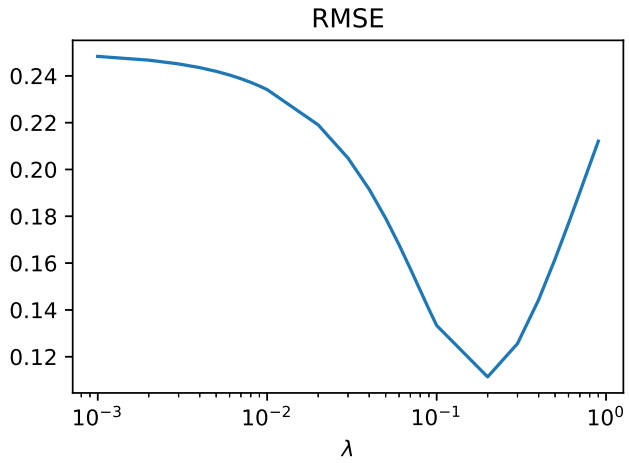


Figure 5.4: TV reconstruction error vs the λ parameter. There is an optimal value of λ parameter ($\lambda \approx 0.2$) for which the reconstruction error is minimum.

5.3.2 Test case 2: a synthetic 4D-Flow MRI dataset

A synthetic 4D-Flow MRI dataset was generated by following the approach taken in Chapter 3 and as briefly explained here. A CFD simulation was conducted first to simulate an actual intra-cranial aneurysm geometry where the boundary conditions were taken from a realistic pulsatile. The high-resolution time snapshots of the CFD simulation were down-sampled into a typical 4D-Flow grid to form the noise-free reference dataset. The synthetic dataset was acquired by adding noise to the noise-free reference dataset in *k-space* [60]. The size of the 4D-Flow grid was $25 \times 31 \times 86$.

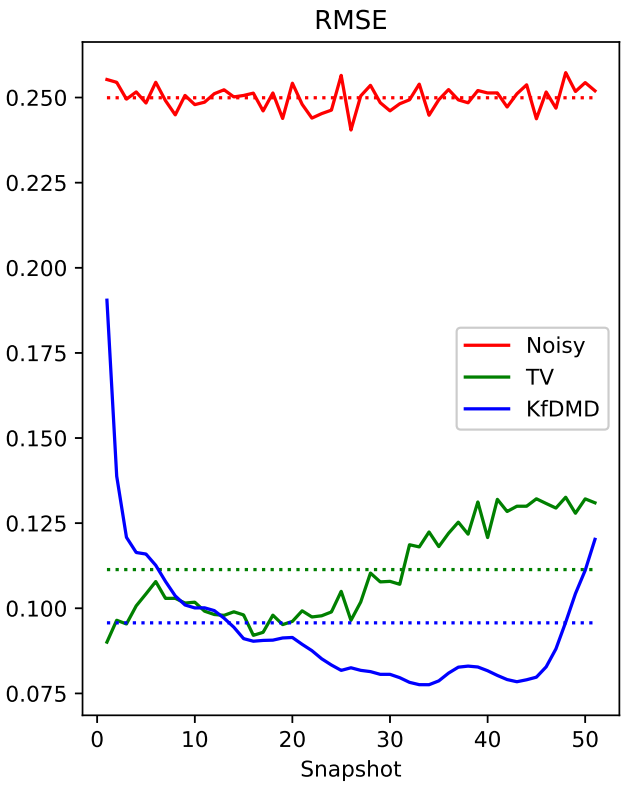


Figure 5.5: Error of reconstruction for the unforced Duffing dataset. Per-snapshot (solid lines) and overall (dotted) RMSE values of the noisy and the reconstructed datasets are shown.

There were a total of 50 snapshots in the dataset. A geometric mask was defined to distinguish the aneurysm from the surrounding tissues. According to the mask, the aneurysm contained 6516 spatial locations which account for 9.8% of the total number of points in the grid.

The number of dynamic modes was taken as 3 as resulted from SVHT. Figure 5.6 shows sample slices from the noise-free reference, noisy, and reconstructed datasets. The reconstruction errors are shown as well. Like the first dataset, several runs of TV and DFW-sms were performed to find and pick the best for comparison while KfDMD results were acquired by a single run. The simulation times were about 127 seconds for TV, about 19 seconds for DFW-sms, and not more than 1.3 seconds for KfDMD. This confirms KfDMD is one order of magnitude faster than TV and two orders of magnitude faster than DFW-sms. Figure 5.7 shows the plot of per-snapshot RMSE values as the bold lines and the overall RMSE values as the dotted lines.

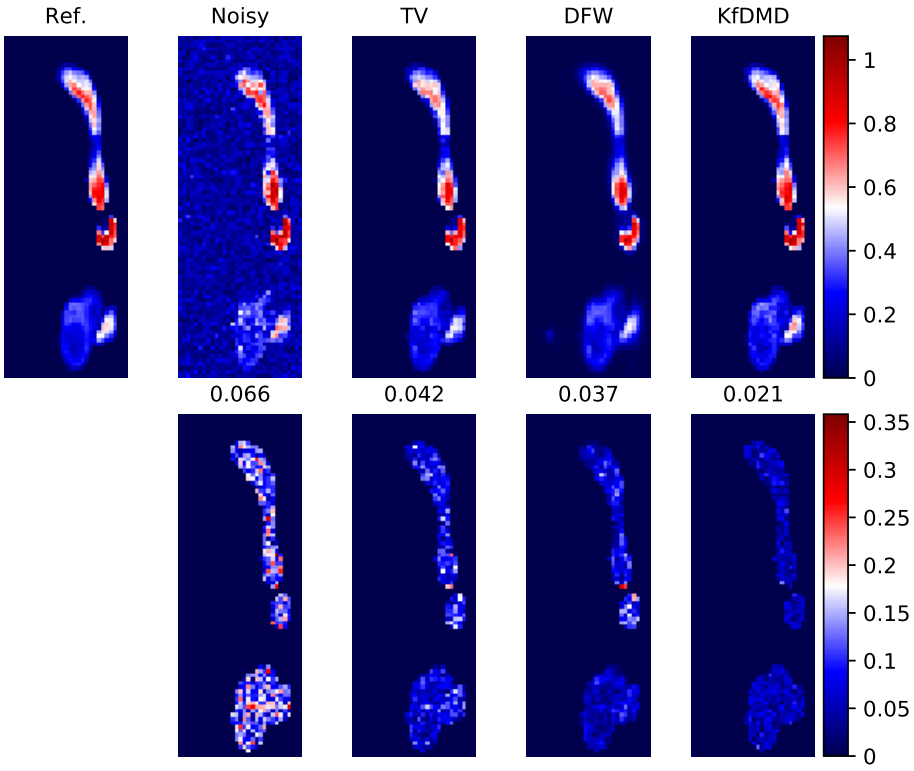


Figure 5.6: The results for the synthetic 4D-Flow MRI dataset reconstruction. The top row shows sample slices where at each point, the norm of the velocity is shown. The bottom row shows the norm of the difference between the datasets and the noise-free reference. The bottom row figures are titled with the corresponding overall RMSE values. All areas outside the geometry mask are zeroed before RMSE values are calculated.

5.3.3 Test case 3: in-vivo 4D-Flow MRI datasets

Two datasets generated by in-vivo 4D-Flow MRI scans were used in this test case. In this case, unlike the other two test cases, the noise-free datasets are unknown and so, there is no metric to make a comparison based on and evaluate the results. Each dataset consisted of 20 consecutive snapshots. The spatial resolutions were $30 \times 40 \times 103$ and $22 \times 30 \times 48$. For either dataset, the geometry of the aneurysm was provided as a binary mask. The geometries of the aneurysms contained 4900 and 1566 spatial locations which respectively accounted for 4.0% and 4.9% of the spatial locations scanned. The simulation times for the first in-vivo dataset were about 100 seconds for TV, about 13 seconds for DFW-sms, and about 1 second for KfDMD. Similarly, the simulations times for the second in-vivo dataset were about 26 seconds for TV, 3-4 seconds for DFW-sms, and less than 0.3 seconds for KfDMD. Two sample slices of the noisy datasets are shown in Figures 5.8 and 5.9.

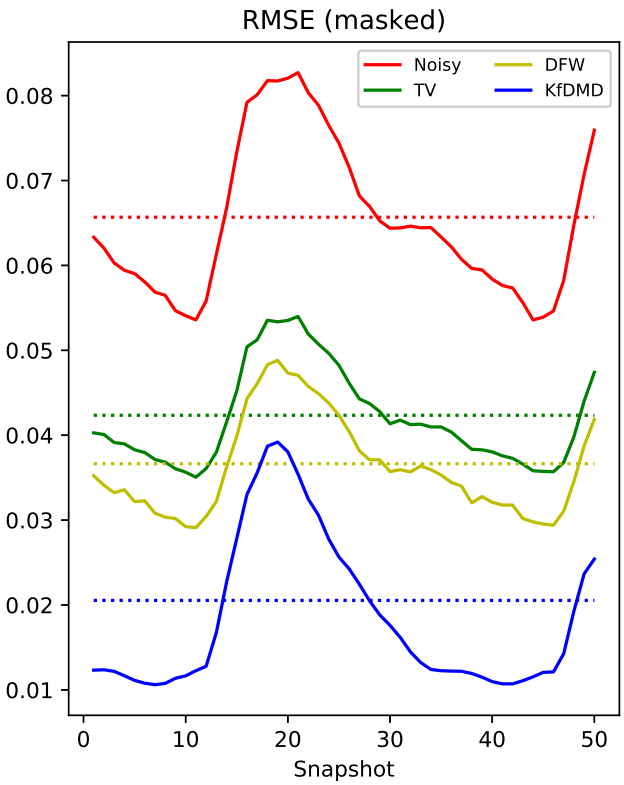


Figure 5.7: Plot of reconstruction error for the synthetic 4D-Flow MRI dataset. Per-snapshot (solid lines) and overall (dotted) RMSE values of the noisy and the reconstructed datasets are shown.

5.4 Discussion

Three tests were conducted to compare the proposed denoising method versus TV and DFW methods which are well-known state-of-the-art denoising methods. In contrast to TV and DFW, our method considers the dynamics of the underlying dynamic system while denoising. This is done by using the dominant dynamic modes and eigenvalues detected by fbDMD to implement the forward model. Also, our method is parameter-free and runs very fast. It didn't take more than a few seconds for any of the test cases to run our method.

The first test was based on a 2D dataset generated from the unforced Duffing equation. As Figure 5.5 shows, the per-snapshot RMSE of KfDMD result is initially high but it drops quickly and goes lower than the RMSE of TV as the simulation marches forward in time. It starts to go up again over the last few snapshots. This is because DMD approximates the nonlinear dynamics as a linear auto-regressive model and so, it fails to estimate the next snapshot properly in the presence of highly nonlinear dynamics. The dotted lines in Figure 5.5 represent the overall RMSE values.

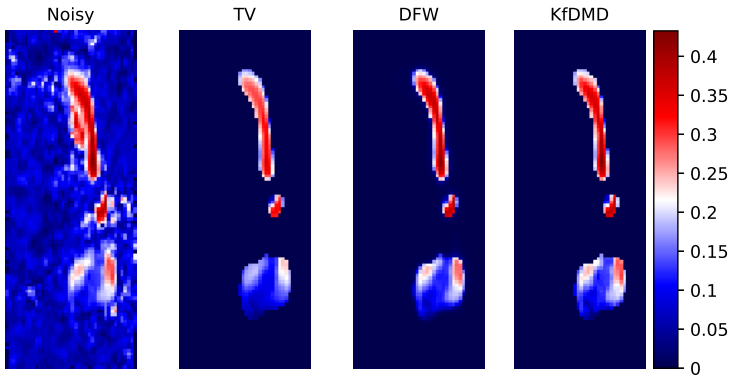


Figure 5.8: The results for the first in-vivo 4D-Flow MRI dataset. The binary mask was used to exclude all points outside the vessel. The spatial resolution is $30 \times 40 \times 103$.

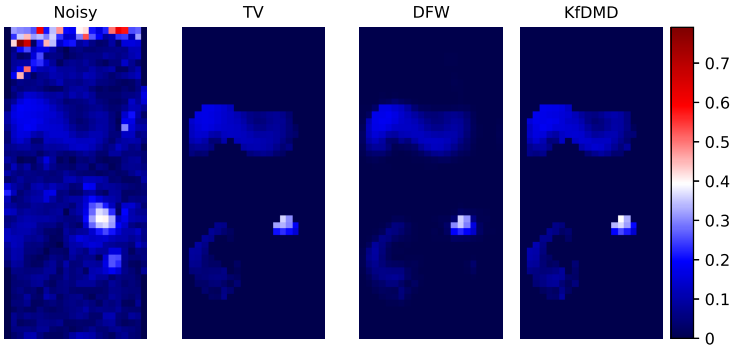


Figure 5.9: The results for the second in-vivo 4D-Flow MRI dataset. The binary mask was used to exclude all points outside the vessel. The spatial resolution is $22 \times 30 \times 48$.

While both methods have resulted in more than 50% reduction in overall RMSE, the overall RMSE of KfDMD has seen more drop than TV's (0.096 versus 0.111). The last snapshot (#51) in Figure 5.3 shows the effectiveness of the proposed algorithm in recovering fine features. Even though most fine features seem to be washed out by the noise, the KfDMD was still able to reconstruct some of them by taking advantage of knowing the dominant dynamic features and their corresponding growth/decay rate and frequency of oscillation.

The second test was based on a synthetic 4D-Flow MRI dataset consisting of 50 snapshots. The plot of per-snapshot RMSE values in Figure 5.7 reveals the effectiveness of KfDMD in denoising the noisy dataset where it has resulted in lower reconstruction error in all snapshots. The overall RMSE of KfDMD is less than DFW's and TV's and less than one-third of the noisy dataset's (0.021, 0.037, 0.042, and 0.066, respectively). This test clearly shows the effectiveness of KfDMD in denoising without having to set any parameters.

The third test was based on two actual 4D-Flow MRI in-vivo datasets. Due to the lack of the desired solution, it is not possible to find the RMSE values and make a comparison based on them. By observing the reconstruction results, it seems KfDMD has been able to reveal more fine features than TV and DFW. Since KfDMD was shown to perform better than TV for the first test case and better than both TV and DFW for the second test case and also since it is a parameter-free method, we tend to believe it performed better for the third test case as well but there is no proof.

In summary, KfDMD outperforms TV and DFW in terms of denoising a dynamic dataset regarding the defined metric. It runs very quickly and is parameter-free. Our method relies on fbDMD to approximate the dynamics of the given dataset. The fbDMD algorithm (Algorithm 4) involves finding the square root of a matrix. The square root of a matrix is not unique and so, picking the best square root remains an open question. If the dynamics are highly nonlinear, fbDMD will fail to identify them properly and so, the forward model derived from fbDMD will be inaccurate. This might affect the performance of our method. One possible workaround is to use a variant of DMD which takes into consideration the nonlinear dynamics as well (such as extended DMD or kernel-based DMD). It is also possible to use a more advanced variant of the Kalman filter such as extended Kalman filter or unscented Kalman filter.

5.5 Conclusion

In this chapter, a new method for denoising 4D-Flow MRI data was proposed. The proposed method is fast, parameter-free, and not computationally-intensive. It was compared against TV and DFW-sms methods in three test cases. In the first two cases, where the noise-free dataset is available, our method was shown to perform better than TV and DFW-sms (when applicable) methods in terms of the defined error metric.

Chapter 6

Dynamic Reconstruction of a 2D Flow Field Based on Sparse Measurements Taken in The Fourier Space: A Preliminary Study

6.1 Introduction

Magnetic Resonance Imaging (MRI) is a widely-used technique in medical imaging for diagnostic purposes. It provides high soft-tissue contrast and is non-invasive. However, a major drawback of the technique is its lengthy data acquisition time. During data acquisition, the patient is supposed to stay steady for the whole duration of acquisition. This can get frustrating since depending on the size of the area to be scanned, data acquisition may take several minutes. Things get even worse when a time-varying phenomenon like blood flow is scanned.

One possible workaround is to reduce the acquisition time by reducing the number of phase-encoding steps [78]. MRI data acquisition is performed in the frequency domain. Inspired by the idea of compressed sensing, the data acquisition time may be reduced by sampling a portion of the frequency domain rather than the whole and then, trying to reconstruct the scanned region through post-processing. This results in shorter scan time at the expense of further processing time and the

need for higher computation power. In 2017, Lustig et al. [79] introduced the idea of incorporating sparse sampling in MR imaging. They exploited the implicit sparseness of the representation of MR images in terms of wavelet basis. In 2016, Kustner et al. [78] proposed a scheme called ESPReSSo aimed at partially sampling the k-space and reconstructing the full image through compressed sensing [78]. These methods were implemented for single images and no dynamics were involved.

In this chapter, a scheme is proposed for reconstructing a 2D incompressible fluid flow by taking a series of sparse noisy measurements performed in the frequency domain while accounting for the dynamics of the flow as well. Our scheme takes advantage of the KfDMD method introduced in Chapter 5 for denoising the noisy measurements and reconstructing the noisy measurements based on the set of POD basis vectors. Then, the POD vectors are reconstructed in Cartesian space in full by taking the POD vectors in the frequency domain as the sparse measurements through l_1-regularization.

Since a 2D incompressible fluid flow is studied, a set of divergence-free basis vectors are developed and used as the reconstruction basis. The derivation and implementation of our custom set of divergence-free basis vectors are explained in the Appendix.

6.2 Method

Consider a 2D incompressible flow velocity field defined over a rectangular mesh of equally-spaced nodes as shown in Figure 6.1. The figure shows the wake behind a cylinder. The dimensions of the domain shown in Figure 6.1 are $l_x \times l_y$ which is discretized as an $n_x \times n_y$ grid. Each velocity component is represented as an $n_y \times n_x$ matrix, so $2n_xn_y$ real values should be specified at each moment of time in order to describe the state of the discretized flow field in full. Nonetheless, the $2n_xn_y$ real values are not fully independent since the flow field is governed by Navier-Stokes equation and conservation of mass (continuity). This implies the information rank of the flow field is much smaller than $2n_xn_y$ and so, it might have a sparse representation in terms of a proper set of basis vectors.

Suppose the measurement of the velocity components is performed in the frequency domain. Also, assume the measurements are performed sparsely such that rather than the whole frequency domain, only a fraction is scanned. The measurements are further assumed to be altered by random

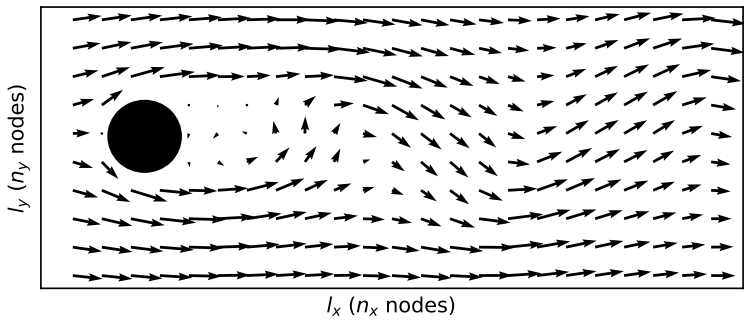


Figure 6.1: A sample 2D incompressible flow velocity field. The size of the field is $l_x \times l_y$ which is discretized as an $n_x \times n_y$ grid of equally-spaced nodes.

Gaussian white noise in the frequency domain. Figure 6.2 shows the amplitude of the frequency components of the sample flow field shown in Figure 6.1. As seen, the amplitudes are larger at lower frequencies (near the origin which is at the center).

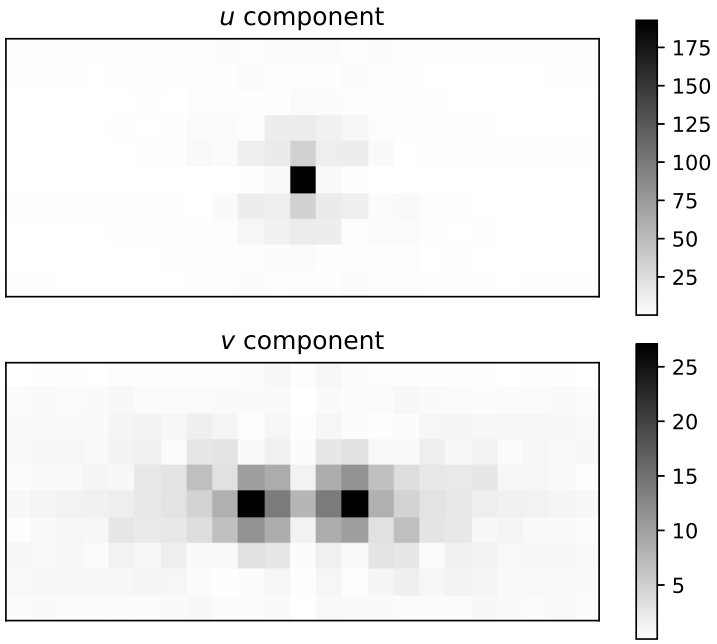


Figure 6.2: The amplitude plots of the sample 2D incompressible flow velocity field in the frequency domain. The amplitudes are larger at lower frequencies (near the center).

The aim is to reconstruct the noise-free velocity components based on a given set of basis vectors in the Cartesian domain given the sparse noisy measurements from the frequency domain and while preserving the dynamics of the flow field. In the method proposed here, the divergence-free DCT-based basis vectors will be used as the reconstruction basis (see the Appendix for derivation and implementation steps). The inverse discrete transform in 2D is given as equation (A9). Using the

divergence-free basis vectors as the reconstruction basis guarantees the reconstructed flow field will satisfy the continuity equation since the reconstruction is performed as a linear combination of the basis vectors and the continuity equation (A1) is linear as well.

In the sample domain shown in Figure 6.1 there is no flow passing through the cylinder. This means when the domain is discretized, the nodes that fall inside the cylinder must be excluded from the reconstruction. This is usually accomplished through defining the discretized geometry as a boolean mask. The geometry mask can be presented as an $n_y \times n_x$ matrix of 0 or 1 values such that a value of 0 (1) at a node means the corresponding location in the domain is excluded from (included in) the flow field.

Figure 6.3 shows the flowchart of the proposed method. The approach is to take the sparse noisy measurements in the frequency domain and then, apply the KfDMD method introduced in Chapter 5 to denoise the noisy measurements. The KfDMD method results in the set of POD basis vectors and the corresponding snapshot-wise projection coefficients. Next, the POD vectors are reconstructed in the Cartesian domain using the divergence-free DCT-based basis vectors. Finally, the reconstructed POD vectors are linearly combined according to the snapshot-wise projection coefficients to reconstruct the divergence-free flow field in full. Note that in contrast to the POD vectors obtained from KfDMD, their fully-reconstructed versions are neither normal nor orthogonal.

By showing the geometry mask as the matrix $\mathbf{G}$ and using the inverse discrete transform equations (A9), the components of the fully-reconstructed POD vector i may be represented as

$$\begin{cases} \tilde{\mathbf{U}}_i = \mathbf{G} \odot \left[l_x \mathbf{C}_s \mathbf{X}_i \mathbf{C}_r^T + l_x \mathbf{S}_s \mathbf{Y}_i \mathbf{D}_r^T \right] \\ \tilde{\mathbf{V}}_i = \mathbf{G} \odot \left[l_y \mathbf{D}_s \mathbf{X}_i \mathbf{S}_r^T + l_y \mathbf{C}_s \mathbf{Y}_i \mathbf{C}_r^T \right] \end{cases} \tag{6.1}$$

where $\odot$ is the element-wise product of two matrices. The measurements in the frequency domain are sparse. Let's assume the sparse measurements are taken according to a sampling mask which is also presented as a boolean matrix. By showing the sparse measurements of the horizontal and vertical components of the POD vector i as $\mathbf{\Omega}_{\tilde{U}_i}$ and $\mathbf{\Omega}_{\tilde{V}_i}$ respectively, the errors between the

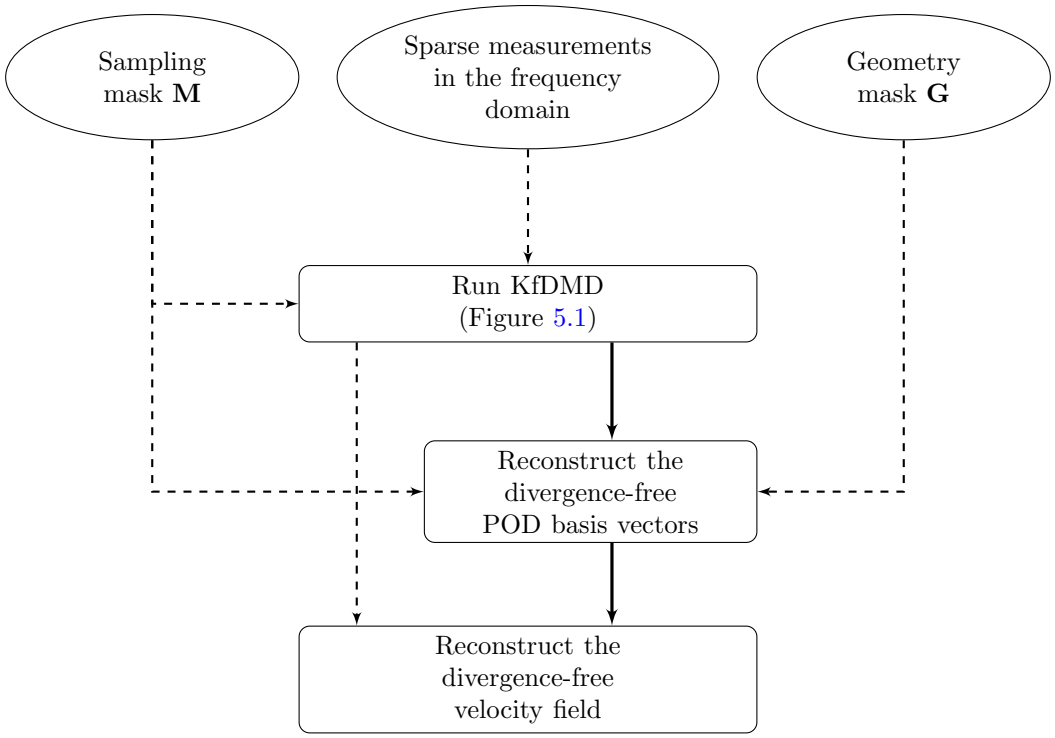


Figure 6.3: The flowchart of the proposed method.

reconstructed components and the measurements are calculated as

$$\begin{cases} \mathbf{E}_{\tilde{U}_i} \triangleq \mathbf{M} \odot \mathcal{F}_{2D}\left\{\tilde{\mathbf{U}}_i\right\} - \mathbf{\Omega}_{\tilde{U}_i} = \mathbf{M} \odot \left(\mathbf{F}_s^T \tilde{\mathbf{U}}_i \mathbf{F}_r\right) - \mathbf{\Omega}_{\tilde{U}_i} \\ \mathbf{E}_{\tilde{V}_i} \triangleq \mathbf{M} \odot \mathcal{F}_{2D}\left\{\tilde{\mathbf{V}}_i\right\} - \mathbf{\Omega}_{\tilde{V}_i} = \mathbf{M} \odot \left(\mathbf{F}_s^T \tilde{\mathbf{V}}_i \mathbf{F}_r\right) - \mathbf{\Omega}_{\tilde{V}_i} \end{cases} \tag{6.2}$$

where $\mathbf{M}$ is the sampling matrix, $\mathcal{F}_{2D}$ is the 2D Fourier transform, and $\mathbf{F}_s$ and $\mathbf{F}_r$ are the matrices of the forward 2D Fourier transform. The two matrices $\mathbf{\Omega}_{\tilde{U}_i}$ and $\mathbf{\Omega}_{\tilde{V}_i}$ are sparse and have zeros wherever no sample is taken. The error of reconstruction is defined as

$$E_i \triangleq \frac{1}{2}\left\|\Re\mathbf{E}_{\tilde{U}_i}\right\|_F^2 + \frac{1}{2}\left\|\Im\mathbf{E}_{\tilde{U}_i}\right\|_F^2 + \frac{1}{2}\left\|\Re\mathbf{E}_{\tilde{V}_i}\right\|_F^2 + \frac{1}{2}\left\|\Im\mathbf{E}_{\tilde{V}_i}\right\|_F^2 \tag{6.3}$$

The unknown real-valued matrices $\mathbf{X}_i$ and $\mathbf{Y}_i$ are found by solving the following minimization problem

$$\mathbf{X}_i, \mathbf{Y}_i = \underset{\mathbf{X}_i, \mathbf{Y}_i}{\operatorname{argmin}} \left(E_i + \beta\left(\|\mathbf{X}_i\|_1 + \|\mathbf{Y}_i\|_1\right)\right) \tag{6.4}$$

where β is the l_1-regularization parameter. The following equations are helpful if a gradient-based

minimization algorithm is used

$$\frac{\partial E_i}{\partial \mathbf{X}_i} = l_x \mathbf{C}_s^T \left(\mathbf{G} \odot \Re \left(\mathbf{F}_s \bar{\mathbf{E}}_{\tilde{U}_i} \mathbf{F}_r^T \right) \right) \mathbf{C}_r + l_y \mathbf{D}_s^T \left(\mathbf{G} \odot \Re \left(\mathbf{F}_s \bar{\mathbf{E}}_{\tilde{V}_i} \mathbf{F}_r^T \right) \right) \mathbf{S}_r \tag{6.5}$$

$$\frac{\partial E_i}{\partial \mathbf{Y}_i} = l_x \mathbf{S}_s^T \left(\mathbf{G} \odot \Re \left(\mathbf{F}_s \bar{\mathbf{E}}_{\tilde{U}_i} \mathbf{F}_r^T \right) \right) \mathbf{D}_r + l_y \mathbf{C}_s^T \left(\mathbf{G} \odot \Re \left(\mathbf{F}_s \bar{\mathbf{E}}_{\tilde{V}_i} \mathbf{F}_r^T \right) \right) \mathbf{C}_r \tag{6.6}$$

Note that equation (6.4) should be solved for each POD vector individually. This means the POD vectors may be reconstructed simultaneously.

6.3 Results

A test was performed to demonstrate the usefulness of the proposed method in denoising and reconstruction of an incompressible 2D flow field. The test dataset was generated by downsampling the dataset used in section 4.3.6 from 449×199 to 89×39. The dataset consists of 151 snapshots. The geometry mask shown in Figure 6.4 was used to exclude the cylinder from the flow region.

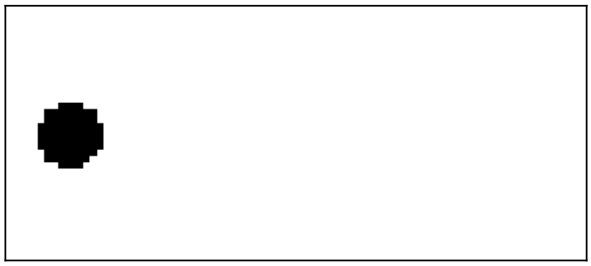


Figure 6.4: The 89×39 geometry mask of the downsampled 2D test dataset. The dark nodes fall inside the cylinder and are excluded from the flow region.

The test dataset was transformed to the frequency domain by applying 2D Fourier transform. Then, white Gaussian noise with the standard deviation of twice the maximum flow speed was added to the transformed data. Three different types of sampling mask were tested, namely *Normal*, *Full-disk*, and *Half-disk*. The Normal sampling masks were generated by randomly distributing the sampling points all over the mask with the normal distribution. For the Full-disk masks, half of the sampling points were located around the center forming a disk and the rest were uniformly distributed over the mask. The Half-disk masks have the same disk at the center as the Full-masks, except the rest were spread over the top half plane rather than the whole plane. Since the input data is real-valued, the bottom half plane is the same as the conjugate of the top half plane rotated

180°. So, in the absence of measurement noise, taking measurements from the top half plane is enough. For each mask type, three masks corresponding to 5%, 10%, and 20% sampling were generated to study the effect of sampling percent as well. Figure 6.5 depicts the sampling masks used.

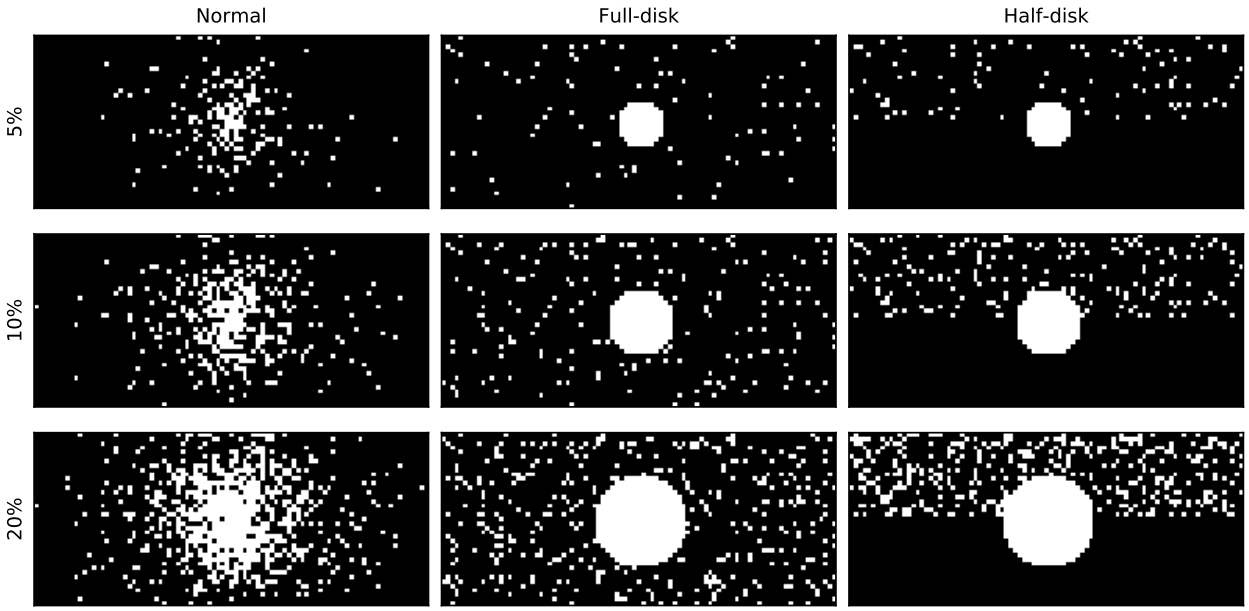


Figure 6.5: The sampling masks are shown. White dots show the sampling spots. The data in the frequency domain is sampled according to the chosen mask. The numbers on the left represent the sampling ratio.

For each sampling mask, the noisy transformed data was sampled and taken as the input data. The reduced-order DMD-based Kalman filter and smoother introduced in chapter 5 was employed to denoise the sampled data in the frequency domain. The noise standard deviation was assumed to be unknown. To study the effect of the l_1-regularization parameter β (equation (6.4)) on the quality of data reconstruction, eleven different β values were tested.

The root mean square error (RMSE) defined below was used as the comparison metric

$$\text{RMSE} \triangleq \frac{1}{\sqrt{2mn}} \sqrt{\sum_{k=0}^{m-1} \left[\left\| \mathbf{G} \odot \left(\hat{\mathbf{U}}_k - \mathbf{U}_k \right) \right\|_F^2 + \left\| \mathbf{G} \odot \left(\hat{\mathbf{V}}_k - \mathbf{V}_k \right) \right\|_F^2 \right]} \tag{6.7}$$

where m is the number of snapshots, n is the number of points the geometry mask $\mathbf{G}$ contains, $\hat{\mathbf{U}}_k$ and $\hat{\mathbf{V}}_k$ are the reconstructed velocity components of snapshot k, and $\mathbf{U}_k$ and $\mathbf{V}_k$ are the reference velocity components of snapshot k. Figure 6.6 shows the plots of RMSE values for the

three types of sampling mask and the three sampling ratios versus eleven different values of the l_1-regularization parameter β. As seen, the RMSE values decrease as the β value decreases. In all cases, the reconstruction error remains almost the same at $\beta = 0.002$ and below.

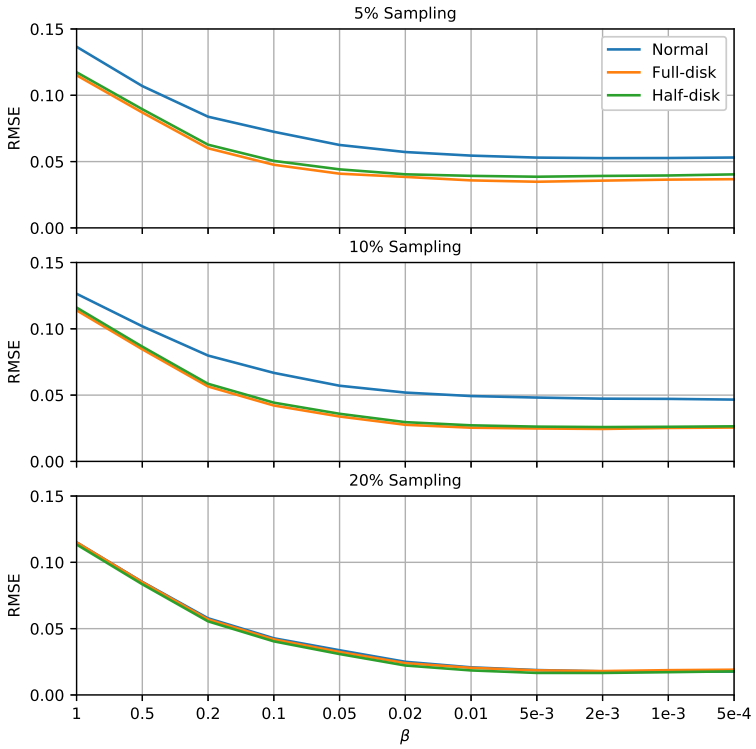


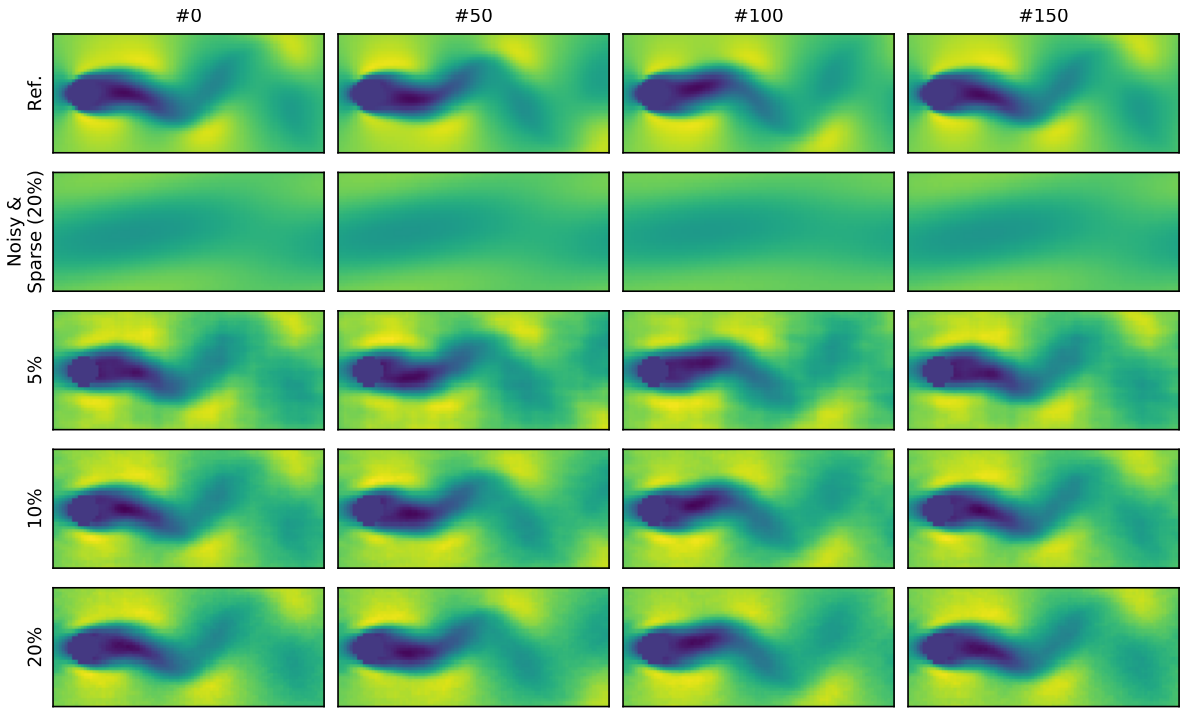
Figure 6.6: The RMSE values versus mask type and sampling ratio for various values of the l_1-regularization parameter β.

Figure 6.7 shows the reconstruction results for $\beta = 0.002$ using a Full-disk mask.

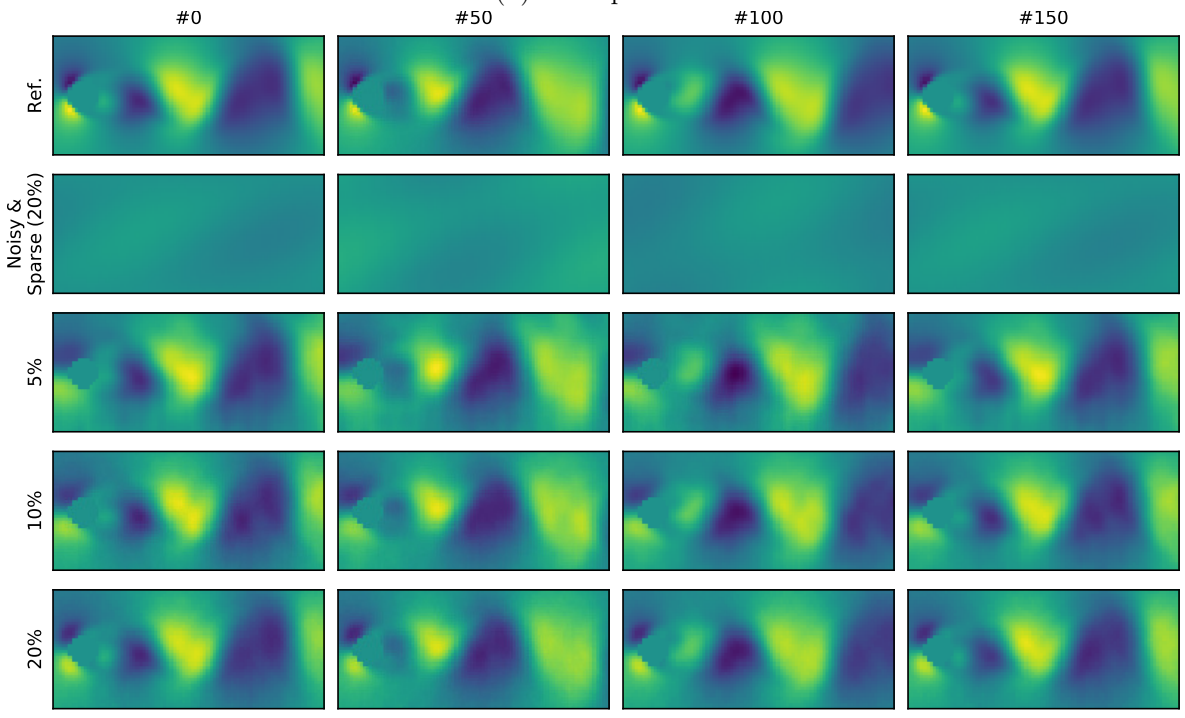
6.4 Discussion

As a proof of concept, the method proposed here was tested with a simulated dataset showing the wake behind a cylinder in 2D. In order to study the effect of sampling mask and sampling ratio, three different sampling masks and sampling ratios were tested. Also, the effect of the sparsity level was studied by using eleven different values of the l_1-regularization parameter.

As Figure 6.6 shows, the error decreases as the sparsity level decreases in all cases (lower β means denser solution). As previously seen in Figure 3.2, there is an optimal level of sparsity which results in the perfect balance between least-square fit and sparseness of the solution. Similarly, we would expect to see an increase in the reconstruction error if we tried lower β values.



(**a**) u component



(**b**) v component

Figure 6.7: The results of reconstruction for $\beta = 0.002$ using the Full-disk mask. The snapshot numbers are shown at the top. The top row shows the reference. The second row shows the sample snapshots as reconstructed by applying the inverse Fourier transform on the noisy and sparsely-sampled data (20% sampling). The remaining rows show the results of reconstruction for different sampling ratios as shown at the left of each row.

With 5% and 10% sampling, the reconstruction errors of the Normal mask seem to be significantly higher than the errors of the Full-disk and Half-disk masks. The reconstruction errors decrease as the sampling ratio increases, which is as expected. In all cases, the Full-disk and Half-disk masks show the same performance.

With 20% sampling, no notable difference between the masks is observed. As seen in Figure 6.5, the Normal mask with 20% sampling has a very good overlap with the disk at the center. This is probably why no significant difference between the masks is observed with 20% sampling. Comparing the errors of 5% sampling versus 10% sampling shows using a Full-disk or Half-disk mask with 5% sampling results in slightly lower reconstruction error than the Normal mask with 10% sampling. This justifies the importance of fully sampling the low-frequency components.

Figure 6.7 shows that with all sampling ratios, the reconstructed snapshots follow the dynamics of the reference dataset. Even with 5% sampling, the proposed scheme is able to reconstruct the snapshots with good agreement with the reference. Some artifacts are seen in the reconstruction which almost disappear as the sampling ratio increases.

In summary, the proposed scheme shows effectiveness in reconstructing a 2D incompressible flow field by taking noisy sparse measurements in the frequency domain. The denoising ability of KfDMD is promising. Also, the proposed divergence-free DCT-based basis vectors seem impressive in reconstructing divergence-free flow features.

6.5 Conclusion

In this chapter, the preliminary results of a new scheme were presented. The proposed scheme aims at reconstructing a 2D incompressible fluid flow by taking a series of sparse noisy measurements performed in the frequency domain. The reconstruction method accounts for the dynamics of the flow as well. The method was evaluated by using a 2D flow past a cylinder as the test dataset. Three different sampling masks and three different sampling ratios were studied. The effect of sparsity coefficient on reconstruction error was studied as well. As part of the method, a set of divergence-free basis vectors were developed and used as the reconstruction basis.

Chapter 7

Conclusion

7.1 Summary

During these studies, various applications of DMD, POD, compressed sensing, Kalman filter and smoother, and lasso regularization were investigated. The main focus was to analyze physical systems solely based on some collected data. Four novel methods were devised for data denoising and reconstruction. Several tests were conducted to evaluate the effectiveness of the proposed methods.

In Chapter 3, a novel algorithm was developed that could be used to couple patient-specific CFD simulation with 4D-Flow measurements by use of POD and lasso regularization. The efficacy of the method was tested using a numerical phantom. This method is capable of reconstructing data to any arbitrary high resolution (depending on the resolution of the CFD mesh) and reconstruct many flow details. Therefore, this method can potentially improve the ability to accurately derive relevant secondary hemodynamic parameters such as wall shear stresses and pressure gradients at a much higher resolution than 4D-Flow. See [2] for the journal article published based on this chapter.

In Chapter 4, a novel approach for dynamic reconstruction of a given dataset based on DMD and a set of basis vectors and by taking a random sub-sample of the fully-sampled dataset was proposed. The proposed approach was compared against csDMD in terms of reconstruction error for three test cases. The results of the tests show that, while the two methods performed similarly on the dataset with a few numbers of dynamic modes, the proposed method outperformed csDMD

in terms of both denoising and gap-filling. The third test also showed per-snapshot reconstruction error of DMDct has less variation than csDMD reconstruction. See [73] for the journal article published based on this chapter.

In Chapter 5, a new method for denoising 4D-Flow MRI data was proposed. The proposed method is fast, parameter-free, and not computationally-intensive. As part of the method, a new approach to estimating the noise covariance matrices using full state observations and fbDMD-based reconstruction was presented. The method was compared against TV and DFW methods in three test cases. In the first two cases, where the noise-free dataset was available, our method was shown to perform better than TV and DFW methods (when applicable) in terms of the defined error metric.

In Chapter 6, the preliminary results of a new scheme were presented. The proposed scheme aims at reconstructing a 2D incompressible fluid flow by taking a series of sparse noisy measurements performed in the frequency domain. The reconstruction method accounts for the dynamics of the flow as well. The method was evaluated by using a 2D flow past a cylinder as the test dataset. Three different sampling masks along with three sampling ratios were studied. The effect of sparsity coefficient on reconstruction error was studied as well. As part of the method, a set of divergence-free basis vectors were developed and used as the reconstruction basis.

7.2 Future work

The method proposed in Chapter 3 relies on a CFD simulation for generating the POD vectors used as the reconstruction basis. Running the CFD simulation is a time-consuming and computationally-intensive task. Devising an alternative method for finding the POD vectors which is not as complicated as a CFD simulation may improve the applicability of the proposed method. The DMDct method introduced in Chapter 4 is limited to 2D problems but it is possible to extend its application to 3D problems. Also, the divergence-free DCT-based basis vectors presented in the Appendix may replace the DCT basis vectors to make a DMD-based method for dynamic denoising and gappy data reconstruction of incompressible flow fields. The method proposed in Chapter 6 is the first step towards a DMD-based reconstruction method for sparse MRI. The proposed method relies on a reduced-order model for denoising. It will be desirable if the denoising is performed in the Cartesian

space while taking the sparse noisy measurements from the Fourier space without incorporating the reduced-order model. In order to make the proposed scheme practical, the measurements should rather be more similar to the MRI data acquisition method than a pure Fourier transform. Finally, investigating the possibility of using the divergence-free wavelets as the reconstruction basis is recommended.

Bibliography

[1] Cheng Yue Wang, Qi Gao, Hong Ping Wang, Run Jie Wei, Tian Li, and Jin Jun Wang. Divergence-free smoothing for volumetric PIV data. *Experiments in Fluids*, 57(1):1–23, 2016. ISSN 07234864. doi: 10.1007/s00348-015-2097-1.

[2] Mojtaba F. Fathi, Ali Bakhshinejad, Ahmadreza Baghaie, David Saloner, Raphael H. Sacho, Vitaliy L. Rayz, and Roshan M. D'Souza. Denoising and Spatial Resolution Enhancement of 4D Flow MRI Using Proper Orthogonal Decomposition and Lasso Regularization. *Computerized Medical Imaging and Graphics*, 2018. ISSN 0895-6111. doi: 10.1016/J.COMPMEDIMAG.2018.07.003.

[3] Emmanuel J Candès and Michael B Wakin. An introduction to compressive sampling. *IEEE signal processing magazine*, 25(2):21–30, 2008.

[4] Annapurna Pradhan, Nibedita Pati, Suvendu Rup, Avipsa S Panda, and Lalit Kumar Kanoje. A comparative analysis of compression techniques–the sparse coding and BWT. *Procedia Computer Science*, 92:106–111, 2016.

[5] Anindya Chatterjee. An introduction to the proper orthogonal decomposition. *Current science*, pages 808–817, 2000.

[6] R. E. Kalman, Kalman, and R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35, 1960. ISSN 00219223. doi: 10.1115/1.3662552.

[7] Gaetan Kerschen, Jean-claude Golinval, Alexander F Vakakis, and Lawrence A Bergman. The

method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: an overview. *Nonlinear dynamics*, 41(1-3):147–169, 2005.

[8] Arthur Gelb. *Applied optimal estimation*. MIT press, 1974. ISBN 0262570483.

[9] E Wan. Sigma-Point Filters: An Overview with Applications to Integrated Navigation and Vision Assisted Control. In *2006 IEEE Nonlinear Statistical Signal Processing Workshop*, pages 201–202, 2006. ISBN VO -. doi: 10.1109/NSSPW.2006.4378854.

[10] Herbert E Rauch, F Tung, Charlotte T Striebel, and Others. Maximum likelihood estimates of linear dynamic systems. *AIAA journal*, 3(8):1445–1450, 1965.

[11] Galen Andrew and Jianfeng Gao. Scalable training of L1 -regularized log-linear models. In *Proceedings of the 24th international conference on Machine learning - ICML '07*, pages 33–40, jan 2007.

[12] Peter J. Schmid and Joern Sesterhenn. Dynamic mode decomposition of numerical and experimental data. *Bulletin of the American Physical Society*, 53:2008, 2008.

[13] Peter J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656(July 2010):5–28, 2010.

[14] Kevin K Chen, Jonathan H Tu, and Clarence W Rowley. Variants of Dynamic Mode Decomposition: Boundary Condition, Koopman, and Fourier Analyses. *Journal of Nonlinear Science*, 22(6):887–915, dec 2012.

[15] J Nathan Kutz, Steven L Brunton, Dirk M Luchtenburg, Clarence W. Rowley, and Jonathan H. Tu. On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1(2):391–421, 2014.

[16] Clarence W. Rowley, Igor Mezić, Shervin Schlatter, Philipp Bagheri, and Dan S Henningson. Spectral analysis of nonlinear flows. *Journal of Fluid Mechanics*, 641(Rowley 2005):115–127, 2009.

[17] Mihailo R. Jovanovic, Peter J. Schmid, and Joseph W. Nichols. Sparsity-promoting dynamic mode decomposition. *Physics of Fluids*, 26(2):1–22, 2014.

[18] Matthew O. Williams, Ioannis G. Kevrekidis, and Clarence W. Rowley. A Data-Driven Approximation of the Koopman Operator: Extending Dynamic Mode Decomposition. *Journal of Nonlinear Science*, 25(6):1307–1346, 2015.

[19] M. O. Williams, C. W. Rowley, and I. G. Kevrekidis. A kernel-based method for data-driven Koopman spectral analysis. *arXiv preprint arXiv:1411.2260*, 2015.

[20] Florimond Guéniat, Lionel Mathelin, and Luc R. Pastur. A dynamic mode decomposition approach for large and arbitrarily sampled systems. *Physics of Fluids*, 27(2), 2015.

[21] J. Nathan Kutz, Jonathan H. Tu, Joshua L. Proctor, and Steven L. Brunton. Compressed sensing and dynamic mode decomposition. *Journal of Computational Dynamics*, 2(2):165–191, 2016.

[22] Scott T M Dawson, Maziar S. Hemati, Matthew O. Williams, and Clarence W. Rowley. Characterizing and correcting for the effect of sensor noise in the dynamic mode decomposition. *Experiments in Fluids*, 57(3):1–19, 2016. ISSN 07234864. doi: 10.1007/s00348-016-2127-7.

[23] 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, and Norbert J. Pelc. Time-resolved three-dimensional phase-contrast MRI. *Journal of Magnetic Resonance Imaging*, 17(4):499–506, 2003. ISSN 10531807. doi: 10.1002/jmri.10272.

[24] Lars Wigström, Lars Sjöqvist, and Bengt Wranne. Temporally resolved 3D phase-contrast imaging. *Magnetic Resonance in Medicine*, 36(5):800–803, 1996. ISSN 07403194. doi: 10.1002/mrm.1910360521.

[25] AF Stalder, MF Russe, A Frydrychowicz, J Bock, J Hennig, and M Markl. Quantitative 2d and 3d phase contrast mri: optimized analysis of blood flow and vessel wall parameters. *Magnetic resonance in medicine*, 60(5):1218–1231, 2008.

[26] Darren P Lum, Kevin M Johnson, Russell K Paul, Aquilla S Turk, Daniel W Consigny, Julie R Grinde, Charles A Mistretta, and Thomas M Grist. Transstenotic pressure gradients: Measurement in swine—retrospectively ecg-gated 3d phase-contrast mr angiography versus endovascular pressure-sensing guidewires 1. *Radiology*, 245(3):751–760, 2007.

[27] Merih Cibis, Kelly Jarvis, Michael Markl, Michael Rose, Cynthia Rigsby, Alex J Barker, and Jolanda J Wentzel. The effect of resolution on viscous dissipation measured with 4d flow mri in patients with fontan circulation: Evaluation using computational fluid dynamics. *Journal of biomechanics*, 48(12):2984–2989, 2015.

[28] Petter Dyverfeldt, Malenka Bissell, Alex J. Barker, Ann F. Bolger, Carl-Johan Carlhäll, Tino Ebbers, Christopher J. Francios, Alex Frydrychowicz, Julia Geiger, Daniel Giese, Michael D. Hope, Philip J. Kilner, Sebastian Kozerke, Saul Myerson, Stefan Neubauer, Oliver Wieben, and Michael Markl. 4D flow cardiovascular magnetic resonance consensus statement. *Journal of cardiovascular magnetic resonance : official journal of the Society for Cardiovascular Magnetic Resonance*, 17(1):72, dec 2015.

[29] AH Andersen and JE Kirsch. Analysis of noise in phase contrast mr imaging. *Medical Physics*, 23(6):857–869, 1996.

[30] Susanne Schnell, Julio Garcia, Can Wu, and Michael Markl. Dual-velocity encoding phase-contrast mri: extending the dynamic range and lowering the velocity to noise ratio. In *Proc. Intl. Soc. Mag. Reson. Med*, volume 23, page 4546, 2015.

[31] Fraser M Callaghan, Rebecca Kozor, Andrew G Sherrah, Michael Vallely, David Celermajer, Gemma A Figtree, and Stuart M Grieve. Use of multi-velocity encoding 4d flow mri to improve quantification of flow patterns in the aorta. *Journal of Magnetic Resonance Imaging*, 43(2): 352–363, 2016.

[32] Mat A Bernstein, Xiaohong Joe Zhou, Jason A Polzin, Kevin F King, Alexander Ganin, Norbert J Pelc, and Gary H Glover. Concomitant gradient terms in phase contrast mr: analysis and correction. *Magnetic resonance in medicine*, 39(2):300–308, 1998.

[33] M Markl, R Bammer, MT Alley, CJ Elkins, MT Draney, A Barnett, ME Moseley, GH Glover, and NJ Pelc. Generalized reconstruction of phase contrast mri: analysis and correction of the effect of gradient field distortions. *Magnetic resonance in medicine*, 50(4):791–801, 2003.

[34] Peter G Walker, Gregory B Cranney, Markus B Scheidegger, Gena Waseleski, Gerald M Pohost, and Ajit P Yoganathan. Semiautomated method for noise reduction and background phase

error correction in mr phase velocity data. *Journal of Magnetic Resonance Imaging*, 3(3): 521–530, 1993.

[35] J Bock, BW Kreher, J Hennig, and M 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.

[36] Matt A Bernstein, Mladen Grgic, Thomas J Brosnan, and Norbert J Pelc. Reconstructions of phase contrast, phased array multicoil data. *Magnetic resonance in medicine*, 32(3):330–334, 1994.

[37] Qing-San Xiang. Temporal phase unwrapping for cine velocity imaging. *Journal of Magnetic Resonance Imaging*, 5(5):529–534, 1995.

[38] Loic Boussel, Vitaliy Rayz, Alastair Martin, Gabriel Acevedo-Bolton, Michael T Lawton, Randall Higashida, Wade S Smith, William L Young, and David Saloner. Phase-contrast magnetic resonance imaging measurements in intracranial aneurysms in vivo of flow patterns, velocity fields, and wall shear stress: comparison with computational fluid dynamics. *Magnetic Resonance in Medicine*, 61(2):409–417, 2009.

[39] Mark Hofman, Frans C Visser, Albert C Van Rossum, Ger QM Vink, Michiel Sprenger, and Nico Westerhof. In vivo validation of magnetic resonance blood volume flow measurements with limited spatial resolution in small vessels. *Magnetic resonance in medicine*, 33(6):778–784, 1995.

[40] Young-Gyun Jeong, Yong-Tae Jung, Moo-Seong Kim, Choong-Ki Eun, and Sang-Hwan Jang. Size and location of ruptured intracranial aneurysms. *Journal of Korean Neurosurgical Society*, 45(1):11–15, 2009.

[41] Sebastian Schmitter, Susanne Schnell, Kâmil Uğurbil, Michael Markl, Van de Moortele, et al. Towards high-resolution 4d flow mri in the human aorta using kt-grappa and b1+ shimming at 7t. *Journal of Magnetic Resonance Imaging*, 44(2):486–499, 2016.

[42] Susanne Schnell, Michael Markl, Pegah Entezari, Riti J Mahadewia, Edouard Semaan, Zoran Stankovic, Jeremy Collins, James Carr, and Bernd Jung. k-t grappa accelerated four-

dimensional flow mri in the aorta: effect on scan time, image quality, and quantification of flow and wall shear stress. *Magnetic resonance in medicine*, 72(2):522–533, 2014.

[43] Barbara M Johnston, Peter R Johnston, Stuart Corney, and David Kilpatrick. Non-newtonian blood flow in human right coronary arteries: steady state simulations. *Journal of biomechanics*, 37(5):709–720, 2004.

[44] N Shahcheraghi, HA Dwyer, AY Cheer, AI Barakat, and T Rutaganira. Unsteady and three-dimensional simulation of blood flow in the human aortic arch. *Journal of biomechanical engineering*, 124(4):378–387, 2002.

[45] Vitaliy L Rayz, Loic Boussel, Gabriel Acevedo-Bolton, Alastair J Martin, William L Young, Michael T Lawton, Randall Higashida, and David Saloner. Numerical simulations of flow in cerebral aneurysms: comparison of cfd results and in vivo mri measurements. *Journal of biomechanical engineering*, 130(5):051011, 2008.

[46] Pouya Dehghani Tafti, Ricard Delgado-Gonzalo, Aurélien F Stalder, and Michael Unser. Variational enhancement and denoising of flow field images. In *Biomedical Imaging: From Nano to Macro, 2011 IEEE International Symposium on*, pages 1061–1064. IEEE, 2011.

[47] Emrah Bostan, Stamatios Lefkimmiatis, Orestis Vardoulis, Nikolaos Stergiopulos, and Michael Unser. Improved variational denoising of flow fields with application to phase-contrast MRI data. *IEEE Signal Processing Letters*, 22(6):762–766, 2015. ISSN 1070-9908.

[48] Frank Ong, Martin Uecker, Umar Tariq, Albert Hsiao, Marcus T. Alley, Shreyas S. Vasanawala, and Michael Lustig. Robust 4D flow denoising using divergence-free wavelet transform. *Magnetic Resonance in Medicine*, 73(2):828–842, 2015.

[49] Erwan Deriaz and Valérie Perrier. Divergence-free and curl-free wavelets in two dimensions and three dimensions: Application to turbulent flows. *Journal of Turbulence*, 7(3):1–37, 2006. ISSN 14685248. doi: 10.1080/14685240500260547.

[50] Julia Busch, Daniel Giese, Lukas Wissmann, and Sebastian Kozerke. Reconstruction of divergence-free velocity fields from cine 3D phase-contrast flow measurements. *Magnetic resonance in medicine*, 69(1):200–210, 2013.

[51] Samuel M Song, Sandy Napel, Gary H Glover, and Norbert J Pelc. Noise reduction in three-dimensional phase-contrast MR velocity measurementsl. *Journal of Magnetic Resonance Imaging*, 3(4):587–596, 1993.

[52] Vinicius C Rispoli, Jon F Nielsen, Krishna S Nayak, and Joao LA Carvalho. Computational fluid dynamics simulations of blood flow regularized by 3d phase contrast mri. *Biomedical engineering online*, 14(1):110, 2015.

[53] Andrei Nikolaevich Tikhonov, AV Goncharsky, VV Stepanov, and Anatoly G Yagola. *Numerical methods for the solution of ill-posed problems*, volume 328. Springer Science & Business Media, 2013.

[54] Suhas Patankar. *Numerical heat transfer and fluid flow*. CRC press, 1980.

[55] Gaetan Kerschen, Jean-claude Golinval, Alexander F Vakakis, and Lawrence A Bergman. The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: an overview. *Nonlinear dynamics*, 41(1-3):147–169, 2005.

[56] Ali Bakhshinejad, Ahmadreza Baghaie, Alireza Vali, David Saloner, Vitaliy L. Rayz, and Roshan M. D'Souza. Merging computational fluid dynamics and 4D Flow MRI using proper orthogonal decomposition and ridge regression. *Journal of Biomechanics*, 58:162–173, jun 2017.

[57] Ali Bakhshinejad, Vitaliy Rayz, and Roshan M. D'Souza. Reconstructing blood velocity profiles from noisy 4d-pcmr data using ensemble kalman filtering. In *Biomedical Engineering Society (BMES) 2016 Annual Meeting, Minneapolis, MN, USA*, 2016.

[58] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996. ISSN 00359246.

[59] L Sirovich. Method of snapshots. *Q. Appl. Math*, 45:561–571, 1987.

[60] Kevin M. Johnson and Michael Markl. Improved SNR in phase contrast velocimetry with five-point balanced flow encoding. *Magnetic Resonance in Medicine*, 63(2):349–355, 2010. ISSN 07403194. doi: 10.1002/mrm.22202.

[61] Gene H Golub and Christian Reinsch. Singular value decomposition and least squares solutions. *Numerische mathematik*, 14(5):403–420, 1970.

[62] Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Montreal, Canada, 1995.

[63] R. Everson and L. Sirovich. Karhunen–Loève procedure for gappy data. *Journal of the Optical Society of America A*, 12(8):1657, 1995.

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

[65] Alexander Yakhot, Tomer Anor, and George Em Karniadakis. A reconstruction method for gappy and noisy arterial flow data. *IEEE Transactions on Medical Imaging*, 26(12):1681–1697, 2007.

[66] G Berkooz, P Holmes, and J L Lumley. The Proper Orthogonal Decomposition in the Analysis of Turbulent Flows. *Annual Review of Fluid Mechanics*, 25(1):539–575, 1993.

[67] S. S. Ravindran. A reduced-order approach for optimal control of fluids using proper orthogonal decomposition. *International Journal for Numerical Methods in Fluids*, 34(5):425–448, nov 2000.

[68] Ali Bakhshinejad, Ahmadreza Baghaie, Vitaliy L Rayz, and Roshan M D'Souza. A proper orthogonal decomposition approach towards merging CFD and 4D-PCMR flow data. In *The 28th Society for Magnetic Resonance Angiography*, 2016.

[69] D. Needell and J. A. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, 2009.

[70] Matan Gavish and David L. Donoho. The Optimal Hard Threshold for Singular Values is 4/sqrt(3). *IEEE Transactions on Information Theory*, 60(8):5040–5053, 2014.

[71] J. Nathan Kutz, Steven L. Brunton, Bingni W. Brunton, and Joshua L. Proctor. *Dynamic Mode Decomposition*. SIAM, 2016. ISBN 978-1-61197-449-2.

[72] Robert Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996. ISSN 00359246.

[73] Mojtaba Fathi, Ali Bakhshinejad, Ahmadreza Baghaie, and Roshan D'Souza. Dynamic Denoising and Gappy Data Reconstruction Based on Dynamic Mode Decomposition and Discrete Cosine Transform. *Applied Sciences*, 8(9):1515, 2018.

[74] G. V. Iungo, C. Santoni-Ortiz, M. Abkar, F. Porté-Agel, M. A. Rotea, and S. Leonardi. Data-driven Reduced Order Model for prediction of wind turbine wakes. *Journal of Physics: Conference Series*, 625(1), 2015. ISSN 17426596. doi: 10.1088/1742-6596/625/1/012009.

[75] John L Crassidis and John L Junkins. *Optimal estimation of dynamic systems*. Chapman and Hall/CRC, 2004.

[76] Yilun Chen, Ami Wiesel, and Alfred O. Hero. Shrinkage estimation of high dimensional covariance matrices. *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2937–2940, 2009. doi: 10.1109/ICASSP.2009.4960239.

[77] Kaare Brandt Petersen, Michael Syskind Pedersen, and Others. The matrix cookbook. *Technical University of Denmark*, 7(15):510, 2008.

[78] T. Kustner, C. Wurslin, S. Gatidis, P. Martirosian, K. Nikolaou, N. F. Schwenzer, F. Schick, B. Yang, and H. Schmidt. MR Image Reconstruction Using a Combination of Compressed Sensing and Partial Fourier Acquisition: ESPReSSo. *IEEE Transactions on Medical Imaging*, 35(11):2447–2458, 2016. ISSN 1558254X. doi: 10.1109/TMI.2016.2577642.

[79] Michael Lustig, David Donoho, and John M Pauly. Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 58(6):1182–1195, 2007.

Appendix

The Divergence-Free DCT-Based Basis Vectors

In this section, the derivation steps of the proposed basis functions and the corresponding forward and inverse discrete transforms are presented. The idea is first introduced for the case of a 2D velocity field. Later, it is shown how to extend the idea to 3D velocity fields.

2D velocity fields

Consider a divergence-free velocity field defined over a 2D Cartesian region where $x_{min} \leq x \leq x_{max}$ and $y_{min} \leq y \leq y_{max}$. The divergence-free velocity field must satisfy this equation:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{A1}$$

where u and v are the velocity components along the x (horizontal) and y (vertical) directions, respectively. Let's linearly map the x and y coordinates to a new set of Cartesian coordinates r and s as below

$$\begin{cases} l_x \triangleq x_{max} - x_{min} \\ l_y \triangleq y_{max} - y_{min} \end{cases} \rightarrow \begin{cases} x = x_{min} + \frac{l_x}{\pi} r \\ y = y_{min} + \frac{l_y}{\pi} s \end{cases} \tag{A2}$$

where $r, s \in [0, \pi]$. The divergence-free equation (A1) may be rewritten in terms of the new coordinates r and s as

$$\frac{\partial}{\partial r}\left(\frac{u}{l_x}\right) + \frac{\partial}{\partial s}\left(\frac{v}{l_y}\right) = 0 \tag{A3}$$

Now consider the pairs of functions f_{pq} and g_{pq} defined as

$$\begin{cases} f_{pq}(x_1, x_2) \triangleq \cos(px_1)\cos(qx_2) \\ g_{pq}(x_1, x_2) \triangleq \frac{p}{q}\sin(px_1)\sin(qx_2) = px_2\sin(px_1)\,\mathrm{sinc}\left(\frac{q}{\pi}x_2\right) \end{cases} \tag{A4}$$

where p and q are arbitrary constants. It is clearly seen the definition of the function f_{pq} above is inspired by the definition of the discrete cosine transform (DCT) basis vectors. It can be easily verified

$$\frac{\partial}{\partial x_1} f_{pq} + \frac{\partial}{\partial x_2} g_{pq} = 0 \tag{A5}$$

So, these pairs of functions satisfy the divergence-free condition (A3) for any arbitrary choice of p and q constants. Figure A1 shows the first few basis functions corresponding to $p, q = 0, 1, 2$. In general, p and q don't need to be integers and can be any arbitrary real number.

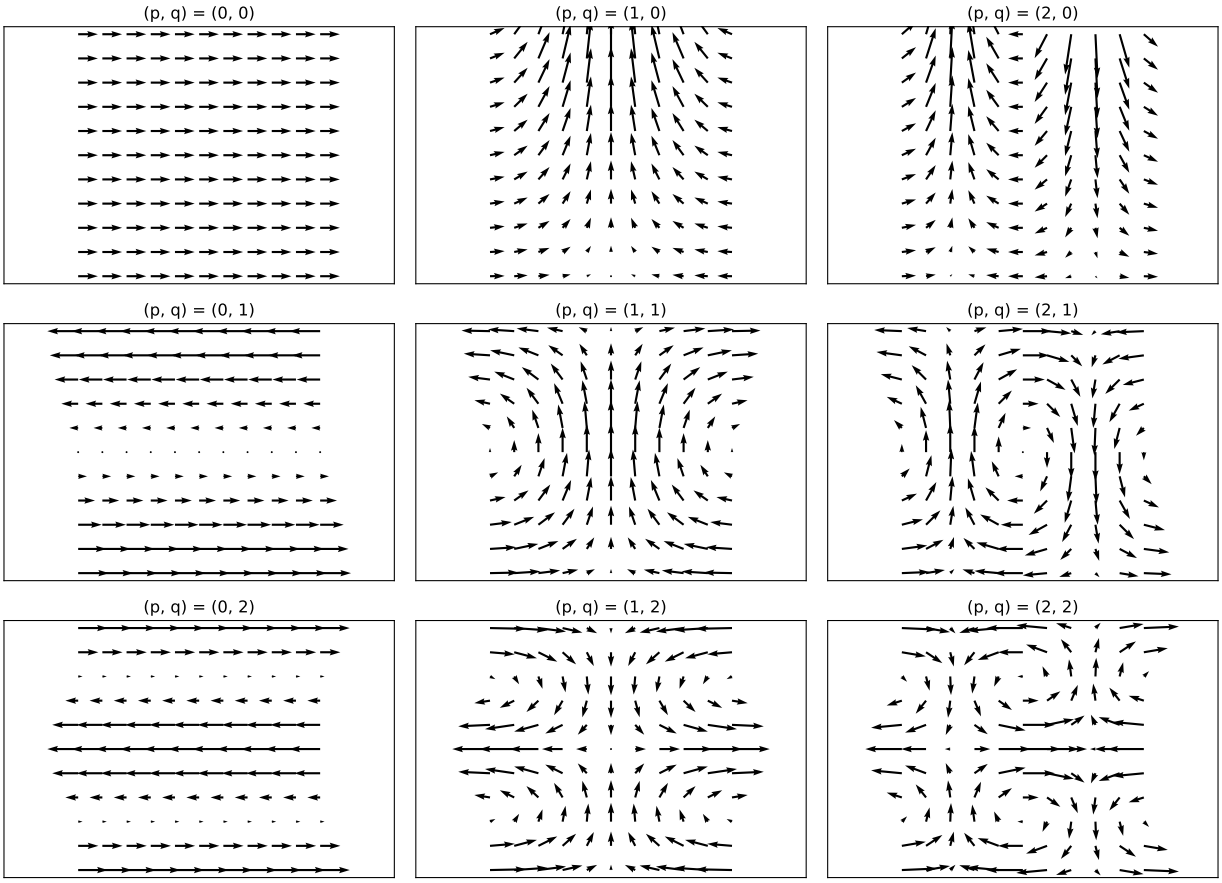


Figure A1: Plot of the first few divergence-free basis functions. The corresponding p and q constants are indicated.

Using these basis functions, the following form may be used for representing the components of a divergence-free velocity field in 2D

$$\begin{cases} \hat{u}(r,s) = l_x \sum_{p,q} a_{pq} f_{pq}(r,s) + l_x \sum_{p',q'} a'_{p'q'} g_{q'p'}(s,r) \\ \hat{v}(r,s) = l_y \sum_{p,q} a_{pq} g_{pq}(r,s) + l_y \sum_{p',q'} a'_{p'q'} f_{q'p'}(s,r) \end{cases} \tag{A6}$$

where a_{pq} and $a'_{p'q'}$ are the constant coefficients to be determined.

Discretization

Suppose the Cartesian domain of (r,s) is discretized at $n_r \times n_s$ equally-spaced locations (r_i, s_j) defined as

$$\begin{cases} r_i \triangleq \left(i + \frac{1}{2}\right) \delta_r \quad , \ 0 \leq i < n_r, \ i \in \mathbb{N}_0, n_r \in \mathbb{N} \\ s_j \triangleq \left(j + \frac{1}{2}\right) \delta_s \quad , \ 0 \leq j < n_s, \ j \in \mathbb{N}_0, n_s \in \mathbb{N} \end{cases}, \begin{cases} \delta_r \triangleq \frac{\pi}{n_r} \\ \delta_s \triangleq \frac{\pi}{n_s} \end{cases} \tag{A7}$$

where n_r and n_s are the number of divisions along r and s directions, respectively. In this scheme, the r and s coordinates are uniformly divided into n_r and n_s pieces of respective lengths δ_r and δ_s and then, the middle points of all pieces are taken as the equidistant sampling points.

Using this scheme, the velocity component $u(r,s)$ can be discretized and represented as an $n_s \times n_r$ matrix $\mathbf{U}$ defined as

$$\mathbf{U}_{n_s \times n_r} = [u_{ji}] \ \text{ where } u_{ji} = u(r_i, s_j) \tag{A8}$$

The same applies to the velocity component $v(r,s)$ too.

Inverse discrete transform

Using matrix notation, the inverse discrete transform equations may be written as

$$\begin{cases} \mathbf{U} = l_x \mathbf{C}_s \mathbf{X} \mathbf{C}_r^T + l_x \mathbf{S}_s \mathbf{Y} \mathbf{D}_r^T \\ \mathbf{V} = l_y \mathbf{D}_s \mathbf{X} \mathbf{S}_r^T + l_y \mathbf{C}_s \mathbf{Y} \mathbf{C}_r^T \end{cases} \tag{A9}$$

where

$$\mathbf{c}_p \triangleq \frac{1}{\sqrt{n_r}} \begin{bmatrix} \cos(pr_0) \\ \vdots \\ \cos(pr_{n_r-1}) \end{bmatrix}, \quad \mathbf{C}_r \triangleq \begin{bmatrix} \mathbf{c}_0 & \dots & \mathbf{c}_{n_r-1} \end{bmatrix} \tag{A10}$$

$$\mathbf{s}_p \triangleq \frac{1}{\sqrt{n_r}} \begin{bmatrix} \sin(pr_0) \\ \vdots \\ \sin(pr_{n_r-1}) \end{bmatrix}, \quad \mathbf{S}_r \triangleq \begin{bmatrix} \mathbf{s}_0 & \dots & \mathbf{s}_{n_r-1} \end{bmatrix} \tag{A11}$$

$$\mathbf{d}_p \triangleq \frac{1}{\sqrt{n_r}} \begin{bmatrix} r_0 \operatorname{sinc}\left(\frac{p}{\pi} r_0\right) \\ \vdots \\ r_{n_r-1} \operatorname{sinc}\left(\frac{p}{\pi} r_{n_r-1}\right) \end{bmatrix}, \quad \mathbf{D}_r \triangleq \begin{bmatrix} \mathbf{d}_0 & \dots & \mathbf{d}_{n_r-1} \end{bmatrix} \tag{A12}$$

$$\mathbf{c}'_q \triangleq \frac{1}{\sqrt{n_s}} \begin{bmatrix} \cos(qs_0) \\ \vdots \\ \cos(qs_{n_s-1}) \end{bmatrix}, \quad \mathbf{C}_s \triangleq \begin{bmatrix} \mathbf{c}'_0 & \dots & \mathbf{c}'_{n_s-1} \end{bmatrix} \tag{A13}$$

$$\mathbf{s}'_q \triangleq \frac{1}{\sqrt{n_s}} \begin{bmatrix} \sin(qs_0) \\ \vdots \\ \sin(qs_{n_s-1}) \end{bmatrix}, \quad \mathbf{S}_s \triangleq \begin{bmatrix} \mathbf{s}'_0 & \dots & \mathbf{s}'_{n_s-1} \end{bmatrix} \tag{A14}$$

$$\mathbf{d}'_q \triangleq \frac{1}{\sqrt{n_s}} \begin{bmatrix} s_0 \operatorname{sinc}\left(\frac{q}{\pi} s_0\right) \\ \vdots \\ s_{n_s-1} \operatorname{sinc}\left(\frac{q}{\pi} s_{n_s-1}\right) \end{bmatrix}, \quad \mathbf{D}_s \triangleq \begin{bmatrix} \mathbf{d}'_0 & \dots & \mathbf{d}'_{n_s-1} \end{bmatrix} \tag{A15}$$

3D velocity fields

The proposed 2D divergence-free basis functions may be extended to 3D as explained here. A given divergence-free velocity field defined over a 3D Cartesian region must satisfy this equation:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0 \tag{A16}$$

where u, v, and w are the velocity components along the x, y, and z directions, respectively. The x, y, and z coordinates are linearly mapped to the new set of Cartesian coordinates r, s, and t as

below

$$\begin{cases} l_x \triangleq x_{max} - x_{min} \\ l_y \triangleq y_{max} - y_{min} \\ l_z \triangleq z_{max} - z_{min} \end{cases} \rightarrow \begin{cases} x = x_{min} + \frac{l_x}{\pi} r \\ y = y_{min} + \frac{l_y}{\pi} s \\ z = z_{min} + \frac{l_z}{\pi} t \end{cases} \tag{A17}$$

where $r, s, t \in [0, \pi]$. The divergence-free equation (A16) becomes

$$\frac{\partial}{\partial r}\left(\frac{u}{l_x}\right) + \frac{\partial}{\partial s}\left(\frac{v}{l_y}\right) + \frac{\partial}{\partial t}\left(\frac{w}{l_z}\right) = 0 \tag{A18}$$

The pair of divergence-free functions f_{pqm} and g_{pqm} are defined as

$$\begin{cases} f_{pqm}(x_1, x_2, x_3) \triangleq f_{pq}(x_1, x_2)\cos(mx_3) = \cos(px_1)\cos(qx_2)\cos(mx_3) \\ g_{pqm}(x_1, x_2, x_3) \triangleq g_{pq}(x_1, x_2)\cos(mx_3) = px_2 \sin(px_1)\,\mathrm{sinc}\left(\frac{q}{\pi}x_2\right)\cos(mx_3) \end{cases} \tag{A19}$$

where p, q, and m are arbitrary constants.

Using these basis functions, the following form may be used for representing the components of a divergence-free velocity field in 3D

$$\begin{cases} \hat{u}(r,s,t) = l_x \sum_{p,q,m} a_{pqm} f_{pqm}(r,s,t) + l_x \sum_{p'',q'',m''} a''_{p''q''m''} g_{m''p''q''}(t,r,s) \\ \hat{v}(r,s,t) = l_y \sum_{p',q',m'} a'_{p'q'm'} f_{q'm'p'}(s,t,r) + l_y \sum_{p,q,m} a_{pqm} g_{pqm}(r,s,t) \\ \hat{w}(r,s,t) = l_z \sum_{p'',q'',m''} a''_{p''q''m''} f_{m''p''q''}(t,r,s) + l_z \sum_{p',q',m'} a'_{p'q'm'} g_{q'm'p'}(s,t,r) \end{cases} \tag{A20}$$

where a_{pqm}, $a'_{p'q'm'}$, and $a''_{p''q''m''}$ are the constant coefficients to be determined.

Discretization

The same discretization scheme as the 2D case may be used

$$\begin{cases} r_i \triangleq \left(i + \frac{1}{2}\right)\delta_r & , \ 0 \le i < n_r, \ i \in \mathbb{N}_0, n_r \in \mathbb{N} \\ s_j \triangleq \left(j + \frac{1}{2}\right)\delta_s & , \ 0 \le j < n_s, \ j \in \mathbb{N}_0, n_s \in \mathbb{N} \\ t_k \triangleq \left(k + \frac{1}{2}\right)\delta_t & , \ 0 \le k < n_t, \ j \in \mathbb{N}_0, n_t \in \mathbb{N} \end{cases}, \begin{cases} \delta_r \triangleq \frac{\pi}{n_r} \\ \delta_s \triangleq \frac{\pi}{n_s} \\ \delta_t \triangleq \frac{\pi}{n_t} \end{cases} \tag{A21}$$

The velocity components may be represented as 3D tensors. For example, the velocity component $u(r,s,t)$ is discretized and represented as an $n_t \times n_s \times n_r$ tensor $\mathbb{U}$ defined as

$$\mathbb{U}_{n_t \times n_s \times n_r} = [u_{kji}] \text{ where } u_{kji} = u(r_i, s_j, t_k) \tag{A22}$$

The same applies to the velocity components $v(r,s,t)$ and $w(r,s,t)$ as well.

Inverse discrete transform

Using 3D tensor notation defined above, the inverse discrete transform equations in 3D may be written as

$$\begin{cases} \mathbb{U} = l_x \mathbf{C}_t \otimes (\mathbf{C}_s \otimes (\mathbf{C}_r \otimes \mathbb{X})) + l_x \mathbf{S}_t \otimes (\mathbf{C}_s \otimes (\mathbf{D}_r \otimes \mathbb{Z})) \\ \mathbb{V} = l_y \mathbf{C}_t \otimes (\mathbf{C}_s \otimes (\mathbf{C}_r \otimes \mathbb{Y})) + l_y \mathbf{C}_t \otimes (\mathbf{D}_s \otimes (\mathbf{S}_r \otimes \mathbb{X})) \\ \mathbb{W} = l_z \mathbf{C}_t \otimes (\mathbf{C}_s \otimes (\mathbf{C}_r \otimes \mathbb{Z})) + l_z \mathbf{D}_t \otimes (\mathbf{S}_s \otimes (\mathbf{C}_r \otimes \mathbb{Y})) \end{cases} \tag{A23}$$

where

$$\mathbf{c}''_m \triangleq \frac{1}{\sqrt{n_t}} \begin{bmatrix} \cos(mt_0) \\ \vdots \\ \cos(mt_{n_t-1}) \end{bmatrix}, \quad \mathbf{C}_t \triangleq \begin{bmatrix} \mathbf{c}''_0 & \dots & \mathbf{c}''_{n_r-1} \end{bmatrix} \tag{A24}$$

$$\mathbf{s}''_m \triangleq \frac{1}{\sqrt{n_t}} \begin{bmatrix} \sin(mt_0) \\ \vdots \\ \sin(mt_{n_t-1}) \end{bmatrix}, \quad \mathbf{S}_t \triangleq \begin{bmatrix} \mathbf{s}''_0 & \dots & \mathbf{s}''_{n_t-1} \end{bmatrix} \tag{A25}$$

$$\mathbf{d}''_m \triangleq \frac{1}{\sqrt{n_t}} \begin{bmatrix} t_0 \operatorname{sinc}\left(\frac{m}{\pi} t_0\right) \\ \vdots \\ t_{n_t-1} \operatorname{sinc}\left(\frac{m}{\pi} t_{n_t-1}\right) \end{bmatrix}, \quad \mathbf{D}_t \triangleq \begin{bmatrix} \mathbf{d}''_0 & \dots & \mathbf{d}''_{n_t-1} \end{bmatrix} \tag{A26}$$

Curriculum Vitae

Mojtaba F. Fathi

linkedin.com/in/mojif • mojtaba.fathi@gmail.com

Education

Earned Degrees

Master of Science in Mechatronics June 2012
Sharif University of Technology Tehran, Iran

Master of Science in Mechanical Engineering June 2002
K.N.Toosi University of Technology Tehran, Iran

Bachelor of Science in Mechanical Engineering August 1999
University of Tehran Tehran, Iran

Dissertation Title

Applications of Dynamic Mode Decomposition and Sparse Reconstruction in The Data-Driven Dynamic Analysis of Physical Systems

Professional Experience

Lab Equipment Design

Controls Lab Equipment Designer Summer 2015
University of Wisconsin-Milwaukee, Mechanical Engineering Dept. Milwaukee, WI

Revamping *Control Systems Lab.* equipment

- Designed a new controller box based on Arduino
- Assembled the controller boxes and auxiliary parts
- Developed a Simulink toolbox for the new controller box

Teaching Experience

Graduate Teaching Assistant 2013–Present
University of Wisconsin-Milwaukee, Mechanical Engineering Dept. Milwaukee, WI

- Introduction to Control Systems
- Modeling and Analysis of Dynamic Systems
- Mechatronics
- CAD

Instructor 2009–2009
Azad University, Civil Engineering Dept. Parand, Iran

- Computer Programming Course

Publications and Research

Journal Articles (Peer Reviewed)

Fathi, M., Bakhshinejad, A., Baghaie, A., & D'Souza, R. (2018). Dynamic Denoising and Gappy Data Reconstruction Based on Dynamic Mode Decomposition and Discrete Cosine Transform. *Applied Sciences*, 8(9), 1515.

Fathi, M. F., Bakhshinejad, A., Baghaie, A., Saloner, D., Sacho, R. H., Rayz, V. L., & D'Souza, R. M. (2018). Denoising and Spatial Resolution Enhancement of 4D Flow MRI Using Proper Orthogonal Decomposition and Lasso Regularization. *Computerized Medical Imaging and Graphics*.

Baghaie, A., Schnell, S., Bakhshinejad, A., **Fathi, M. F.**, D'Souza, R. M., & Rayz, V. L. (2018). Curvelet Transform-based volume fusion for correcting signal loss artifacts in time-of-Flight Magnetic Resonance Angiography data. *Computers in biology and medicine*.

Seo, N. J., **Fathi, M. F.**, Hur, P., & Crocher, V. (2016). Modifying Kinect placement to improve upper limb joint angle measurement accuracy. *Journal of Hand Therapy*, 29(4), 465-473.

Omrani, E., Tafti, A. P., **Fathi, M. F.**, Moghadam, A. D., Rohatgi, P., D'Souza, R. M., & Yu, Z. (2016). Tribological study in microscale using 3D SEM surface reconstruction. *Tribology International*, 103, 309-315.

Seo, N. J., Enders, L. R., Motawar, B., Kosmopoulos, M. L., & **Fathi-Firoozabad, M.** (2015). The extent of altered digit force direction correlates with clinical upper extremity impairment in chronic stroke survivors. *Journal of biomechanics*, 48(2), 383-387.

Abstract/Posters

Mojtaba F. Fathi, Ali Bakhshinejad, David Saloner, Raphael H. Sacho, Vitaliy L. Rayz, Roshan M. D'Souza. (2018). Denoising 4D-Flow MRI With Dynamic Mode Decomposition and Kalman Filter. *The 8th World Congress of Biomechanics*, Dublin, Ireland.

Mojtaba F. Firoozabad, Pilwon Hur, Na Jin Seo. (2014). Determination of the optimal location of kinect sensor for upper-limb virtual rehabilitation. *The 7th World Congress of Biomechanics*, Boston, MA.