Dynamic Mode Decomposition and Koopman Operator

(Data-Driven Modeling of Complex Dynamical Systems)

Master of Science Thesis in Applied and Computational Mathematics

Hugo Gonçalo Antunes Moreira

June 2019

Acknowledgments

I would like to thank my supervisors, Prof. Guttorm Alendal and Anna Oleinik, for their patience, support, guidance and interest on following me and the subject of this thesis. Thank you to all my colleagues, for making me feel welcomed and as part of the student adventure. A special mention to Yafee Ishraq, for all his support on the long study days and evenings, and for the precious input while reading my drafts. Enormous gratitude for my mother and sister who from a long distance always expressed their support and understanding. And finally, a very special thank you to my daughter, Lara, whose infinite patience and understanding for my unexpected decision to return to university meant the world to me.

Abstract

Data-driven schemes are in high demand, given the growing abundance and accessibility to large amounts of measurements from historical records, numerical simulations, and experimental data. However, despite the abundance of data, modeling high-dimensional complex dynamical systems remains a challenge. In this thesis we present a data-driven method for modeling dynamical systems called the Dynamic Mode Decomposition (DMD). This is a recent method that has first emerged in the fluid mechanics community as a tool for analyzing the dynamics of nonlinear systems. However, given its ability to provide an accurate decomposition of a complex system into spatiotemporal coherent structures, it gained popularity and interest from other fields where complex nonlinear processes cannot be accurately characterized by known governing equations, or that exhibit a rich multiscale dynamic properties. This method relies on the fact that many of these systems evolve on a low-dimensional attractor that may be characterized by dominant spatiotemporal coherent structures. The confidence that the DMD is useful to characterize non-linear dynamics is given by theoretical framework provided by Koopmans theory, which will also be presented in the thesis. Short examples are used to illustrate the DMD application and the Koopmans operator theory. Finally, two data-sets generated from two different fields (from a 2D ocean model, and a neuron strip experiment) are tested using the DMD. We will use the decomposition results to identify structures which we may relate to a physical phenomena, and discuss the performance.

Contents

Notation

Vectors

$\boldsymbol{v}$ is a vector of dimension n in a space $\mathbb{C}^n$, where $n \in \mathbb{N}$;

$\boldsymbol{v}^*$ is the complex conjugate of $\boldsymbol{v}$;

$\overline{v}$ is the average of the components of $\boldsymbol{v}$;

$\text{diag}(A)$ is the vector with the diagonal components of a matrix A;

Matrices

A is a $n \times m$ matrix;

a_{ij} is the $(i,j)^{th}$ entry of a matrix A;

$\mathbb{M}^{n \times m}$ is the space of $n \times m$ matrices;

$\det(A)$ is the determinant of matrix A;

A^* is the complex conjugate transpose of matrix A;

A^+ is the Moore-Penrose pseudo-inverse of matrix A, computed as $A^+ = (A^*A)^{-1}A^*$;

Functions

$\boldsymbol{f}$ is a function $\boldsymbol{f} : \mathbb{S} \to \mathbb{F}$;

$\boldsymbol{f}_x$ is the partial derivative w.r.t. x, that is, $\boldsymbol{f}_x = \partial \boldsymbol{f} / \partial x$;

$\boldsymbol{f}_{xx}$ is the second order partial derivative w.r.t. x, that is, $\boldsymbol{f}_{xx} = \partial^2 \boldsymbol{f} / \partial x^2$;

$\boldsymbol{f}_{xy}$ is the second order partial derivative w.r.t. x and y, that is, $\boldsymbol{f}_{xy} = \partial^2 \boldsymbol{f} / (\partial x \partial y)$;

$\dot{\boldsymbol{f}}$ is the time derivative of $\boldsymbol{f}$, that is, $\dot{\boldsymbol{f}} = d\boldsymbol{f}/dt$;

$\ddot{\boldsymbol{f}}$ is the second time derivative of $\boldsymbol{f}$, that is, $\ddot{\boldsymbol{f}} = d^2\boldsymbol{f}/dt^2$;

Acronyms/Abbreviations

DMD Dynamic Mode Decomposition

FFT Fast Fourier Transform

PDE Partial Differential Equation

POD Proper Orthogonal Decomposition

ODE Ordinary Differential Equation

SVD Singular Value Decomposition

Introduction

The description of the dynamics of complex systems involves the construction of models to accurately simulate high-dimensional processes such as, for example, the hydrodynamics of the ocean through Navier-Stokes equations. Additionally, given the large scale complexity of such processes, high computational cost is required for solving such models. In order to reduce these costs while preserving an acceptable numerical accuracy, reduced order modeling schemes are of great importance.

One other scenario is the incomplete knowledge or even the unavailability of access to the governing equations that can accurately describe the system we wish to model. The spread of infectious diseases, neuron networks, or other biological processes, are examples of such systems. A dynamical system, in the abstract sense, is an evolution rule that describes how one state develops into another over the course of time.

We consider that $\boldsymbol{f}$ is associated with an autonomous continuous dynamical system, in particular,

$$\dot{\boldsymbol{y}}(\boldsymbol{x}, t) = \boldsymbol{f}(\boldsymbol{y}(x, t)), \tag{0.1}$$

where $\boldsymbol{y}(\boldsymbol{x}, t)$ is the state of the system at time t and $\boldsymbol{x}$ the spatial distribution over some domain Ω, and $\boldsymbol{f}$ is a vector field that maps smooth manifold $\mathcal{M} \subset \mathbb{R}^n$ into itself.

Since we are interested in numerical solutions of (0.1), we also consider the discrete-time dynamical systems which can be induced by considering a flow map $\boldsymbol{F} : \mathcal{M} \to \mathcal{M}$, which maps the state $\boldsymbol{y}_k$ at time k to a future state $\boldsymbol{y}_{k+1}$ by

$$\boldsymbol{F}(\boldsymbol{y}_k) = \boldsymbol{y}_k + \int_{k\Delta t}^{(k+1)\Delta t} \boldsymbol{f}(\boldsymbol{y}(\boldsymbol{x}, \tau))d\tau. \tag{0.2}$$

For simplicity, we will use $\boldsymbol{f}$ instead of $\boldsymbol{F}$, if it is clear from the context. The discrete representation of the autonomous dynamical system then takes the form

$$\boldsymbol{y}_{k+1} = \boldsymbol{f}(\boldsymbol{y}_k). \tag{0.3}$$

When having access to data, the main goal here is to model complex dynamical systems with a reduced computational cost and extract the relevant and meaningful dynamical structures, while maintaining its accuracy within an acceptable margin.

The dynamic mode decomposition (DMD), first proposed by Schmid in [22], is a purely data-driven, equation-free method that extracts dynamic information (in the form of eigenvalues and eigenfunctions) from data generated by numerical simulations or experimental data. It does not require the knowledge of the governing equations for the dynamical system, *e.g.* (0.1) and (0.3), relying solely on the gathered input data to extract its dynamic modes. The growing interest on the application of this method is related to its potential usage as a diagnostic tool, for model order reduction, as a future-state predictor and for control applications. Furthermore, the connection with the Koopman spectral analysis of non-linear dynamical systems provided the DMD with the theoretical framework so that it can be used as a tool for the analysis of general non-linear systems.

This thesis is organized as follows. In Chapter 1 we briefly present the Proper Orthogonal Decomposition (POD) technique, which lays the concepts of model reduction which capitalizes on the existence of low rank dominant dynamics in the system, to obtain an optimal basis functions spanning a lower-dimensional subspace.

Chapter 2 introduces the Koopman operator theory, which provides a mathematical foundation for the application of the DMD to data generated by nonlinear systems. At the end of Chapter 2 we introduce four short practical examples to illustrate some of the underlying concepts behind the Koopman operator.

In Chapter 3 we present the DMD, where we introduce the algorithmic formulation, its variations, and a theoretical framework which connects the DMD with the Koopman operator. Finally, we revisit the short examples introduced in the previous chapter to establishing the practical connection between the Koopman operator and the DMD method.

In Chapter 4 we present two different applications where we measure the results and test the performance of the method using the concepts and techniques introduced in the previous chapters. Each application correspond to two very distinct fields: the first one is for a 2D hydrodynamic velocity field model which data was generated from a numerical simulation of the Bergen Ocean Model [1]; and the second corresponds to data collected from a neural field experiment [28].

Chapter 5 we give a summary of the results, make concluding remarks and discuss future work.

Chapter 1

Proper Orthogonal Decomposition

The main motivation to present the Proper Orthogonal Decomposition (POD) technique in the context of this thesis is to introduce the concept of a reduced-order models and snapshot methods. Although the focus of this thesis is not on this technique, for the sake of completeness we present a brief description of the method. In this chapter, we closely follow [18, 20].

The POD, broadly speaking, is a technique of finding in a optimal way a basis which spans an ensemble of data collected from an experimental or numerical simulation of a dynamical system. This method has been often used in developing low-dimensional models of fluids [23]. The idea is, given a set of data that lies in a vector space $\mathcal{V}$, to find a subspace $\mathcal{V}_r$ of fixed dimension r such that the error in the projection onto the subspace is minimized.

1.1 Optimality of the POD Basis

Suppose we have a set of data $y(t) \in \mathbb{R}^d$, with $0 \leq t \leq T$. We seek a projection $P_r : \mathbb{R}^d \to \mathbb{R}^d$ of fixed rank r that minimizes the total error

$$\int_0^T \|y(t) - P_r y(t))\|^2 \, dt. \tag{1.1}$$

To solve this problen, we introduce the $d \times d$ matrix

$$R = \int_0^T y(t) y(t)^* dt, \tag{1.2}$$

and find the eigenvalues and eigenvectors of R, given by

$$R\phi_j = \lambda_j \phi_j, \quad \lambda_1 \geq \cdots \geq \lambda_q \geq 0. \tag{1.3}$$

Since R is symmetric, positive-semidefinite, all the eigenvalues λ_j are real and nonnegative, and the eigenvectors ϕ_j may be chosen to be orthonormal. The vectors ϕ_j are called the POD modes. The optimal subspace of dimension r is spanned by $\{\phi_1, \ldots, \phi_r\}$, and the optimal projection P_r is given by

$$P_r = \sum_{j=1}^{r} \phi_j \phi_j^*. \tag{1.4}$$

1.2 Computation of reduced-order models

Having determined P_r, assume now that we are determining the solutions of a system described by equations (0.1). To capitalize on the POD modes one can form reduced order models using Galerkin projection (see, *e.g.* L.C.Evans [10]), which specifies that $\dot{y}(t) = P_r \boldsymbol{f}(y(t))$, *i.e.*, projecting the original vector field $\boldsymbol{f}$ onto the r-dimensional subspace. We then write

$$\boldsymbol{y}_r(t) = \sum_{j=1}^{r} \alpha_j(t)\phi_j, \tag{1.5}$$

Substituting (1.5) on (0.1) and multiplying by ϕ_j^*, we obtain

$$\dot{\alpha}_j(t) = \phi_j^* f(\boldsymbol{x}(t)), \quad j = 1, ..., r \tag{1.6}$$

which is a set of r ODE that describe the evolution of $\boldsymbol{x}_r(t)$. In other words, the determination of solutions to (0.1) which previously involved solving a set of d ODE, can be now reduced to a set of r ODE.

1.3 Snapshot-based methods

When analyzing a time series of data on a spatial grid, it is often beneficial to use snapshot-based methods. The two or three-dimensional vector field data at time t_k is then rearranged into a single column vector. If we consider a discrete set of snapshots $\{\boldsymbol{y}(t_k)\}_{k=0}^{q} \in \mathbb{R}^d$ generated by some physical process described by the equations (0.3), the ensemble of snapshots can be expressed as in the following matrix,

$$X = \begin{pmatrix} y_{0,1} & y_{1,1} & \cdots & y_{q,1} \\ y_{0,2} & y_{1,2} & \cdots & y_{q,2} \\ \vdots & \vdots & \vdots & \vdots \\ y_{0,d} & y_{1,d} & \cdots & y_{q,d} \end{pmatrix}. \tag{1.7}$$

In this framework, the POD can be formulated using the SVD of the matrix X, that is, snapshots X,

$$X = U\Sigma V, \tag{1.8}$$

where $U \in \mathbb{C}^{n\times n}, V \in \mathbb{C}^{q\times q}$, and $\Sigma \in \mathbb{R}^{n\times q}$. U and V are unitary matrices, and Σ is a diagonal rectangular matrix with positive singular values $\{\sigma_1, \sigma_2, ..., \sigma_r\}$, where r denotes the number of positive singular values.

The column vectors in $U = \{\phi_1, \phi_2, ..., \phi_n\}$ and $V = \{\varphi_1, \varphi_2, ..., \varphi_q\}$ contain the orthogonal eigenvectors of XX^T and X^TX, respectively, as in

$$\begin{cases} X = U\Sigma V^T \\ X^T = V\Sigma^* U^T \end{cases} \Rightarrow \begin{cases} XX^T = U\Sigma\Sigma^T U^T \\ X^TX = V\Sigma^T\Sigma V^T \end{cases} \Rightarrow \begin{cases} XX^TU = U\Lambda \\ X^TXV = V\Lambda, \end{cases}$$

where $\Lambda = \Sigma\Sigma^T = \Sigma^T\Sigma = \sum_{k=1}^{q} \lambda_q$.

The singular values of the snapshot matrix X are then associated with the eigenvalues of the matrices X^TX and XX^T by the relation $\lambda_k = \sigma_k^2$.

Since XX^T is symmetric and positive-semidefinite, all the eigenvalues λ_k are real and non-negative, and by virtue of the properties of the SVD, the eigenvectors U are orthonormal.

The main result is that the optimal POD subspace of dimension l is spanned by $\{\phi_1, \phi_2, ..., \phi_l\}$, and the optimal projection is given by $P_r = \sum_{k=1}^{q} \phi_k\phi_k^T$.

The basis vectors, re-written as $U_r = (\phi_1, \ldots, \phi_r)$, are called the POD modes, where the $r \leq \min(n, q)$ is the number of nonzero singular-values, corresponding to the rank of X.

1.4 Dimension reduction

We now set our goal to determine an optimal subspace of $\mathcal{V}$ which is of the lowest possible dimension $r << l$, while maintaining a good approximation to the original data set. In other words, we seek a reduced-order system such that the exact solution of $\boldsymbol{y}_k$ can be approximated by a linear combination of r basis vectors, where U_r.

As proposed in [15], we measure the approximation by using the relative information content referred as *energy*, defined as

$$I(r) = \frac{\sum_{i=1}^{r} \sigma_i^2}{\sum_{j=1}^{l} \sigma_j^2}, \tag{1.9}$$

The goal is to choose the smallest r such that $I(r)$ is still sufficiently close to 1. For example, if the subspace U_r should contain a percentage γ of the information in U, then one should chose r such that

$$r = \min_r \Big(I(r) \geq \frac{\gamma}{100} \Big).$$

1.5 Conclusions

In this chapter we have seen that the POD method identifies an optimal orthogonal basis of spatially and temporal correlated modes U and V, respectively, capitalizing on the SVD method of decomposing a data matrix. It is essentially a model reduction technique, and it is based on the assumption that the evolution in time of the dynamics of the system is governed by a reduced number of dominant modes.

Associated with the Galerkin projection, the dimension of the system of governing equations we have to solve in order to determine its solution can be greatly reduced. However, even for a reduced dimension r, these systems may still be expensive to simulate.

To connect the POD to the feature method of this thesis, the DMD method, we emphasize on what makes them distinct: we will see that the DMD not only provides the modes of the system, as POD, but it also associates these correlated spatial modes with a temporal frequency and a possible growth or decay rate.

Chapter 2

Koopman Operator

Much of the interest surrounding the DMD method comes from the strong connection to nonlinear dynamical systems through Koopman spectral theory. In this chapter we follow closely [5, 6, 18, 20].

In the field of dynamical systems, the composition operator

$$\mathcal{K}\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{g} \circ \boldsymbol{f}(\boldsymbol{x}). \tag{2.1}$$

is often referred to as the Koopman operator. Here, $\boldsymbol{f}$ is associated with evolution rule and $\boldsymbol{g}$ belongs to a Hilbert space

$$L^2(\mathcal{M}) = \left\{ \boldsymbol{g} : \mathcal{M} \to \mathbb{C} | \boldsymbol{g} \text{ measurable, and } \int_{\mathcal{M}} \left| \boldsymbol{g}^2 \right| dx < +\infty \right\},$$

with the $\langle f, g \rangle_{L^2(\mathcal{M})} := \int_{\mathcal{M}} f g d\mathcal{M}$, which induces the norm, $\|\boldsymbol{g}\|_{L^2}^2 = \langle \boldsymbol{g}, \boldsymbol{g} \rangle_{L^2(\mathcal{M})}$. We call this function $\boldsymbol{g}$ observable.

It was shown in the original work by Koopman [14] that for a map $\boldsymbol{f}$ which is invertible and measure-preserving, $\mathcal{K}$ is a unitary operator, belonging to a Hilbert space with the usual definition of inner product and induced norm in L^2. We refer to [19] for extension of these results.

Using the definition of the Koopman operator to the discrete case, having the bounded operator $\mathcal{K} : \mathcal{H} \to \mathcal{H}$ (see Appendix C) acting on all possible measurements of the state $\boldsymbol{x}_k \in \mathcal{M}$, $\boldsymbol{g}(\boldsymbol{x}_k) \in \mathbb{C}$, we write

$$\mathcal{K}\boldsymbol{g}(\boldsymbol{x}_k) = \boldsymbol{g}(\boldsymbol{x}_{k+1}), \tag{2.2}$$

where $\boldsymbol{g}(\boldsymbol{x}_{k+1}) \in \mathbb{C}$ are all possible measurements of the state $\boldsymbol{x}_{k+1}$ at time $k+1$.

In the continuous case, using the chain rule on (2.1) the Koopman operator definition is expressed as

$$\mathcal{K}\boldsymbol{g}(\boldsymbol{x}) = \nabla \boldsymbol{g}(\boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}). \tag{2.3}$$

The underlying idea behind this transformation is that a dynamical system, mapped from a possible nonlinear finite-dimensional space $\mathcal{M}$ onto a infinite-dimensional Hilbert space $\mathcal{H}$, $\boldsymbol{g} \in \mathcal{H}(\mathcal{M}, \mathbb{C})$, can now be represented by a infinite-dimensional linear operator $\mathcal{K}$ as in (2.2) in the discrete case, or (2.3) in the continuous case.

2.1 Spectral Decomposition of the Koopman operator

To represent the solution of a dynamical system in the discrete-time case (0.3), the spectral decomposition of the linear Koopman operator $\mathcal{K}$ is

$$\mathcal{K}\varphi_j(\boldsymbol{x}) = \lambda\varphi_j(\boldsymbol{x}), \quad j = \{1, 2, \dots\}, \tag{2.4}$$

where $\varphi_j : \mathcal{M} \to \mathbb{R}$, are the eigenvectors and $\lambda_j \in \mathbb{C}$ the eigenvalues of the *Koopman* operator $\mathcal{K}$.

If the vector observable $\boldsymbol{g}(\boldsymbol{x})$ lies within the span of the eigenfunctions $\{\varphi_j\}_{j=1}^n$, where n may be infinite, then $\boldsymbol{g}$ may be expanded in terms of the eigenfunctions,

$$\boldsymbol{g}(\boldsymbol{x}) = \sum_{j=1}^{n} v_j \varphi_j(\boldsymbol{x}), \tag{2.5}$$

where $\{v_j\}_{j=1}^n$ is a set of scalar coefficients called *Koopman modes* of the map $\boldsymbol{f}$.

The dynamics of $\boldsymbol{g}(\boldsymbol{x}_k)$ decomposition can be obtained by first iterating (2.2) relative to the initial condition of the state $\boldsymbol{x}$, yielding

$$\begin{aligned}
[\mathcal{K}\boldsymbol{g}](\boldsymbol{x}_0) &= \boldsymbol{g}(\boldsymbol{x}_1) \\
[\mathcal{K}^2\boldsymbol{g}](\boldsymbol{x}_0) &= \boldsymbol{g}(\boldsymbol{x}_2) \\
&\vdots \\
[\mathcal{K}^k\boldsymbol{g}](\boldsymbol{x}_0) &= \boldsymbol{g}(\boldsymbol{x}_k),
\end{aligned}$$

then plugging in (2.5),

$$\mathcal{K}^k\boldsymbol{g}(\boldsymbol{x}_0) = \mathcal{K}^k \sum_{j=1}^{n} v_j \varphi_j(\boldsymbol{x}_0), \tag{2.6}$$

and finally, from (2.4)

$$\boldsymbol{g}(\boldsymbol{x}_{k+1}) = \sum_{j=1}^{n} \lambda_j^k v_j \varphi_j(\boldsymbol{x}_0). \tag{2.7}$$

The Koopman eigenvalues $\{\lambda_j\}_{j=1}^n$ characterize the growth rate and frequency of each corresponding Koopman mode v_j. The phase of λ_j determines its frequency and its magnitude the rate of growth. φ_j is the eigenfunction of $\mathcal{K}$ which is a function of the initial condition.

Let us now consider the vector of observable functions $\boldsymbol{g}(\boldsymbol{x}) \in \mathcal{H}$, such that $\boldsymbol{g} : \mathcal{M} \to \mathcal{D}^n$, where $\mathcal{D}^n \subset \mathbb{C}$, and $\boldsymbol{x} \in \mathbb{R}^m$ and $\boldsymbol{f}$ is as previously defined in (0.1). If $\mathcal{D}$ is an invariant subspace spanned by the eigenfunctions of the Koopman operator $\{\varphi_j(\boldsymbol{x})\}_{j=1}^n$, where $n < \infty$, such that a linear operator $\boldsymbol{K} : \mathcal{D}^n \to \mathcal{D}^n$, then $\boldsymbol{K}$ is also finite-dimensional.

That is, if there is a restriction $\boldsymbol{g} = (g_1, \ldots, g_n)^T$, where $n < \infty$, which induces $\boldsymbol{K} : \mathcal{D}^n \to \mathcal{D}^n$, where $\mathcal{D}^n$ is a subspace of $\mathbb{C}$, then the expression

$$[K\boldsymbol{g}](\boldsymbol{x}_k) = \sum_{j=1}^{n} \lambda_j v_j \varphi_j(\boldsymbol{x}_k),$$

holds. Equivalently,

$$\boldsymbol{g}(\boldsymbol{x}_{k+1}) = \sum_{j=1}^{m} \lambda_j^k v_j \varphi_j(\boldsymbol{x}_0). \tag{2.8}$$

In practice, the goal is to find an invariant finite-dimensional Hilbert sub-space ($\mathcal{D}^n$) to where our nonlinear dynamical system can be mapped, while conserving all its dynamical characteristics.

2.2 Examples of Koopman modes

In this section following [5, 20], two examples illustrate that the eigenvalues and eigenfunctions of the Koopman operator are related to the eigenmodes for linear systems and to the discrete Fourier transform for periodic solutions.

2.2.1 Koopman modes for linear systems

Let us consider a special case when $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$ is linear, that is, $\boldsymbol{f}(\boldsymbol{x}) = A\boldsymbol{x}$. Note that n may be infinity. Consider that A has a complete set of eigenvectors and corresponding eigenvalues denoted by $\boldsymbol{v}_j$ and λ_j, respectively, such that,

$$A\boldsymbol{v}_j = \lambda_j \boldsymbol{v}_j, \;\; j = 1, ..., n. \tag{2.9}$$

Let $\boldsymbol{w}_i$ be the eigenfunctions of the adjoint A^*, such that, $A^*\boldsymbol{w}_j = \overline{\lambda_j}\boldsymbol{w}_j$, with $\overline{\lambda_j}$ the corresponding eigenvalue. Next, we define the scalar valued function

$$\varphi_j(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{w}_j \rangle, \;\; j = 1, ..., n.$$

Then, since

$$\boldsymbol{K}\varphi_j(\boldsymbol{x}) = \varphi(A\boldsymbol{x}) = \langle A\boldsymbol{x}, \boldsymbol{w}_j\rangle = \langle \boldsymbol{x}, A^*\boldsymbol{w}_j\rangle = \left\langle \boldsymbol{x}, \overline{\lambda}_j\boldsymbol{w}_i\right\rangle = \lambda_j \langle \boldsymbol{x}, \boldsymbol{w}_j\rangle = \lambda_j\varphi_j(\boldsymbol{x}),$$

φ_i are eigenfunctions of $\boldsymbol{K}$.

Now, for any $\boldsymbol{x} \in \mathcal{M}$, as long as A has a full set of eigenvectors, we may write,

$$\boldsymbol{x} = \sum_{j=1}^{n} \langle \boldsymbol{x}, \boldsymbol{w}_j\rangle \boldsymbol{v}_j = \sum_{i=1}^{n} \varphi_j(\boldsymbol{x})\boldsymbol{v}_j$$

From these expressions and (2.5), for linear systems, the Koopman modes $\boldsymbol{v}_j$ coincide with the eigenvectors of A.

2.2.2 Koopman modes for periodic systems

Consider a nonlinear system which solution to (0.3) is periodic. Assume the set of vectors generated by $\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k)$ is $X = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{m-1})$, such that $\boldsymbol{x}_{k+m} = \boldsymbol{x}_k$ for any k, where $X \in \mathcal{M}$.

A way to analyze this solution is to take its Fourier transform. Let us consider the set of vectors $F = (\widehat{\boldsymbol{x}}_0, \ldots, \widehat{\boldsymbol{x}}_{m-1})$, where $F \in \mathbb{C}$, that satisfy

$$\boldsymbol{x}_k = \sum_{j=0}^{m-1} e^{2\pi ijk/m}\widehat{\boldsymbol{x}}_j, \quad k = 0, \ldots, m-1. \tag{2.10}$$

Let us now define a set of functions $\varphi : \mathcal{M} \to \mathbb{C}$,

$$\varphi_j(\boldsymbol{x}_k) = e^{2\pi ijk/m}, \quad j, k = 0, \ldots, m-1. \tag{2.11}$$

Acting on all functions $\varphi_j(\boldsymbol{x}_k)$ with a linear operator $\boldsymbol{K}$, we obtain

$$\boldsymbol{K}\varphi_j(\boldsymbol{x}_k) = \varphi_j(f(\boldsymbol{x}_k)) = \varphi(\boldsymbol{x}_{k+1}) = e^{2\pi ij(k+1)/m} = e^{2\pi ij/m}e^{2\pi ijk/m} = e^{2\pi ij/m}\varphi(\boldsymbol{x}_k). \tag{2.12}$$

By comparing this result with (2.4), we see that φ_j are the eigenfunctions of the Koopman operator $\boldsymbol{K}$, with eigenvalues $\lambda_j = e^{2\pi ij/m}$.

Recalling (2.10), and plugging in (2.11), we get

$$\boldsymbol{x}_k = \sum_{j=0}^{m-1} \varphi_j(\boldsymbol{x}_k)\widehat{\boldsymbol{x}}_j. \tag{2.13}$$

This expression is equivalent in form to (2.5). Thus, if we restrict our phase space to the periodic orbit S, the Koopman modes are the vectors given by the discrete Fourier transform $\widehat{\boldsymbol{x}}_j$ and the phases of the corresponding eigenvalues $\lambda_j = e^{2\pi ij/m}$ are the frequencies given by $2\pi j/m$.

As discussed in papers [16, 20], this result can be generalized to non-periodic systems, when the dynamics are restricted to any attractor.

2.3 Examples of Simple Applications

To illustrate applications of the Koopman Operator theory, we introduce 4 short examples of nonlinear dynamical systems.

In the first three examples we look into nonlinear ODE. Although examples 2 and 3 are not distinct when it comes to the process for determining the linear operator $\boldsymbol{K}$ and the conclusions in the context of this chapter are also similar, each of them will be useful in different ways on Chapter 3.

In example 4 we solve the Burgers' equation, and look into the connection between the analytical solution using the Fourier transform and Koopman theory.

2.3.1 Example 1 - Nonlinear ODE

Let us consider a nonlinear homogeneous ODE defined by

$$\begin{cases} \dot{x_1} = & \mu x_1 \\ \dot{x_2} = & \xi(x_2 - x_1^2), \end{cases} \tag{2.14}$$

where, the vector $\boldsymbol{x} \in \mathcal{M} \subseteq \mathbb{R}^2$, and μ a constant. Consider now the set of observables,

$$\boldsymbol{g}(\boldsymbol{x}) := (x_1, x_2, x_1^2)^T, \tag{2.15}$$

where, $\boldsymbol{g} : \mathcal{M} \to \mathcal{D}$. A change of variables $\boldsymbol{z} := \boldsymbol{g}(\boldsymbol{x})$, results in

$$\begin{cases} \dot{z_1} = \dot{x_1} = \mu x_1 = \mu z_1 \\ \dot{z_2} = \dot{x_2} = \xi(x_2 - x_1^2) = \xi(z_2 - z_3) \\ \dot{z_3} = 2x_1\dot{x_1} = 2x_1\mu x_1 = 2\mu x_1^2 = 2\mu z_3, \end{cases} \tag{2.16}$$

which, in matrix form, we write

$$\begin{pmatrix} \dot{z_1} \\ \dot{z_2} \\ \dot{z_3} \end{pmatrix} = \begin{pmatrix} \mu & 0 & 0 \\ 0 & \xi & -\xi \\ 0 & 0 & 2\xi \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix}. \tag{2.17}$$

We now have a linear system of ODE of the form $\dot{z} = \boldsymbol{K}z$, as in (2.3).

For first order linear system of ODE in (2.17), the solution is of the form $\boldsymbol{z}(t) = c_1 e^{\lambda_1 t}\varphi_1 + c_2 e^{\lambda_2 t}\varphi_2 + c_3 e^{\lambda_3 t}\varphi_3$, where λ_j and φ_j are the eigenvalues and eigenvectors, respectively, of $\boldsymbol{K}$, and c_j are constants.

For the analytical solution of (2.17), being a linear system, we start by determining the eigenvalues of $\boldsymbol{K}$, with

$$\det(A - \lambda I) = 0.$$

The eigenvalues are

$$\lambda_1 = \mu; \quad \lambda_2 = \xi; \quad \lambda_3 = 2\xi,$$

and the corresponding eigenvectors,

$$\varphi_1 = (1, 0, 0)^T; \quad \varphi_2 = (0, 1, 0)^T; \quad \varphi_3 = (0, -1, 2)^T.$$

The solution of (2.17), given by the states x_1 and x_2, which we can obtain from the transformation of variables $\boldsymbol{z} := \boldsymbol{g}(\boldsymbol{x})$, $(x_1, x_2) = (z_1, z_2)$, is thus,

$$\begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} = c_1 \begin{pmatrix} 1 \\ 0 \end{pmatrix} e^{\mu t} + c_2 \begin{pmatrix} 0 \\ 1 \end{pmatrix} e^{\xi t} + c_3 \begin{pmatrix} 0 \\ -1 \end{pmatrix} e^{2\xi t}. \tag{2.18}$$

Assuming that the parameters ξ and μ are real and negative (so that the solution $\boldsymbol{x}(t)$ is stable and not oscillatory).

It is important to note that this particular choice of observables (2.15) allowed us to find an invariant finite-dimensional Hilbert sub-space $\mathcal{D} = span\{\varphi_1, \varphi_2, \varphi_3\}$, where the solution to (2.17) was easy to obtain. Determining the solution $\boldsymbol{x}$ in the original space $\mathcal{M}$ was made easy by the selection of the observables, since that for $\boldsymbol{g}^{-1} : \mathcal{D} \to \mathcal{M}$, $\boldsymbol{g}^{-1}(x_1, x_2, x_1^2)^T = (x_1, x_2)^T$.

2.3.2 Example 2 - Logistic Map

In the next example, we consider the logistic map

$$x_{k+1} = \mu x_k (1 - x_k), \tag{2.19}$$

where $x \in [0, 1] \subset \mathbb{R}$ is the variable and $\mu \in [0, 4] \subset \mathbb{R}$ a parameter. A chaotic behavior of this nonlinear system can arise from the choice of the parameter μ.

Let us select a new mapping with the observables, as we did on the previous example, and select the nonlinear term in (2.19) such that,

$$\boldsymbol{g}(\boldsymbol{x}_k) = (x_k, x_k^2)^T. \tag{2.20}$$

We define the change of variables $\boldsymbol{y}_k := \boldsymbol{g}(\boldsymbol{x}_k)$. Then, it yields

$$\begin{cases} y_{k+1,1} &= \mu x_k - \mu x_k^2 \\ y_{k+1,2} &= (\mu x_k - \mu x_k^2)^2 \quad = \mu x_k^2 - 2\mu^2 x_k^3 + \mu^2 x_k^4, \end{cases} \tag{2.21}$$

which induces third and forth order polynomials of the state x_k. To have a linear representation of these measurements, we will add to the vector of observables these induced nonlinearities, that is,

$$\boldsymbol{g}(\boldsymbol{x}) = (x_k, x_k^2, x_k^3, x_k^4)^T. \tag{2.22}$$

However, taking the same steps, we find that these added observables will induce polynomials of order six and eight, thus continuing the cycle and extending it to infinity.

In matrix form,

$$\begin{pmatrix} x_{k+1} \\ x_{k+1}^2 \\ x_{k+1}^3 \\ x_{k+1}^4 \\ x_{k+1}^5 \\ \vdots \end{pmatrix} = \begin{pmatrix} \mu & -\mu & 0 & 0 & 0 & 0 & 0 & 0 & \dots \\ 0 & \mu^2 & -2\mu^2 & \mu^2 & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & \mu^3 & -3\mu^3 & 3\mu^3 & \mu^3 & 0 & 0 & \dots \\ 0 & 0 & 0 & \mu^4 & -4\mu^4 & 6\mu^4 & 4\mu^4 & \mu^4 & \dots \\ 0 & 0 & 0 & 0 & \mu^5 & -5\mu^5 & 10\mu^5 & -10\mu^5 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} x_k \\ x_k^2 \\ x_k^3 \\ x_k^4 \\ x_k^5 \\ \vdots \end{pmatrix}. \tag{2.23}$$

In this case, the dimension for the Koopman operator matrix reaches infinite, which doesn't violate Koopmans theory, since it allows infinite-dimensional space of all possible measurements of state $\boldsymbol{x}$. However, the infinite-dimensional nature of the problem makes it, in computational terms, unattainable to solve.

To test if with a truncation in (2.23) we can still achieve a good approximation, we solve the resulting linear system and compare it with the exact solution.

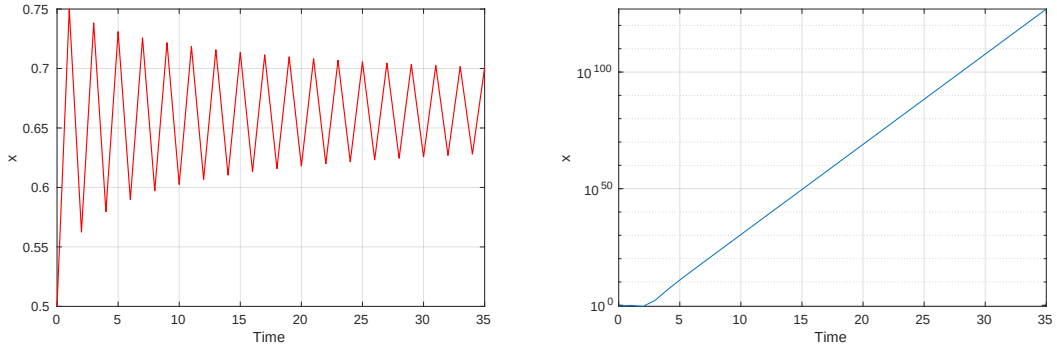
Testing in Matlab©for $k = 7$ with $x_0 = 0.5$, as seen in Figures 2.1 and 2.2, it is obvious to conclude that the simple truncation of the system is not a good method to obtain approximate solutions.

2.3.3 Example 3 - Van der Pol

The Van der Pol oscillator is expressed as

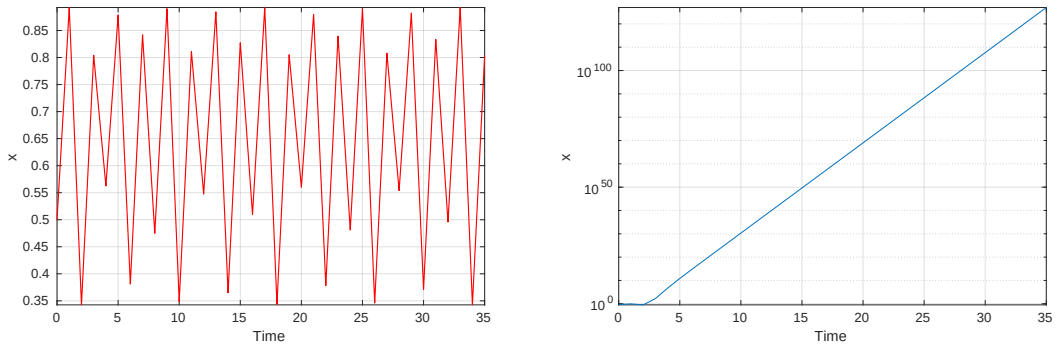
$$\ddot{y} - \mu(1 - y^2)\dot{y} + y = 0, \tag{2.24}$$

where $y \in \mathbb{R}$ is the state corresponding to the position coordinate, which is a function of time t, and $\mu \in \mathbb{R}_0^+$ is a scalar. This scalar parameter gives us a measure on how strongly



(a) (b)

Figure 2.1: *Solution $\boldsymbol{x}_k$, for $\mu = 3$ and $\boldsymbol{x}_0 = 0.5$. In (a) the exact solution, and in (b) the solution obtained from the truncated system.*



(a) (b)

Figure 2.2: *Solutions $\boldsymbol{x}_k$, for $\mu = 3.57$ and $\boldsymbol{x}_0 = 0.5$. In (a) the exact solution, and in (b) the solution obtained from the truncated system.*

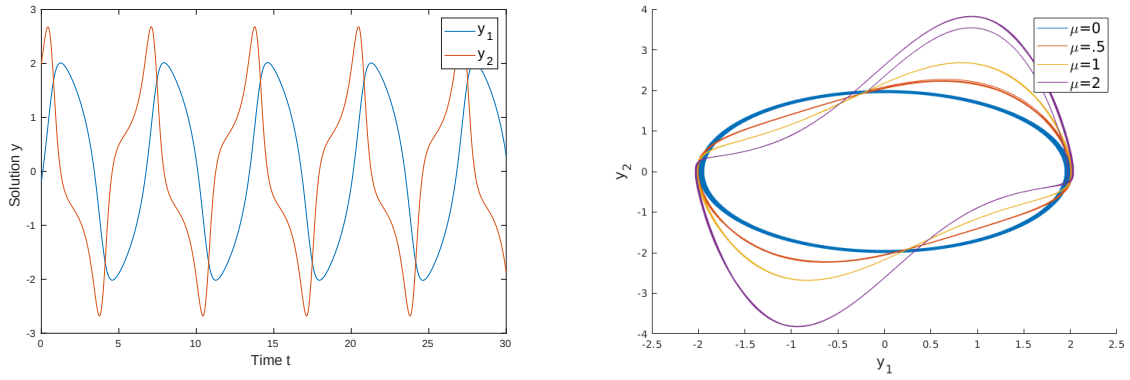
this system is non-linear. A weight of $\mu = 0$ would give us a linear system, the simple harmonic motion $\ddot{y} + y = 0$.

The second order ODE, can be transformed into a first order system of ODE,

$$\begin{cases} \dot{y_1} &= & y_2 \\ \dot{y_2} &= & \mu(1 - y_1^2)y_2 - y_1. \end{cases} \tag{2.25}$$

The numerical solution of (2.25), obtained from Matlab©, using the command ***ode*45**, can be visualized in Figure 2.3.

Just as we did in section 2.3.2, we expand our vector of observables with the nonlinear terms of (2.25). For convenience, we define a change of variables for the vector of



(a) ($\mu = 1$)

(b) Limit cycle in phase space for values of μ

Figure 2.3: *The solution of the Van der Pol Oscillator for different values of μ in (b) and the time evolution of the states $\boldsymbol{y}$ with $\mu = 1$.*

observables as $\mathbf{z} := \mathbf{g}(\mathbf{y})$, where

$$\mathbf{g}(\mathbf{y}) = (y_1, y_2, y_1^2 y_2, y_1 y_2^2, y_1^2 y_2^2, y_1 y_2^3, y_1^3 y_2, \dots)^T. \tag{2.26}$$

In matrix form, we expand (2.25) with (2.26), as

$$\begin{pmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \dot{z}_3 \\ \dot{z}_4 \\ \dot{z}_5 \\ \dot{z}_6 \\ \dot{z}_7 \\ \vdots \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & \dots \\ -1 & \mu & -\mu & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & \mu/2 & (2+\mu/4) & -\mu/2 & 0 & 0 & \dots \\ 0 & 0 & -2\mu & 2\mu & 1/5 & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & 2 & 2\mu & -2 & \dots \\ 0 & 0 & 0 & 0 & 3\mu & -3 & 0 & \dots \\ 0 & 0 & 3 & 0 & 0 & 0 & \mu & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \\ z_6 \\ z_7 \\ \vdots \end{pmatrix}. \tag{2.27}$$

Just as in the previous example, using the nonlinear terms as observables to determine the transformation $\mathbf{g}(\mathbf{y})$, defined in (2.26) fails to determine a finite-dimensional Koopman operator.

2.3.4 Example 4 - Burgers' Equation (PDE)

In this example we will focus on solving a PDE, and in particular, the Burger's equation

$$u_t + uu_x - \epsilon u_{xx} = 0, \tag{2.28}$$

with diffusive regulation and a nonlinear advection. Equivalently,

$$u_t = (\epsilon u_x - \frac{u^2}{2})_x. \tag{2.29}$$

Let us consider the solutions $u(x,t)$ to the one-dimensional (2.28), over a domain $x \in \big[-L, L\big]$, and

$$u_t = f(u), \tag{2.30}$$

where the function $f : \mathbb{R} \to \mathbb{R}$ is given by

$$f(u) = \epsilon u_{xx} - u u_x, \tag{2.31}$$

with Dirichlet boundary conditions $u(\pm L, t) = 0$.

The exact solution for this problem can be obtained by the application of the Cole-Hopf transformation of variables. We will see that this is equivalent, in the context of Koopmans theory, as defining a vector of observables.

The Cole-Hopf transformation, $h\big(u(x,t)\big) = v(x,t)$, where $h : \mathbb{R} \to \mathcal{M} \subseteq \mathbb{R}$ is given by

$$h\big(u(x,t)\big) = \exp\left(-\frac{1}{2\epsilon} \int_{-\infty}^{x} u(\xi, t) d\xi \right). \tag{2.32}$$

We now have a transformation that maps a strongly nonlinear PDE to a linear diffusion equation expressed as

$$v = \epsilon v_{xx}. \tag{2.33}$$

Expression (2.33) can be derived by starting from

$$u = -2\epsilon \frac{v_x}{v}, \tag{2.34}$$

which is equivalent to (2.32). From this we can obtain

$$\begin{aligned} u_t &= -2\epsilon \left(\frac{v_x}{v}\right)_t = -2\epsilon \left[\frac{v_{xt} v - v_t v_x}{v^2}\right] = -2\epsilon \left(\frac{v_t}{v}\right)_x, \\ u_x &= -2\epsilon \left(\frac{v_x x}{x} - \left(\frac{v_x}{v}\right)^2\right), \text{ and} \\ u^2 &= 4\epsilon^2 \left(\frac{v_x}{v}\right)^2, \end{aligned}$$

which we plug into (2.29), so that

$$-2\epsilon \left(\frac{v_t}{v}\right)_x = -2\epsilon^2 \left(\frac{v_{xx}}{v}\right)_x \Rightarrow v_t = \epsilon v_{xx}, \text{ from integration.}$$

The solution to (2.33) can be found by applying the Fourier transform in x, which is given by

$$\widehat{v}_t = -\epsilon k^2 \widehat{v}, \tag{2.35}$$

where $\widehat{v}$ denotes the Fourier transform of $v(x,t)$, and k the wavenumber.

We denominate the Fourier transform as $F(v)$, mapping $F : \mathcal{M} \to \mathcal{F} \subseteq \mathbb{C}$. The solution in the Fourier domain to the ODE of (2.35) is easily found as,

$$\widehat{v} = \widehat{v}_0 \exp(-\epsilon k^2 t), \tag{2.36}$$

where $\widehat{v}_0 = \widehat{v}(k,0)$ is the Fourier transform of the initial condition $v(x,0)$.

To establish the equivalency with Koopmans theory, the vector of observables can be derived as follows,

$$g\big(u(x,t)\big) = F \circ h\big(u(x,t)\big) = F\big(v(k,t)\big) = \widehat{v}(k,t).$$

In summary, the observable $g(u) = \hat{v}$ maps the function (2.30) to the Fourier space $\mathcal{F}$ as $g(u_t) = g(f(u))$, such that

$$\boldsymbol{K} g(u) = g(f(u)),$$

where $\boldsymbol{K} : \mathcal{D} \to \mathcal{D} \subseteq \mathcal{F}$ is the Koopman linear operator which, from (2.36), is given by

$$\boldsymbol{K} = \exp(-\epsilon k^2 t). \tag{2.37}$$

Computation of a particular solution

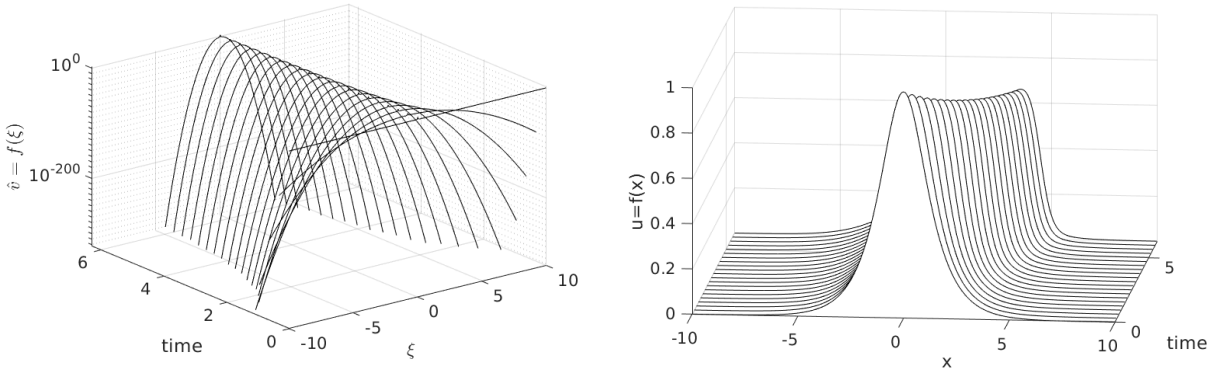
To illustrate this example, let us consider an initial condition for the Burgers' equation as $u_0 = sech(x)$, with $x \in [-10, 10]$ and parameter $\epsilon = 0.1$.

To find the solution u for the Burger's equation, we only need to compute the evolution of our observable $\widehat{v}$ given by (2.36) in the Fourier space (Figure 2.4(a)). To compute the solution of the Burgers equation $u(x,t)$, at any point in time, we have to use (2.34) to map the solution $\widehat{v}$ back to the original space (Figure 2.4(b)).

Explicit observables and Koopman operators that can be constructed analytically are, however, uncommon.

2.4 Comments and Conclusions

These examples highlighted that the right set of observables $\boldsymbol{g}(\boldsymbol{x})$ is crucial to analytically determine an invariant finite-dimensional Hilbert subspace $\mathcal{D}$ to which the nonlinear dy-



(a) Solution $\widehat{v} = K\widehat{v}_0$.

(b) Solution u.

Figure 2.4: *In (b) the solution $u(t)$ of the Burgers equation mapped from the solution to the linear problem in the Fourier space depicted in (a).*

namical system can be mapped. Furthermore, the determination of this set of observable functions may not be trivial, as showed in Example 4. One added difficulty we saw in the same example, expressed in (2.34), is that that the transformation of the observable function back to the original state space may also require additional calculations.

In the next chapter we present the purely data-driven DMD method which approximates the Koopman operator. This is extremely useful for enabling evaluation of the operator from data since it provides the mathematical framework of Koopmans theory with a computationally tractable algorithm.

To illustrate practical applications of this connection, the examples presented here will be revisited at the end of the next Chapter.

Chapter 3

Dynamic Mode Decomposition

Following the definitions in [13, 18], suppose we have two sets of data,

$$X = (\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_m), \quad Y = (\boldsymbol{y}_0, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_m), \tag{3.1}$$

such that $\boldsymbol{y}_k = \boldsymbol{f}(\boldsymbol{x}_k)$, where $\boldsymbol{f}$ is a map associated with the evolution of a dynamical system (0.1). The DMD computes the leading eigendecomposition of the best-fit linear operator A relating the data $Y \approx AX$.

The DMD modes, also called dynamic modes, are the eigenvectors of A, and each DMD mode corresponds to a particular eigenvalue of A.

Algorithmically, the DMD can be described as a method that inputs discrete data (3.1) generated from a dynamical system (0.1) and outputs the eigenvalues and eigenvectors satisfying

$$A\phi_j = \lambda_j \phi_j. \tag{3.2}$$

Having the low-rank approximations of the eigenvalues and eigenvectors for A, a continuous solution of $\dot{\tilde{\boldsymbol{y}}}(t) = A\tilde{\boldsymbol{y}}(t)$, where $\tilde{\boldsymbol{y}}(t) \approx \boldsymbol{y}(t)$, can be constructed as a function of time. This we call the linear model that approximates the solution of (0.1), which is the main goal for this chapter, thus

$$\boldsymbol{y}(t) \approx \sum_{j=1}^{r} v_j \phi_j(\boldsymbol{x}) \exp(\omega_j t), \tag{3.3}$$

where $\boldsymbol{x}$ is the state vector, v_j corresponds to a scalar, $\phi_j(\boldsymbol{x})$ the eigenfunctions of A, and, assuming that the data was collected with a uniform sampling time Δt,

$$\omega_j = \log(\lambda_j)/\Delta t. \tag{3.4}$$

A direct result of the formulation of the expansion of the solution as in (3.3) is that one now has access to characteristic spatiotemporal features of the system. The rate of growth/decay and frequency of oscillations of each DMD mode is given by the eigenvalue ω_j and the time dependent term $\exp(\omega_j t)$ gives us the dynamics associated to each mode $\phi_j(\boldsymbol{x})$ scaled with a constant v_j.

The determination of matrix A is trivial when the data-set (3.1) is generated by a linear dynamical system. However, when we have non-linear systems, we are not guaranteed to obtain good approximations from the simple application of the DMD method. In [20] it was showed that the DMD approximates the Koopman operator. This fact highlighted the important role played by the observables and their associated evolution manifolds.

3.1 Connection with Koopman Operator

In the first papers over the DMD it was required that the data was a sequential time series. A sequential time series is an ordered sequence such that $(\boldsymbol{z}_0, \ldots, \boldsymbol{z}_q)$ is generated by $\boldsymbol{z}_{k+1} = \boldsymbol{f}(\boldsymbol{z}_k)$. However, a more general definition of data was proposed in [13]. In the new definition, data is a set of pairs $\{(\boldsymbol{x}_0, \boldsymbol{y}_0), \cdots, (\boldsymbol{x}_m, \boldsymbol{y}_m)\}$, which are a not necessarily ordered. The emphasis of this new definition is on the linear-consistency property of the data which provides a theoretical framework for the algorithm and the connection with Koopman operator theory (see Section 3.3.3).

However, since the sequential time-series collected data is only a particular case of this new definition, we keep this assumption on all examples and applications throughout the thesis.

Let us now assume that A has a full set of eigenvectors, so that we can write the expansion

$$\boldsymbol{x}_k = \sum_{j=1}^{l} c_{jk}\phi_j, \tag{3.5}$$

where c_{jk} are some constants (See Appendix B for details). Then,

$$\boldsymbol{y}_k \approx A\boldsymbol{x}_k \tag{3.6}$$

$$\approx \sum_{j=1}^{l} Ac_{jk}\phi_j \tag{3.7}$$

$$\approx \sum_{j=1}^{l} \lambda_j c_{jk}\phi_j. \tag{3.8}$$

Comparing this result with the Koopman operator spectral decomposition expression (2.7), we find that the DMD modes ϕ_j correspond to the Koopman modes v_j, the DMD eigenvalues to the Koopman eigenvalues, and the constant c_{jk} to the eigenfunctions $\varphi_j(\boldsymbol{z}_k)$.

This Koopman analogy is what provides a mathematical foundation for applying the DMD to data generated by nonlinear systems.

3.2 Formulation in terms of the Frobenius companion matrix

The theory behind the algorithm presented in this section is based on the Frobenius companion matrix as proposed in [20, 21, 22].

Consider data in the snapshot matrix $\mathcal{X}$ represented as

$$\mathcal{X} = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_m), \tag{3.9}$$

where $\boldsymbol{x}_k \in \mathbb{R}^n$, and matrices X and Y as in (3.1), with $\boldsymbol{y}_k = \boldsymbol{x}_{k+1}$. Herein we assume that the snapshot $\mathcal{X}$ is an ordered sequence of data separated by a constant sampling time Δt.

We start by assuming that $\boldsymbol{x}_{k+1} = A\boldsymbol{x}_k$, and n is so large that we cannot compute eigenvalues of A directly.

A standard method for computing estimates of the eigenvalues of A is a Krylov method, which starts with an initial vector $\boldsymbol{x}_0$ (often random), and then computes iterates of $\boldsymbol{x}_0$. After $m-1$ iterations, one has a collection of m orthonormal vectors that span a Krylov subspace given by

$$K_n(A, \boldsymbol{x}_0) = span\left\{\boldsymbol{x}_0, A\boldsymbol{x}_0, \ldots, A^{m-1}\boldsymbol{x}_0\right\}. \tag{3.10}$$

The Arnoldi method is a type of Krylov method which involves computing the action of A on arbitrary vectors. See, *e.g.* [25] for more details.

Matrix A is not available, however, as we only have access to a data-set (3.1). P.J.Schmid in [22] proposes a variation of the Arnoldi algorithm which does not require the explicit knowledge of A. For that, we start by assuming the special case where the mth iterate $\boldsymbol{x}_m$ is a linear combination of the previous iterates, *i.e.*,

$$\boldsymbol{x}_m = A\boldsymbol{x}_{m-1} = c_0\boldsymbol{x}_0 + \ldots + c_{m-1}\boldsymbol{x}_{m-1}, \tag{3.11}$$

where $\boldsymbol{c} = (c_0, \ldots, c_{m-1})$ is a vector of constants. This yields,

$$AX = XC, \tag{3.12}$$

where

$$C = \begin{pmatrix} 0 & & \cdots & & c_0 \\ 1 & 0 & \cdots & & c_1 \\ & \ddots & \ddots & & \vdots \\ & & 1 & 0 & c_{m-2} \\ & & & 1 & c_{m-1} \end{pmatrix} \tag{3.13}$$

is a Frobenius companion matrix of dimension $(m \times m)$. The eigenvalues of C are then a subset of the eigenvalues of A (see box below).

To verify this, let the pair (φ_j, λ_j) be the eigenvectors and eigenvalues of C. Then, let T be a square matrix whose columns are the m linearly independent eigenvectors of C, and Λ a diagonal matrix with the corresponding eigenvalues $\{\lambda_j\}_{j=1}^{m}$. As T is invertible, since its columns are linearly independent, the eigendecomposition of C is thus, $C = T\Lambda T^{-1}$.

Starting from (3.12), and plugging in the eigendecomposition of C,

$$\begin{aligned} & AX = XC \\ \Leftrightarrow\ & AX = XT^{-1}\Lambda T \\ \Leftrightarrow\ & AXT^{-1} = XT^{-1}\Lambda TT^{-1} \\ \Leftrightarrow\ & AXT^{-1} = XT^{-1}\Lambda \end{aligned}$$

where XT^{-1} is the matrix of the eigenvectors of A with eigenvalue Λ. Moreover, v_j are the columns of $V = XT^{-1}$.

Due to the properties of power iteration, the linearity of the sequence (3.10) will occur gradually with the increase of m. So, if the m-th iterate is not a linear combination of the previous iterations, we write the residual,

$$\boldsymbol{r} = A\boldsymbol{x}_{m-1} - \sum_{j=0}^{m-1} c_j \boldsymbol{x}_j.$$

Equivalently,

$$\boldsymbol{r} = \boldsymbol{x}_m - X\boldsymbol{c}. \tag{3.14}$$

In this case, (3.12) becomes

$$AX = XC + \boldsymbol{r}\boldsymbol{e}^T, \tag{3.15}$$

where $\boldsymbol{e} = (0, ..., 1)$.

From (3.14) we know that the residual r is minimum when it is orthogonal to $span\{\boldsymbol{x}_0, ..., \boldsymbol{x}_{m-1}\}$, then c is chosen such that $\min_c |\langle \boldsymbol{r}, X\rangle|$.

The eigenvalues of C are now the approximations to the eigenvalues of A, called the *Ritz* values, and the corresponding approximate eigenvectors are given by $\varphi_j = X\phi_j$, called the *Ritz* vectors. See, *e.g*, [25].

The following theorem proven in [20], summarizes the above.

Theorem 3.2.1. *Consider a set of data cX as in (3.9), and let λ_j, ϕ_j be the empirical Ritz values and vectors of this sequence. Assume that λ_j are distinct. Then*

$$\boldsymbol{x}_k = \sum_{j=1}^{m} \lambda_j^k \phi_j, \quad k = \{0, \ldots, m-1\}, \tag{3.16}$$

$$\boldsymbol{x}_m = \sum_{j=1}^{m} \lambda_j^m \phi_j + \boldsymbol{r}, \quad \boldsymbol{r} \perp \{\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{m-1}\}. \tag{3.17}$$

Next, we resume the results from above in a form of an algorithm:

Algorithm 1

1: INPUT: Define X from (3.1);
2: Find constants c_i such that $\min_c |\langle \boldsymbol{r}, X\rangle|$.
3: Define the companion matrix C from (3.13);
4: Find eigenvalues and eigenvectors which satisfy $C\phi_j = \lambda_j \phi_j$;
5: OUTPUT:

- DMD modes ϕ_j
- Eigenvalues λ_j

Comments

From the properties associated with the Krylov methods, if $\boldsymbol{x}_k = A^k \boldsymbol{x}_0$, then the *Ritz* values λ_j are the same as the ones determined after m steps of the *Arnoldi* method, and ϕ_j are the corresponding eigenvectors.

3.3 SVD based algorithm

In the previous section, Algorithm 1 was formulated in terms of the *Frobenius* companion matrix (3.13). However, an alternative based on SVD was first proposed in [22] due to the unstability of Algorithm 1, since determining eigenvalues of C is an ill-conditioned problem.

In [13] the algorithm is modified so that it has a more general application by expanding on the definition of the data. This approach is also used to strengthen the connection with the Koopman operator (see Section 3.3.3).

3.3.1 Projected DMD

In this subsection, we assume X and Y as in (3.1), where $\boldsymbol{y}_k = \boldsymbol{x}_{k+1}$.

We start by preprocessing the data matrix X, applying the SVD, and plug in (3.15), so that

$$\begin{aligned} AX = Y \Leftrightarrow \\ \Leftrightarrow\ AU\Sigma V^* = Y \\ \Leftrightarrow\ U^*AU\Sigma V^* = U^*Y \\ \Leftrightarrow\ U^*AU\Sigma V^*V = U^*YV \\ \Leftrightarrow\ U^*AU = U^*YV\Sigma^{-1}. \end{aligned}$$

Then, let $\widetilde{A} := U^*AU$, so that we have,

$$\widetilde{A} = U^*YV\Sigma^{-1}. \tag{3.18}$$

Since the matrix U contains the proper orthogonal modes of X, $\widetilde{A}$ is a projection of the linear operator A onto the POD basis functions U. One feature obtained with this variation is that we can now restrict the projection basis U, similar to the POD method.

The modal structures are then to be extracted from the matrix $\widetilde{A}$,

$$\phi_i = U\xi_j, \tag{3.19}$$

where ξ_j is the jth eigenvector of $\widetilde{A}$, i.e., $\widetilde{A}\xi_j = \lambda_j\xi_j$, and U is the unitary matrix of the right singular vectors of the snapshot sequence X.

Algorithm 2 summarizes the method.

In [13] the DMD mode obtained from (3.22) in Algorithm 2 is referred to as *projected DMD modes*. The origin for this nomeclature is discussed in Section 3.3.3.

3.3.2 Exact DMD

A variation of this algorithm, called *Exact DMD* in the formulation presented in [13], proposed a more general definition of data, while emphasizing on its linear consistency property (see Section 3.3.3). Matrices X and Y are defined in (3.1).

Algorithm 2 (SVD based DMD)

1: INPUT: Matrices X and Y as defined in (3.1), where $\boldsymbol{y}_k = \boldsymbol{x}_{k+1}$, $k = 0, \ldots, m-1$;

2: Compute the reduced or truncated SVD of X,

$$X = U\Sigma V^* \tag{3.20}$$

3: Define the matrix

$$\widetilde{A} = U^* Y V \Sigma^{-1} \tag{3.21}$$

4: Compute eigenvalues and eigenvectors that satisfy $\widetilde{A}\xi_j = \Lambda\xi_j$, where $\Lambda = \mathrm{diag}(\lambda_j)$;

5: OUTPUT:

- DMD modes,

$$\phi_j = U\xi_j, \tag{3.22}$$

- Eigenvalues λ_j.

For the data-set given by (3.1), we define the operator

$$A = YX^+, \tag{3.23}$$

where X^+ is the pseudoinverse of X. The DMD of the pair (X, Y) is given by the eigendecomposition of A, *i.e.*, the DMD modes and eigenvalues are the eigenvectors and eigenvalues of A.

Algorithm 3 (Exact DMD)

1: INPUT: Rearrange the data $\{(\boldsymbol{x}_0, \boldsymbol{y}_0), \ldots, (\boldsymbol{x}_{m-1}, \boldsymbol{y}_{m-1})\}$ into the matrices X and Y, as in (3.1),

2: Compute the reduced or truncated SVD of X (3.20);

3: Define the matrix

$$\widetilde{A} = U^* Y V \Sigma^{-1} \tag{3.24}$$

4: Compute eigenvalues and eigenvectors which satisfy $\widetilde{A}\xi_j = \lambda_j\xi_j$, and define $W = (\xi_1, \ldots, \xi_m)$;

5: OUTPUT:

- DMD mode given by

$$\Phi = YV\Sigma^{-1}W, \tag{3.25}$$

  where $\Phi = (\phi_1, \ldots, \phi_m)$;

- Eigenvalues λ_j.

Theorem 3.3.1, proven in [13], shows that Algorithm 3 identifies the eigenpairs of matrix A, therefore the denomination for the modes in (3.25) as *Exact DMD*.

Theorem 3.3.1. *Each pair (ϕ, λ) generated by expression (3.25) from Algorithm 3, is an eigenvalue/eigenvector pair of A. Furthermore, the algorithm identifies all of the non-zero*

eigenvalues of A.

3.3.3 Exact and Projected DMD

Let us first define the linear consistency of matrices. Two $n \times m$ matrices X and Y are linearly consistent if, whenever $Xc = 0$, then $Yc = 0$. That is, X and Y are linearly consistent if and only if the nullspace of Y, which we denote by $\mathcal{N}(Y)$ contains the nullspace of X, or equivalently, $\mathcal{N}(X) \subset \mathcal{N}(Y)$. Theorem 3.3.2, proven in [13], follows:

Theorem 3.3.2. *Define $A = YX^{+}$. Then $Y = AX$ if and only if X and Y are linearly consistent.*

Note that algorithms 2 and 3 are nearly similar, the difference being on the terms used in (3.25) and (3.22). Theorem 3.3.3 proven in [13] addresses this difference.

Theorem 3.3.3. *Let $\widetilde{A}\xi = \lambda\xi$, with $\lambda \neq 0$, and let P_X denote the orthogonal projection onto the image of X. Then, $\widehat{\phi} := U\xi$ is an eigenvector of $P_X A$ with eigenvalue λ. Furthermore, if ϕ is given by (3.25), then $\widehat{\phi} = P_X\phi$.*

From that it follows that the modes determined from (3.22) by the Algorithm 2, which we now refer as $\widehat{\phi}$, are the projection of the modes determined from (3.25) by Algorithm 3 onto the range of X, therefore the reference to (3.22) as *Projected DMD*. Denoting the orthogonal projection onto the range of X as P_X, if vectors $\boldsymbol{y}_k$ lie in the span of the vectors $\boldsymbol{x}_k$, then $P_X A = A$, and the projected modes (3.22) and exact DMD modes (3.25) are identical.

Although both solutions (3.22) and (3.25) converge if X and Y have the same column spaces, in all the examples and applications presented in the thesis we implemented Algorithm 3, given its more general formulation.

3.4 Standing Waves and Time Delay Embedding

In this section, we show that one of the shortcomings of this method manifests when the data is generated by a standing wave. This can be solved using a technique based on the methods developed in [24]. For this section, we closely follow [13].

Consider the data set $Z = (\boldsymbol{z}_0, \ldots, \boldsymbol{z}_q)$ generated by a standing wave defined by

$$\boldsymbol{z}_k = \cos(k\theta)\boldsymbol{q}, \quad \boldsymbol{z}_k \in \mathbb{R}^n \text{and} \quad k = 0, \ldots, m, \tag{3.26}$$

$\boldsymbol{q} \in \mathbb{R}^n$ is a constant vector, and θ denotes the frequency of oscillation.

DMD applied to data-set Z

To determine the DMD modes and eigenvalues from the data we apply the DMD to matrices

$$X = \boldsymbol{q}\boldsymbol{x}^T, \quad Y = \boldsymbol{q}\boldsymbol{y}^T$$

where the components of $\boldsymbol{x}$ and $\boldsymbol{y}$ are $x_k = \cos(k\theta)$ and $y_k = \cos\big((k+1)\theta\big)$, respectively.

The unitary matrix U obtained from the application of the SVD on matrix X is a single column, corresponding to a singular value matrix Σ of dimension 1×1, given that $\boldsymbol{q}$ is a constant vector and $rank(X) = 1$. The matrix $\widetilde{A}$, as a consequence, is a matrix of dimension 1×1. The output of the DMD will, therefore, be precisely one real-valued DMD eigenvalue.

The eigenvalue in the continuous space obtained from (3.4), is of the form $\omega = \alpha + i\beta$, where α is the real part and β the imaginary part. Given that $\beta = 0$, since the eigenvalue is real, then $e^{\omega t} = e^{\alpha t}\left[\cos(\beta) + i\sin(\beta)\right] = e^{\alpha t}$, from which we see that the oscillatory nature of the data in (3.26) is not captured by the method.

Linear consistency of the input data

The failure of this method can be underlined by Theorem 3.3.2, when we look at the linear consistency of data. X and Y are not linearly consistent, unless $\theta = n\pi$, where $n \in \mathbb{N}_0^+$.

This can be seen from the definition of linearly consistency ($\mathcal{N}(X) = \mathcal{N}(Y)$). If we consider a vector $\boldsymbol{a} = (-\cos\theta, 1, 0, \ldots, 0)^T$, then $\boldsymbol{a} \in \mathcal{N}(X)$, since $X\boldsymbol{a} = 0$. However, $Y\boldsymbol{a} \neq 0$, unless $\theta = n\pi$, which comes from $Y\boldsymbol{a} = \boldsymbol{q}(-\cos^2(\theta) + \cos(2\theta)) = \boldsymbol{q}\sin^2\theta$. Therefore, from theorem 3.3.2, There is no A such that $Y = AX$.

However, by appending to Z a time-shifted value of itself, that is, $\widetilde{\boldsymbol{z}}_k = (\boldsymbol{z}_k, \boldsymbol{z}_{k+1})^T$, results in that the new updated matrices X and Y becoming linearly consistent, thus enabling to identify the correct dynamics, which we will verify next.

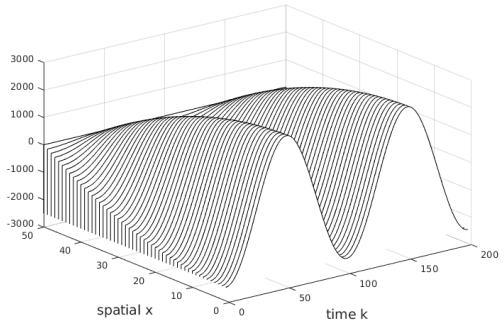
DMD to modified data-set

We illustrate these results with Matlab©, computing the DMD algorithm for the data in (3.26), where $\boldsymbol{q} \in \mathbb{R}^{50}$. For this simple example, we considered $\boldsymbol{q} = -50^2 + \boldsymbol{x}^2, \boldsymbol{x} \in [0, 50]$.

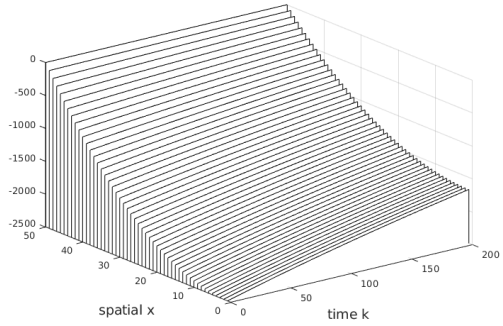
Figure 3.1 resumes the results. Notice in Figure 3.1(d) the location of the estimated DMD eigenvalues. When we add the time-delayed values, the data matrix is now of rank 2, and the determined eigenvalues are complex conjugate, thus denoting a oscillatory behavior.

The linear consistency of the data in this example is easy to compute in Matlab©. By determining the null space using the command **null** on the constructed matrix X, and knowing that, by definition $\mathcal{N}(X) \subset \mathcal{N}(Y)$, then $Y\mathcal{N}(X) \leq \epsilon$, where ϵ is a value that we may consider negligible (for example, $\epsilon_{machine}$).

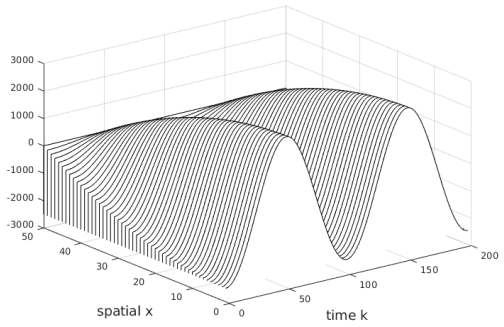
When we use the data directly from (3.26), $\|Y\mathcal{N}(X)\| \approx \mathcal{O}(10^3)$, whereas, by adding the time-delayed values, we obtain $\|Y\mathcal{N}(X)\| \approx \mathcal{O}(10^{-10})$. Thus confirming that by adding the time-delayed terms the values obtained are negligible, hence the linear consistency of matrices X and Y.



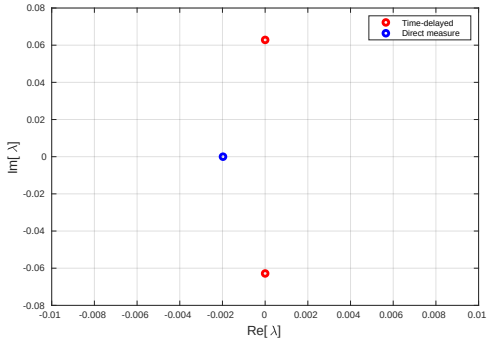
(a) Exact Solution



(b) DMD



(c) DMD with time delay



(d) Eigenvalues

Figure 3.1: *In (d) the location of the continuous time eigenvalues, where we see, marked in red, the two complex conjugate eigenvalues obtained by appending to* $\boldsymbol{z}$ *the time-shifted value of itself. This resulted in the standing wave DMD approximation in (c). In (b) is the approximated solution obtained from the DMD without the time-shifted augmented data, and in (c) the exact solution for the Burgers equation.*

Conclusions

The time delay strategy we used in this example is a particularization of the idea behind Takens theorem, which extended results can be read in [24]. The main idea is that we can extend the matrix of snapshots with the time-delayed observations, and by doing this, we may be able to recover dynamics which are not directly measured but "embedded" in the measured states, hence the name *Time Delay Embedding* used in this theorem. More on this method is also in [13]. The difficulty of this strategy is that we are enlarging the data matrix dimension, thus increasing the algorithms computational cost. We will often apply the results of Takens theorem on examples and applications throughout the thesis. The theory, however, given the limited extent of time permitted for this thesis, can be found on the already mentioned literature.

3.5 Examples of Simple Applications

In this section, to illustrate the methods applications and to evidence the connections to the Koopman operator, we revisit the examples used in Chapter 2.

We define the relative error ε_j as

$$\varepsilon_j = \frac{\|\tilde{\boldsymbol{y}}_j - \boldsymbol{y}_j\|_2}{\|\boldsymbol{y}_j\|_2}, \tag{3.27}$$

where, $\tilde{\boldsymbol{x}}_j$ is the state vector estimated by DMD at time j, and $\boldsymbol{x}_j$ the reference state vector. To measure the performance of the method, we use the averaged relative error, given by

$$\overline{\varepsilon} = \frac{1}{m}\sum_{j=1}^{m} \varepsilon_j, \tag{3.28}$$

where m is the number of time samples.

For the dimension reduction of step 2 in Algorithm 3, we used the same expression as in chapter 1 with (1.9), which we recall here, as

$$I(r) = \frac{\sum_{i=1}^{r} \sigma_i^2}{\sum_{j=1}^{l} \sigma_j^2},$$

As a threshold criteria, by default, we are using $I(r) \geq 0.99$, if not mentioned otherwise.

All Matlab©code needed for the computation of the problems presented in these examples are included in Appendix D.

3.5.1 Example 1 - Nonlinear ODE

The system (2.14), which we recall here, was defined as

$$\begin{cases} \dot{x} - \mu x = 0 \\ \ddot{y} - \lambda(\dot{y} - \dot{x}^2) = 0. \end{cases}$$

In this example we use the discrete values generated from Matlab© with the **ode45** function to construct a snapshot matrix.

DMD with $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{x}$

In the first implementation, the observables are the set of linear measurements of the states, *i.e.* $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{x}$.

The DMD generates the modes and eigenvalues with which we can construct the linear model (3.3), and the obtained solution is seen in Figure (3.2(a)). We can observe that the DMD produces an approximation to the exact solution, and the states trajectory correctly converges on the attractor at the origin. However, the approximation is not accurate.

DMD with $\boldsymbol{g}(\boldsymbol{x}) = (x_1, x_2, x_1^2)^T$

For the second test, we used the set of observables as derived in Chapter 2 for this same example, that is, $\boldsymbol{g}(\boldsymbol{x}) = (x_1, x_2, x_1^2)^T$. The linear model approximation obtained from the determined DMD modes and eigenvalues generated the solution as seen in Figure (3.2(b)). This is a better approximation to the exact solution, as expected, since the DMD method determines the eigenvalues and eigenvectors of a linear operator A as discussed in section 3.2. The mean relative error obtained is $\bar{\varepsilon} = \mathcal{O}(10^{-8})$.

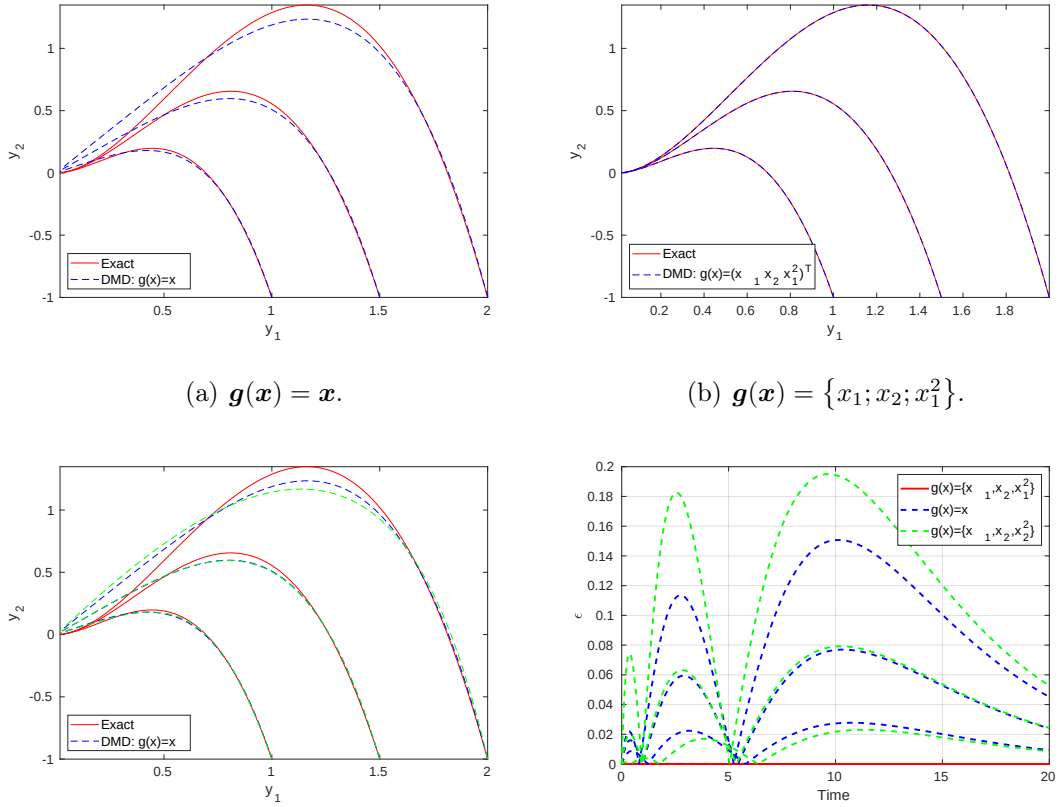
DMD with $\boldsymbol{g}(\boldsymbol{x}) = (x_1, x_2, x_2^2)^T$

If we select the wrong state as an observable, however, the relative error ε_j shows higher values than with $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{x}$, as seen in Figure 3.2(c).

Figure 3.2(d) resumes the evolution of the relative error ε_j for each of these different settings.

Conclusions

From this very simple example it is possible to illustrate some important characteristics discussed in the last two chapters:



(a) $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{x}$.

(b) $\boldsymbol{g}(\boldsymbol{x}) = \{x_1; x_2; x_1^2\}$.

(c) $\boldsymbol{g}(\boldsymbol{x}) = \{x_1; x_2; x_2^2\}$.

(d) Evolution of error ε_j

Figure 3.2: *In (d) the evolution of the relative error ϵ for the three possible choices of observables, in which is evident that the wrong choice has worse results (in green) than the the linear measurement of the states (in blue). (a), (b) and (c) maps the trajectories of the estimated states, when compared with the exact solution.*

- the right choice of observables is crucial to approximate the Koopman operator;
- the wrong choice of observables can in fact produce larger errors than only using linear measures of the states;
- if the dynamical system is linear, as the one obtained in (2.17), the DMD algorithm approximates the exact solution with $\overline{\varepsilon} \to 0$.

3.5.2 Example 2 - Logistic Map

As discussed in section 2.3.2, the infinite-dimensionality of the Koopman operator $\boldsymbol{K}$ made the calculation of the solution to the system unattainable. Truncating $\boldsymbol{K}$ did not produce satisfactory results, either. In this section we now use the truncation of the infinite set of

the observables previously determined, apply the DMD and verify the results.

The solutions shown were obtained using the data set $\left\{x_k^{(1)}\right\}_{k=0}^{q}$ which corresponds to the discrete values generated from Matlab©by computing (2.19) with parameter $\mu = 3$, and the data-set $\left\{x_k^{(2)}\right\}_{k=0}^{q}$ which corresponds to $\mu = 3.8$. The initial condition on both cases was defined as $x_0 = 0.5$

Define the set of Observables

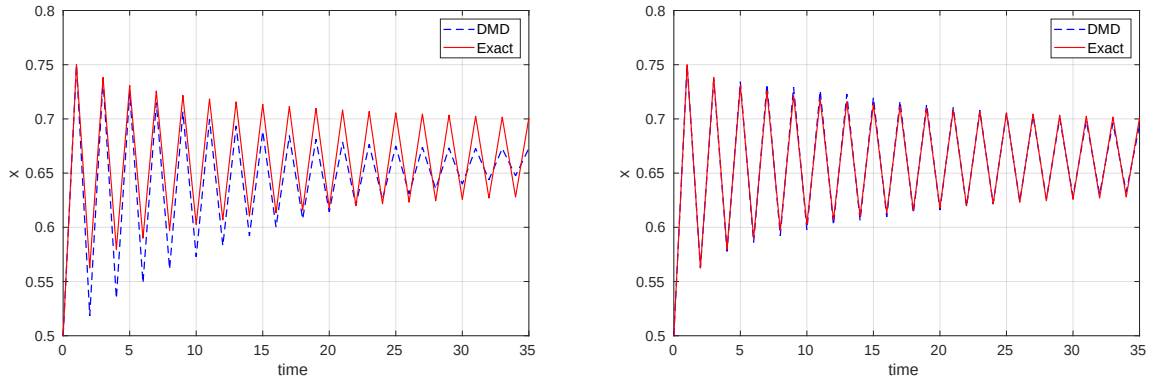
As in the previous example, we test the algorithm using different sets of observables. The selected set of observables are the ones determined in section 2.3.2, that is,

$$\boldsymbol{g}(x_k) = (x_k, x_k^2, \ldots, x_k^p)^T, \tag{3.29}$$

where, p is the order of the polynomial.

Results

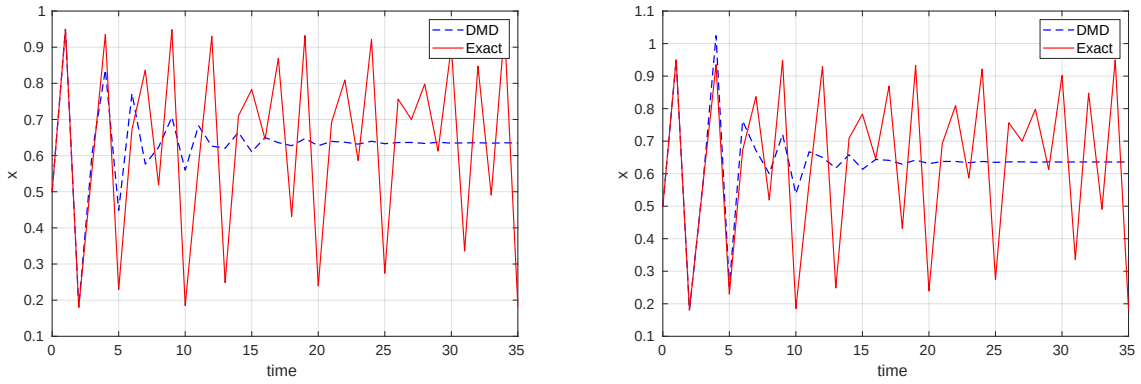
In Figure 3.3 the solutions obtained by DMD with different p using data-set $\boldsymbol{x}^{(1)}$. As the order p increases, the solution obtained from the linear model approximates the exact solution.



(a) $p = 2$ (b) $p = 4$

Figure 3.3: *Evolution of the exact solution of the logistic equation with parameters* $\mu = 3, x_0 = 0.5$*, marked in red, compared with the obtained from the linear model determined by the DMD marked in blue. In (a) the solution obtained using the data matrix augmented with* $p = 2$ *order of polynomials of the state* x*. In (b) the solution using the data matrix with the polynomials up to order* $p = 4$*.*

In Figure 3.4, the solutions obtained from DMD, with different p using data-set $\boldsymbol{x}^2$. The linear model fails to capture the exact solution.



(a) p=7 (b) p=11

Figure 3.4: *Evolution of the exact solution of the logistic equation with parameters* $\mu = 3.8, x_0 = 0.5$ *marked in red, compared with the obtained from the linear model estimated by the DMD marked in blue.*

Figure 3.5 plots the evolution of the error $\bar{\varepsilon}$ by curve defined with the parameter μ as the polynomial order p increases. Note the convergence of the algorithm with p in the tested conditions and the low accuracy in the case where $\mu = 3.8$.

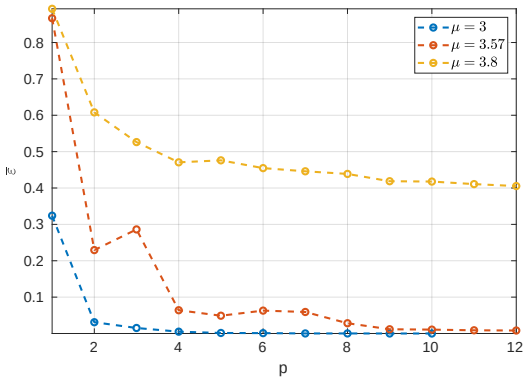


Figure 3.5: *Comparison of the evolution of the relative error* ϵ *with the order* p *used in the data matrices. For* $\mu = 3.8$ *the relative error does not converge to zero.*

Conclusions

In this example, we used the results from section 2.3.2, where the Koopman theory framework enabled us to find a infinite set of observables with a polynomial structure.

When the behavior of the exact solution exhibited a stable convergence, as with the set $\boldsymbol{x}^{(1)}$, the DMD was able to determine a linear model that could approximate it fairly accurately. However, when the system does not converge to an attractor, or a limit cycle, as with the set $\boldsymbol{x}^{(2)}$, DMD fails to find a fitting linear model. This can be simple to intuit, since a finite-dimensional linear space does not admit multiple fixed-points or attracting structures. Extended results on this can be found in [3, 17].

3.5.3 Example 3 - Van der Pol

The Van der Pol equation, transformed into a first order ODE, as formulated in (2.25), is given as

$$\begin{cases} \dot{y_1} &= & y_2 \\ \dot{y_2} &= & \mu(1-y_1^2)y_2 - y_1. \end{cases}$$

The nonlinearity is driven by the parameter μ. In this example, $\mu = 1.2$, except when indicated otherwise.

Observables based on polynomials of order p

The numerical solution we used to obtain the data-set for computing the DMD was generated with the **ode45** command in Matlab©(code in Appendix D). This was also used as a reference to compute the mean relative error $\bar{\epsilon}$.

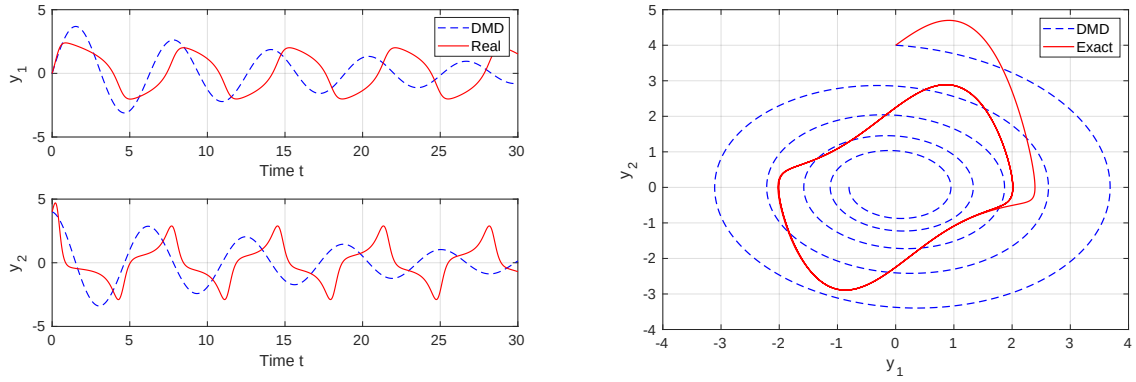
As in the previous example, the vector of observables (2.26) also denotes an increasing order polynomial structure, except we now have a 2-dimensional system with variables y_1 and y_2.

As previously, we use a truncation of the vector of observables defined as

$$P_p(\boldsymbol{y}) = (y_1, y_2, y_1y_2, y_1^2, y_2^2, y_1^2y_2, \ldots, y_1^p, y_2^p)^T.$$

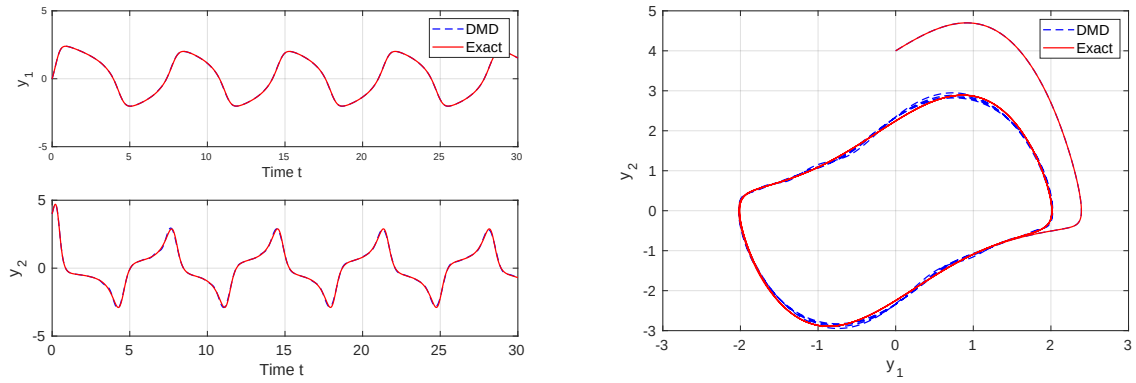
In Figures 3.6 and 3.7, we plot the solutions obtained from the linear model with $p = 1$ and $p = 12$, respectively.

As the order p of polynomials used to construct the observable $\boldsymbol{g}(\boldsymbol{x})$ increases, the approximation to the exact solution improves until $p = 12$. As we increase p further, however, the condition number of the matrix Σ, determined in step 2 of the DMD, also increases, and consequently the determination of ϕ by (3.25) becomes a ill-conditioned problem. Figure 3.8 with the relative error $\bar{\epsilon}$ reflects the above.



(a) (b)

Figure 3.6: *Comparison of the exact solution of the Van der Pol equations with parameter* $\mu = 1.2$ *and initial condition* $\boldsymbol{y}_0 = (0, 4)$*, in red, and the obtained from the DMD using as the data matrix* $\boldsymbol{g}(\boldsymbol{y}) = \boldsymbol{y}$*, in blue. In (a) the evolution in time of states* y_1 *and* y_2*, and in (b) the state space plot.*



(a) (b)

Figure 3.7: *Comparison of the exact solution of the Van der Pol equations with parameter* $\mu = 1.2$ *and initial condition* $bmy_0 = (0, 4)$*, in red, and the obtained from the DMD using as the data matrix* $\boldsymbol{g}(\boldsymbol{y}) = P_{12}(\boldsymbol{y})$*, in blue. In (a) the evolution in time of states* y_1 *and* y_2*, and in (b) the state space plot.*

Observables based on time-delay embedding

As a different approach, the observable matrix is now augmented with time-shifted measurement copies of the snapshot states, based on the embedded time delay technique discussed in section 3.4.

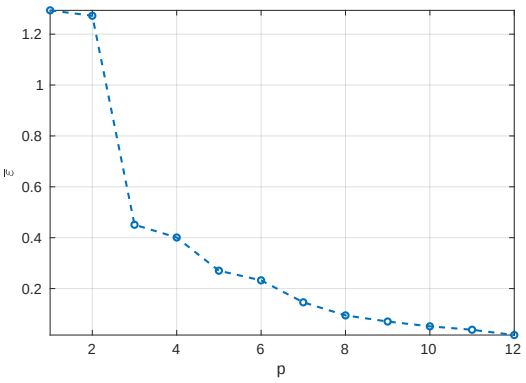


Figure 3.8: *Mean relative error evolution $\bar{\epsilon}$ with p. As the order p of the polynomials used for the data matrix increases, the mean relative error converges.*

The observable matrix is now constructed as

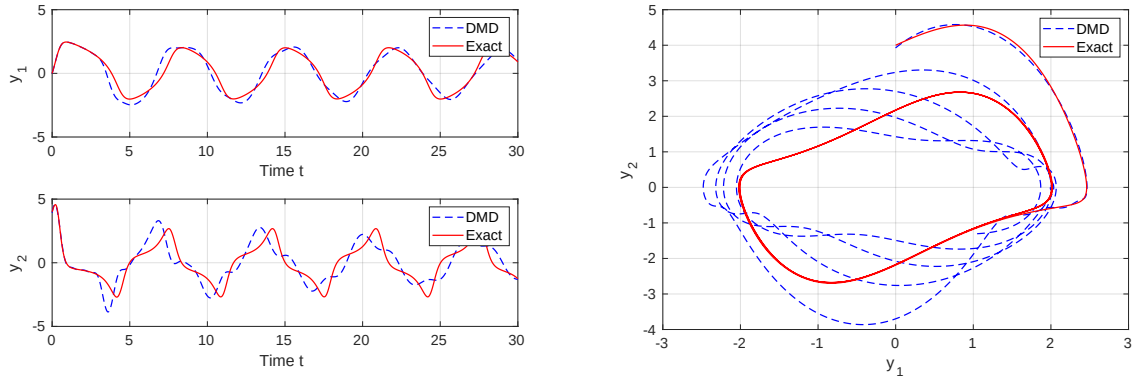
$$\boldsymbol{g}(\boldsymbol{y}) = \begin{pmatrix} \boldsymbol{y}_0 & \boldsymbol{y}_1 & \cdots & \boldsymbol{y}_{m-d} \\ \boldsymbol{y}_1 & \boldsymbol{y}_2 & \cdots & \boldsymbol{y}_{m-1} \\ \vdots & \vdots & \cdots & \vdots \\ \boldsymbol{y}_d & \boldsymbol{y}_{d+1} & \cdots & \boldsymbol{y}_m \end{pmatrix}. \tag{3.30}$$

where the parameter d determines the number of time-shifted copies of the measurements $\boldsymbol{y}$ to stack on the observables matrix (3.30).

As the number of time-shifted stacked copies d increases, the solution obtained from the determined linear model by the DMD approximates the reference solution. This can be observed in Figures 3.9, and 3.10. The approximation to the exact curves depicting the temporal evolution of the states y_1 and y_2 improves with d (Figures 3.9(a) and 3.10(a)). Similarly, the phase space on Figures 3.9(b) and 3.10(b) show how the states trajectories increasingly adjust to the shape of the limit cycle in the exact solution.

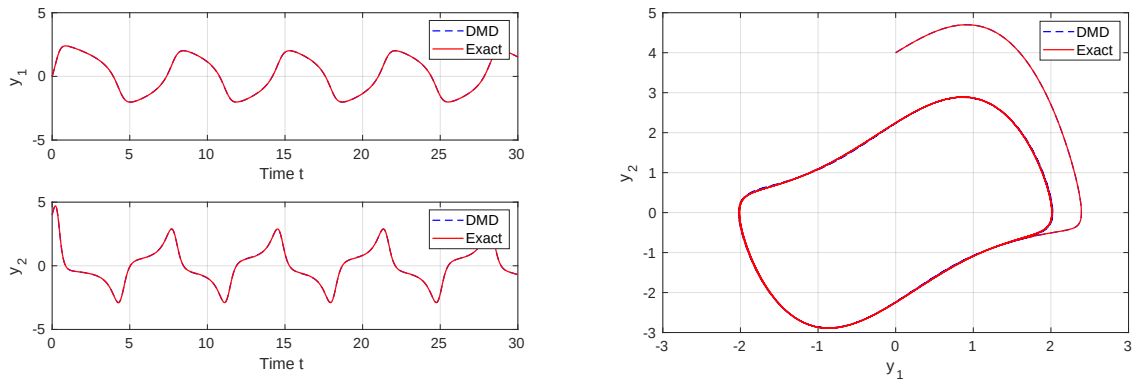
Initial conditions

We verified that the initial condition had an effect on the value of d necessary time-shifted measurements until $\bar{\varepsilon}$ converged. Figure 3.11, plotting $\bar{\varepsilon}$ against d, for each initial condition illustrates that the furthest the initial condition $\boldsymbol{y}_0$ is from the limit cycle, the higher is the number of d of time-shifted measurements required.



(a) (b)

Figure 3.9: *Comparison of the exact solution of the Van der Pol equations with parameter* $\mu = 1.2$ *and initial condition* $\boldsymbol{y}_0 = (0, 4)$*, in red, and the obtained from the DMD using as the data matrix* $d = 200$ *time-shifted vectors, in blue. In (a) the time evolution of the states* y_1 *and* y_2*, and in (b) the state space plot.*



(a) (b)

Figure 3.10: *Comparison of the exact solution of the Van der Pol equations in red, and the obtained from the DMD now using* $d = 400$ *time delayed vectors in blue. In (a) the time evolution of the states* y_1 *and* y_2*, and in (b) the state space plot.*

Nonlinearity

Since the parameter μ gives us a measure of how strongly the Van der Pol system is nonlinear, we changed this parameter and measured the relative error $\bar{\varepsilon}$ for each value of d. In Figure 3.14 we can see the results and verify that the more strongly nonlinear the system is, the higher is the number d of time-shifted measurements are required until the linear model fits the measured data with acceptable accuracy.

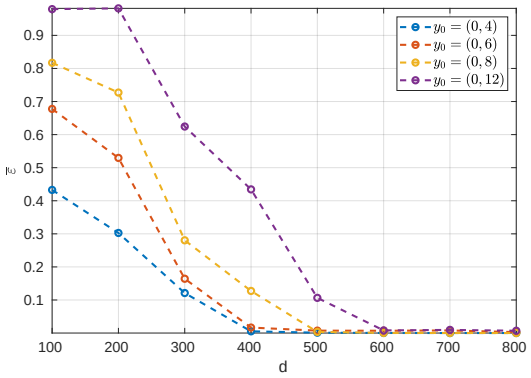
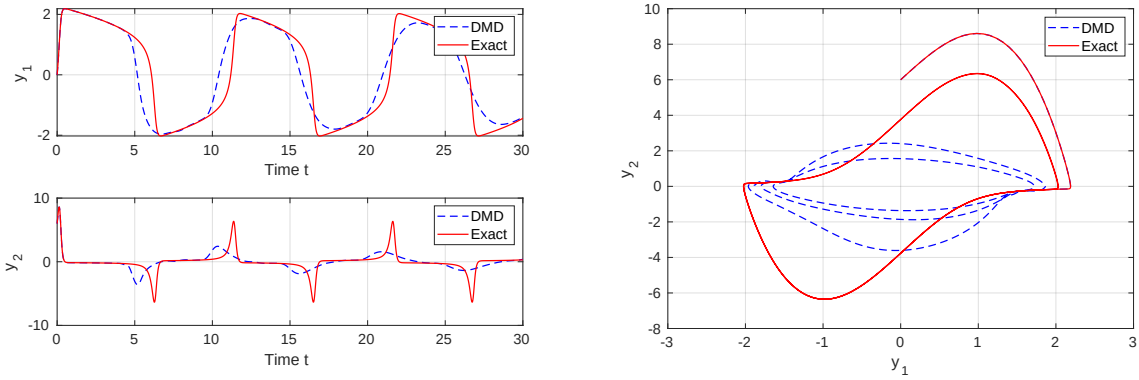


Figure 3.11: *Comparison of mean relative error $\bar{\epsilon}$ by initial condition $\boldsymbol{y}_0$. The furthest is it from the limit cycle, the mode time delays d are necessary until $\bar{\epsilon}$ converges.*

Figures 3.12 and 3.13 show the solution estimated from the DMD compared with the exact solution, considering $\mu = 4$ and initial condition $\boldsymbol{y}_0 = (0, 6)$



(a) (b)

Figure 3.12: *Comparison of the exact solution of the Van der Pol equations with parameter $\mu = 4$ and initial condition $\boldsymbol{y}_0 = (0, 6)$ in red, and the obtained from the DMD using $d = 400$, in blue. In (a) the time evolution of the states y_1 and y_2, and in (b) the state space plot.*

3.5.4 Example 4 - Burgers' equation

In this example we approximate the solution of a PDE, the Burgers' equation $u_t + uu_x - \epsilon u_{xx} = 0$, using the DMD with the discrete data generated from a numerical simulation of this system. To obtain this data we applied the Fast Fourier Transform method (FFT) using Matlab©. The solution obtained by FFT is our reference when computing the mean relative error $\bar{\epsilon}$.

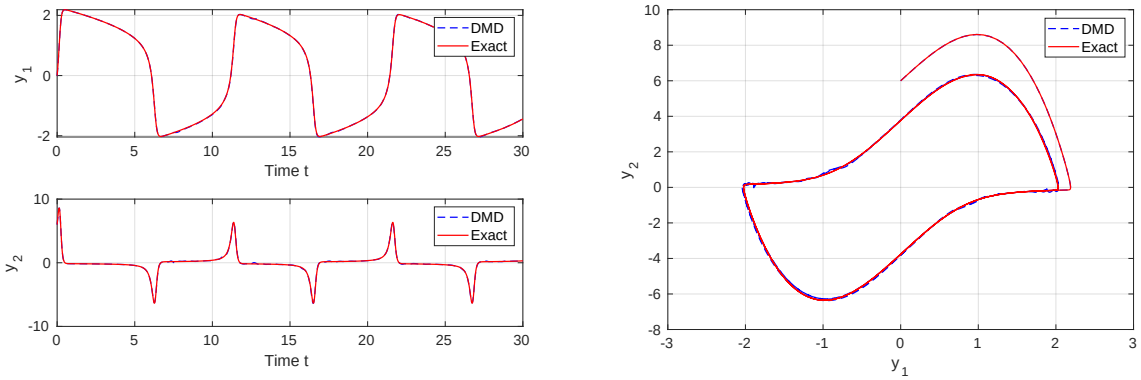


Figure 3.13: *Comparison of the exact solution of the Van der Pol equations with parameter* $\mu = 1.2$ *and initial condition* $\boldsymbol{y}_0 = (0, 4)$ *in red, and the obtained from the DMD using* $d = 600$*, in blue. In (a) the time evolution of the states* y_1 *and* y_2*, and in (b) the state space plot.*

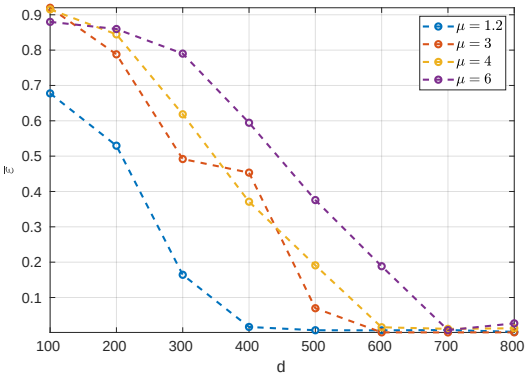
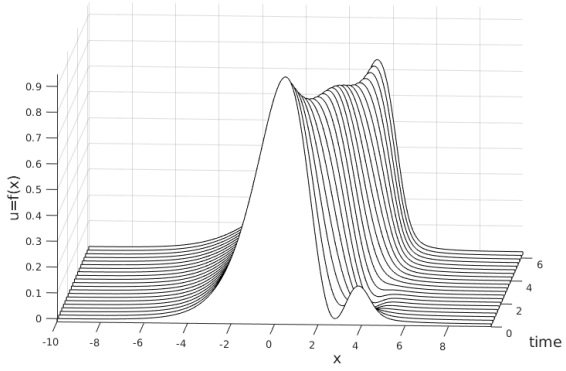


Figure 3.14: *Comparison of the relative error* $\bar{\varepsilon}$ *with parameter* μ*. The higher is the value of* μ*, the higher needs* d *to be, before* $\bar{\epsilon}$ *converges.*

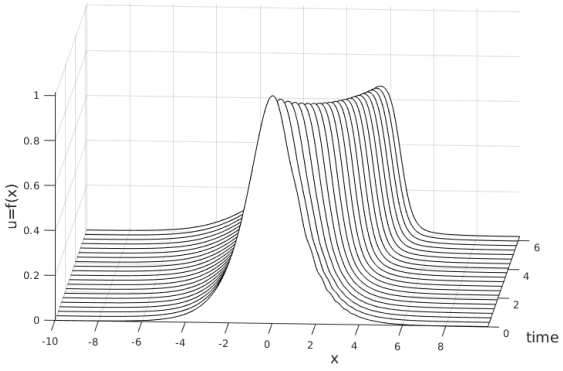
We now consider the set of observables as the linear measure of the states u, so that $\boldsymbol{g}(\boldsymbol{u}) = \boldsymbol{u}$, and evaluate the results obtained from the DMD by changing the truncation criteria at the step 2 of Algorithm 3 corresponding to the SVD of matrix X.

The output of the DMD then provides r eigenvalues and the corresponding DMD modes, where r corresponds to our truncation criteria. Figures 3.15(a) and 3.15(b) plot the solutions obtained from the linear model estimated by the DMD with $r = 3$ and $r = 7$, respectively. For comparison, Figure 3.15(c) shows the numerical solution obtained by FFT.

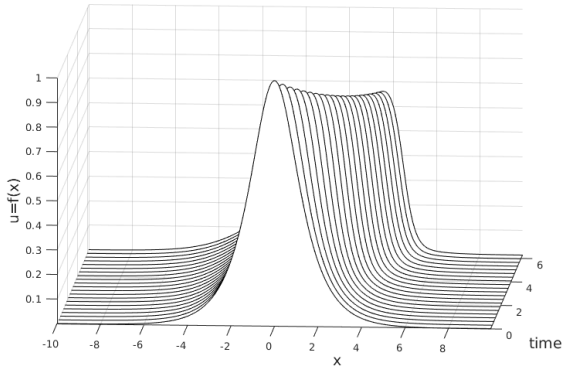
In Figure 3.16(a) we plot the evolution of the relative error $\bar{\epsilon}$ with the truncation criteria r. and Figure 3.16(b) plots the normalized weight of each singular value determined by



(a) DMD with 3 modes



(b) DMD with 7 modes



(c) Exact

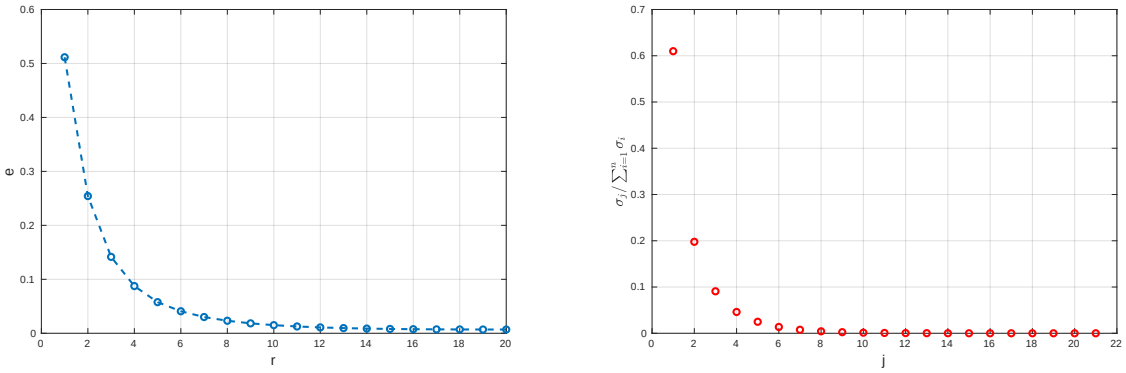
Figure 3.15: *In (a) the solution of the linear model estimated by the DMD with $r = 3$, while in (b) $r = 7$, and in (c) the exact solution for Burgers equation.*

$\sigma_j / \sum_{i=1}^{l} \sigma_i$. We see that, for the Burgers equation, as the weight of the singular values sharply decreases, the relative error $\bar{\epsilon}$ follows the same trend.

3.6 Conclusions

From the short examples presented, we have seen that the proper choice of a set of observables is crucial for the success of the method. In order to obtain reasonably accurate results, the construction of the observables was based in techniques such as the embedded time delay for example 3, or by using the measure of the states in polynomial form, tested in every case but example 4. However, in systems such as the Burgers equation, the direct measure of the states delivered sufficiently accurate results.

The concepts discussed in the previous Chapters 1 and 2, on the POD method and the



(a) Evolution of error with r

(b) Singular Values

Figure 3.16: *In (a) the evolution of the relative error with the increase of rank r, and in (b), the normalized Singular Values from the SVD of the data matrix consisting of the state measurements $\boldsymbol{u}$ obtained from the exact solution of the Burgers equation, with $\boldsymbol{u}_0 = sech(\boldsymbol{x})$.*

Koopman operator, respectively, are related to the DMD in different but complementary ways. The DMD incorporates the concept of spatial dimensionality-reduction technique of the POD, and the connection with the Koopmans operator theory, which provides the method with a framework which gives us the confidence to use it as useful method to characterize non-linear dynamics.

Chapter 4

Application - Case Studies

In this chapter we use the methods and techniques discussed earlier on two particular applications.

In the first application, presented in Section 4.1, the numerical data set is generated by numerical simulations by the Bergen Ocean Model (see [1]). This data-set consist of 2D velocity fields at the sea floor. The goal is to describe the data using the DMD modes and eigenvalues within acceptable accuracy. We test different methods for choosing observables, compare the outcomes and discuss the results.

The second application, at Section 4.2, we apply the DMD on the data generated by two-population neural field model with added noise. We consider two examples of pattern formation (for details see [28]). The goal here is to test if the method is capable of recovering the stable patterns from a noisy data.

To evaluate the accuracy of the results, we use the relative error ε_j and the mean relative error $\bar{\varepsilon}$ as defined in (3.27) and (3.28), respectively.

4.1 2D Velocity Field Data

In the discretized area covered by a $n \times m$ spatial grid, we represent the state vector of the velocity field by $\boldsymbol{y}_k \in \mathbb{R}^{n \cdot m}$, where the subscript $k = 0, \ldots, q$ represents the discrete time. At each grid point $l = 1, \ldots, n, \ldots, n \cdot m$, $\boldsymbol{x}_l = (u_l, v_l)^T \in \mathbb{R}^2$ is a vector where $u_l \in \mathbb{R}$ corresponds to the horizontal component of the velocity, and $v_l \in \mathbb{R}$ is the vertical component in an euclidean coordinate system. The velocity field data, at each time k is then given by $\boldsymbol{y}_k = (\boldsymbol{x}_{k,1}, \ldots, \boldsymbol{x}_{k,n \cdot m})^T$. The data-set is, hence, ordered as the matrix

$\mathcal{X} = (\boldsymbol{y}_0, \ldots, \boldsymbol{y}_q)$, which expanded with its components gives

$$\mathcal{X} = \begin{pmatrix} u_{0,1} & u_{1,1} & \cdots & u_{q,1} \\ v_{0,1} & v_{1,1} & \cdots & v_{q,1} \\ u_{0,2} & u_{1,2} & \cdots & u_{q,2} \\ \vdots & \vdots & \vdots & \vdots \\ v_{0,n\cdot m} & \cdots & \cdots & v_{q,n\cdot m} \end{pmatrix}.$$

In section 4.1.1, we apply the DMD to the data-set $\mathcal{X}_1$, which consists of values of the velocity field obtained from a $32[Km] \times 32[Km]$ area with $800[m]$ resolution corresponding to a 40×40 grid over a time interval of $\tau = \big[0, 1500\big][h]$ with a $\Delta\tau = 0.7143[h]$.

Here, we are mostly interested in the gains in accuracy we can obtain as we test different techniques for selecting the observables for approximating the Koopman operator.

In Section 4.1.2 the DMD is applied to the data-set $\mathcal{X}_2$ which consists of values of the velocity field obtained from a $40[Km] \times 41.6[Km]$ area covered by a 53×51 spatial grid over a time interval of $\tau = \big[0, 1500\big][h]$, sampled at $\Delta\tau = 0.01429[h]$. The choice of technique to construct the matrix of observables to use in the DMD is based on the results of Section 4.1.1.

4.1.1 Choice of Observables

To minimize the effects of external forcing terms, such as wind or other time-dependent terms that may change the equilibrium state of the dynamical system, the data in $\mathcal{X}_1$ is a subset of $\mathcal{X}$ with a shorter time interval, in this case $48[h]$, resulting in $q = 68$ snapshots, so that those forcing terms can be considered as invariant.

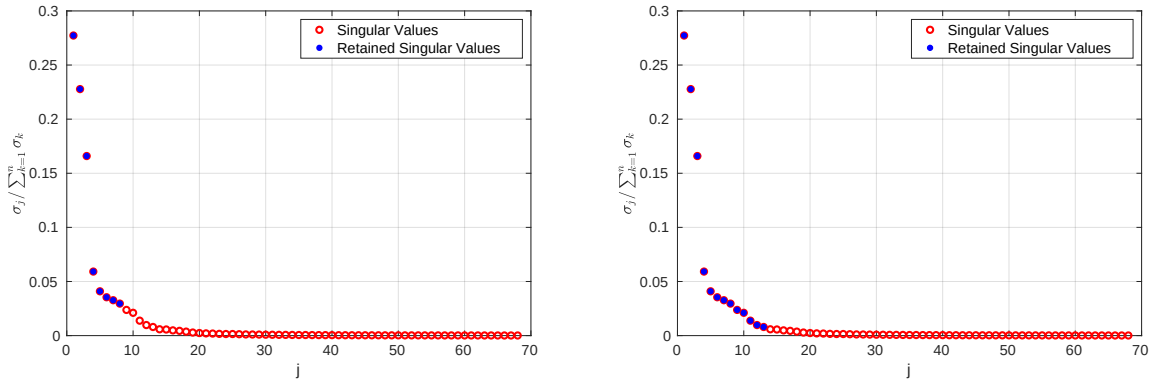
The selection of the right set of observables is important to best approximate the Koopman operator. For that purpose, we tested three strategies as summarized next:

- Using the direct measure of the states as the observables, *i.e.*, $\boldsymbol{g}(\boldsymbol{y}) = \boldsymbol{y}$;
- Augmenting the data matrix with n-th order polynomial measurements of the states, as defined in (3.29), *i.e.*, $\boldsymbol{g}(\boldsymbol{y}) = P_n(\boldsymbol{y})$;
- Augmenting the data matrix using the embedded time delay technique as in (3.30).

Linear measurements of the state

We first construct the snapshot matrix as in (1.7) from the data $\mathcal{X}_1$. As the observables are exactly the measurement of the states $\boldsymbol{x}$, the resulting matrix of snapshots is of dimension 3362×68.

Step 2 of Algorithm 3 is where we can reduce the dimensionality of our problem. As in Section 3.5, the reduced rank which we denote by r, is chosen so that the expression of energy (1.9) holds. We test the results with the energy threshold set to $I(r) < 0.99$ and $I(r) < 0.999$. Figure 4.1 plots the normalized singular values obtained in step 2 of Algorithm 3. The singular values that are retained are marked in blue.



(a) $I(r) < 0.99$ (b) $I(r) < 0.999$

Figure 4.1: *Singular Values distribution normalized by* $\sigma_j / \sum_{k=1}^{n} \sigma_k$. *In (a) marked in blue are the* r *retained singular values, set by the threshold* $I(r) < 0.99$ *and in (b) the equivalent for* $I(r) < 0.999$

After obtaining the DMD modes and eigenvalues from the Algorithm 3, Figure 4.2 plots the eigenvalues $\omega = \log(\lambda)/\Delta t$ (3.3). Since the eigenvalue $\omega_k = \alpha_k + i\beta_k \in \mathbb{C}$, the real part α_j gives the growth rate and the imaginary part β_j gives the frequency associated with the eigenfunction $\varphi_k(\boldsymbol{y})$.

We define

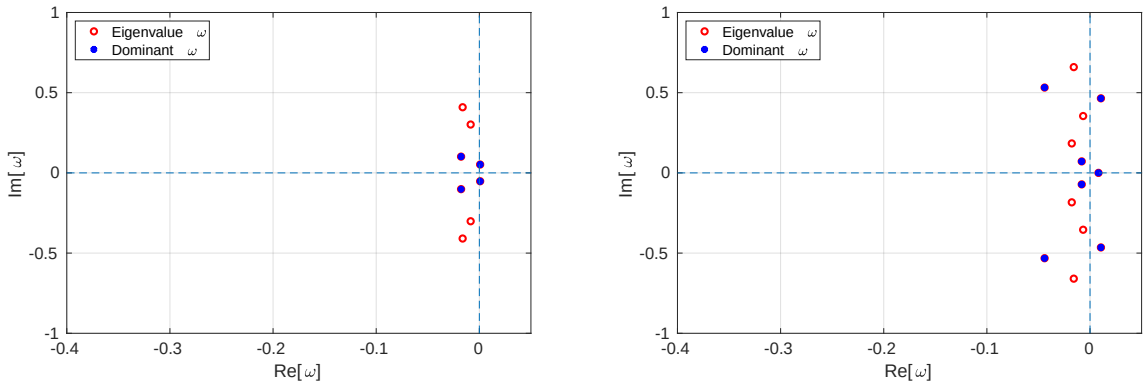
$$w_j = \frac{v_j}{\sum_{l=1}^{n} v_l}, \tag{4.1}$$

where w_j is a normalized weight of each scalar v_j. The eigenvalues in Figure 4.2, marked in blue, correspond to those whose weight of the scalar v_j is $w_j > 0.05$.

Note that by choosing r such that $I(r) < 0.999$, Figures (4.4) and (4.3), show that the relative error ϵ is generally lower than when the choice of r is $I(r) < 0.99$. The mean

relative error for these two cases results in $\overline{\varepsilon}_{0.99} = 0.46892$ and $\overline{\varepsilon}_{0.999} = 0.11439$. This suggests that the low energy modes have an important effect on the system dynamics.

Regarding the location of the eigenvalues, another observation from both cases is that they seem to cluster near the imaginary axis close to the zero real value, denoting a slow growth/decay of eigenvalues associated with the DMD modes.



(a) $I(r) < 0.99$ (b) $I(r) < 0.999$

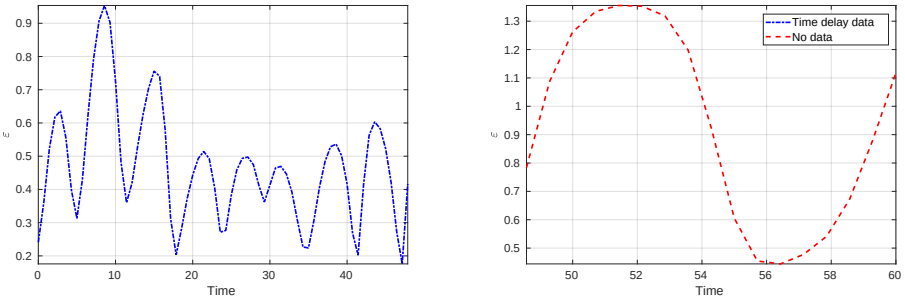
Figure 4.2: *In (a) the distribution of the estimated continuous time eigenvalues ω corresponding to the threshold $I(r) < 0.99$ and in (b), the estimation for a threshold $I(r) < 0.999$. Marked in blue are the dominant eigenvalues, using as a criteria $w_j > 0.05$*

Figures 4.3(a) and 4.4(a) show the evolution of the relative error ε_j in time j when the data was measured. Figures 4.3(b) and 4.4(b), show the evolution of the error when the system evolves beyond the measured data. We conclude that the current model is not useful for long-term future-states predictions.

Polynomial based states observables

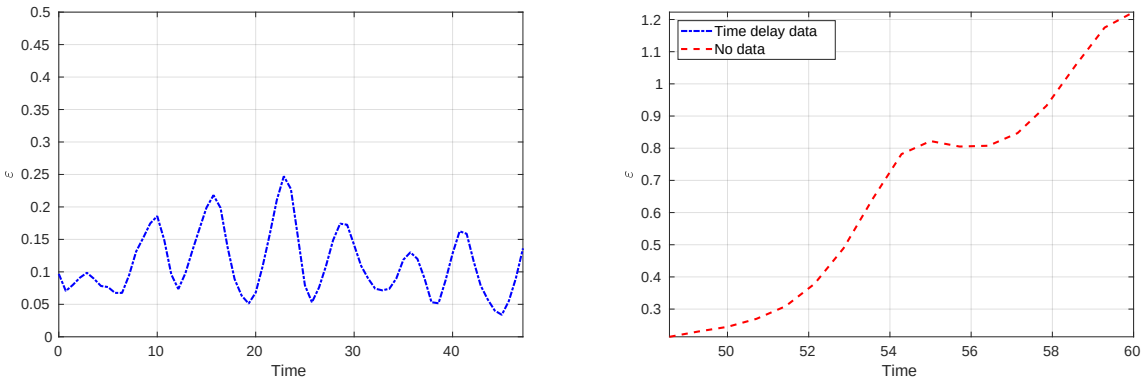
Here we applied the DMD using the observables matrix augmented with a set of polynomials defined as $P_p(\boldsymbol{y}) := (\boldsymbol{y}, \boldsymbol{y}^2, \ldots, \boldsymbol{y}^p)^T$. The tests for $p = 4, 10$, and 20, did not show any improvement from the previous results. When $I(r) < 0.99$, $\overline{\varepsilon}_{P_p} = 0.47043$, and when $I(r) < 0.999$, $\overline{\varepsilon}_{P_p} = 0.11144$, for $p = 4, 10, 20$.

Note that the observables matrix dimension increases, as $\dim(P_p) = 2(n \cdot m)p \times p = 3362 \cdot p \times 68$. Despite of the increased computation cost the method did not give better accuracy.



(a) (b)

Figure 4.3: *Evolution of the relative error ϵ, when the threshold for determining r is $I(r) < 0.99$. In (a) is when the estimated linear model evolves during the time corresponding to the acquired data, whereas in (b) it corresponds to a future-state prediction.*



(a) Measured Data (b) Predicted Data

Figure 4.4: *Evolution of the relative error ϵ, when the threshold for determining r is $I(r) < 0.999$. In (a) is when the estimated linear model evolves during the time corresponding to the acquired data, whereas in (b) it corresponds to a future-state prediction.*

Embedded time delay

Testing DMD using the embedded time delay technique, as we did in section 3.5 for the Van der Pol equations example, produced the best results in terms of accuracy, as we show next.

The observables matrix is now constructed as in (3.30), that is,

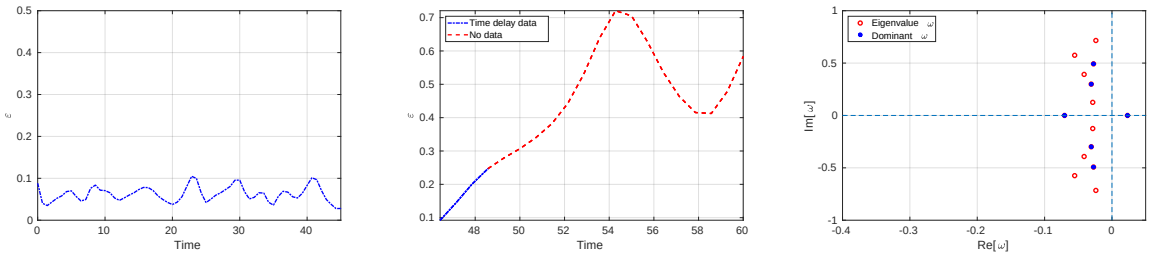
$$\boldsymbol{g}(\boldsymbol{x}) = \begin{pmatrix} g_0\Big(\{\boldsymbol{y}_j\}_{j=0}^{q-d}\Big) \\ g_1\Big(\{\boldsymbol{y}_j\}_{j=1}^{q-d+1}\Big) \\ \\ \vdots \\ g_d\Big(\{\boldsymbol{y}_j\}_{j=d}^{q}\Big) \end{pmatrix} = \begin{pmatrix} \boldsymbol{y}_0 & \boldsymbol{y}_1 & \cdots & \boldsymbol{y}_{q-p} \\ \boldsymbol{y}_1 & \boldsymbol{y}_2 & \cdots & \boldsymbol{y}_{q-p+1} \\ \vdots & \vdots & \cdots & \vdots \\ \boldsymbol{y}_p & \boldsymbol{y}_{p+1} & \cdots & \boldsymbol{y}_q \end{pmatrix}, \tag{4.2}$$

where d denotes the number of copies of time-shifted measurements to stack on the observables matrix (4.2). We tested for $d = 4, 10$, and 20, so that we get the same matrix dimensions as we obtained in the previous polynomial based matrix of observables.

To determine r we considered only the threshold $I(r) < 0.999$.

Note that as the number d changes, the observables matrix dimension also changes according to $\dim\big(\boldsymbol{g}(\boldsymbol{y})\big) = \big(nm \cdot (d+1)\big) \times \big(q-d\big)$.

From (4.2), the approximation to the states $\{\boldsymbol{y}_j\}_{j=0}^{q-d}$ is obtained from the identity $g_0(\boldsymbol{y}) = \boldsymbol{y}$. Figures 4.5(a) and 4.6(a) plot the relative error until time $k = q - d$, whereas in Figures 4.5(b) and 4.6(b) includes the relative error from the time $k = q - d + 1$.

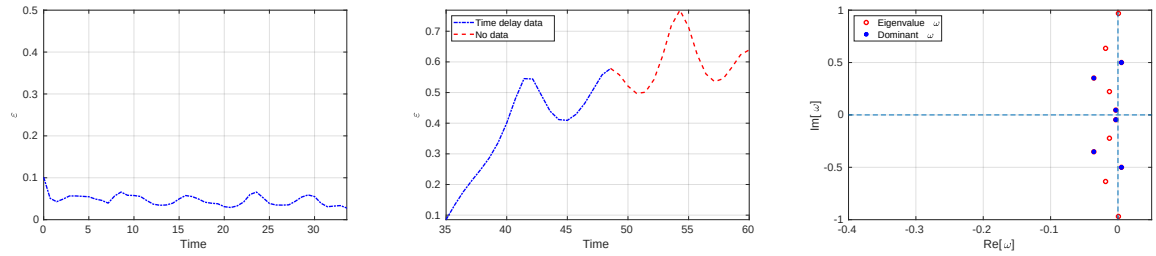


(a) Measured data (b) Prediction (c) Eigenvalues

Figure 4.5: *Using $d = 4$ time delayed vectors, (a) shows the evolution of the relative error ϵ corresponding to $q - d$ time data-points, whereas (b) shows, marked in blue, the evolution of ϵ during the last d measured data points of $\mathcal{X}_1$, and marked in red the corresponding to future-state prediction. In (c) the continuous time eigenvalues locations, where marked in blue are the dominant eigenvalues.*

Figures 4.5(c) and 4.6(c) show that as the value d increases, the continuous time eigenvalues tend to cluster closer to the axis where the real part of its value is zero.

The mean relative error obtained with the increase of d denoted an improvement, as showed by the values of $\overline{\varepsilon}_{d=4} = 0.0689$, $\overline{\varepsilon}_{d=10} = 0.0607$, and $\overline{\varepsilon}_{d=20} = 0.0517$.



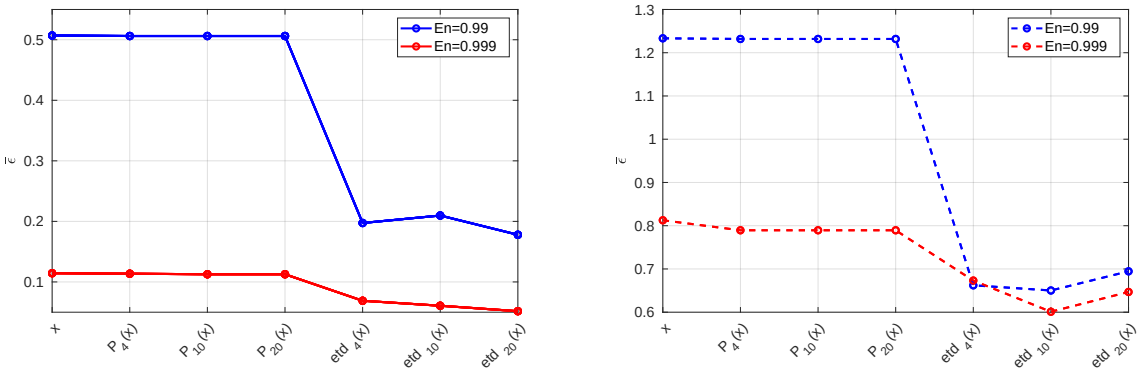
(a) Measured data (b) Prediction (c) Eigenvalues

Figure 4.6: *Equivalent to Figure 4.5, but now with the number of time delays* $d = 20$.

Relative Error Comparison

Figure 4.7 resumes the mean relative errors for the three used techniques with the different parameters. The embedded time delay technique has shown the most accurate results.

However, when $j \geq (q - p)$ as the dimension of the matrix of observables has p less columns, the accuracy reduces sharply, and the predictions produced, as observed in the red curve in Figures 4.5(b) and 4.6(b), are not reliable.



(a) Measured Data (b) Predicted Data

Figure 4.7: *Summarizing the results, the evolution of the mean relative error* $\bar{\epsilon}$ *with each set of chosen observables. In (a), corresponding to the acquired data, and in (b) for future-state predictions. in the* x*-axis,* x *corresponds to the linear measurement of the states,* $P_n(x)$ *corresponds to the* n*th order polynomials agumenting the data matrix, and* etd_d*, corresponds to the embedded time delay technique with* d *time-shifted copied ensembles.*

Comments and Conclusions

From the results obtained, we summarize the conclusions as follow:

1. The choice of threshold in the energy criteria to perform a low-rank truncation of the data have shown a significant impact on the accuracy results. This result also seems to suggest that the low-energy singular values have an important effect on the dynamics.

2. The strategy of augmenting the observable matrix with polynomial based measurements of the states did not produce any improved results from the DMD when compared to the simple linear measurement of the states. This particular choice of observables was not suitable to find an appropriate approximation to the Koopman operator.

3. From the three strategies tested, the embedded time delay showed the most accurate results. This came at a cost, however, since the required number of data points, or the time interval of measurements, had to be larger than the actual time interval for the more accurate estimation. The data matrix is also of larger dimension.

From these observations, for the subsequent tests, the rank r is to be determined by $I(r) < 0.999$ and the embedded time delay technique is used to construct the snapshot matrix of observables $\boldsymbol{g}(\boldsymbol{x})$.

4.1.2 DMD on matrix of observables: Results and analysis

Data-set sampling time

For this section the data-set $\mathcal{X}_2$ is sampled at $\Delta\tau = 0.01429[h]$. However, we can, in fact, chose to alter the current $\Delta\tau$, since the oversampling brings a greater cost in computation with no significant gain in the accuracy.

If we refer to the Nyquist theorem, it states that, in order to adequately reproduce a signal, it should be periodically sampled at a rate that is at least 2 times the highest frequency we wish to record. In the previous tests, which data was sampled at $\Delta\tau = 0.7143$, the highest frequency the algorithm managed to capture corresponded to $\omega \approx 2\pi/9$, which would imply a $\Delta\tau < 4.5[h]$.

However, since we are using the embedded time delay technique, there is a trade-off to consider. From (4.2), we see that higher d reduces the number of columns by the same value, that corresponds to the total number of snapshots, which is now reduced to $q - d$. If

we consider the Nyquist sampling as the sole reference, we would create greater restrictions on the number of time-shifted observables.

Data-set duration

For these runs of Algorithm 3, we considered the data with $\Delta\tau = 0.2857[h]$. For each test we used data spanning the equivalent to $5 \times 24[h]$, from which we used the equivalent to $d = 299$. This results in an increase from the original data snapshot matrix of dimension 5406×420 to a 1621800×121 matrix of observables. The consequence of this choice is that we are using the method to extract spatial structures and associated temporal responses for a time frame of $34[h]$, while using $120[h]$ of data. This is a cost we are willing to take in this specific case, since we are interested in finding fundamental spatial structures and temporal patterns that we can extract from a complex system such as this one. If we were aiming to make future state predictions, the strategy would have to be adjusted.

Experiments description

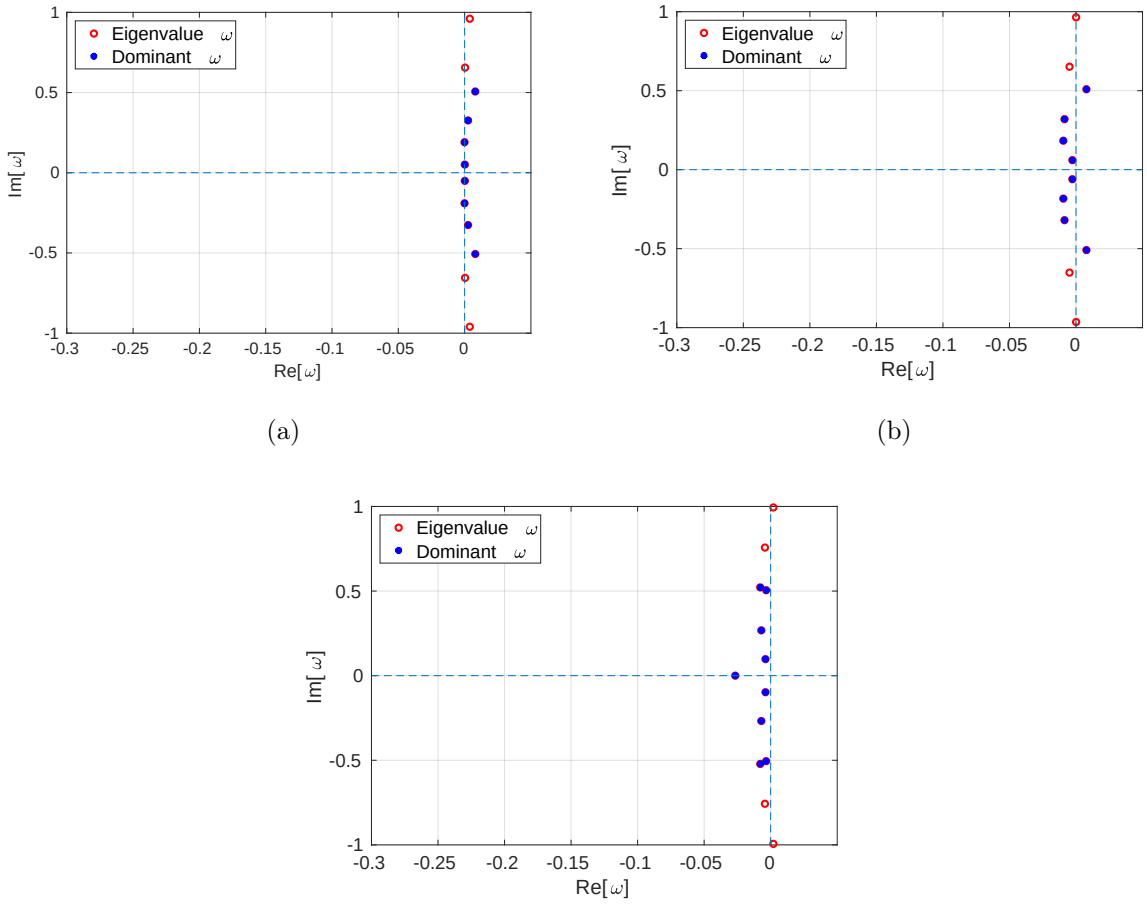
We run this setup at three different time frames, namely $\tau^{(1)} = \big[0, 120\big][h]$, $\tau^{(2)} = \big[500, 620\big][h]$ and $\tau^{(3)} = \big[1000, 1120\big][h]$. Each time frame corresponds to a experiment. For simplicity, we use the notation for time t, such that, $t = 0$ at $\tau_0^{(i)}$, where i is the number of the experiment, and the subscript 0 in τ denotes the initial time. The data-sets corresponding each experiment are denoted by $\mathcal{X}_2^{(i)}$, which are subsets of $\mathcal{X}_2$.

The goal is to analyze the DMD modes and eigenvalues resulting from each experiment, and to identify spatiotemporal coherent structures that persist in each of these three different time frames.

In Figure 4.8 we show the maps of the continuous time eigenvalues ω_j obtained from the DMD from each of the three experiments. We are focusing on the most dominant eigenvalues which we rate from (4.1). Marked in blue, the eigenvalues which hold the threshold $w_j > 0.01$.

Experiment 1: $\tau = \big[0, 120\big][h]$

With the data-set $\mathcal{X}_2^{(1)}$, the construction of the observable matrix (4.2) is based on the stacking of $d = 399$ time-shifted observables. Figure 4.9 depicts the selected 4 dominant complex conjugate eigenvalues and corresponding DMD modes determined from the DMD.



(a) (b)

(c)

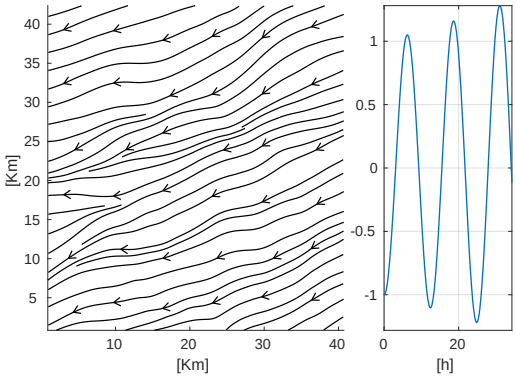
Figure 4.8: *Location of the time continuous eigenvalues estimated by the DMD, when (a)* $\tau = [0, 120]$*, (b)* $\tau = [500, 620]$*, and (c)* $\tau = [1000, 1120]$*.*

Since the eigenvalues are complex conjugate, we only represent the ones with positive imaginary part.

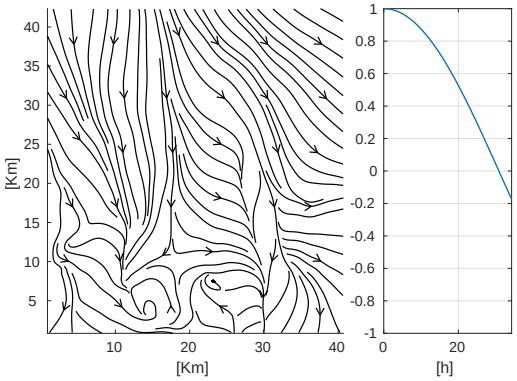
The right panel in each subplot shows the dynamics related to the term $\exp(\omega_k t)$ in (3.3). Since ω_k is a complex number, we observe a oscillatory behavior with a periodicity of $12.3[h]$ ($T_k = 2\pi/\beta_k$). In the left panel we represent the streamlines that result from a constant vector field defined by the DMD mode $\phi_k(\boldsymbol{x})$. Figures 4.9(b), 4.9(c) and 4.9(d) are the representations of the remaining modes, not including their complex conjugate.

Figure 4.10 shows the result of the sum of the DMD modes with the dynamics from Figure 4.9 at time $t = 17[h]$ when compared with the original data . The color-map corresponds to kinetic energy, given by $k_j = u_j^2 + v_j^2$.

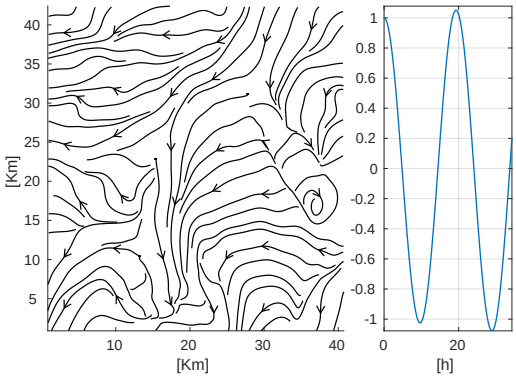
One can observe that, using only 4 modes, the streamlines generated by the vector field determined from the linear model exhibits a very similar structure to the original data.



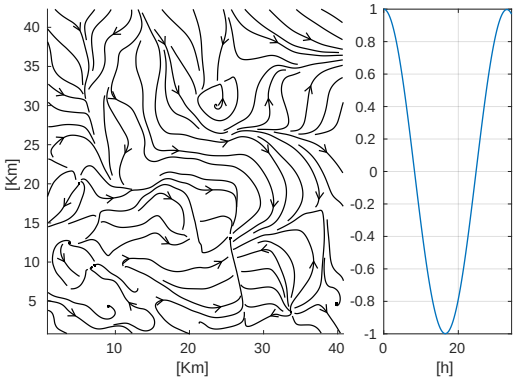
(a) DMD mode 1 with $v_1 \approx 11.7$ and $T_1 \approx 12.4(h)$



(b) DMD mode 3 with $v_3 \approx 17.5$ and $T_2 \approx 124(h)$



(c) DMD mode 5 with $v_5 \approx 5.6$ and $T_5 \approx 19.3(h)$



(d) DMD mode 7 with $v_7 \approx 12.8$ and $T_1 \approx 33(h)$

Figure 4.9: *Dominant modes and eigenvalues estimated by the DMD from data of Experiment 1. On the left graph of each plot, the streamlines generated from the DMD mode ϕ_j corresponding to the eigenvalue ω_j, and on the right side the evolution in time of* $\exp(\omega_j t)$

The relative error evolution in time is in Figure 4.11(a), with $\bar{\epsilon}^{(1)} \approx 0.0648$. Figure 4.11(b) shows the evolution of the relative error corresponding to the time interval when the data points were used for the time-shifted components in the observable matrix (in blue), and the future-state predictions (in red).

Experiment 2: $\tau = \big[500, 620\big]\,[h]$

In this experiment we now use the data-set $\mathcal{X}_2^{(2)}$ with the same parameter d and threshold $I(r) < 0.999$ as used in Experiment 1. Figure 4.12 depicts the modes corresponding to the eigenvalues selected in Figure 4.8(b).

In Figure 4.13, for comparison, we show the reconstructed velocity field at time $t = 17[h]$ using 4 modes, side by side with the original data. We can observe that the model was

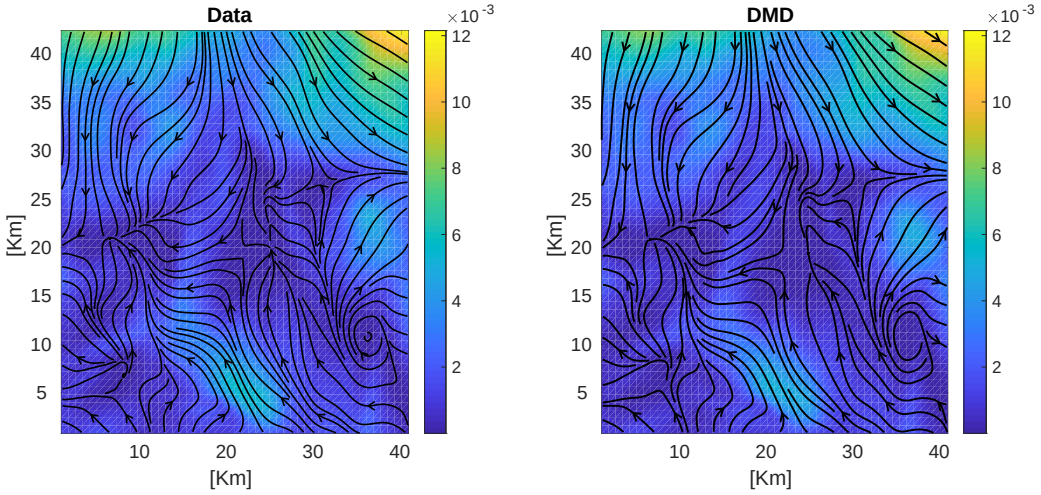
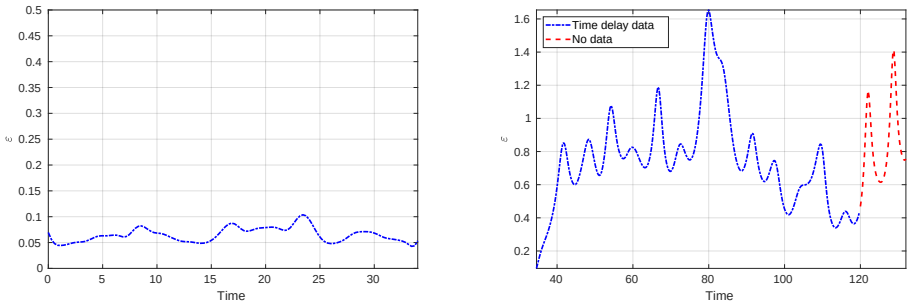


Figure 4.10: *Solution of Experiment 1 at time $t = 17[h]$. The left plot corresponds to the exact solution, and on the right the obtained from the linear model estimated by the DMD.*



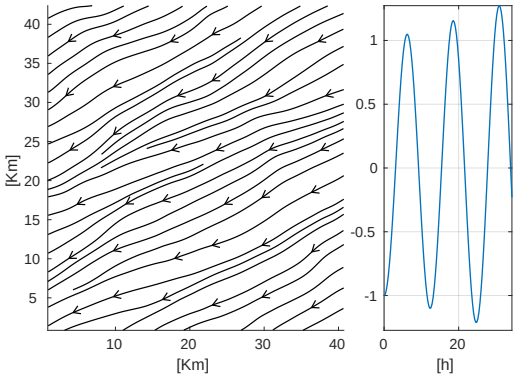
(a) $\bar{\epsilon} = 0.0648$ (b)

Figure 4.11: *Evolution in time of the relative error ϵ obtained with the DMD estimated linear model in Experiment 1. In (a) during the acquired data $\mathcal{X}_2^{(1)}$, and in (b) the combination of the last d acquired data-points, marked in blue, and the future-state prediction in red.*

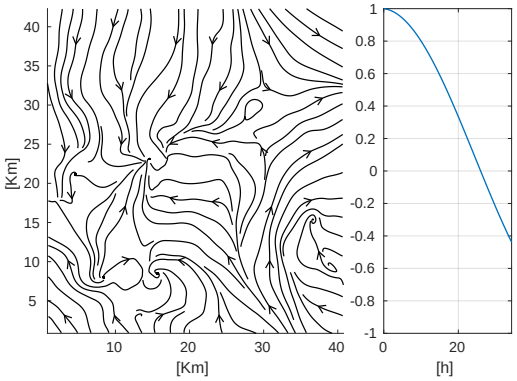
able to reconstruct the main characteristic structures of the original dynamical system.

Figure 4.14(a) shows the relative error evolution ϵ over the time corresponding to the measured data with mean relative error $\bar{\varepsilon}^{(2)} = 0.0644$. Figure 4.14(b) shows the relative error for the future-state prediction (in red), and the relative error during the d time delays (in blue).

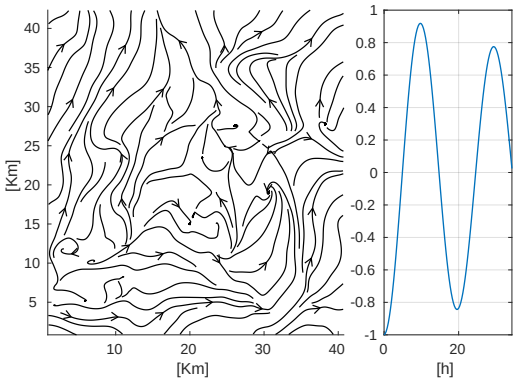
Comparing with the modes in Figure 4.9, note that the DMD modes and eigenvalues corresponding to Figure 4.12(a) and Figure 4.9(a) exhibit a very similar structure and periodicity ($T = 12.4[h]$).



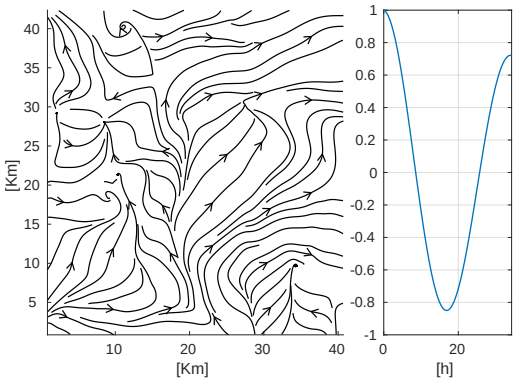
(a) DMD mode 1 with $v_1 \approx 15.8$ and $T_1 \approx 12.3(h)$



(b) DMD mode 3 with $v_3 \approx 21.2$ and $T_3 \approx 104(h)$



(c) DMD mode 5 with $v_5 \approx 6.63$ and $T_1 \approx 19.7(h)$



(d) DMD mode 5 with $v_1 \approx 16.6$ and $T_1 \approx 34.3(h)$

Figure 4.12: *Dominant modes and eigenvalues estimated by the DMD from data of Experiment 2. On the left graph of each plot, the streamlines generated from the DMD mode ϕ_j corresponding to the eigenvalue ω_j, and on the right side the evolution in time of* $\exp(\omega_j t)$

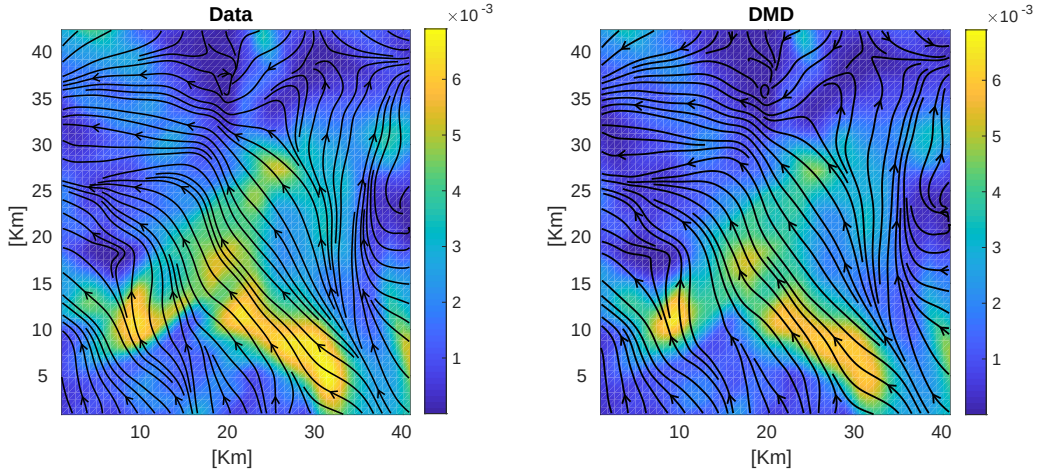
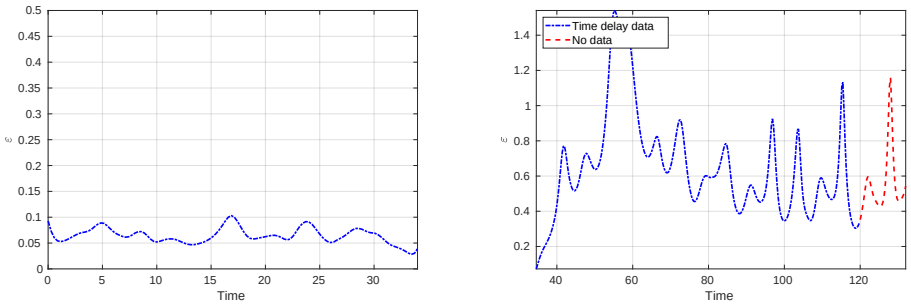


Figure 4.13: *Solution of Experiment 2 at time $t = 17[h]$. The left plot corresponds to the exact solution, and the right one the obtained from the linear model estimated by the DMD.*



(a) Measured Data (b) Prediction

Figure 4.14: *Evolution in time of the relative error ϵ obtained with the DMD estimated linear model in Experiment 2. In (a) during the acquired data $\mathcal{X}_2^{(2)}$, and in (b) the combination of the last d acquired data-points, marked in blue, and the future-state prediction in red.*

Experiment 3: $\tau = \big[1000, 1120\big][h]$

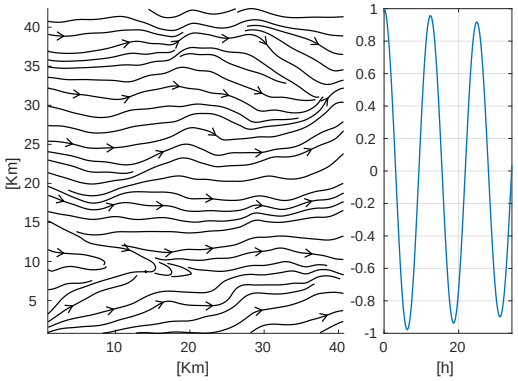
Similarly to the procedure of the previous experiments, the DMD modes and dynamics can be seen in Figure 4.15. In Figure 4.16, the dynamics as estimated using the dominant modes, at time $t = 17[h]$, side by side with the original data.

Figure 4.17 plots the relative errors. The mean relative error corresponding to Figure 4.17(a) was $\bar{\varepsilon}^{(3)} = 0.0587$.

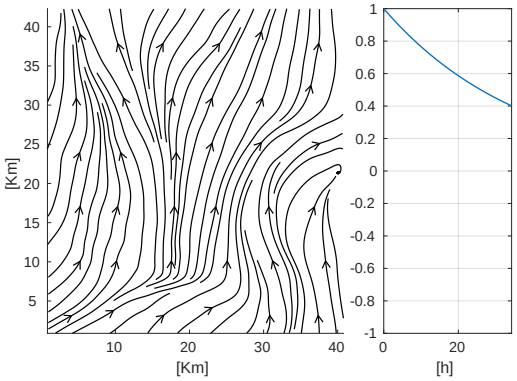
4.1.3 Comments and Conclusions

From the three results presented we can identify one dominant mode that persists in each of the three different experiments. The eigenvalue has a period of $T \approx 12.4(h)$, and the corresponding DMD mode $\phi(\boldsymbol{y})$ shows a laminar-like structure. This suggest that we may have identified the M_2 tidal mode component of this velocity field [7].

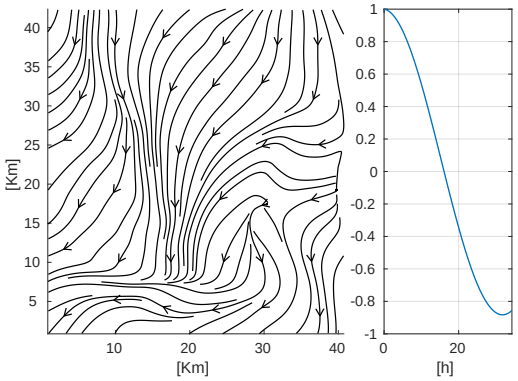
When generating a solution $\widetilde{\boldsymbol{y}}_k$, $k = 0, 1, \ldots$, from the linear model approximation obtained by DMD, the accuracy when compared to the original data $\boldsymbol{y}_k$ showed values consistently under 10%, even when using only 4 modes, and we were able to reproduce fairly complex spatiotemporal structures. In fact, if we average the mean relative error $\bar{\epsilon}$ obtained by each experiment, $\bar{\epsilon}_{experiments} = 0.0626$. Have we used all the determined modes, this average relative error would drop to $\bar{\epsilon}_{all\ modes} = 0.0332$.



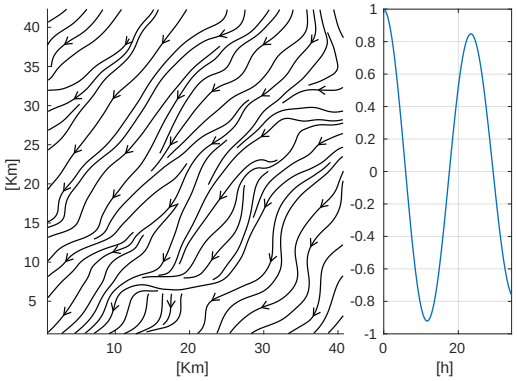
(a) DMD mode 1 with $v_1 \approx 13.4$ and $T_1 \approx 12.4(h)$



(b) DMD mode 3 with $v_3 \approx 41.6$ and $T_3 \approx \infty(h)$



(c) DMD mode 5 with $v_5 \approx 34.4$ and $T_3 \approx 64.4(h)$



(d) DMD mode 7 with $v_7 \approx 11.9$ and $T_3 \approx 23.5(h)$

Figure 4.15: *Dominant modes and eigenvalues estimated by the DMD from data of Experiment 3. On the left graph of each plot, the streamlines generated from the DMD mode ϕ_j corresponding to the eigenvalue ω_j, and on the right side the evolution in time of* $\exp(\omega_j t)$

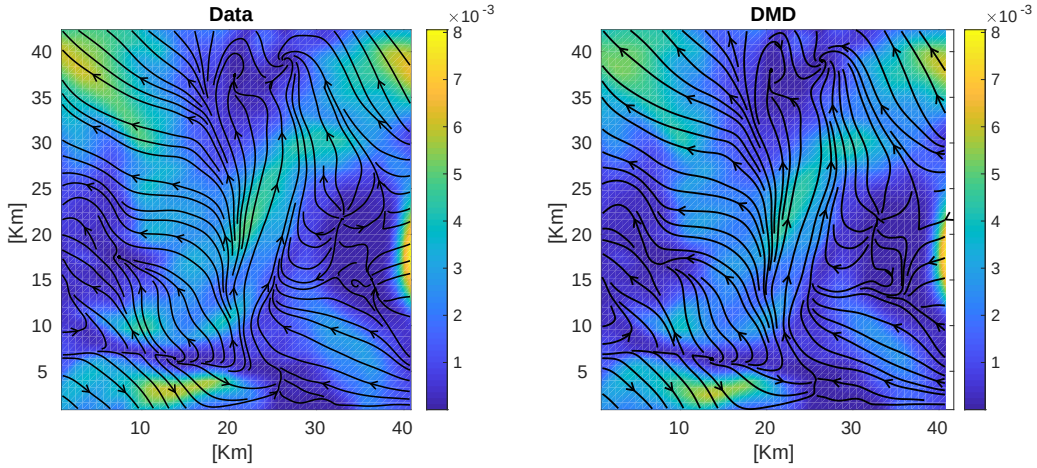
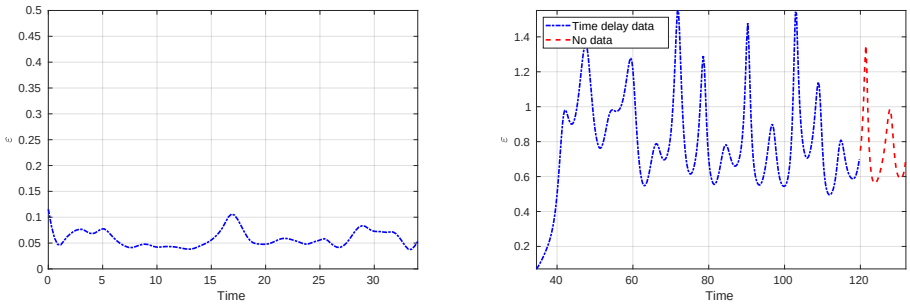


Figure 4.16: *Solution of Experiment 3 at time $t = 17[h]$. The left plot corresponds to the exact solution, and the right one the obtained from the linear model estimated by the DMD.*



(a) Measured Data (b) Prediction

Figure 4.17: *Evolution in time of the relative error ϵ obtained with the DMD estimated linear model in Experiment 3. In (a) during the acquired data $\mathcal{X}_2^{(3)}$, and in (b) the combination of the last d acquired data-points, marked in blue, and the future-state prediction in red.*

4.2 Neural Field Experiment

In [28] a two-population firing-rate model describing the dynamics of excitatory and inhibitory neural activity in one spatial dimension is investigated with respect to formation of patterns, in particular stationary periodic patterns and spatiotemporal oscillations. The model is described by means of the coupled system of two nonlinear integro-differential equations,

$$\begin{aligned} \frac{\partial}{\partial t}u_e &= -u_e + \omega_{ee} \otimes P_e(u_e - \theta_e) - \omega_{ie} \otimes P_i(u_i - \theta_i) \\ \tau\frac{\partial}{\partial t}u_i &= -u_i + \omega_{ei} \otimes P_e(u_e - \theta_e) - \omega_{ii} \otimes P_i(u_i - \theta_i) \end{aligned} \tag{4.3}$$

where $\otimes$ denotes the convolution operator

$$[f \otimes g](x) = \int_\Omega f(x-y)g(y)dy, \quad \Omega \subseteq \mathbb{R}.$$

Here u_e and u_i denote the membrane potentials of excitatory and inhibitory elements, respectively, at the spatial point x and time t. The region Ω is the spatial region occupied by the neurons. The functions ω_{mn} $(m, n = e, i)$ model the coupling strengths in the network, while P_m $(m = e, i)$ are the firing rate functions modelled by sigmoidal functions with parameterized maximum inclinations. The parameter τ is the relative inhibition time while θ_e and θ_i are the threshold values for firing of the excitatory and the inhibitory neurons, respectively. For the certain parameter regimes, evolution of perturbed constant

states $u_i = u_e = const$ result in different oscillatory patterns, see data $X = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{100})$ in Figure 4.18 and data $Y = (\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{200})$ in Figure 4.25 and [28]. Here we test if the DMD can reconstruct these stable oscillatory patterns.

4.2.1 Computing the DMD

For that end, since we are only interested on the slow decaying or growing eigenvalues, we ignored all modes which eigenvalues with negative real values are further away from the imaginary axis, regardless of the value of the scalar v_j. That is, given the continuous time eigenvalues $\omega_j = \alpha_j + i\beta_j$, we select the modes which growth rate given by α_j are closer to (or greater than) zero.

For these computations, the reduced rank r was determined by the threshold $I(r) < 0.999$. We are selecting the eigenvalues with growth rate $\alpha_j \geq -0.01$, and the weight $w_j \geq 0.1$ using (4.1).

To this data, we added Gaussian white noise. This added noise only affects the measurement but does not interact with the true dynamics of the system, that is, we assume that the states $\boldsymbol{x}_k$ take the form

$$\widehat{\boldsymbol{x}}_k = \boldsymbol{x}_k + \boldsymbol{n}_k,$$

where $\boldsymbol{n}_k$ is a random noise vector, and the subscript $k = 0, 1, \ldots, m$ corresponds to the discrete time instant. We take each component of $\boldsymbol{n}_k$ to be independent and normally distributed with zero mean and a given variance, *i.e.*, $\boldsymbol{n}_k \sim \mathcal{N}(\boldsymbol{0}, N)$.

In [8] it was shown that the computation of the DMD eigenvalues is biased by the presence of sensor noise. In this same work, several methods are presented for debiasing within the standard DMD framework. A different approach is also presented in [2].

In this current application case, however, we instead test the effect of using the time delay technique in the noise affected data.

Data-set X

Figure 4.19 shows $\widehat{X}$ when $N = \text{diag}(0.2)$.

Linear measurements of the states

We first run the DMD algorithm considering that the observables matrix is the linear measurement of states $\boldsymbol{g}(\boldsymbol{x}_k) = \boldsymbol{x}_k$.

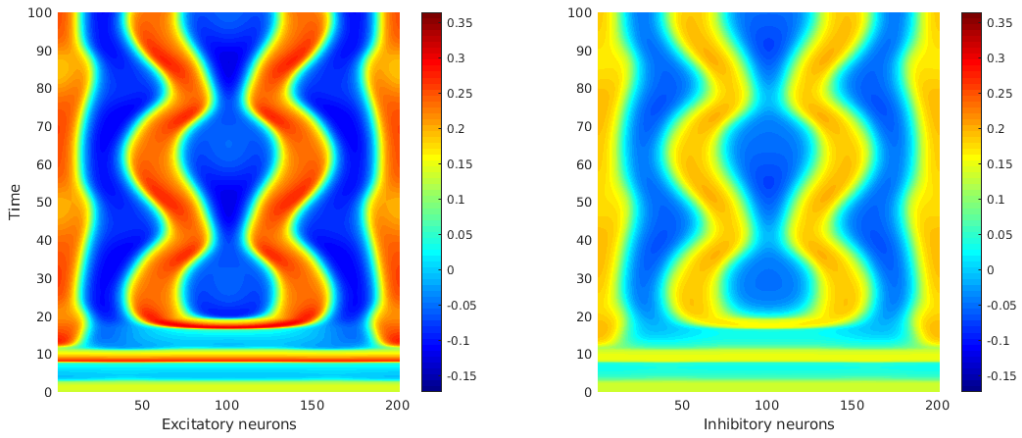


Figure 4.18: *Data X*

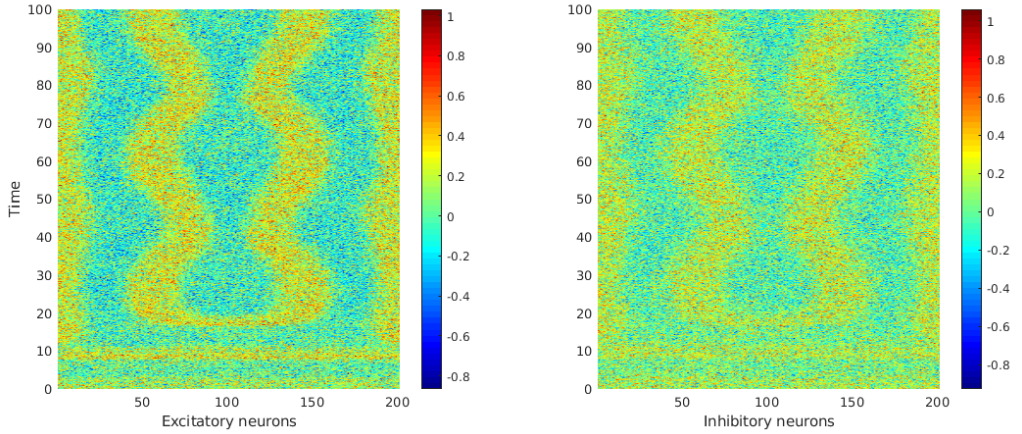
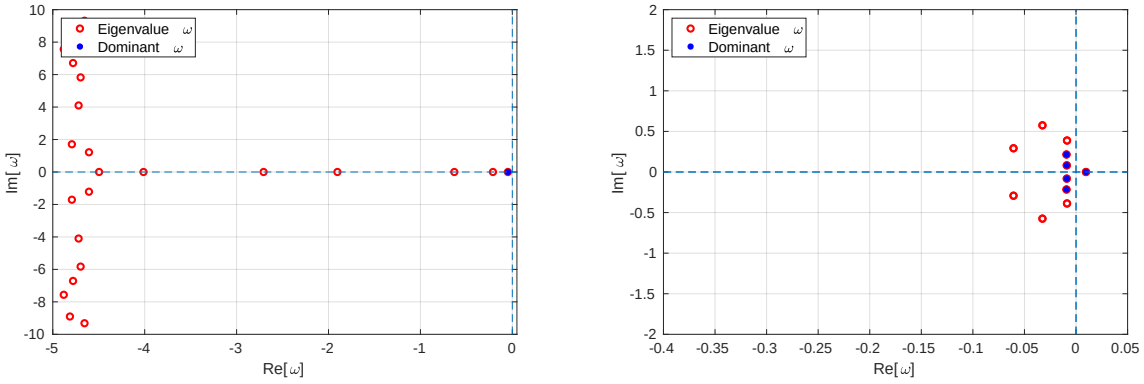


Figure 4.19: *Data $\widehat{X}$ with added noise $\boldsymbol{n}_k$.*

The resulting continuous time eigenvalue (ω_k) map is in Figure 4.20(a). The dominant eigenvalues are real valued and far from zero, thus denoting a fast decay. From this we can immediately see that the method failed to capture the dynamics, since all states rapidly decay to zero. For comparison, Figure 4.20(b) shows the eigenvalues obtained by the DMD from the data X not affected by noise. Here, the eigenvalues are closer to the imaginary axis, and we can also find complex conjugates with $\alpha_j \geq 0.05$.

Although the obtained DMD model did not result in an accurate approximation for the dynamical system either as $t \to +\infty$, the crucial point is that the added noise produced a very noticeable effect on the determination of the DMD eigenvalues. More on the effect of sensor noise in the DMD can be read in [8].



(a) (b)

Figure 4.20: *Eigenvalue map considering $g(\boldsymbol{x}) = \boldsymbol{x}$. In (a) the eigenvalues were estimated by the DMD from the noisy data $\widehat{X}$. The eigenvalues which are closest to the imaginary axis are only real valued. The dominant marked in blue, does not even satisfy the condition we set for the real value, since $\omega_X = -0.049$. In (b) we see the dominating eigenvalues estimated by the DMD from computing the noiseless data X, $\omega_{\widehat{X}} = \{-0.0097, -0.009 \pm i0.0082, -0.0092 \pm i2154\}$.*

Embedded time delay

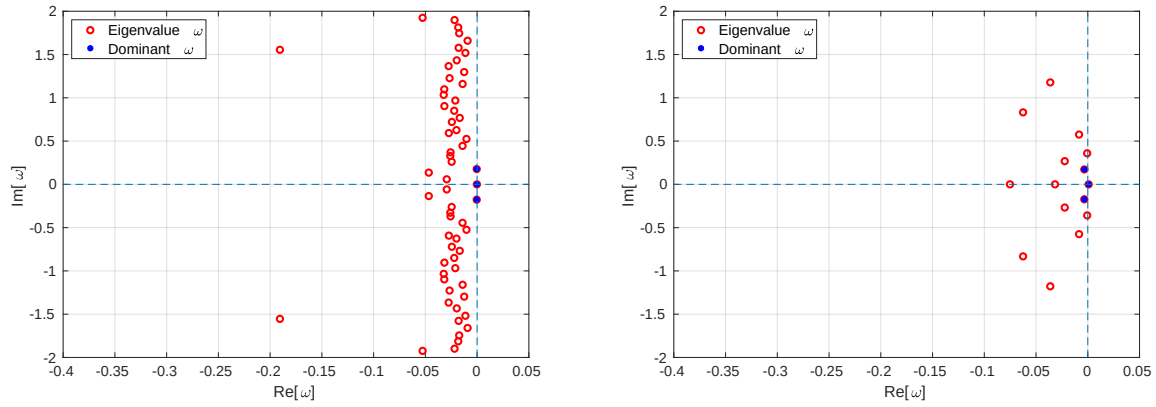
Next, we considered the observables snapshot matrix constructed from time-shifted copies of the measurements X, as in (3.30) and set $d = 19$.

Figure 4.21(a) maps the DMD continuous time eigenvalues obtained from $\widehat{X}$. For comparison, Figure 4.21(b) maps the DMD eigenvalues obtained from the data not affected by noise. We see that the dominant eigenvalues in both cases, marked in blue, are very similar.

Figure 4.22 displays the modes and eigenvalues extracted from $\widehat{X}$ by the DMD. As in the previous section, we only plot one of the complex conjugate pairs, hence the two plots to modes and eigenvalues (ϕ_j, λ_j) correspond to $j = 1$ and $j = 3$. The upper two graphs show the time evolution of $\boldsymbol{x}(j,t) = v_j \varphi_j \exp(\omega_j t)$, for each mode j. The left plot corresponds to the excitatory neurons, and the right to the inhibitory neurons. The bottom graphic plots the time dependent term $\exp(\omega_j t)$ of the linear model (3.3).

The sum of the 3 modes results in the approximation $\widetilde{\boldsymbol{x}}(t)$, plotted in Figure 4.23. As expected, the transient dynamics occurring from t_0 until $t \approx 20$ is not captured by the model, given that we excluded all eigenvalues which growth rate $\alpha_j < -0.01$. However, a stable pattern is shown in the approximated dynamics.

The nonzero real values of the eigenvalues indicates that, even if slow, there is a growth



(a) (b)

Figure 4.21: *Continuous eigenvalue location with* $d = 19$. *In (a), the dominant eigenvalues, estimated from* $\widehat{X}$, $\omega_{\widehat{X}} = \{0.0011, -0.0031 \pm i0.1699\}$, *and in (b) the estimated from* X, $\omega_X = \{0.0008, -0.0033 \pm i0.1726\}$

of the DMD mode 1 corresponding to ω_1, and a decay of modes 2 and 3 corresponding to ω_2 and ω_3, so that in the long run as $t \to \infty$, the accuracy of the approximation $\tilde{\boldsymbol{x}}$ to the exact solution tends to degrade.

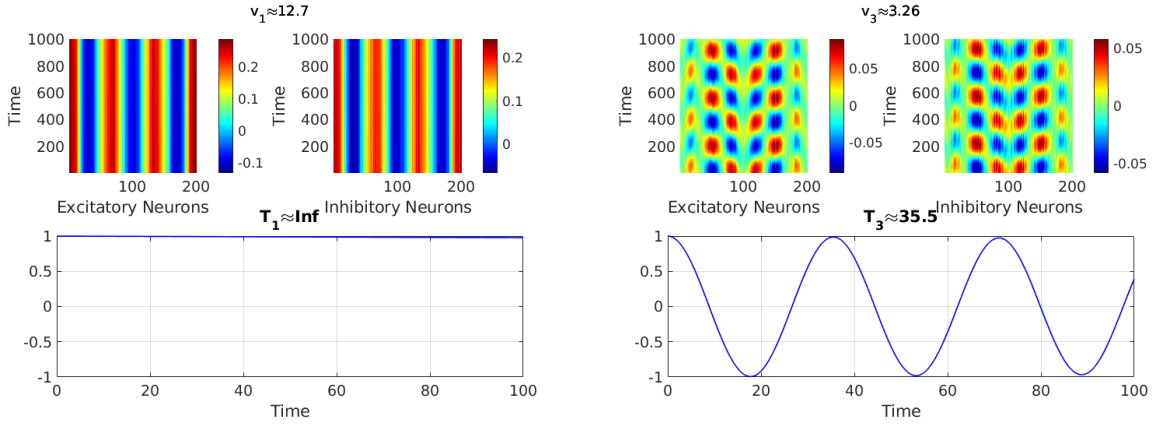
For the complex conjugate pair of eigenvalues the phase $\theta = \arctan(Im(\omega_j)/Re(\omega_j))$ is non-zero as well. In fact, by inspecting Figure 4.24(a), one verifies an oscillatory behavior of the relative error ϵ_j curve. Moreover, comparing Figures 4.23 and 4.18, we can observe a phase delay on the oscillations of the model generated solution. To test this hypothesis, we added to the model a phase delay corresponding to $\theta_1 = 0, \theta_2 = -1.5587, \theta_3 = 1.5587$. The linear model was now corrected to $\tilde{\boldsymbol{x}}(t) = \sum_{j=1}^{3} v_j \phi_j(\boldsymbol{x}) \exp(\omega_j t + i\theta_j)$.

Figure 4.24(a) plots the evolution of relative error ε when the approximation $\widetilde{\boldsymbol{x}}$ is obtained with only 3 modes. In Figure 4.24(b) the evolution of the relative error ϵ_j corrected with a phase delay θ. The results indicate that the phase θ is in fact one of the factors that contributes to the relative error ϵ_k.

Data-set Y

Figure 4.26 plots the noise affected data $\widehat{Y}$, when $N = 0.1$.

Once again, the observables used to approximate the Koopman eigenfunctions are based on the time-delay technique. The number of time-shifted observables of Y was set to $d = 399$.



(a) Mode 1 (b) Mode 2

Figure 4.22: *Plots of the time evolution of each of the DMD modes extracted from data* $\widehat{X}$ *(on top) and corresponding time evolution of the eigenvalue dependent term of the DMD (at the bottom). (a) corresponds to the real valued eigenvalue* ω_1*, and (b) to the complex valued* ω_3 *equivalent to its complex conjugate* ω_2*.*

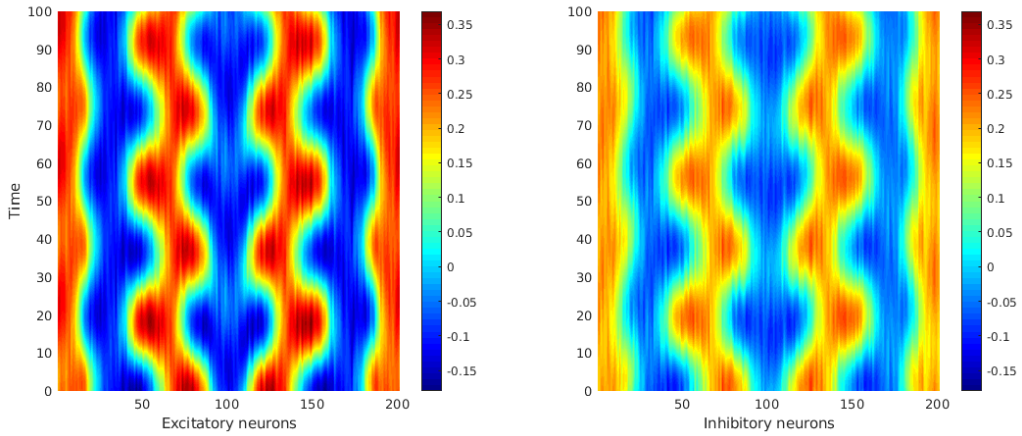


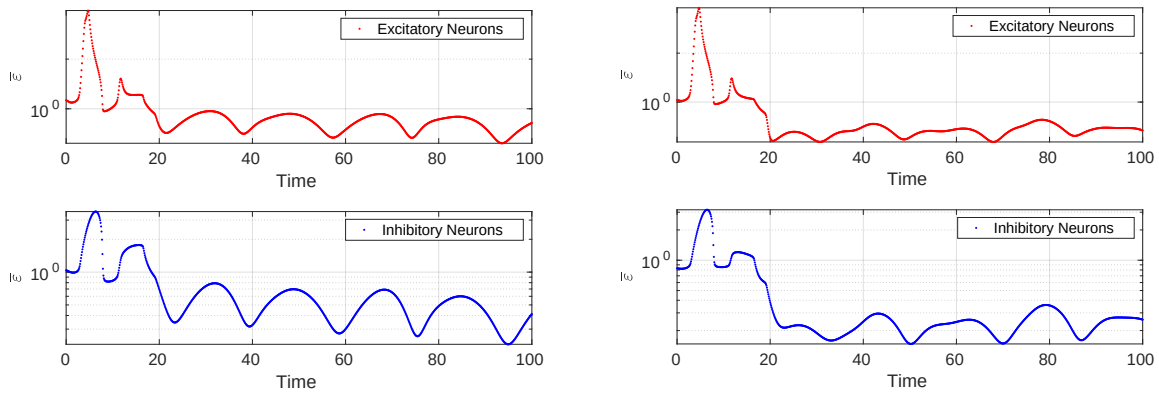
Figure 4.23: *Solution* $\widehat{x}(t)$ *generated by the DMD from the computation of data* $\widehat{X}$*.*

After running the DMD algorithm on the data $\widehat{Y}$, we reconstructed the continuous time dependent approximation $\widetilde{\boldsymbol{y}}(t)$ and plotted its evolution in Figure 4.29. Figure 4.27 plots the dominant DMD modes and dynamics.

Finally, Figure 4.30 plots the evolution of the relative error.

4.2.2 Comments and Conclusions

In this section, as we set the goals we proposed to achieve, the choices of parameters and criteria focused on the growth rate of the eigenvalues estimated by the DMD. Choosing dif-



(a) (b) With phase θ correction

Figure 4.24: *Evolution in time of the relative error $\bar{\epsilon}$ corresponding to the data-set $\widehat{X}$. The plot in (b) corresponds to the linear model corrected with the phase delay θ, whereas (a) corresponds to the linear model estimated directly from the DMD.*

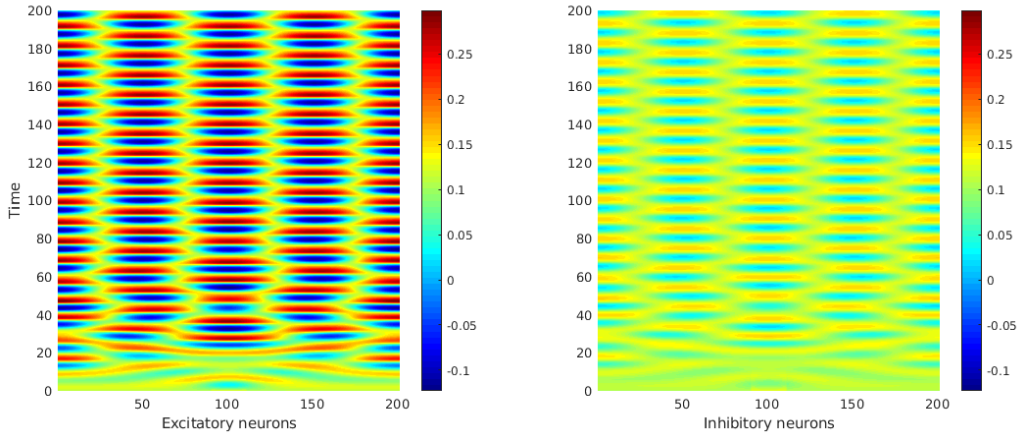


Figure 4.25: *Data Y*

ferent criteria to select dominant modes depends on what is the specific goal of the study when analyzing a particular system. For example, a different goal could be identifying specific oscillatory patterns in the system, which we could characterize by the frequency of oscillation measured by the imaginary component of the eigenvalues. Since the focus was only on the identification of the most dominating stationary periodic patterns and spatiotemporal oscillations, the resulting linear model when only composed of the corresponding modes and eigenvalues, was not adequate to be used as an accurate model of the full-dimensional system.

Comparing the computation of the two data-sets X and Y, augmented with the time-

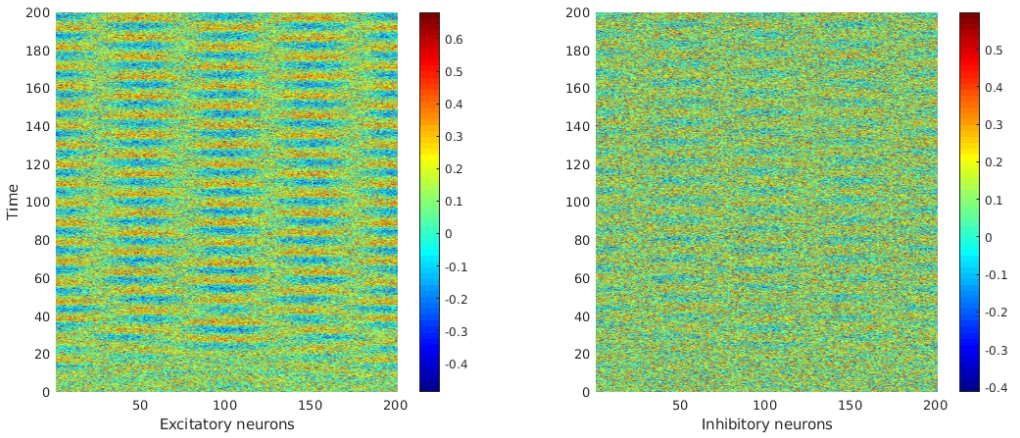
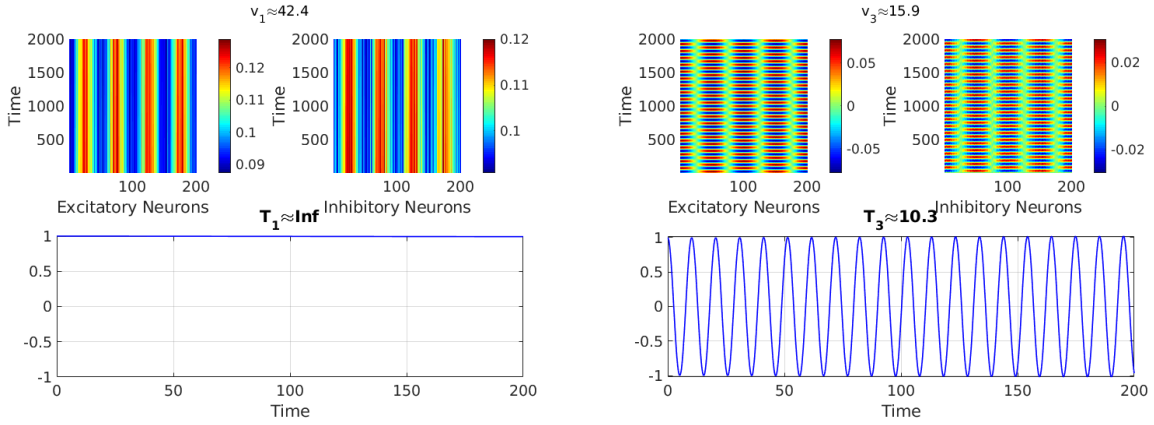


Figure 4.26: *Data $\widehat{Y}$, with the added noise $\boldsymbol{n}_k$*



(a) Mode 1

(b) Mode 2

Figure 4.27: *Plots of the time evolution of each of the DMD modes extracted from data $\widehat{Y}$ (on top) and corresponding time evolution of the eigenvalue dependent term of the DMD (at the bottom). (a) corresponds to the real valued eigenvalue ω_1, and (b) to the complex valued ω_3 equivalent to its complex conjugate ω_2.*

delayed components, we observed that a much larger number $d_Y = 399$ and data points (total of 200), when compared to $d_X = 19$ and 100 data points for the data-set X, was needed before achieving a satisfactory result. If we recall the observations from the Van der Pol equations example while using embedded time delay in Chapter 3, the number of time-shifted observables d necessary to find an approximation solution increased the more strongly nonlinear the Van der Pol equations were. It is plausible that this case follows the same trend.

One other aspect we tested with these two cases was the effect of added Gaussian white

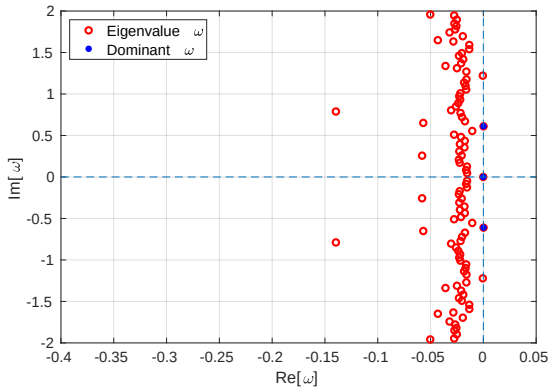


Figure 4.28: *Continuous eigenvalue location with* $d = 399$*, which dominant eigenvalues correspond to* $\omega_{\widehat{Y}} = \{0.0, 0.0001 \pm i0.6100\}$*.*

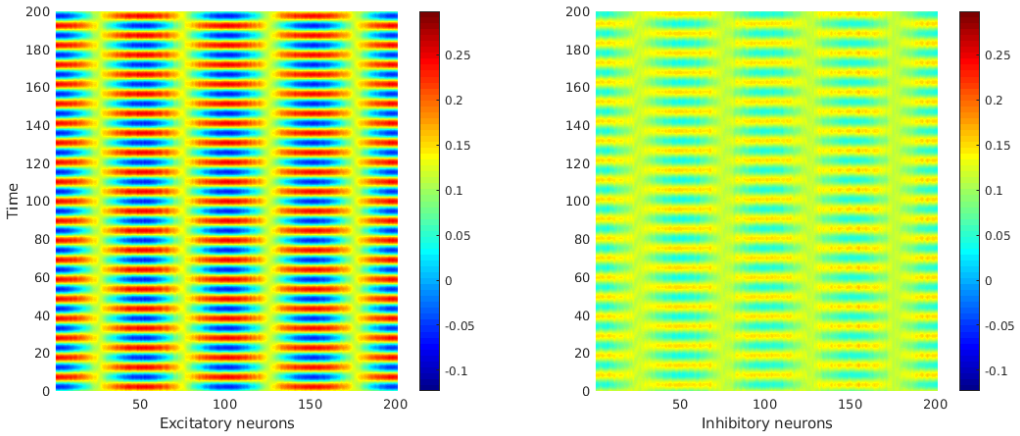


Figure 4.29: *DMD*

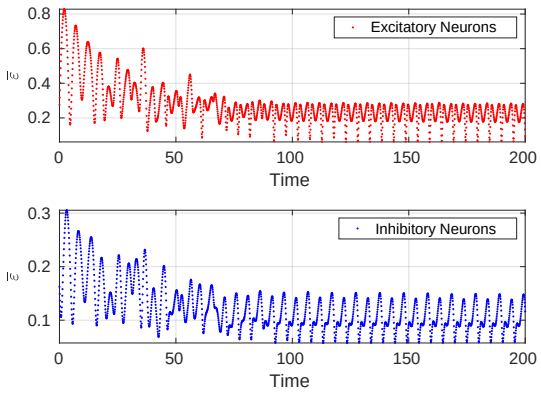


Figure 4.30: *Evolution in time of the relative error* $\bar{\epsilon}$ *corresponding to the data-set* $\widehat{Y}$

noise to the measured data. The addition of time-shifted observables resulted in a solution where the effect of noise was reduced, while successfully recovering the dominant modes and eigenvalues characteristic of the dynamical system.

Chapter 5

Discussion and Conclusions

5.1 Summary

Although the Dynamic Mode Decomposition is the feature method presented in this thesis, it is the strong connection to the Koopman theory that allows to show that the DMD is useful for characterizing nonlinear dynamics, and, furthermore, to develop strategies to improve this characterization.

The DMD is a purely data-driven method that computes approximations to the Koopman eigenvalues, eigenfunctions and modes from a set of snapshots. It differs from the POD method in that the latter is used essentially to determine an optimal set of orthogonal basis functions, a new set coordinates in which space the solution of the problem evolves. The DMD, however, besides providing the modes, not necessarily orthogonal, also provides the associated eigenvalues, with which we are able to construct a time dependent model of the system.

The fact that the DMD does not pose restrictions on the linearity of the dynamical system, nor the knowledge of its governing equations, makes it possible to use in different disciplines, such as in Fluid Mechanics or in Neuroscience. The choice of the applications presented in Chapter 4 is a reflection of these points.

5.2 Results

The short examples included in Chapters 2 and 3 illustrated the different characteristics related to the Koopman operator and the DMD method and to introduce strategies and

techniques which we later used in the main two applications. Particularly, the embedded time delay [3, 24, 27].

One tested feature of the DMD method was the effects on the accuracy by reducing the order of the system. The criteria for the reduction was based on the energy threshold expressed in (1.9). However, in the ocean model application the difference between choosing a threshold of 99% and 99.9% was significant, suggesting that low-energy modes in this system may have significant impact on the dynamics.

Throughout the examples and the applications presented in the thesis, the importance of selecting the correct set of observables was evident. On our applications, the data matrix of snapshots augmented with the copied time-shifted linear measurements of the states, consistently showed the most accurate results. This came at a cost, however, which was more obvious ocean model application, as the addition of time-shifted measurements had the consequence of increasing the data matrix dimension. The computational cost of running the DMD algorithm consequently increased as well. Moreover, to be able to use this technique, we require the access to extra snapshots for the evolving system beyond the time window we are studying.

In both applications, we were able to identify dominant spatiotemporal patterns and temporal responses characteristic of the studied dynamical systems purely by processing data. In the ocean model case, we captured a structure resembling the M_2 tidal mode, and in the neuron field, two main dominant modes.

The approximation of the Koopman modes and eigenvalues generated by the DMD Algorithm 3 allowed us to construct a time depended linear model, which can potentially output the dynamical system state at any point in time, including future-state predictions. However, the results obtained for that end in the Bergen Ocean Model application were not good. The fact that this particular system is characteristically subjected to external time dependent forces that may change its equilibrium state, poses challenges that the strategies we used in this thesis do not cover.

In the Neural field experiment, one aspect that showed interesting results was with the DMD method combined with the embedded time delay technique was applied to data affected by Gaussian white noise. Although there are methods developed to correct the bias of the DMD in computing the eigenvalues when in presence of noise in the data in [8] and [2], the embedded time delay technique did reduce these effects, and we were able to recover the stable patterns that most characterized the dynamical system.

5.3 Future work

Next we list some of the strategies and studies which would be interesting to pursue.

Singular value thresholding

The DMD algorithm is based on the SVD of the data. In all the examples we tested, the criteria for the truncation of the singular values was based on (1.9). The importance of low-energy modes is difficult to estimate, and the simple truncation with this criteria may be not optimal.

There are other methods for determination of the singular values truncation threshold. In [12] is proposed an optimal truncation of singular values in the case of a data matrix of measurements that contain additive white noise.

Nonuniform Sampling

All the DMD applications we presented were based on a uniform step size sampled data. The work in [4] developed compressive sampling strategies for computing DMD from heavily sub-sampled data. These strategies may be useful to chose different sampling strategies as well.

Taking the Neural field application as an example: to improve the capture of the transient dynamics, a more dense distribution of measurements at the initial states of the system, and more sparse as it evolves to a steady state, may potentially allow us to collect richer data, while keeping a lower overall data storage. A Chebyshev polynomial to select the sample points, for example, may be an adequate candidate for this problem.

Multi-resolution Dynamic Mode Decomposition

The multi-resolution DMD is a technique inspired on the idea of wavelets, this method is structured as a recursive computation of DMD to remove low frequency features from a given collection of snapshots. The recursive nature of the method, and the linear properties of the DMD decomposition allows to apply DMD to progressively extract its modes to shorter snapshot sampling windows. Refer to [26, 27] for more on this method.

Extended and Kernel Methods

To expand on the possible alternative sets of observables to approximate the Koopman operator, [26, 27] proposes techniques called Extended DMD and Kernel DMD. The core idea is to choose a large and sufficiently diverse set of candidates with the expectation that it will include enough features for an accurate reconstruction of the Koopman operator.

Chaotic systems

Capitalizing on the work in [24] on the embedded time delays, [3] presents a data-driven decomposition of chaos as an intermittenly forced linear system.

Effect of noise

One result which may be interesting to pursue is the noise reduction on the data when DMD was used with the data matrix augmented with the time delay vectors as observables. Even if not a dedicated method for noise filtering, it is, nevertheless, a useful side-effect.

Bibliography

[1] Ali, A., Frøysa, H. G., Avlesen, H., and Alendal, G. (2016). Simulating spatial and temporal varying co2 signals from sources at the seafloor to help designing risk-based monitoring programs. *Journal of Geophysical Research: Oceans*, 121(1):745–757. 2, 46

[2] Askham, T. and Kutz, J. N. (2017). Variable projection methods for an optimized dynamic mode decomposition. *ArXiv e-prints*. 62, 72

[3] Brunton, S., Brunton, B., L. Proctor, J., Kaiser, E., and Nathan Kutz, J. (2016). Chaos as an intermittently forced linear system. *Nature Communications*, 8. 37, 72, 74

[4] Brunton, S. L., Proctor, J. L., and Kutz, J. N. (2013). Compressive sampling and dynamic mode decomposition. *ArXiv e-prints*. 73

[5] Budišić, M., Mohr, R., and Mezic, I. (2012). Applied koopmanism. *Chaos (Woodbury, N.Y.)*, 22:047510. 9, 11

[6] Chen, K. K., Tu, J. H., and Rowley, C. W. (2012). Variants of dynamic mode decomposition: Boundary condition, koopman, and fourier analyses. *Journal of Nonlinear Science*, 22(6):887–915. 9

[7] Davies, A. M. and Furnes, G. K. (1980). Observed and computed m2 tidal currents in the north sea. *Journal of Physical Oceanography*, 10(2):237–257. 59

[8] Dawson, S. T. M., Hemati, M. S., Williams, M. O., and Rowley, C. W. (2016). Characterizing and correcting for the effect of sensor noise in the dynamic mode decomposition. *Experiments in Fluids*, 57(3):42. 62, 63, 72

[9] Du, J., Fang, F., Pain, C., Navon, I., Zhu, J., and Ham, D. (2013). Pod reduced-order unstructured mesh modeling applied to 2d and 3d fluid flow. *Computers and Math-*

ematics with Applications, 65(3):362 – 379. Efficient Numerical Methods for Scientific Applications.

[10] Evans, L. C. (1999). Partial differential equations and monge-kantorovich mass transfer (surveypaper. In *Current Developments in Mathematics, 1997, International Press*. 5, 83

[11] Friedman, B. (1962). *Principles and Techniques of Applied Mathematics*. Dover. 81

[12] Gavish, M. and Donoho, D. L. (2014). The optimal hard threshold for singular values is $4/\sqrt{3}$. *IEEE Transactions on Information Theory*, 60(8):5040–5053. 73

[13] H. Tu, J., Rowley, C., Luchtenburg, D., Brunton, S., and Nathan Kutz, J. (2013). On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1. 22, 23, 27, 28, 29, 32

[14] Koopman, B. O. (1931). Hamiltonian systems and transformation in hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318. 9

[15] Ly, H. V. and Tran, H. T. (2002). Proper orthogonal decomposition for flow calculations and optimal control in a horizontal cvd reactor. *Quarterly of Applied Mathematics*, 60(4):631–656. 6

[16] Mezic, I. (2005). Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41(1):309–325. 13

[17] Mezic, I. (2017). Koopman operator spectrum and data analysis. 37

[18] Nathan Kutz, J., Brunton, S., Brunton, B., and L. Proctor, J. (2016). *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*. 4, 9, 22, 81

[19] Nordgren, E. A. (1978). Composition operators on hilbert spaces. In Bachar, J. M. and Hadwin, D. W., editors, *Hilbert Space Operators*, pages 37–63, Berlin, Heidelberg. Springer Berlin Heidelberg. 9

[20] Rowley, C. W., Mezic, I., Bagheri, S., Schlatter, P., and Henningson, D. S. (2009). Spectral analysis of nonlinear flows. *Journal of Fluid Mechanics*, 641:115–127. 4, 9, 11, 13, 23, 24, 26

[21] Ruhe, A. (1984). Rational krylov sequence methods for eigenvalue computation. *Linear Algebra and its Applications*, 58:391 – 405. 24

[22] Schmid, P. J. (2010). Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28. 2, 24, 26

[23] Sirovich, L. (1987). Turbulence and the dynamics of coherent structures. i - coherent structures. ii - symmetries and transformations. iii - dynamics and scaling. 45. 4

[24] Takens, F. (2006). *Detecting Strange Attractors in Turbulence. Lecture Notes in Mathematics*, volume 898, pages 366–381. 29, 32, 72, 74

[25] Trefethen, L. N. and Bau, D. (1997). *Numerical Linear Algebra*. SIAM: Society for Industrial and Applied Mathematics. 24, 26, 79, 81

[26] Williams, M., Rowley, C., and G. Kevrekidis, I. (2014). A kernel-based approach to data-driven koopman spectral analysis. 73, 74

[27] Williams, M. O., Kevrekidis, I. G., and Rowley, C. W. (2015). A data–driven approximation of the koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25(6):1307–1346. 72, 73, 74

[28] Wyller, J., Blomquist, P., and Einevoll, G. T. (2007). Turing instability and pattern formation in a two-population neuronal network model. *Physica D: Nonlinear Phenomena*, 225(1):75 – 93. 2, 46, 61, 62

Appendix A

SVD

Definitions from [25].

A.1 Description and Definition

The Singular Value Decomposition (SVD) is a matrix factorization, applicable to both complex and real matrices.

$$A = U\Sigma V^*, \tag{A.1}$$

where the matrix $A \in \mathbb{C}^{n\times m}$ has a decomposition in which $U \in \mathbb{C}^{n\times n}$ is unitary, $V \in \mathbb{C}^{m\times m}$ is unitary, and $\Sigma \in \mathbb{R}^{n\times m}$ is diagonal.

- $U = \{u_i\}$, where u_i are the columns defined as *left singular vectors*, with r linearly independent and orthonormal vectors in space $\mathbb{C}^n$;
- $V = \{v_i\}$, where v_i are the columns defined as *right singular vectors*, with r linearly independent and orthonormal vectors in space $\mathbb{C}^m$; and
- $\Sigma = diag(\sigma_i)$, whose entries σ_i, defined as *singular values* are nonnegative and nonincreasing in their ordering, that is $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \geq 0$.

A.2 Existence and Uniqueness

Every matrix has a unique SVD, as proven in [25] in the following theorem:

Theorem A.2.1. *Every matrix $A \in \mathbb{C}^{m\times n}$ has a singular value decomposition (A.1). Furthermore, the singular values $\{\sigma_i\}$ are uniquely determined, and, if A is square and the*

σ_j are distinct, the left and right singular vectors $\{u_j\}$ and $\{v_j\}$ are uniquely determined up to complex signs.

A.3 Matrix Properties

- The rank of A is r, the number of nonzero singular values.
- $range(A) = \{u_1, \ldots, u_r\}$ and $null(A) = \{v_{r+1}, \ldots, v_n\}$.
- $\|A\|_2 = \sigma_1$ and $\|A\|_F = \sqrt{\sigma_1^2 + \cdots + \sigma_r^2}$
- The nonzero singular values of A are the square roots of the nonzero eigenvalues of A^*A or AA^*.(These matrices have the same nonzero eigenvalues.)
- If $A = A^*$, then the singular values of A are the absolute values of the eigenvalues of A.
- For $A \in \mathbb{C}^{m\times m}$, $|det(A)| = \prod_{i=1}^{m} \sigma_i$

Appendix B

Eigenvalue Problem

Definitions from [11, 18, 25].

B.1 Definitions

1. Let the square matrix $A \in \mathbb{C}^{m\times m}$ be a linear transformation on a subspace S of $\mathbb{C}^m$. A nonzero vector $x \in \mathbb{C}^m$ is an *eigenvector* of A and $\lambda \in \mathbb{C}$ is the corresponding *eigenvalue* if

$$Ax = \lambda x. \tag{B.1}$$

   The set of all eigenvalues of the matrix A is the *spectrum* of A;

2. The characteristic polynomial of $A \in \mathbb{C}^{m\times m}$ is the polynomial of degree m defined as

$$p_A(z) = det(zI - A). \tag{B.2}$$

   λ is an eigenvalue of A if and only if $p_A(\lambda) = 0$.

3. Algebraic multiplicity of an eigenvalue λ of A is the multiplicity as a root of $p_A(z) = (z - \lambda_1)(z - \lambda_2)\dots(z - \lambda_m)$;

   If $A \in \mathbb{C}^{m\times m}$, then A has m eigenvalues, counted with algebraic multiplicity. In particular, if the roots of p_A are simple (algebraic multiplicity of 1), then A has m distinct eigenvalues;

4. Similarity Transformations: If X is nonsingular, then A and $X^{-1}AX$ have the same characteristic polynomial, eigenvalues, and algebraic multiplicities.

5. Unitary Diagonalization: In the cases when matrix $A \in \mathbb{C}^{m\times m}$ not only have m linearly independent eigenvectors, but are also orthogonal, then A is unitarily diagonizable,*i.e.*, there exists a unitary Q, such that

$$A = Q\Lambda Q^*; \tag{B.3}$$

6. A hermitian matrix ($A = A^*$) is unitarily diagonizable, and its eigenvalues are real;

7. A matrix is unitarily diagonizable if and only if it is normal ($A^*A = AA^*$).

B.2 First order ODE

The analytical solution of a system of continuous first order linear differential equations

$$\dot{x} = Ax, \tag{B.4}$$

is of the form,

$$x(t) = e^{At}x_0 = \sum_j \boldsymbol{q}_j e^{\lambda_j t}\boldsymbol{q}_j^{-1}x_0, \tag{B.5}$$

where x_0 is the initial condition for the vector $x(t=0)$, $A \in \mathbb{R}^{n\times n}$ a square matrix, and λ_j and $\boldsymbol{q}_j$ the eigenvalues and eigenvectors of A.

Stability

Let the eigenvalue $\lambda = a + ib$, then, $e^{\lambda t} = e^{at}\left[\cos(bt) + i\sin(bt)\right] \leq e^{at}$.

$$\text{As } t \to +\infty, \begin{cases} a < 0, & e^{at} < M \ \text{ (stable)} \\ a > 0, & e^{at} \to +\infty \ \text{ (unstable).} \end{cases}$$

Discrete Case

The solution to

$$x_{k+1} = \tilde{A}x_k, \tag{B.6}$$

where $\tilde{A} = e^{At}$, is

$$x_k = Q\tilde{\Lambda}^k Q^{-1}x_0. \tag{B.7}$$

Stability

$$\text{As } k \to +\infty, \begin{cases} \tilde{\Lambda} < 1, & x_k < M \ \text{ (stable)} \\ \tilde{\Lambda} > 1, & x_k \to +\infty \ \text{ (unstable).} \end{cases}$$

Appendix C

Functional analysis

Definitions from [10].

C.1 Banach Spaces

Let X denote a linear space.

Definition: A mapping $\|\cdot\| : X \to [0, \infty)$ is a norm if

1. $\|u + v\| \leq \|u\| + \|v\| \, \forall u, v \in X$,
2. $\|\lambda u\| = |\lambda| \, \|u\| \, , \forall u \in X, \lambda \in \mathbb{R}$,
3. $\|u\| = 0 \iff u = 0$.

Definitions:

1. A sequence $\{u_k\}_{k=1}^{\infty} \subset X$ converges to $u \in X$, that is $u_k \to u$, if
$$\lim_{k \to \infty} \|u_k - u_l\| = 0.$$
2. A sequence $\{u_k\}_{k=1}^{\infty} \subset X$ is called a a Cauchy sequence provided for each $\epsilon > 0$ there exists $N > 0$ such that
$$\lim_{k \to \infty} \|u_k - u_l\| < \epsilon \ \ \forall k, l \geq N.$$
3. X is complete if each Cauchy sequence in X converges; that is, whenever $\{u_k\}_{k=1}^{\infty}$ is a Cauchy sequence, there exists $u \in X$ such that $\{u_k\}_{k=1}^{\infty}$ converges to u.
4. A Banach space X is a complete, normed linear space.

C.2 Hilbert spaces

Let $\mathcal{H}$ be a real linear space.

Definition: A mapping $\langle \cdot, \cdot \rangle : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$ is called an inner product if

1. $\langle u, v \rangle = \langle v, u \rangle \, \forall u, v \in \mathcal{H}$,

2. the mapping $u \rightarrowtail \langle u, v \rangle$ is linear for each $v \in \mathcal{H}$,

3. $\langle u, u \rangle \geq 0, \forall u \in \mathcal{H}$,

4. $\langle u, u \rangle = 0 \iff u = 0$.

Definition: If $\langle \cdot, \cdot \rangle$ is an inner product, the associated norm is

$$\|u\| := \langle u, u \rangle^{1/2}, u \in \mathcal{H}.$$

Definition: A Hilbert space $\mathcal{H}$ is a Banach space endowed with an inner product which generates the norm.

For example, the space $L^2(\Omega)$ is a Hilbert space, with

$$\langle f, g \rangle = \int_\Omega f g dx.$$

Definitions:

1. Two elements $u, v \in \mathcal{H}$ are orthogonal if $\langle u, v \rangle = 0$.

2. A countable basis $\{w_k\}_{k=1}^\infty \subset \mathcal{H}$ is orthonormal if

$$\begin{cases} \langle w_k, w_l \rangle = 0, k \neq l \\ \|w_k\| = 1. \end{cases}$$

If $u \in \mathcal{H}$ and $\{w_k\}_{k=1}^\infty \subset \mathcal{H}$ is an orthonormal basis, we can write

$$u = \sum_{k=1}^\infty \langle u, w_k \rangle \, w_k.$$

Definition: If S is a subspace of $\mathcal{H}$, $S^\perp = \{u \in \mathcal{H} | \, \langle u, v \rangle = 0, \forall v \in \mathcal{H}\}$ is the subspace orthogonal to S.

C.3 Bounded linear operators

Let X and Y be real Banach spaces.

Definitions:

1. A mapping $A : X \to Y$ is a linear operator provided

$$A[\lambda u + \mu v] = \lambda Au + \mu Av, \quad \forall u, v \in X, \lambda, \mu \in \mathbb{R}.$$

2. The range of A is $R(A) := \{v \in Y | v = Au, \text{ for some } u \in X\}$ and the null space of A is $\mathcal{N}(A) := \{u \in X | Au = 0\}$.

Definition: A linear operator $A : X \to Y$ is bounded and continuous if

$$\|A\| < \infty.$$

Definitions:

1. If $A : \mathcal{H} \to \mathcal{H}$ is a bounded, linear operator, its adjoint $A^* : \mathcal{H} \to \mathcal{H}$ satisfies

$$\langle Au, v \rangle = \langle u, A^* v \rangle, \ \forall u, v \in \mathcal{H}.$$

2. A is symmetric if $A^* = A$.

Appendix D

Code

D.1 Main functions

```
function [Phi lambda] = computeDMD(Data,r)
% Compute DMD - Code adapted from JKutz, S.Brunton, B.Brunton,
% and J.Proctor (Dynamic Mode Decomposition)
        X = Data(:,1:end-1);
        X2 = Data(:,2:end);
        [U,S,V] = svd(X,'econ');
% Compute DMD (Phi are eigenvectors)
        U = U(:,1:r);
        S = S(1:r,1:r);
        V = V(:,1:r);
        Atilde = U'*X2*V*pinv(S);
        [W,eigs] = eig(Atilde);
        Phi = X2*V*pinv(S)*W;
        lambda = diag(eigs);
```

```
function r = reduced_form(Cs,tolerance)
% Determine minimum rank r, s.t. sum S(r)/sum S(n) >= tolerance
        if nargin < 2
                tolerance = .99;
        end
        S=svd(Cs);
```

```
        denominator = sum (S.^2);
        epson_E = 0;
        r=0;
        while(epson_E < tolerance)
                r = r + 1;
                epson_E = sum(S(1:r).^2)/denominator;
        end
```

```
function y_dmd=time_series(Phi,lambda,z0,t)
% reconstruction of the dynamical system from eigenvalues (lambda)
% and eigenvectors (phi), with z0=Phi\x0, and t the vector time
% x(t)=Phi*exp(omega*t)*z0
        dt=t(2)-t(1); % time step
% frequency in continuos space
        omega=log(lambda)/dt;
        r_ord=length(Phi(1,:));
% Reconstruction of time series from eigenpairs
        time_dynamics=zeros(r_ord,length(t));
        for iter=1:length(t)
                time_dynamics(:,iter)=z0.*(exp(omega*t(iter)));
        end
        y_dmd=Phi*time_dynamics;
```

```
function error=relError(y_approx,y,t)
        m=size(y,1);
        n=length(t)
        difference = real(y_approx-y);
        error=[];
        for iter = 1:length(t)
                error(iter)=norm(difference(:,iter))/norm(y(:,iter));
        end
```

```
function H=timeDelay(X,p)
% Construction of the Hankel matrix for the embedded time delay.
```

```
% X is the snapshot data matrix, and p the number of time delays.
        [n,m]=size(X);
        H=[];
        for j = 1:p
                H = [H ; X(:,j:j-p+end)];
        end
```

D.2 Code for Examples

```
%% Example 1
mu = -.2; lambda = -.5; % Parameters
t=0:0.001:1000; % Time
% initial conditions
x0 = [1.5; -1]

% numerical solution
[t,x]=ode45(@(t,x)[mu*x(1);lambda*(x(2)-x(1)^2)], t, x0);

% Koopman linear operator
y0=[x0; x0(1).^2]; % initial condition
A = [mu 0 0; 0 lambda -lambda; 0 0 2*mu];
[t,y] = ode45(@(t,y) A*y,tspan, y0); % Solution

% DMD
y0 = [x0; x0(1).^2]; % initial condition
yc = [x(:,1),x(:,2),x(:,1).^2];
r = reduced_form(yc);
[Phi,lambda] = computeDMD(y,r);
```

```
%% Example 2 - Logistic mapping
% r = [ <(r-1)/r oscillatory>,<4 attractors>,<chaotic>]
r=[3 3.56995 3.8];
```

```
x(1:length(r),1) = 0.5 % Initial condition
Tf=35; % Final time
T=0:1:Tf; % Time
dt=T(2)-T(1);
% Constructing a matrix X with all different parameters 'r'
for iter=2:Tf+1
        x(:,iter)=...
                r'.*(x(:,iter-1).*(ones(length(r),1)-x(:,iter-1)));
end
mu=1; % Select logistic curve (position in vector r)

%% Koopman linear operator
mu_par=r(mu);
% Koopman operator truncated at x_7
K=diag([mu_par mu_par^2 mu_par^3 mu_par^4 mu_par^5 mu_par^6 mu_par
    ^7])+...
diag([-mu_par -2*mu_par^2 -3*mu_par^3 -4*mu_par^4 -5*mu_par^5 -6*
    mu_par^6],1)+...
diag([0 mu_par^2 3*mu_par^3 6*mu_par^4 10*mu_par^5],2)+...
diag([0 0 mu_par^3 -4*mu_par^4],3);


% Construct the observables matrix
% g(x) = {x x^2 x^3 x^4 x^5 x^6 x^7 ...}
ord=10; % order of polynomial
g=zeros(ord,length(x(1,:)),length(r));
for iter=1:length(r)
        g(1,:,iter)=x(iter,:);
        for order=2:ord
                g(1:order,:,iter)=...
                        [g(1:(order-1),:,iter);x(iter,:).^order];
        end
```

```
        end
        g0=g(:,1,mu); % Observables initial condition
        %% DMD
        r_ord = reduced_form(g(:,:,mu));
        [Phi,lambda] = computeDMD(g(:,:,mu),r_ord);
        z0=Phi\g0;
        % reconstruct the linear model
        y_dmd=time_series(Phi,lambda,z0,T);
```

```
% Example 3 - Van der Pol
        mu = 2.5; % Parameter
        t = 0:0.01:30; % time discretization
        y0 = [0;4];% Initial condition
        % Numerical solution - Reference
        [t,y] = ode45('rhs_vdPol',t,y0,[],mu);

        % Funtion for the rhs of vanderpol
        function rhs = rhs_vdPol(t,x,dummy,mu)
                rhs=[x(2) ; mu*(1-x(1)^2)*x(2)-x(1)];
        %% DMD
        % Construct the observables matrix (based on the
        % determined koopman operator)
        g=[]; % initializing observables
        p=14; % order of polynomial
        for iter=1:p
                n=0;
                for m=iter:-1:0
                        g=[g;y_1.^m.*y_2.^n];
                        n=n+1;
                end
        end
        g1=g(:,1); % Initial condition
        r_g = reduced_form(g);
```

```
25        % determine eigenpairs from DMD
26        [Phi_g,lambda_g] = computeDMD(g,r_g);
27        z0_g = Phi_g\g1;
28        % reconstruct linear model
29        y_dmd_g =time_series(Phi_g,lambda_g,z0_g,t);
30
31        % Embedded time delay
32        p=100; % number of time delays
33        H=timeDelay(y',p);
34        r = reduced_form(H,.999); % tolerance can be adjusted
35        [Phi,lambda] = computeDMD(H,r);
36        H1=H(:,1); % Initial condition
37        z0 = Phi\H1;
38        y_dmd =time_series(Phi,lambda,z0,t); % reconstruct linear model
```

```
1  % Example 4
2         %Burgers' equation
3         %u_t +uu_x-eu_xx=0
4         L=20; n=512; % L is the length and n the number of discrete points
5         x=linspace(-L/2,L/2,n+1); x=x(1:n); % spatial discretization
6         k=fftshift((2*pi/L)*[-n/2:n/2-1].'); % fouries modes scaled from L
7         steps=20; % time discretization
8         t=linspace(0,2*pi,steps+1); dt=t(2)-t(1);
9         %Parameters
10        [X,T]=meshgrid(x,t);
11        epsilon=.1; % constant
12        % initial condition
13        u0=sech(x);
14        v0=exp(-(epsilon/2)*atan(sinh(x)));
15        % Solving Burgers equation with FFT
16        [t,u]=ode45(@(t,u)rhs_burgers(t,u,k,epsilon),t,u0);
17        % Solving Heat equation with FFT
18        [t,v]=ode45(@(t,v)rhs_heat(t,v,k,epsilon),t,v0);
```

```
19
20      %% DMD
21      r = reduced_form(u,.999);
22      %Determine DMD eigenfunctions and eigenvalues
23      [Phi,lambda] = computeDMD(u',r);
24      z0=Phi\u0.';
25      % reconstruct the linear model
26      u_dmd=time_series(Phi,lambda,z0,t);
27
28      function du_dt = rhs_burgers(t,u,k,epsilon)
29      % function to determine rhs of Burgers equation using FFT
30              u_hat=fft(u);
31              du_hat=1i*k.*u_hat;
32              ddu_hat=-(k.^2).*u_hat;
33              du=ifft(du_hat);
34              ddu=ifft(ddu_hat);
35              du_dt= -u.*du+epsilon*ddu;
36      end
37      function dvdt=rhs_heat(t,v,k,epsilon)
38      % function to determine rhs of the heat equation using FFT
39              v_hat=fft(v);
40              ddv_hat=-(k.^2).*v_hat;
41              ddv=ifft(ddv_hat);
42              dvdt=epsilon*ddv;
43      end
```