Department of Mathematics
TUM School of Computation, Information and Technology
Technical University of Munich

# Fast Eigensolvers for Koopman Operator Approximation

Saksham Malhotra

Thesis for the attainment of the academic degree

**Master of Science**

at the TUM School of Computation, Information and Technology of the Technical University of Munich

**Supervisor:**
Prof. Dr. Hans-Joachim Bungartz

**Advisor:**
Dr. Felix Dietrich

**Submitted:**
Munich, 10. Oktober 2023

I hereby declare that this thesis is entirely the result of my own work except where otherwise indicated. I have only used the resources given in the list of references.

Munich, 10. Oktober 2023 [redacted]

## Zusammenfassung

Der Koopman-Operator ist ein beliebter Rahmen für die Untersuchung dynamischer Systeme. Datengesteuerte numerische Algorithmen, die auf der dynamischen Moduszerlegung basieren, approximieren Eigenfunktionen des Operators, die für die Systemanalyse nützlich sind. Die Eigenfunktionen des Koopman-Operators bilden bei punktweiser Multiplikation eine abelsche Gruppe. Das Hauptziel dieser Arbeit ist die Entwicklung von Eigensolvern für Koopman-Operator-Approximationen, die diese algebraische Struktur der Koopman-Eigenfunktionen ausnutzen. Ein effizienter Eigenlöser-Algorithmus für den Koopman-Operator, der die Eigenfunktionen erweitert, würde die numerische Analyse von hochdimensionalen nichtlinearen dynamischen Systemen verbessern. Wir stellen einen theoretischen Rahmen mit einem Fehlermaß und Schranken vor, der diese multiplikative Eigenschaft zur Erweiterung von Eigenfunktionen nutzt. Wir entwickeln einen iterativen Algorithmus zur Ermittlung von Eigenwerten und Eigenvektoren der Koopman-Matrix und verwenden diese zusammen mit den Fehlergrenzen zur Approximation und Erweiterung von Eigenfunktionen. Wir demonstrieren die Effizienz des Eigenlösers, indem wir ihn auf nichtlineare Systeme anwenden.

## Abstract

The Koopman operator is a popular framework for the study of dynamical systems. Data-driven numerical algorithms based on dynamic mode decomposition approximate eigenfunctions of the operator that are useful for system analysis. The eigenfunctions of the Koopman operator form an Abelian group under pointwise multiplication. The main goal of this thesis is to develop eigensolvers for Koopman operator approximations that exploit this algebraic structure of the Koopman eigenfunctions. An efficient eigensolver algorithm for the Koopman operator that extends eigenfunctions would improve the numerical analysis of high-dimensional non-linear dynamical systems. We present a theoretical framework with an error measure and bounds, using this multiplicative property to extend eigenfunctions. We develop an iterative algorithm for finding eigenvalues and eigenvectors of the Koopman matrix and use them along with the error bounds to approximate and extend eigenfunctions. We demonstrate the efficiency of the eigensolver by applying it to non-linear systems.

# Contents

# 1 Introduction

The Koopman framework has become increasingly popular for studying dynamical systems. It was first introduced by Koopman and Von Neumann, who showed that each system can be associated with a linear operator [Koo31; Neu32]. The operator, when applied to complex-valued functions of the system states, results in the functions evaluated at a future state. Therefore, the dynamics can be studied in the function space instead of the state space. Thus, the Koopman operator framework is understood as studying the *dynamics of observables* instead of the dynamics of states. The Koopman operator is formulated as an operator that advances the observable's value. Moreover, the linear nature of the operator makes it tractable for finite-dimensional matrix approximations. The Koopman framework addresses the challenge of dealing with non-linear dynamical systems. The classical geometric perspective allows analysis techniques to only be applied in the neighbourhood of fixed points, periodic orbits, and attractors of these non-linear systems by using local linearisations [Gle94]. The Koopman framework presents a global analysis of the system, thereby allowing for prediction and control away from fixed points and periodic orbits.

Several techniques have been developed to approximate the Koopman operator. These techniques either rely on analytical methods that make use of the *Koopman PDE* and solve it using Laurent and Taylor series or use *data-driven approximations* to compute a finite-dimensional matrix representation. Data-driven approximations have become a reliable tool for studying complex dynamical systems where the governing equations of the system are not known, or the available models are oversimplified to describe complex behaviour. These data-driven techniques only require the sampled states of the system to make the approximation without explicit system equations. The most common techniques are based on *Dynamic mode decomposition* and approximate a finite-dimensional Koopman matrix by solving a least-squares problem [ROW+09; Tu+14]. Several extensions have been proposed, such as the *Extended Dynamic mode decomposition* that formulates the finite-dimensional approximation as acting on a finite set of basis functions called the *dictionary basis* [WKR15]. Other techniques similar to EDMD have been developed for measure-preserving dynamical systems [Col23], and deep-learning-based approaches for *learning* eigenfunctions have also been proposed [LKB18]. The data-only approach has also led to applications of the operator framework to complex algorithms formulated as dynamical systems [DTK20; Man+20].

The spectral properties of the operator play an essential role in system analysis. The development of Koopman mode analysis used for studying the system's evolution has applications in multiple areas [Eis+10; Mez13]. The level sets of eigenfunctions have been used to extract and analyse invariant and periodic structures in the state space [BM12]. Additionally, stability analysis techniques have been developed that use the Koopman spectrum [MM13]. Therefore, the efficient computation of the operator's spectrum through finite-dimensional matrix approximations is important.

As the Koopman operator is infinite-dimensional, it can have an infinite number of eigenfunctions. The multiplicative property of the Koopman eigenfunctions structures a subset of this set by stating that the eigenfunctions of the Koopman operator form an Abelian semigroup under pointwise multiplication. The current techniques compute eigenfunctions by using general eigensolver algorithms on the finite-dimensional matrix to obtain the eigenvectors and using these eigenvectors to find eigenfunctions in a finite basis of functions. This approach does not use the multiplicative property to extend the set of eigenfunctions or develop efficient eigensolver algorithms that reduce the computation effort. The need for efficient eigensolver algorithms for Koopman approximations becomes clear when we consider that high-dimensional complex systems require a high-dimensional matrix approximation to capture the dynamics accurately. Additionally, extending eigenfunctions enables us to discover eigenfunctions that do not lie in the span of the finite dictionary basis used for the approximation, thereby capturing the non-linear dynamics more closely.

This work aims to develop algorithms for efficient computation and extension of eigenfunctions of the Koopman operator based on their multiplicative property. We define an error measure for the extended eigenfunctions, present an error analysis and design an iterative eigensolver algorithm. The remainder of the thesis is organised as follows: Chapter 2 lays down the mathematical background for Koopman operator theory and its numerical approximation, and algorithms used for general eigenvalue computation of matrices. Section 2.1 defines the Koopman operator and its spectral properties, section 2.2 introduces two numerical algorithms (DMD and EDMD) for Koopman operator approximation and section 2.3 gives a brief overview of algorithms used for eigenvalue and eigenvector computation of general matrices. Chapter 3 develops the theory and methods for extending eigenfunctions, defining an error measure– *trajectory error* for eigenfunctions, formulating error bounds and developing an iterative algorithm for Koopman approximations. Section 3.1 defines the extended eigenfunctions based on the multiplicative property and defines a measure of error for extended eigenfunctions. Section 3.2 presents the error analysis and derives upper bounds for the trajectory error. Section 3.3 presents an algorithm for extending eigenfunctions for a single eigenpair based on the error bounds and applies the error analysis and algorithm to continuous and discrete linear systems. Finally, Section 3.4 develops an iterative algorithm for Koopman operators based on the power method, Arnoldi method and trajectory error bounds. The algorithm is then applied to two different non-linear systems.

# 2 Mathematical Foundations

This chapter briefly overviews the Koopman operator theory and general eigensolvers. We present Koopman operator theory and its spectral properties. Then, we describe two principal algorithms used for its numerical approximation. We also discuss the main algorithms used for computing eigenvectors and eigenvalues of general matrices.

## 2.1 Koopman operator theory

### 2.1.1 The Koopman operator

We define the Koopman operator for discrete and continuous dynamical systems [BMM12].

**Definition 1.** Given the state space $M$ and a discrete dynamical system $x_{n+1} = T(x_n)$ where $T : M \longrightarrow M$, $M \subseteq \mathbb{R}^D$ and $\mathcal{F}$ is the space of observables such that $f \in \mathcal{F}, f : M \longrightarrow \mathbb{C}$, the Koopman operator, $\mathcal{K} : \mathcal{F} \longrightarrow \mathcal{F}$ is defined as

$$\mathcal{K}f(x_n) = f(T(x_n)),\ n \in \mathbb{N}. \tag{2.1}$$

**Definition 2.** Given the state space $M$ and a continuous dynamical system $\dot{x} = T(x)$ where $T : M \longrightarrow M$, $M \subseteq \mathbb{R}^d$ and $\mathcal{F}$ is the space of observables such that $f \in \mathcal{F}, f : M \longrightarrow \mathbb{C}$, the Koopman semigroup, $\{\mathcal{K}^t\}_{t\in\mathbb{R}^+}$, $\mathcal{K}^t : \mathcal{F} \longrightarrow \mathcal{F}$ is defined as

$$\mathcal{K}^t f(x) = f(T^t(x)),\ x \in M, \tag{2.2}$$

where $T^t : M \longrightarrow M$ is the flow of the system. The generator of the Koopman semigroup, $\mathcal{A}_{\mathcal{K}} : \mathcal{F} \longrightarrow \mathcal{F}$ is defined as

$$\mathcal{A}_{\mathcal{K}}f = \lim_{t\longrightarrow 0} \frac{\mathcal{K}^t f - f}{t}. \tag{2.3}$$

The vector field of the continuous dynamical system and the Koopman generator have the following relationship:

$$[\mathcal{A}_{\mathcal{K}}f](x) = \langle T, \nabla f(x)\rangle,\ \forall f \in \mathcal{F}. \tag{2.4}$$

The proof of equation 2.4 is given in Appendix A.2.1.

The Koopman operator transfers the dynamics from the state space to the observable/function space. In the observable space, the dynamics becomes linear. For continuous systems, the flow is induced by $\mathcal{K}^t$ and given by

$$\mathcal{K}^t = e^{\mathcal{A}_{\mathcal{K}}t},$$

where $\mathcal{A}_{\mathcal{K}}$ is the Koopman generator. Therefore, the application of $\mathcal{K}^t$ to an observable $f$ at state $x$ advances the value of the observable $f$ at state $T^t(x)$ for continuous systems. For discrete systems, the application of operator $\mathcal{K}$ to an observable $f$ at state $x_n$ advances the value of the observable $f$ at state $x_{n+1}$.

### 2.1.2 Spectral properties of the Koopman operator

$\mathcal{F}$ is a vector space, and the Koopman operator is a linear operator. Therefore, the spectral properties of the operator are critical for understanding the behaviour of the operator. Furthermore, eigenvalues and eigenfunctions are important for Koopman mode analysis.

We assume that $\mathcal{F}$ is a Banach space under some norm, and that $\mathcal{K}$ is a bounded, and hence continuous operator on this space [Row].

**Definition 3.** The eigenvalue $\lambda \in \mathbb{C}$ and eigenfunction $\phi : M \longrightarrow \mathbb{C}$ of Koopman operator $\mathcal{K}$ for a discrete system is defined by the following relation:

$$(\mathcal{K}\phi)(x_n) = \lambda\phi(x_n),\ n \in \mathbb{N}. \tag{2.5}$$

**Definition 4.** The eigenvalue $\lambda \in \mathbb{C}$ and eigenfunction $\phi : M \longrightarrow \mathbb{C}$ of generator $\mathcal{A}_{\mathcal{K}}$ for the Koopman semigroup, $\{\mathcal{K}^t\}_{t\in\mathbb{R}^+}$ for a continuous system is defined by the following relation:

$$(\mathcal{A}_{\mathcal{K}}\phi)(x) = \lambda\phi(x). \tag{2.6}$$

Then, the eigen equation for $\mathcal{K}^t, t \geq 0$ is

$$(\mathcal{K}^t\phi)(x) = e^{\lambda t}\phi(x). \tag{2.7}$$

The eigenfunctions are $\mathbb{C}$-valued observables on $M$ and the eigenpairs $(\lambda_i, \phi_i)$ depend on the state space $M$, dynamics $T$, and the function space $\mathcal{F}$.

The eigenpairs of the Koopman operator for both discrete and continuous systems have the multiplicative property described in Proposition 1.

**Proposition 1.** *(Multiplicative property of Koopman eigenpairs) $\mathcal{F}$ is a subset of all $\mathbb{C}$-valued functions on $M$ that forms a vector space.*

- *If $\phi_1, \phi_2 \in \mathcal{F}$ are eigenfunctions of $\mathcal{K}$ with eigenvalues $\lambda_1, \lambda_2$ then $\phi_1\phi_2$ is an eigenfunction of $\mathcal{K}$ with eigenvalue $\lambda_1\lambda_2$ i.e. set of eigen functions is closed under pointwise multiplication.*
- *If $p \in \mathbb{R}^+$ and $\phi$ is an eigenfunction with eigenvalue $\lambda$ then $\phi^p$ is an eigen function with eigenvalue $\lambda^p$.*
- *Set of eigenfunctions forms an Abelian semigroup with constant function that is one everywhere as identity.*

*Proof.* Assume that $\phi_1, \phi_2$ are eigenfunctions of $\mathcal{K}$ with eigenvalue K, then

$$\begin{aligned}(\mathcal{K}\phi_1\phi_2)(x) &= (\phi_1\phi_2)(Tx) = \phi_1(Tx)\phi_2(Tx)\\ &= (\mathcal{K}\phi_1)(x)(\mathcal{K}\phi_2)(x) = \lambda_1\phi_1(x)\lambda_2\phi_2(x)\\ &= \lambda_1\lambda_2(\phi_1\phi_2)(x).\end{aligned}$$

Let $\mathcal{K}\phi(x) = \lambda\phi(x)$, Then for $p \in \mathbb{Z}^+$,

$$\begin{aligned}(\mathcal{K}\phi^p)(x) &= \phi^p(Tx) = (\phi(Tx))^p\\ &= (\lambda\phi(x))^p = \lambda^p\phi^p(x).\end{aligned}$$

The constant function $\phi(x) = 1$ is a koopman eigenfunction with eigenvalue 1 and acts as the identity element. Combined with the closure property proved above, the set of eigenfunctions is an Abelian semigroup under pointwise multiplication of functions. □

Under the action of diffeomorphisms, the eigenfunctions of the Koopman operator of the transformed system are related to the Koopman eigenfunctions of the original system. Proposition 2 describes the relation formally.

**Proposition 2.** *(Koopman operator and diffeomorphisms) Given a diffeomorphism $h : M \longrightarrow N$. If $(\lambda, \phi)$ is a Koopman eigenpair of a continuous system, then $(\lambda, \phi \circ h^{-1})$ is a Koopman eigenpair of the transformed system.*

*Proof.* Let $\{\mathcal{K}^t\}_{t\in\mathbb{R}^+}$ be the Koopman semigroup of the system. Then $(\mathcal{K}^t\phi)(x) = \phi(T^t(x))\ \forall x \in M$ where $T^t : M \longrightarrow M$ is the flow of the system.

Let $\{\mathcal{K}_h^t\}_{t\in\mathbb{R}^+}$ be the Koopman semigroup of the transformed system and $H^t : N \longrightarrow N$ be the flow of the transformed system. Let $y = h(x),\ x \in M$. Then using $h(T^t(x)) = H^t(h(x))$ and the definition of the eigenfunction,

$$\begin{aligned}(\mathcal{K}_h^t(\phi \circ h^{-1}))(y) &= \phi(h^{-1}(H^t(y)))\\ &= \phi(T^t(x)) = \mathcal{K}^t\phi(x) = e^{\lambda t}\phi(x) = e^{\lambda t}\phi(h^{-1}(y))\\ &= e^{\lambda t}(\phi \circ h^{-1})(y),\ \forall y \in N.\end{aligned}$$

□

**Remark** Some special conditions on the dynamics can lead to the spectrum of the Koopman operator being constrained. For example, if $T$ is measure preserving with $\mathcal{F} = L^2(M,\mu)$, then all eigenvalues of $\mathcal{K}$ are on the unit circle. In addition, $\mathcal{K}$ is guaranteed to have a complete spectral decomposition when the system is measure-preserving.

### Koopman mode decomposition

Assuming a function $f \in \mathcal{F}$ is in the span of eigenfunctions $\{\phi_1, \ldots, \phi_n\}$ of the Koopman operator, then

$$f(x) = \sum_{i=1}^{n} c_i(f)\phi_i(x),$$

where $c_1, \ldots, c_n$ are constants that depend on $f$. The dynamics of $f \in \mathcal{F}$ (for a discrete system) is given by

$$\mathcal{K}^m f(x) = \sum_{i=1}^{n} c_i(f)\lambda_i^m\phi_i(x),\ m \in \mathbb{N}. \tag{2.8}$$

This can be extended to a vector valued observable $F : M \longrightarrow V \subset \mathbb{C}^K$:

$$F = \begin{bmatrix} f_1\\ \vdots\\ f_K\end{bmatrix}$$

$$f_j(x) = \sum_{i=1}^{n} c_i(f_j)\phi_i(x),\ j = 1, \ldots, K.$$

The dynamics in this case is given by

$$[\mathcal{K}^m F](x) = \sum_{i=1}^{n} \lambda_i^m\phi_i(x)\begin{bmatrix} c_i(f_1)\\ \vdots\\ c_i(f_K)\end{bmatrix} = \sum_{i=1}^{n} \lambda_i^m\phi_i(x)C_i(F),\ m \in \mathbb{N}. \tag{2.9}$$

Similarly, the dynamics in the case of a continuous system is given by

$$[\mathcal{K}^t F](x) = \sum_{i=1}^{n} e^{m\lambda_i t}\phi_i(x)\begin{bmatrix} c_i(f_1)\\ \vdots\\ c_i(f_K)\end{bmatrix} = \sum_{i=1}^{n} e^{m\lambda_i t}\phi_i(x)C_i(F),\ t \geq 0. \tag{2.10}$$

The quantity

$$C_i(F) = \begin{bmatrix} c_i(f_1)\\ \vdots\\ c_i(f_K)\end{bmatrix},$$

which is the projection of $F$ onto $\operatorname{span}\{\phi_i\}$ is called the *Koopman mode* of the observable $F$ corresponding to eigenfunction $\phi_i$. The Koopman modes are defined with respect to the eigenfunctions and not eigenvalues as an eigenvalue might have multiple linearly independent eigenfunctions associated with it.

The relationship between the state space and the observable space can be established using the *full-state observable*, $F(x) = x$. By assuming $x \in \operatorname{span}\{\phi_i\}$, we can reconstruct the evolution of $x$ for a discrete system:

$$\begin{aligned} x &= \sum_{i=1}^{n} C_i(x)\phi_i(x) \\ \implies T(x) &= \sum_{i=1}^{n} \lambda_i C_i(x)\phi_i(x). \end{aligned}$$

Therefore, the non-linear dynamics in the state space becomes linear in the observable space.

Now we consider the application of Koopman operator theory to discrete linear systems.

**Example 1.** (Koopman spectrum of a linear system) Consider the discrete linear system

$$x_{m+1} = Ax_m, \; m \in \mathbb{N}, \tag{2.11}$$

where $A : M \longrightarrow M$ is a linear map.

We assume that $A$ has a complete set of eigenvectors $\{v_1, \ldots, v_n\}$ with corresponding right eigenvalues $\{\lambda_1, \ldots, \lambda_n\}$, and left eigenvectors $\{w_1, \ldots, w_n\}$. Then $A^* w_i = \bar{\lambda}_i w_i$ where $A^*$ is the adjoint of $A$. Consider the observable $\phi_i(x) = \langle x, w_i \rangle$. Then for $i = 1, \ldots, n$:

$$\begin{aligned} [\mathcal{K}\phi_i](x) &= \phi_i(Ax) = \langle Ax, w_i \rangle = \langle x, A^* w_i \rangle \\ &= \langle x, \bar{\lambda}_i w_i \rangle = \lambda_i \langle x, w_i \rangle = \lambda_i \phi_i(x). \end{aligned}$$

Therefore, $(\lambda_i, \phi_i)$ is an eigenpair of the Koopman operator.

Consider the observable $F(x) = x$ and assume that $x \in \operatorname{span}\{v_i\}$. Then

$$\begin{aligned} [\mathcal{K}F](x) &= F(Ax) = Ax = A\sum_{i=0}^{n} \langle x, w_i \rangle v_i = \sum_{i=0}^{n} \phi_i(x)\lambda_i v_i \\ \implies [\mathcal{K}^m F](x) &= F(Ax) = x = \sum_{i=0}^{n} \lambda_i^m \phi_i(x) v_i, \; m \in \mathbb{N}. \end{aligned}$$

This implies that eigenvectors $v_i$ of linear map $A$ are the Koopman modes of the system corresponding to the eigenfunction $\phi_i$. This is not a complete set of eigenfunctions. More eigenfunctions can be generated using this set of eigenfunctions using Proposition 1.

## 2.2 Numerical algorithms for Koopman operator approximation

Eigenfunctions of the Koopman operator can be approximated using the explicit formulation, which relies on solving the PDE for the eigenfunction. Assuming continuous and differentiable dynamics, and using (2.4) with an eigenfunction $\phi$, we get

$$[\mathcal{A}_{\mathcal{K}} f](x) = \langle T, \nabla\phi(x) \rangle = \lambda\phi(x).$$

It is possible to solve this PDE for eigenfunctions using standard techniques such as Taylor or Laurent series. Methods have been proposed that use this PDE to approximate eigenfunctions [Bol21; KKB17].

Data-driven approaches for eigenfunction approximation rely on finding a finite-dimensional approximation of the Koopman operator. These approximation algorithms use samples generated from the system and do not require an explicit formulation of the dynamical system. This is particularly useful when the system behaviour can only be ascertained using a finite set of initial conditions.

As the Koopman operator operates on an infinite-dimensional function space, this poses challenges for computation. Finding a finite-dimensional approximation involves limiting the function space to an invariant subspace spanned by a finite set of functions. Then, the finite-dimensional approximation is the projection the Koopman operator onto this invariant subspace. The eigenfunctions of the Koopman operator can then also be computed using the spectrum of this finite-dimensional approximation. The standard approach used here is EDMD (Extended dynamic mode decomposition), which is based on its precursor DMD (Dynamic mode decomposition).

### 2.2.1 Dynamic mode decomposition (DMD)

Dynamic mode decomposition uses sampled data points to approximate the operator. It was first formulated using a trajectory– a sequence of observations $\{T^k x\}_{k\in\mathbb{N}}$ generated by the system starting from an initial point.

Given a vector valued observable $F \in \mathcal{F}^m$, we fix $r < \infty$ and define the Krylov subspace,

$$K_r = \operatorname{span}\{\mathcal{K}^j F\}_{j=0}^{r-1}. \tag{2.12}$$

We assume this is a linearly independent set, and define the projection $P_r : \mathcal{F}^m \longrightarrow K_r$ from space of vector-valued observables onto the finite-dimensional space $K_r$. Then, $\mathcal{K}|_{K_r} : K_r \longrightarrow \mathcal{F}^m$ and $P_r\mathcal{K}|_{K_r} : K_r \longrightarrow K_r$ is a finite-dimensional linear operator with matrix representation $A_r : \mathbb{C}^r \longrightarrow \mathbb{C}^r$ in the basis $\{\mathcal{K}^k F\}_{k=0}^{r-1}$.

If $(\lambda, v)$ is an eigenpair of $A_r$ where $v = (v_0, \dots, v_{r-1})^T \in \mathbb{C}^r$, then $\phi = \sum_{j=0}^{r-1} v_j [\mathcal{K}^j F]$ is an eigenfunction of $P_r\mathcal{K}|_{K_r}$ with eigenvalue $\lambda$:

$$\begin{aligned}
P_r\mathcal{K}|_{K_r}\phi &= \sum_{j=0}^{r-1} v_j P_r\mathcal{K}|_{K_r}[\mathcal{K}^j F] \\
&= \sum_{j=0}^{r-1} v_j K_r A_r e_j = K_r A_r v = \lambda K_r v \\
&= \lambda \sum_{j=0}^{r-1} v_j [\mathcal{K}^j F] = \lambda\phi.
\end{aligned}$$

Therefore, the problem of finding eigenfunctions of infinite-dimensional $\mathcal{K}$ is reduced to finding eigenpairs of finite-dimensional matrix $A_r$.

#### DMD for trajectory

The propagation of fixed observable F given a fixed point $x \in M$ is given by the sequence $\{b_j\}_{j=0}^{r}$ where $b_j = \mathcal{K}^j F(x) \in \mathbb{C}^m$. We define

$$B_r = [b_0, \dots, b_{r-1}]. \tag{2.13}$$

As number of points $r$ in the trajectory increases, the vectors $b_j$ become linearly dependent for some value of $r - 1 < \infty$. Then $b_r$ can be expressed as a linear combination of the columns of $B_r$:

$$b_r = \sum_{j=0}^{r-1} c_j b_j + \eta_r,$$

where $c_j$'s are chosen to minimize the residual $\eta_r$. This is equivalent to minimising the projection error of $b_r = \mathcal{K}^r F(x)$ onto the space $K_r = \operatorname{span}\{b_j\}_{j=0}^{r-1}$ and the finite-dimensional approximation defined above is

$$P_r\mathcal{K}|_{K_r}[\mathcal{K}_r F](x) = \sum_{j=0}^{r-1} c_j b_j = \sum_{j=0}^{r-1} c_j [\mathcal{K}^j F](x).$$

In the practical implementation of DMD, this is computed using the least squares approximation.

Using the matrix representation $A_r$ of $P_r\mathcal{K}|_{K_r}$ we get

$$\mathcal{K}B_r = B_rA_r + \eta_r e^T, \tag{2.14}$$

where $e = (0, \dots, 0, 1)^T$ and because of the structure of $B_r$,

$$A_r = \begin{bmatrix} 0 & 0 & \dots & c_0 \\ 1 & 0 & \dots & c_1 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & c_{r-1} \end{bmatrix} \tag{2.15}$$

is the companion matrix.

Eigenfunctions and eigenvalues can then be computed using eigendecomposition of $A_r = V^{-1}\Lambda V$. We define

$$E = B_rV^{-1}. \tag{2.16}$$

Then

$$\mathcal{K}E = E\Lambda + \eta_r e^T V^{-1}. \tag{2.17}$$

If $\left\|\eta_r e^T V^{-1}\right\|$ is small, $UE \approx E\Lambda$. Columns of E are called empirical Ritz vectors, which approximate $\phi_i(p)C_i(F)$ and diagonal entries of $\Lambda$ are the Ritz values, which approximate the corresponding eigenvalues. Proof is given by Rowley et al. [ROW+09]. The quantity $\phi_iC_i(F)$ is called the *DMD mode*.

The companion matrix in canonical basis is given by $A_r = (\mathcal{K}B_r)B_r^{\dagger}$. Therefore, if the trajectory $\{b_0, b_1, \dots, b_r\}$ is arranged in the data matrices $X$ and $Y$ given by

$$X = \begin{bmatrix} b_0 & \dots & b_{r-1} \end{bmatrix}$$
$$Y = \begin{bmatrix} b_1 & \dots & b_r. \end{bmatrix}$$

then

$$A = YX^{\dagger}, \tag{2.18}$$

where $\cdot^{\dagger}$ is the pseudoinverse of the matrix.

The practical implementation of the above algorithm tends to be ill-conditioned because $\mathcal{K}^j b_0$ converges to the eigen spaces corresponding to the eigenvalues with the largest magnitude. This results in the columns of $X$ becoming linearly dependent. Additionally, computing the eigendecompositon of matrix $A$ can be difficult as $A$ has $m^2$ elements, where $m$ can be large for high-dimensitonal state spaces. If $m << r$, then the matrix $A$ has rank $m$. Schmid et al. [SCH10] presents a robust implementation using SVD, which uses a rank deficient least squares problem using SVD. This computes a low-rank matrix $\tilde{A}$ with rank $m$ based on the non-zero singular values of $X$. We compute the reduced SVD, $X = U\Sigma W^*$. Then

$$\begin{aligned} A &= YW\Sigma^{-1}U^* \\ \implies U^*AU &\approx U^*YW\Sigma^{-1} = \tilde{A} \\ \implies \mathcal{K}U &\approx U\tilde{A}, \end{aligned}$$

where $\tilde{A} = U^*YW\Sigma^{-1}$.

$U^*AU = \tilde{A}$ implies that $\tilde{A}$ and $A$ have the same non-zero eigenvalues. Now using eigendecompisiton of $\tilde{A} = \tilde{V}^{-1}\tilde{\Lambda}\tilde{V}$, we get

$$\begin{aligned} \mathcal{K}U &\approx U\tilde{V}^{-1}\tilde{\Lambda}\tilde{V} \\ \implies \mathcal{K}U\tilde{V}^{-1} &\approx U\tilde{V}^{-1}\tilde{\Lambda} \\ \implies \mathcal{K}\tilde{E} &= \tilde{E}\tilde{\Lambda}, \end{aligned}$$

where

$$\tilde{E} = U\tilde{V}^{-1}. \tag{2.19}$$

Therefore, as described earlier, eigenvectors and eigenvalues of $\tilde{E}$ approximate the DMD modes and DMD eigenvalues of $\mathcal{K}$ respectively.

#### DMD for snapshot pairs

The method described above relies on trajectories starting from a fixed point. Tu et al. [Tu+14] present a more general method for DMD that uses data collected as a set of snapshot pairs rather than as a sequential time-series. The snapshot pairs are $\{(x_j, y_j)\}_{j=1}^r$, where $y_j = T(x_j)$ in case of a discrete system and $y_j = T^{\Delta t}(x_j)$ where $T^t$ is the flow and $\Delta t$ is the sampling frequency, in case of a continuous system. The matrices $X$ and $Y$ are defined as

$$\begin{aligned} X &= \begin{bmatrix} x_1 & \dots & x_r \end{bmatrix} \\ Y &= \begin{bmatrix} y_1 & \dots & y_r \end{bmatrix}. \end{aligned}$$

Now we define the matrix $A$ as

$$A = YX^{\dagger}. \tag{2.20}$$

We compute the SVD of $X = U\Sigma^{-1}W^*$ again and define $\tilde{A} = U^*YW\Sigma^{-1}$ and compute the eigendecomposition of $\tilde{A} = \tilde{V}^{-1}\tilde{\Lambda}\tilde{V}$. Then, the DMD modes and eigenvalues are given by eigenvectors and eigenvalues of $\tilde{E} = YW\Sigma^{-1}V^{-1}$. The matrix $\tilde{A}$ is called the *Koopman matrix*. It can be shown that eigenpairs of $\tilde{A}$ are eigenpairs of $A$. Tu et al. [Tu+14] shows the relationship between DMD modes computed using $\tilde{E} = UV^{-1}$ called the *projected DMD modes* and $\tilde{E} = YW\Sigma^{-1}V^{-1}$ called the *exact DMD modes.* Essentially projected DMD modes are the orthogonal projection of exact DMD modes onto the space spanned by the data vectors in $X$.

**Remark** DMD only gets the approximation of the projection of an observable at a point onto the eigen function and not the approximation of the eigen function value at the point.

### 2.2.2 Extended dynamic mode decomposition (EDMD)

Williams et al. [WKR15] developed an extension of DMD, that approximates Koopman eigenfunctions, eigenvalues and Koopman modes using a dataset of snapshot pairs and a finite dictionary of observables. This method is called *EDMD.*

Given snapshot pairs $\{(x_j, y_j)\}_{j=1}^r$, where $y_j = T(x_j)$ in case of a discrete system and $y_j = T^{\Delta t}(x_j)$ where $\Delta t$ is the sampling frequency, in case of a continuous system. For the description of EDMD, we stick to real-valued observables only. We define a dictionary basis $\mathcal{F}_d = \operatorname{span}\{\psi_i\}_{i=1}^d$, $\mathcal{F}_d \subset \mathcal{F}$ with observables $\psi_i : M \longrightarrow \mathbb{R}$. The dictionary vector is defined as

$$\Psi(x) = \begin{bmatrix} \psi_1(x) & \psi(x) \dots & \psi_d(x) \end{bmatrix} \in \mathbb{R}^d. \tag{2.21}$$

EDMD approximates a matrix representation, $K$ of the finite-dimensional approximation of the Koopman operator, $P_r\mathcal{K}|_{K_r}$ with respect to the dictionary basis $\{\psi_i\}$. This matrix representation is called the *Koopman matrix.* Consider a function $\varphi \in \mathcal{F}_d$ given by

$$\varphi = a^T\Psi, \tag{2.22}$$

where $a \in \mathbb{R}^d$. As $\mathcal{F}_d$ is generally not invariant with respect to $\mathcal{K}$, we get

$$(\mathcal{K}\varphi)(x) = \varphi(T(x)) = a^T\Psi(T(x)) = a^TK\Psi(x) + r(x), \tag{2.23}$$

where residual $r \in \mathcal{F}$ and $K \in \mathbb{R}^{d\times d}$. To minimize $r$ and obtain $K$ we solve a least squares problem:

$$\begin{aligned} a^TK &= \underset{a^TK\in\mathbb{R}^{1\times d}}{\arg\min}\ \frac{1}{2}\sum_{i=1}^r |a^T\Psi(T(x)) - a^TK\Psi(x_i)|^2 \\ &= \underset{a^TK\in\mathbb{R}^{1\times d}}{\arg\min}\ \frac{1}{2}\sum_{i=1}^r |a^T\Psi(y_i) - a^TK\Psi(x_i)|^2 \\ &= \left(\underset{K^Ta\in\mathbb{R}^d}{\arg\min}\ \left\|\Psi(Y)^Ta - \Psi(X)^TK^Ta\right\|_2^2\right)^T \\ &= a^T\Psi(Y)\Psi(X)^{\dagger} \end{aligned}$$

where $\Psi(X) = \begin{bmatrix}\Psi(x_1) & \dots & \Psi(x_r)\end{bmatrix}$ and $\Psi(Y) = \begin{bmatrix}\Psi(y_1) & \dots & \Psi(y_r)\end{bmatrix}$.

**Remark** When the dictionary basis dimension is greater than the number of snapshots, $d > r$, the least squares problem to obtain $K$ is not uniquely solvable. For this reason, a truncated SVD is used to get a unique solution for the least squares problem:

$$G = \Psi(X)\Psi(X)^T$$
$$G' = \Psi(Y)\Psi(X)^T.$$

Here $G, G' \in \mathbb{R}^{d\times d}$ and the Koopman matrix $K$ is given by

$$K = G'G^{\dagger}. \tag{2.24}$$

### Approximating eigenfunctions

The eigenfunction approximations are calculated using the eigenvalues and eigenvectors of the Koopman matrix $K$. Using left and right eigenvectors of the Koopman matrix $K$, the eigenfunctions can be approximated in two different ways.

If $w$ is a left eigenvector of the Koopman matrix $K$ with eigenvalue $\lambda$, we define

$$\phi(x) = w^T\Psi(x). \tag{2.25}$$

Using (2.23) we get

$$\mathcal{K}\phi(x) = \phi(T(x)) \approx w^T K\Psi(x) = \lambda w^T\Psi(x) = \lambda\phi(x). \tag{2.26}$$

Therefore, $(\lambda, w^T\Psi)$ is a Koopman eigenpair of $\mathcal{K}$.

If $\{v_j\}_{j=1}^d$ are right eigenvectors of the Koopman matrix $K$ with eigenvalues $\{\lambda_j\}_{j=1}^d$, then eigenfunctions $\{\phi_j\}_{j=1}^d$ on X can be approximated. We define the matrix

$$V = \begin{bmatrix}v_1 & \dots & v_d\end{bmatrix}.$$

The eigenfunctions evaluated at $X$ are given by solving the least squares problem

$$\begin{bmatrix}\phi_1(X)^T \\ \vdots \\ \phi_d(X)^T\end{bmatrix} = \underset{B\in\mathbb{R}^{d\times r}}{\arg\min}\, \|VB - \Psi(X)\|_2^2 = V^{\dagger}\Psi(X). \tag{2.27}$$

The proof is given in Appendix A.2.2.

### Computing Koopman modes

Given the state observable $g(x) = x \in \mathbb{R}^m$, the Koopman modes for it can be approximated using the left eigenvectors $\{w_j\}_{j=1}^d$ of $K$. We define the matrix

$$W = \begin{bmatrix}w_1 & \dots & w_d\end{bmatrix}.$$

The Koopman modes $C_i \in \mathbb{R}^m$, where $x = \sum_{i=0}^d C_i\phi_i(x)$ are then given by

$$\begin{bmatrix}C_1 \\ \vdots \\ C_d\end{bmatrix} = W^{\dagger}g(X). \tag{2.28}$$

Using the eigenvalue and eigenfunction approximations, approximate trajectories of the system can be generated given an initial condition $x_0 \in \mathbb{R}^m$. First we express $x_0$ as a combination of the eigenfunctions, $x_0 = \sum_{j=0}^d C_j(x_0)\phi_j(x_0)$. The trajectory $\{x_k\}_{k\in\mathbb{N}}$ is then given by

$$x_k = T^k(x_0) \approx \sum_{j=0}^{d} \lambda^j C_j(x_0)\phi_j(x_0). \tag{2.29}$$

Williams et al. [WKR15] show that EDMD is related to a Galerkin method used to approximate the operator. In particular, the EDMD approximation converges to the Galerkin method approximation in the large-data limit.

#### Relationship between DMD and EDMD

EDMD is equivalent to DMD if the dictionary of observables is chosen as the identity basis functions $\psi_j(x) = x_j$. The Koopman matrix in DMD is given by equation (2.20):

$$K_{DMD} = YX^{\dagger}.$$

By considering the identity dictionary in EDMD, $\Psi(X) = X$ and $\Psi(Y) = Y$, the Koopman matrix for EDMD is given by

$$K = \Psi(Y)\Psi(X)^{\dagger} = YX^{\dagger}. \tag{2.30}$$

Therefore, DMD is equivalent to EDMD with identity state dictionary and the Koopman modes approximated by EDMD are equal to the *exact DMD modes.* DMD can be thought of as producing approximations of eigenfunctions using only the identity basis functions. This can result in accurate approximations in certain cases such as in the neighbourhood of stable fixed points. For general cases, EDMD produces better approximations compared to DMD by using a larger set of non-linear basis functions. The quality of approximations produced by EDMD heavily depends on the choice of of the dictionary.

## 2.3 General eigensolvers

As approximating the eigenfunctions involves finding the eigenpairs of a matrix, eigensolvers are used in practice. We briefly describe three algorithms– power method, QR algorithm and Arnoldi iteration. We also mention deflation methods for iteratively computing eigenpairs.

### 2.3.1 Power method

Power method is one of the oldest techniques for calculating the dominant eigenpair of a matrix. The method proceeds by generating a sequence of vectors $\{A^k v_0\}$ where $v_0$ is a random initial vector and normalising them appropriately. Under some assumptions on the matrix, this sequence coverges to a dominant eigenvector (vector associated with the largest absolute eigenvalue). The pseudocode is given in Algorithm 1.

**Algorithm 1** (Power method) Given real matrix $A \in \mathbb{R}^{n\times n}$, max iterations $N$ and tolerance tol, compute the dominant eigenpair of $A$

```
$v_0 \leftarrow \text{random}(\mathbb{R}^n) : v_0 \neq 0^n$.
$k \leftarrow 1$.
while $k < N$ and $\|v_k - v_{k-1}\| > \text{tol}$ do
    $v_k \leftarrow A^k v_0$.
    $v_k \leftarrow \frac{v_k}{\|v_k\|}$.
    $k \leftarrow k + 1$.
end while
$v_1 \leftarrow v_k$.
$\lambda_1 \leftarrow \frac{\langle Av_1, v_1\rangle}{\|v_1\|^2}$.
```

Under the assumptions that the dominant eigenvalue $\lambda_1$ is semi-simple (algebraic multiplicity is one and equal to geometric multiplicity) and the initial vector $v_0$ is not orthogonal to the dominant eigenvector, the algorithm converges to the dominant eigenvector $\lambda_1$ associated with the dominant eigenvalue $v_1$ [Saa11].

There exist variations of the power method such as the shifted power method and the inverse iteration that can handle cases where the eigenvalues are close in magnitude (i.e. $|\lambda_1| \approx |\lambda_2|$).

In cases where the dimension of eigenspace of the dominant eigenvalue is greater than one, or the dominant eigenvalue exists in complex conjugate pairs, the power iteration fails to converge to a single single eigenvector and instead converges to a vector in the eigenspace or to a vector in the subspace spanned by the conjugate pair of eigenvectors. For example, if $(\lambda_1, v_1)$ is a complex eigenpair of a real matrix $A$ then $(\bar{\lambda}_1, \bar{v}_1)$ is also an eigenpair. Then

$$A^k x_0 = a_1 \lambda_1^k v_1 + a_2 \bar{\lambda}_1^k \bar{v}_1^k + \sum a_i \lambda_i v_i^k.$$

This converges to the subspace spanned by $\{v_1, \bar{v}_1\}$. For this reason, the power method is not used for general non-Hermitian matrices.

### 2.3.2 QR algorithm

The QR algorithm approximates the Schur factorization of a general non-Hermitian matrix $A = QUQ^T$ where $Q$ is a unitary matrix and $U$ is an upper triangular matrix. The eigenpairs of $U$ can then be used to calculate the eigenpairs of $A$. The pseudocode is given in Algorithm 2.

**Algorithm 2** (QR algorithm) Given real matrix $A \in \mathbb{R}^{n \times n}$ and iterations $m$, compute approximate Schur form

```
$A^0 \leftarrow A$.
$j \leftarrow 0$.
while $j < m$ do
    $A^j \leftarrow Q^j R^j$ (QR decomposition using Gram-Schmidt orthogonalization).
    $A^{j+1} \leftarrow R^j Q^j$.
    $j \leftarrow j + 1$.
end while
```

For larger values of $m$, $A_m$ converges to the Schur form $U$. The eigenvalues of $A$ are then approximated as eigenvalues of $A^m$ and eigenvectors of $A$ are approximated as $Q^0 \dots Q^m u^m$ where $u^m$ is an eigenvector of $A^m$. For general matrices, each iteration of the QR algorithm is expensive as it requires $O(n^3)$ operations to compute the QR factorization. For Hessenberg matrices, QR factorization can be computed in $O(n^2)$ operations [TB22].

### 2.3.3 Arnoldi method

Advanced eigensolver algorithms use a projection method where an eigenvector $v$ is approximated by a vector $\tilde{v}$ belonging to another subspace $\mathcal{K}$. In orthogonal projection methods, the residual vector $A\tilde{v} - \lambda\tilde{v}$ is required to be orthogonal to the subspace $\mathcal{K}$. The Arnoldi method is an orthogonal projection method for general non-Hermitian matrices where the subspace $\mathcal{K}$ is chosen as the Krylov subspace. These methods involve projection onto a Krylov subspace $\mathcal{K}_m$ defined as

$$\mathcal{K}_m(A, v) = \text{span}\{v, Av, \dots, A^{m-1}v\}. \tag{2.31}$$

The algorithm proceeds by building an orthonormal basis of the Krylov subspace. This orthonormalization process is accomplished in practice using the modified Gram-Schmidt orthogonalization. By projection onto the Krylov subspace $\mathcal{K}_m$ using this orthonormal basis, a similarity transformation results in the Hessenburg matrix of dimension $m$. Finding the eigenvectors and eigenvalues of an Hessenberg matrix through QR factorization is efficient. The eigenvector approximations of the original matrix $A$ can then be obtained using the orthonormal basis. The pseudocode is given in Algorithm 3.

**Algorithm 3** (Arnoldi iteration) Given real matrix $A \in \mathbb{R}^{n \times n}$ and Krylov subspace dimension $m$, compute Hessenberg matrix $H$ and orthonormal matrix $V$

```
$v_0 \leftarrow \text{random}(\mathbb{R}^n) : \|v_0\| = 1.$
$j \leftarrow 0.$
while $j < m$ do
    $w \leftarrow Av_j.$
    $i \leftarrow 0.$
    while $i < j$ do
        $h_{ij} \leftarrow \langle w, v_i \rangle.$
        $w \leftarrow w - h_{ij} v_i.$
        $i \leftarrow i + 1.$
    end while
    $h_{j+1,j} \leftarrow \|w\|.$
    $v_{j+1} \leftarrow w / h_{j+1,j}.$
    $j \leftarrow j + 1.$
end while
```

The matrix $H = \begin{bmatrix} h_{11} & \dots & h_{1m} \\ \vdots & \vdots & \vdots \\ h_{m1} & \dots & h_{mm} \end{bmatrix}$ is a Hessenberg matrix. $V = \begin{bmatrix} v_1 \dots v_m \end{bmatrix}$ is an orthornormal matrix and the following relationship holds:

$$V_m^* A V_m = H_m. \tag{2.32}$$

If $y_i$ are the the eigenvectors of $H_m$ with eigenvalues $\lambda_i$, then $V_m y_i$ and $\lambda_i$ are approximations of the eigenvectors and eigenvalues of $A$ respectively. As $m$ increases, the quality of approximation improves [Saa11]. This also leads to high storage and computational requirements. In cases where only the dominant eigenpair is required, a *restarted* version of the algorithm is used where the algorithm is stopped after $m$ iterations, the dominant eigenvalue $\lambda_1$ and eigenvector $y_1$ of the Hessenberg matrix $H_m$ is computed and the algorithm is restarted with initial vector $v_1 = V_m y_1$.

### 2.3.4 Deflation methods

Deflation techniques enable iterative eigenvalue algorithms to be used for finding all the eigenpairs of a matrix. If the dominant eigenvalue $\lambda_1$ and eigenvector $v_1$ of a matrix $A$ have been found using an algorithm, deflation techniques modify the matrix A so that the next application of the algorithm results in the eigenvalue $\lambda_2$ and eigenvector $v_2$. Deflation techniques usually involve a rank one modification to the matrix that retains the original eigenvalues and eigenvectors.

The general procedure is called *Wielandt's deflation* [Saa11]. Given the right eigenvector $v_1$ and eigenvalue $\lambda$ of matrix $A$, the deflated matrix is given by

$$A_1 = A - \sigma v_1 u^T \tag{2.33}$$

where $u$ is an arbitrary vector such that $u^T v_1 = 1$ and $\sigma \in \mathbb{R}$ is an appropriate shift. In this case, the eigenvalues $A_1$ are the same as the eigenvalues of $A$ except the first eigenvalue of $A_1$ which is $\lambda_1 - \sigma$.

Choosing the vector $u$ as the left eigenvector of $A$ preserves both the left and right eigenvectors of $A$. This deflation technique is called *Hotelling's deflation* [Saa11]. Lemma 1 shows that eigenvectors of the deflated matrix are the same as the eigenvectors of the original matrix.

**Lemma 1.** Suppose matrix $A$ has real eigenvalues $\{\lambda_1, \dots, \lambda_n\}$ with corresponding right eigenvectors $\{v_1, \dots, v_n\}$ and left eigenvectors $\{w_1, \dots, w_n\}$. $w_1$ is normalized so that $w_1^T v_1 = 1$. Then, the matrix $A - \lambda_1 v_1 w_1^T$ has eigenvalues $\{0, \lambda_1, \dots, \lambda_n\}$ with corresponding eigenvectors $\{v_1, v_2, \dots, v_n\}$

*Proof.* Let $A_1 = A - \lambda_1 v_1 w_1^T$. Then

$$A_1 v_1 = A v_1 - \lambda_1 v_1 w_1^T v_1 = \lambda_1 v_1 - \lambda_1 v_1 = 0,$$

and for $i \in \{2, \dots, n\}$.

$$A_1 v_i = A v_i - \lambda_1 v_1 w_1^T v_i = \lambda_i v_i,$$

where $w_1^T v_i = 0$ as $\lambda_1 \neq \lambda_i$. Similarly,

$$w_1^T A_1 = w_1^T A - \lambda_1 w_1^T v_1 w_1^T = \lambda w_1^T - \lambda w_1^T = 0,$$

and for $i \in \{2, \dots, n\}$

$$w_i^T A_1 = w_i^T A - \lambda_1 w_i^T v_1 w_1^T = w_i^T A = \lambda_i w_i^T,$$

where $w_i^T v_1 = 0$ as $\lambda_1 \neq \lambda_i$. □

# 3 Fast eigensolvers for Koopman operator approximation

This chapter develops methods for extending the Koopman operator's eigenfunctions and an iterative algorithm for approximating them. We define a measure of error for extended eigenfunctions and formulate upper bounds for it. We then describe an algorithm for extending eigenfunctions, given an eigenvector based on the error bounds and apply these results to linear dynamical systems. Then, we present an iterative algorithm for finding and extending eigenfunctions of the Koopman operator and apply it to two different non-linear systems.

## 3.1 Extending eigenfunctions and trajectory error

### 3.1.1 Extending eigenfunctions

Using the multiplicative property from Proposition 1, additional eigenfunctions of the Koopman operator can be approximated. If $\bar{\phi}_1$ and $\bar{\phi}_2$ are eigenfunction approximations of the Koopman operator $\mathcal{K}$ with eigenvalues $\bar{\lambda}_1$ and $\bar{\lambda}_2$, then

$$\bar{\phi}_{pq} = \bar{\phi}_1^p \bar{\phi}_2^q, \quad p, q \in \mathbb{Z}^+ \tag{3.1}$$

is also an eigenfunction approximation of $\mathcal{K}$ with eigenvalue $\bar{\lambda}_{pq} = \bar{\lambda}^p \bar{\lambda}^q$.

Using EDMD with dictionary $\Psi$, if $K \in \mathbb{R}^{d\times d}$ is the Koopman matrix, with left eigenvector $\{w_i\}_{i=1}^d$ and eigenvalues $\{\bar{\lambda}_i\}_{i=1}^d$. Given $i, j \in \{1, \dots, d\}$, using Equation (2.25) the extended set of eigenfunctions is given by

$$\bar{\phi}_{pq}(x) = (w_i^T \Psi(x))^p (w_j^T \Psi(x))^q \tag{3.2}$$

with eigenvalues $\bar{\lambda}_{pq} = \bar{\lambda}_i \bar{\lambda}_j$ where $p, q \in \mathbb{N}$. For simplicity, we will focus our analysis only on extending powers of eigenfunctions and not combination of eigenfunctions. For an eigenfunction approximation, $\phi = w^T \Psi$ of Koopman operator $\mathcal{K}$ with eigenvalue $\bar{\lambda}$,

$$\bar{\phi}_p(x) = (w^T \Psi(x))^p, \quad p \in \mathbb{N} \tag{3.3}$$

is also an eigenfunction approximation of $\mathcal{K}$ with eigenvalue $\bar{\lambda}_p = \bar{\lambda}^p$.

### 3.1.2 Trajectory error

Taking powers of eigenfunction approximations accumulates errors. Therefore, we need to define an error function for an extended eigenfunction. To measure this error, we define a grid G on our domain:

$$G = \{(a + nh, b + nh) | a, b \in \mathbb{R}; n \in \mathbb{Z}; h \in (0, 1)\}, \tag{3.4}$$

and a norm $||\cdot||_G$ on the grid,

$$||\phi||_G = \sqrt{\frac{1}{|G|} \sum_{x\in G} (\phi(x))^2}. \tag{3.5}$$

Using the relation (2.5) for discrete systems and the relation (2.6) for continuous systems, along with the definition of the Koopman operator, we define the *trajectory error* ($E_{TG}$) in Definition 5.

**Definition 5.** (Trajectory error) Given an extended eigenfunction approximation $\bar{\phi}_p$ with an extended eigenvalue approximation $\bar{\lambda}_p$ for the Koopman operator, the trajectory error of the eigenpair approximation $(\bar{\lambda}_p, \bar{\phi}_p)$ is defined as

$$E_{TG}(\bar{\lambda}_p, \bar{\phi}_p) = \left\|\bar{\phi}_p(T(\cdot)) - \bar{\lambda}_p \phi_p(\cdot)\right\|_G^{(1/p)} \tag{3.6}$$

for discrete systems and as

$$E_{TG}(\bar{\lambda}_p, \bar{\phi}_p) = \left\|\bar{\phi}_p(T^{\Delta t}(\cdot)) - \bar{\lambda}_p \phi_p(\cdot)\right\|_G^{(1/p)} \tag{3.7}$$

for continuous systems.

### 3.1.3 Finding a vector corresponding to an extended eigenfunction

Given an extended eigenfunction $\bar{\phi}_p$ with approximated eigenvalue $\bar{\lambda}_p$, we want to check if a corresponding vector $w_p \in \mathbb{R}^d$ exists in the dictionary basis $\Psi$ so that

$$w_p^T \Psi(x) \approx \bar{\phi}_p(x).$$

We do this by solving the least squares problem on the grid G:

$$w_p = \underset{w \in \mathbb{R}^d}{\arg\min} \left\|\Psi^T w - \bar{\phi}_p\right\|_G^2. \tag{3.8}$$

If the residual in the above problem is large, then the extended eigenfunction does not lie in the dictionary basis. If the residual is small, we can check if the vector obtained is another left eigenvector of the Koopman matrix by checking if $\left\|K^T w_p - \bar{\lambda}_p w_p\right\|_2 \approx 0$.

### 3.1.4 Reconstructing observables using a set of eigenfunctions

Given an observable $g \in \mathcal{F}, g : M \longrightarrow \mathbb{R}$, sampled output on a grid $G$, $\{g(x)\}_{x \in G}$ and a set of eigenfunctions $\{\phi_i\}_{i=1}^n$ we can approximate $g$ as

$$g(x) \approx \sum_{i=0}^{n} c_i \phi_i(x), \tag{3.9}$$

where $c_i \in \mathbb{R}$ are constants given by solving the least squares problem:

$$c = \underset{u \in \mathbb{R}^n}{\arg\min} \left\|g(\cdot) - u_i \phi_i(\cdot)\right\|_G^2. \tag{3.10}$$

## 3.2 Error anlaysis for trajectory error

The two main sources of error in eigenfunction approximation are the error in the eigenvector of the Koopman matrix $K$ due to the eigensolver, and the error in the integration for the flow computation in continuous systems. Eigenvector error is present for both discrete and continuous systems. We can find upper bounds for the trajectory error $E_{TG}$ with respect to these errors.

### 3.2.1 Trajectory error bound with respect to eigenvector error

Let $w$ be a left eigenvector of the Koopman matrix $K \in \mathbb{R}^{d \times d}$. Consider a left eigenvector $w_c \in \mathbb{R}^d$ of $K$ computed by an eigensolver. Let

$$w_c = w + \delta w,$$

where $\delta w \in \mathbb{R}^d$ is the error vector introduced due to the eigensolver.

We can get a posteriori upper bound on the trajectory error in (3.6) for the $p_{th}$-power eigenfunction approximation,

$$\bar{\phi}_p(x) = (w_c^T \Psi(x))^p. \tag{3.11}$$

We assume that the true left eigenvector is normalized so that $\|w\| = 1$. Assuming that $\bar{\lambda}$ is the computed eigenvalue corresponding to $w_c$ and $\bar{\lambda}_p = \bar{\lambda}^p$, Proposition 3 gives the error bound for the trajectory error with respect to the eigenvector error $\|\delta w\|$.

**Proposition 3.**

$$E_{TG}(\bar{\phi}_p, \bar{\lambda}_p) \leq C_{TG}(p, \bar{\lambda})^{(1/p)} \|\delta w\|^{(1/p)}, \tag{3.12}$$

*where*

$$C_{TG}(p, \bar{\lambda}) = \left\| \left\| \psi(T(\cdot)) - \bar{\lambda}\psi(\cdot) \right\| \sum_{i=0}^{p-1} \|\psi(T(\cdot))\|^{p-1-i} \|\psi(\cdot)\|^i \bar{\lambda}^i \right\|_G. \tag{3.13}$$

*Proof.* Using the identity $(x^p - y^p)^2 = (x-y)^2 \big( \sum_{i=0}^{p-1} x^{p-1-i} y^i \big)^2$, and the Cauchy–Schwarz inequality, we get

$$\begin{aligned}
E_{TG}(\bar{\phi}_p, \bar{\lambda}_p)^{2p} &= \left\| \bar{\phi}_p(T(\cdot)) - \bar{\lambda}_p \bar{\phi}_p(\cdot) \right\|_G^2 \\
&= \left\| (w_c^T \Psi(T(\cdot))^p - (\bar{\lambda} w_c^T \Psi(\cdot))^p \right\|_G^2 \\
&= \frac{1}{|G|} \sum_{x \in G} \left( (w_c^T \Psi(Tx))^p - (w_c^T \bar{\lambda} \Psi(x))^p \right)^2 \\
&= \frac{1}{|G|} \sum_{x \in G} \left( w_c^T \Psi(Tx) - w_c^T \bar{\lambda} \Psi(x) \right)^2 \left( \sum_{i=0}^{p-1} (w_c^T \Psi(Tx))^{p-1-i} (w_c^T \bar{\lambda} \Psi(x))^i \right)^2 \\
&= \frac{1}{|G|} \sum_{x \in G} (\delta w^T (\Psi(Tx) - \bar{\lambda} \Psi(x))^2 \left( \sum_{i=0}^{p-1} (w_c^T \Psi(Tx))^{p-1-i} (w_c^T \Psi(x))^i \bar{\lambda}^i \right)^2 \\
&\leq \frac{1}{|G|} \sum_{x \in G} \|\delta w\|^2 \left\| \Psi(Tx) - \bar{\lambda} \Psi(x) \right\|^2 \left( \sum_{i=0}^{p-1} \|w_c\|^{p-1-i} \|\Psi(Tx)\|^{p-1-i} \|w_c\|^i \|\Psi(x)\|^i \bar{\lambda}^i \right)^2 \\
&= \frac{1}{|G|} \sum_{x \in G} \|\delta w\|^2 \left\| \Psi(Tx) - \bar{\lambda} \Psi(x) \right\|^2 \left( \sum_{i=0}^{p-1} \|\Psi(Tx)\|^{p-1-i} \|\Psi(x)\|^i \bar{\lambda}^i \right)^2 \\
&= \|\delta w\|^2 \left\| \left\| \Psi(T(\cdot)) - \bar{\lambda} \Psi(\cdot) \right\| \left( \sum_{i=0}^{p-1} \|\Psi(T(\cdot))\|^{p-1-i} \|\Psi(.)\|^i \bar{\lambda}^i \right) \right\|_G^2 \\
&= C_{TG}(p, \bar{\lambda})^2 \|\delta w\|^2.
\end{aligned}$$

□

**Remark** To keep the trajectory error below $\epsilon$ for a power $p$ we will require that the error in computed eigenvector $\delta w$ has norm such that

$$\|\delta w\| \leq \frac{\epsilon^p}{C_{TG}(p, \bar{\lambda})}. \tag{3.14}$$

### 3.2.2 Trajectory error bound with respect to integration error

For a continuous system, the trajectory error will also depend on the integration error introduced while integrating the system to get the flow $T^{\Delta t}$. Let

$$T^{\Delta t}(x) - x^{\Delta t} = \varepsilon(x), \tag{3.15}$$

where $\epsilon(x) \in \mathbb{R}^n$ is the integration error at $x$, $x^{\Delta t}$ is the accurate flow at $t = \Delta t$, and $T^t(x)$ is the computed flow. We consider the upper bound of the trajectory error with respect to the quantity

$$\epsilon_G = \max_{x \in G} \|\varepsilon(x)\| . \tag{3.16}$$

Using the above, Proposition 4 gives the upper bound for eigenfunction approximation $\bar{\phi}_p$ and eigenvalue approximation $\bar{\lambda}_p$ with respect to the integration error $\epsilon_G$.

**Proposition 4.**

$$E_{TG}(\bar{\phi}_p, \bar{\lambda}_p) \leq \left( (\bar{\lambda}^{\Delta t} M + L\epsilon_G)^p - (\bar{\lambda}^{\Delta t} M)^p \right)^{(1/p)} . \tag{3.17}$$

*where $M = \max_{x \in G} \|\psi(x)\|$ and $L$ is an upper bound on the spectral norm of the Jacobian of $\Psi$ with respect to the $l_2$ norm on the grid $G$,*

$$\|J_\Psi(x)\|_2 \leq L. \tag{3.18}$$

*Proof.* Using the definition of Koopman operator,

$$\begin{aligned}
\phi(T^{\Delta t}(x)) &= \phi(x^{\Delta t} + \varepsilon(x)) \\
&\approx \phi(x^{\Delta t}) + \varepsilon(x)^T \nabla \phi(x^{\Delta t}) \\
&= \bar{\lambda}^{\Delta t} \phi(x) + \varepsilon(x)^T \nabla \phi(x^{\Delta t}).
\end{aligned}$$

Using the bound on the spectral norm on the Jacobian of the dictionary basis $\Psi$, we get

$$\|\nabla \phi(x)\| = \left\|J_\psi(x) w\right\| \leq \left\|J_\psi(x)\right\|_2 \|w\| \leq L, \tag{3.19}$$

where $\|.\|_2$ is the spectral norm. Then using the above equations, the Cauchy–Schwarz inequality and the Binomial theorem, we get

$$\begin{aligned}
E_{TG}(\bar{\phi}_p, \bar{\lambda}_p)^{2p} &= \left\|\phi(T^{\Delta t}(\cdot))^p - (\bar{\lambda}^{\Delta t} \phi(\cdot))^p\right\|_G^2 \\
&= \frac{1}{|G|} \sum_{x \in G} \left( (\phi(T^{\Delta t}(x)))^p - (\bar{\lambda}^{\Delta t} \phi(x))^p \right)^2 \\
&= \frac{1}{|G|} \sum_{x \in G} \left( (\bar{\lambda}^{\Delta t} \phi(x) + \varepsilon(x)^T \nabla \phi(x^{\Delta t}))^p - (\bar{\lambda}^{\Delta t} \phi(x)))^p \right)^2 \\
&= \frac{1}{|G|} \sum_{x \in G} \left( \sum_{k=0}^{p} \binom{p}{k} (\bar{\lambda}^{\Delta t} \phi(x))^{p-k} (\varepsilon(x)^T \nabla \phi(x^{\Delta t}))^k - (\bar{\lambda}^{\Delta t} \phi(x)))^p \right)^2 \\
&= \frac{1}{|G|} \sum_{x \in G} \left( \sum_{k=1}^{p} \binom{p}{k} (\bar{\lambda}^{\Delta t} \phi(x))^{p-k} (\varepsilon(x)^T \nabla \phi(x^{\Delta t}))^k \right)^2 \\
&= \frac{1}{|G|} \sum_{x \in G} \left( \sum_{k=1}^{p} \binom{p}{k} (\bar{\lambda}^{\Delta t} w^T \psi(x))^{p-k} (\varepsilon(x)^T \nabla \phi(x^{\Delta t}))^k \right)^2 \\
&\leq \frac{1}{|G|} \sum_{x \in G} \left( \sum_{k=1}^{p} \binom{p}{k} (\bar{\lambda}^{\Delta t})^{p-k} \|w\|^{p-k} \|\psi(x)\|^{p-k} \|\varepsilon(x)\|^k \left\|\nabla \phi(x^{\Delta t})\right\|^k \right)^2 \\
&\leq \frac{1}{|G|} \sum_{x \in G} \left( \sum_{k=1}^{p} \binom{p}{k} (\bar{\lambda}^{\Delta t})^{p-k} M^{p-k} \|\varepsilon(x)\|^k L^k \right)^2 \\
&\leq \frac{1}{|G|} \sum_{x \in G} \left( \sum_{k=0}^{p} \binom{p}{k} (\bar{\lambda}^{\Delta t})^{p-k} M^{p-k} \|\varepsilon(x)\|^k L^k - (\bar{\lambda}^{\Delta t})^p M^p \right)^2 \\
&\leq \frac{1}{|G|} \sum_{x \in G} \left( (\bar{\lambda}^{\Delta t} M + L \|\varepsilon(x)\| )^p - (\bar{\lambda}^{\Delta t})^p M^p \right)^2
\end{aligned}$$

$$
\begin{aligned}
&= \left\| (\bar{\lambda}^{\Delta t} M + L \left\| \varepsilon(\cdot) \right\| )^p - (\bar{\lambda}^{\Delta t} M)^p \right\|_G^2 \\
&\leq \left( (\bar{\lambda}^{\Delta t} M + L \epsilon_G)^p - (\bar{\lambda}^{\Delta t} M)^p \right)^2 .
\end{aligned}
$$

□

**Remark** To keep the trajectory error below $\epsilon$ for a power $p$, we will require that the integration error $\epsilon_G$ has the following bound:

$$\epsilon_G \leq \frac{1}{L}[(\epsilon^p + (\bar{\lambda}^{\Delta t} M)^p)^{1/p} - \bar{\lambda}^{\Delta t} M] \tag{3.20}$$

**Remark** To calculate $L$, we compute the maximum singular value of the matrix $J_\Psi(x)$ over the grid $G$ and take the maximum value:

$$
\begin{aligned}
&\lambda_{max}(x) = \max\{\lambda : J_\Psi(x) v = \lambda(x) v\} \\
&L = \max_{x \in G} \sqrt{\lambda_{max}(x)} \geq \left\| J_\Psi(x) \right\|_2 .
\end{aligned} \tag{3.21}
$$

## 3.3 Algorithms for computing extended eigenfunctions using error bounds

Given a left eigenvalue and eigenvector of the Koopman operator, we can define an algorithm to find the extended eigenfunctions based on the trajectory error and upper bounds calculated in the last section.

For a discrete system, we do not have integration error and only have error due to eigenvector. In this case, given $\epsilon > 0$, to get $E_{TG}(\bar{\phi}_p, \bar{\lambda}_p) \leq \epsilon$, we require

$$\left\| \delta v \right\| \leq \frac{\epsilon^p}{C_{TG}(p, \bar{\lambda})}. \tag{3.22}$$

Using this relation, Algorithm 4 defines the algorithm to extend eigenpairs of the Koopman operator for a discrete system.

**Algorithm 4** Computing extended eigenpairs for discrete system. Given $\|\delta w\|$, left eigenpair $(\bar{\lambda}, w_c)$ of the Koopman matrix $K$, $\Psi$ as the dictionary basis and desired trajectory error bound $\epsilon$

```
while $p \in \mathbb{N}$ do
    $C_{TG}(p, \bar{\lambda}) \leftarrow \left\| \left\| \Psi(T(\cdot)) - \bar{\lambda}\Psi(\cdot) \right\| \sum_{i=0}^{p-1} \left\| \Psi(T(\cdot)) \right\|^{p-1-i} \left\| \Psi(\cdot) \right\|^i \bar{\lambda}^i \right\|_G .$
    if $\frac{\epsilon^p}{C_{TG}(p, \bar{\lambda})} > \|\delta w\|$ then
        break
    end if
    $\bar{\phi}_p \leftarrow (w_c^T \Psi)^p.$
    $\bar{\lambda}_p = (\bar{\lambda})^p.$
end while
```

For a continuous system, we have integration error in the flow calculation. Given $\epsilon > 0$, to get $E_{TG}(\bar{\phi}_p, \bar{\lambda}_p) \leq \epsilon$, we require

$$\epsilon_G \leq \frac{1}{L}\left( (\epsilon^p + (\bar{\lambda} M)^p)^{1/p} - \bar{\lambda} M \right). \tag{3.23}$$

Using this relation, Algorithm 5 defines the algorithm to extend eigenpairs of the Koopman operator for a continuous system.

**Algorithm 5** Computing extended eigenpairs for continuous system. Given $\epsilon_G$, left eigenpair $(\bar{\lambda}, w_c)$ of the Koopman matrix $K$ and $\psi$ as the dictionary basis and desired trajectory error bound $\epsilon$

```
while $p \in \mathbb{N}$ do
    if $\frac{1}{L}((\epsilon^p + (\bar{\lambda}M)^p)^{1/p} - \bar{\lambda}M) > \epsilon_G$ then
        break
    end if
    $\bar{\phi}_p \leftarrow (w_c^T \psi)^p$.
    $\bar{\lambda}_p = (\bar{\lambda})^p$.
end while
```

## 3.3.1 Example – discrete linear system

Consider the discrete linear system

$$x_{n+1} = Ax_n, \tag{3.24}$$

where $A \in \mathbb{R}^{2\times 2}$ with left eigenpairs $(\lambda_1, w_1)$ and $(\lambda_2, w_2)$.

The Koopman eigenfunctions and eigenvalues for such a system can be written explicitly. As shown in Example 1, the system has eigenvalues and eigenfunctions given by

$$\lambda_p = \lambda_1^p \tag{3.25}$$
$$\phi_p(x) = \big(\langle w_1, x\rangle\big)^p \tag{3.26}$$
$$\lambda_q = \lambda_2^q \tag{3.27}$$
$$\phi_q(x) = \big(\langle w_2, x\rangle\big)^q. \tag{3.28}$$

We can compare the extended eigenfunction with the explicit eigenfunctions. Let $\bar{\phi}_p$ be an extended eigenfunction and $\phi_p$ be an explicit eigenfunction. Then on a grid G, as defined in (3.4), the error is given by $E_G$:

$$G_0 = \{x \in G|\ |\phi_p| < \gamma\} \cup \{x \in G|\ |\bar{\phi}_p| < \gamma\} \tag{3.29}$$
$$c_{mode} = mode\left(\frac{\phi_{p|G/G_0}}{\bar{\phi}_{p|G/G_0}}\right) \tag{3.30}$$
$$E_G(\phi_p, \bar{\phi}_p) = \left\|\phi_p - c_{mode}\bar{\phi}_p\right\|_G^{1/p}, \tag{3.31}$$

where $\gamma$ is some tolerance.

Given a set of points in grid $G$, the trajectory error for the computed extended eigenpair $(\bar{\lambda}_p, \bar{\phi}_p)$ is defined as

$$E_{TG}(\bar{\lambda}_p, \bar{\phi}_p) := \left\|\bar{\phi}_p(A(\cdot)) - \bar{\lambda}_p \phi_p(\cdot)\right\|_G^{1/p}. \tag{3.32}$$

We use

$$A = \begin{bmatrix} 0.9 & -0.1 \\ 0 & 0.8 \end{bmatrix} \tag{3.33}$$

The matrix has left eigenpairs $(0.9, [-1, 1]^T)$ and $(0.8, [0, 1]^T)$. Therefore the explicit Koopman eigenfunctions of the system are given by

$$\phi_p(x) = \left(\frac{x_1 - x_2}{\sqrt{2}}\right)^p \tag{3.34}$$
$$\phi_q(x) = x_2^q, \tag{3.35}$$

with eigenvalues $\lambda_p = (0.9)^p$ and $\lambda_q = (0.8)^q$. To approximate the eigenfunctions using DMD and EDMD, we collect 400 snapshot pairs where the initial conditions are uniformly randomly distributed between $[-2, 2] \times [-2, 2]$.

### Approximating eigenfunctions using DMD

DMD is equivalent to EDMD with the dictionary basis as the identity:

$$\Psi(x) = [x_1, x_2]^T. \tag{3.36}$$

Figure 3.1 shows the results of the DMD approximation. The out-of-sample prediction shows that the DMD approximation can predict trajectories accurately. Using the computed left eigenvectors $w_1$ and $w_2$ of the Koopman matrix, we compute the extended eigenfunctions defined, $\bar{\phi}_p$ and $\bar{\phi}_q$ as defined in (3.3).

We then compare the explicit eigenfunctions of the system given by (3.34) and (3.35) with computed eigenfunctions (3.26) and (3.28). Figure 3.2 and Figure 3.3 show that the extended eigenfunctions agree with the explicit eigenfunctions.

We find a corresponding vector for the extended eigenfunction powers by solving the least squares problem defined in (3.8). Figure 3.4 shows that all powers of the extended eigenfunctions have a high residual, indicating that the extended eigenfunctions do not lie in the identity dictionary space and, therefore, there exists no vector in $\mathbb{R}^d$ which can make them an eigenfunction.

We reconstruct the observable $g(x) = \sin(x_1)\cos(x_2)$ by solving the least squares problem described in (3.10). Then, we calculate the norm error of the reconstruction. Figure 3.5 shows the reconstruction using the original set of eigenfunctions and sets with an increasing number of eigenfunctions. As the number of extended eigenfunctions used for reconstruction increases, the norm error of reconstruction decreases.

We take the grid $G$ with $a = -1, b = 1, n = 100, h = 0.01$ for the trajectory error analysis. Figure 3.6 shows the trend of the error between explicit eigenfunctions and computed eigenfunctions, the trajectory error for the computed extended eigenfunctions and the computed upper bound constant ($C_{TG}$). As the powers of the extended eigenfunctions increase, the error between the explicit eigenfunction and computed eigenfunction increases, and the trajectory error also increases.

We can compute the trajectory error and its upper bound for the extended eigenfunctions by introducing an explicit eigenvector error $\|\delta w\|$. For the Koopman operator of a discrete linear system approximated using DMD, we know the left eigenvectors precisely as they are the left eigenvectors of the matrix $A$. Then we can calculate the trajectory error and its upper bound with computed left eigenvectors $w_c = w + \delta w$ where w is the exact left eigenvector, and $\delta w$ is a random error vector chosen such that $\|\delta w\| = 10^{-6}$. Figure 3.7 shows the trajectory error and its upper bound for different powers of $p$ and $q$. We use a grid G with $h = 0.01,\ a = -1,\ b = 1$ for this calculation.

Finally, we try Algorithm 4 described in the last section using the trajectory errors and upper bounds with respect to eigenvector error to extend eigenfunctions $\bar{\phi}_p$ and $\bar{\phi}_q$ up to powers $p$ and $q$ such that the trajectory error stays below a desired upper bound $\epsilon$. Figure 3.8 shows the results of Algorithm 4 for a desired trajectory error upper bound $\epsilon = 0.1$. As seen in the figure, the powers $p$ and $q$ suggested by the algorithms are close to the actual powers up to which the eigenfunctions can be extended.

### Approximating eigenfunctions using EDMD

We take the dictionary basis as the space of polynomials with degree up to three:

$$\Psi(x_1, x_2) = [x1, x_2, x_1^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3]^T. \tag{3.37}$$

Figure 3.9 shows the results of the EDMD approximation. The out-of-sample prediction shows that the EDMD approximation can predict trajectories accurately. The EDMD approximation gives a Koopman matrix of dimension $9 \times 9$ with nine eigenpairs. Here we take the left eigenvectors $w_1$ and $w_2$ as the vectors corresponding to eigenvalues $\bar{\lambda}_1 = 0.9$ and $\bar{\lambda}_2 = 0.8$. We then compute the extended eigenfunctions, defined $\bar{\phi}_p$ and $\bar{\phi}_q$ as defined in (3.3) using vectors $w_1$ and $w_2$ respectively.

We then compare the explicit eigenfunctions of the system with computed extended eigenfunctions. Figure 3.10 and Figure 3.11 show that the extended eigenfunctions agree with the explicit eigenfunctions.

We find a corresponding vector for the extended eigenfunction powers by solving the least squares problem defined in (3.8). Figure 3.12 shows that the extended eigenfunctions up to power 3 have a low

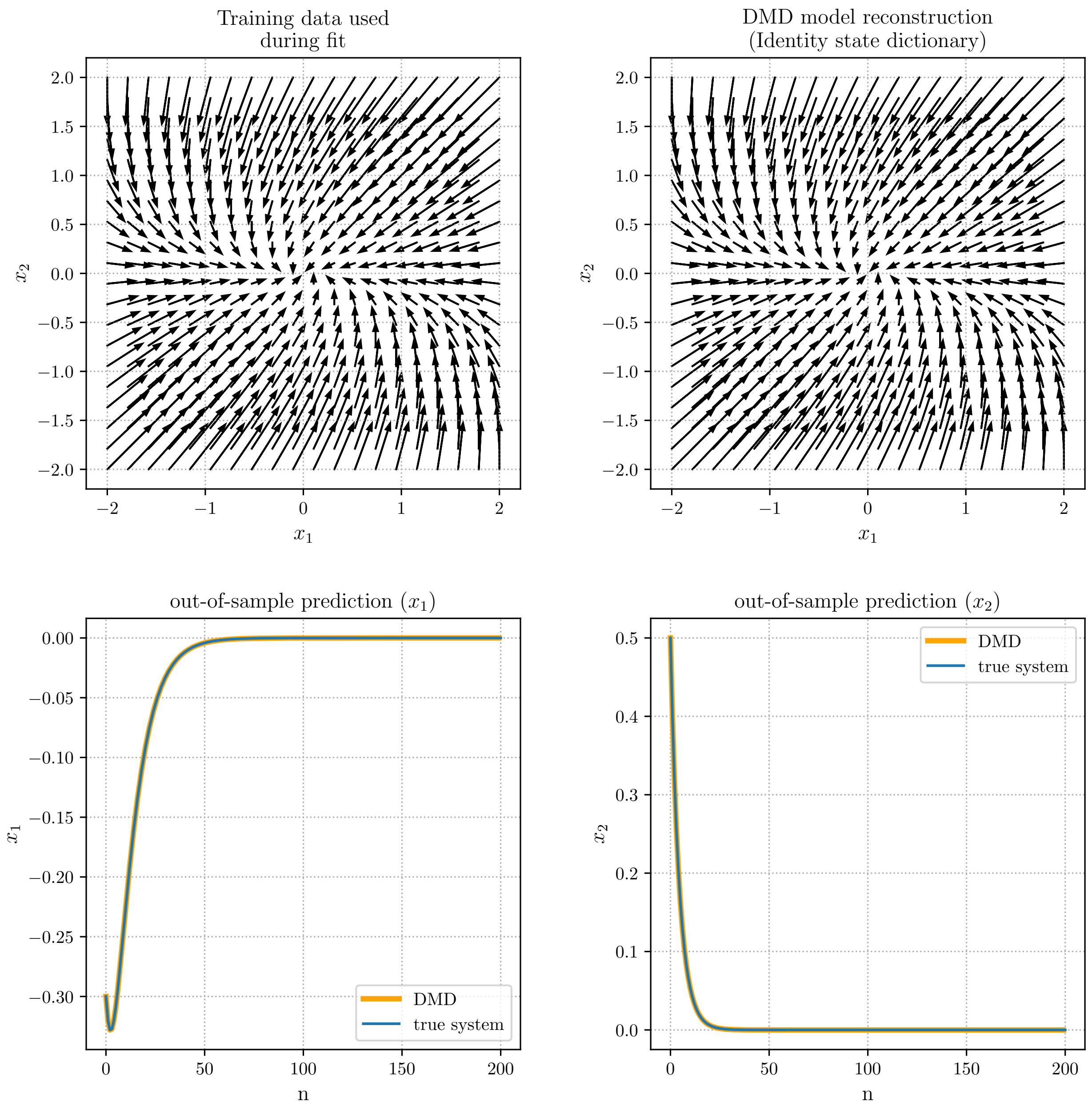

**Figure 3.1** DMD model for the discrete linear system (3.24). Top left shows the training data used for approximation, top right shows the reconstructed data, bottom shows the comparison between predicted trajectory and actual trajectory for one initial condition.

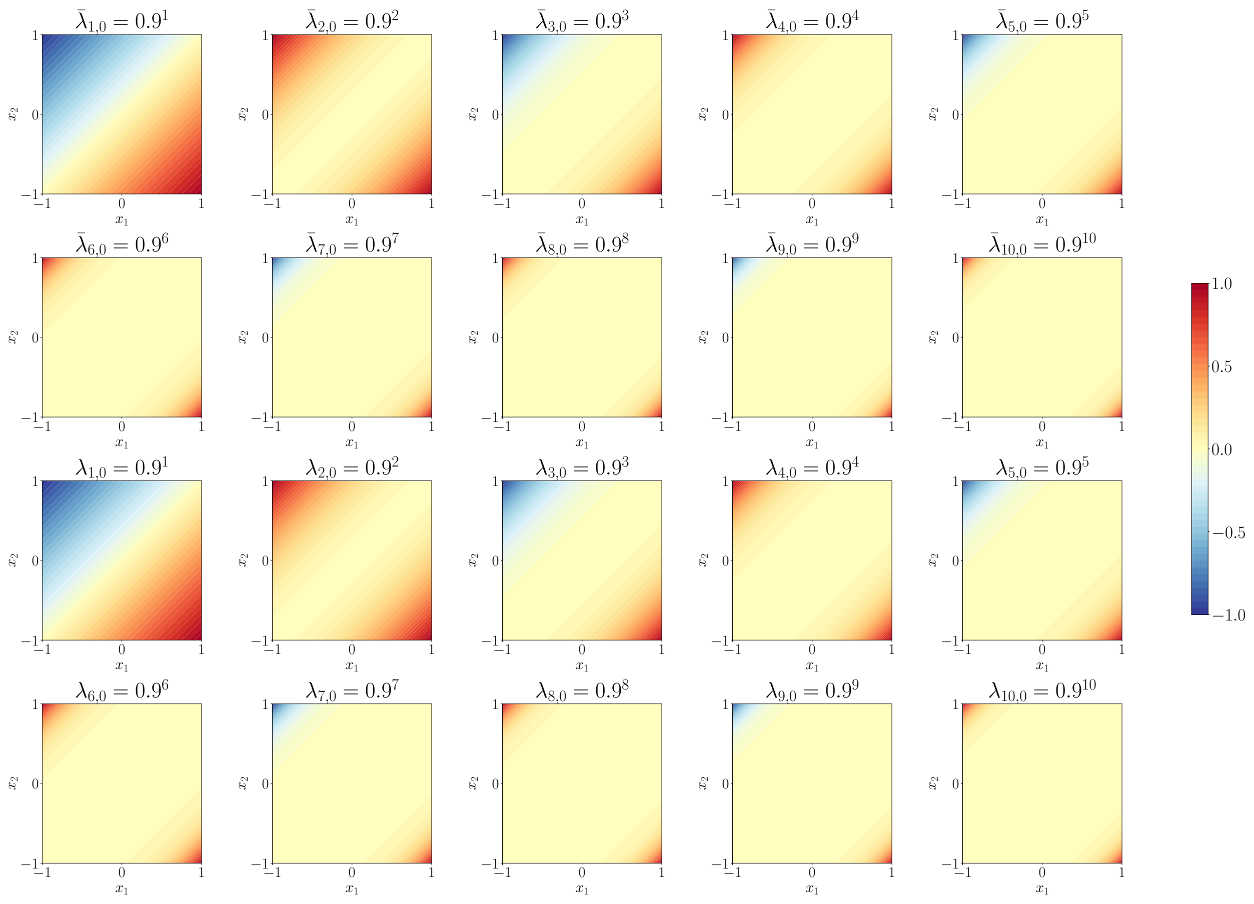

**Figure 3.2** Comparison of explicit eigenfunctions with computed eigenfunctions approximated using DMD eigenvectors for the discrete linear system (3.24). First and second rows show computed eigenfunction contours for powers $1 \le p \le 10$, and third and fourth rows show explicit eigenfunction contours for powers $1 \le p \le 10$. The eigenfunctions have been normalized such that $\|\phi\|_\infty \le 1$.

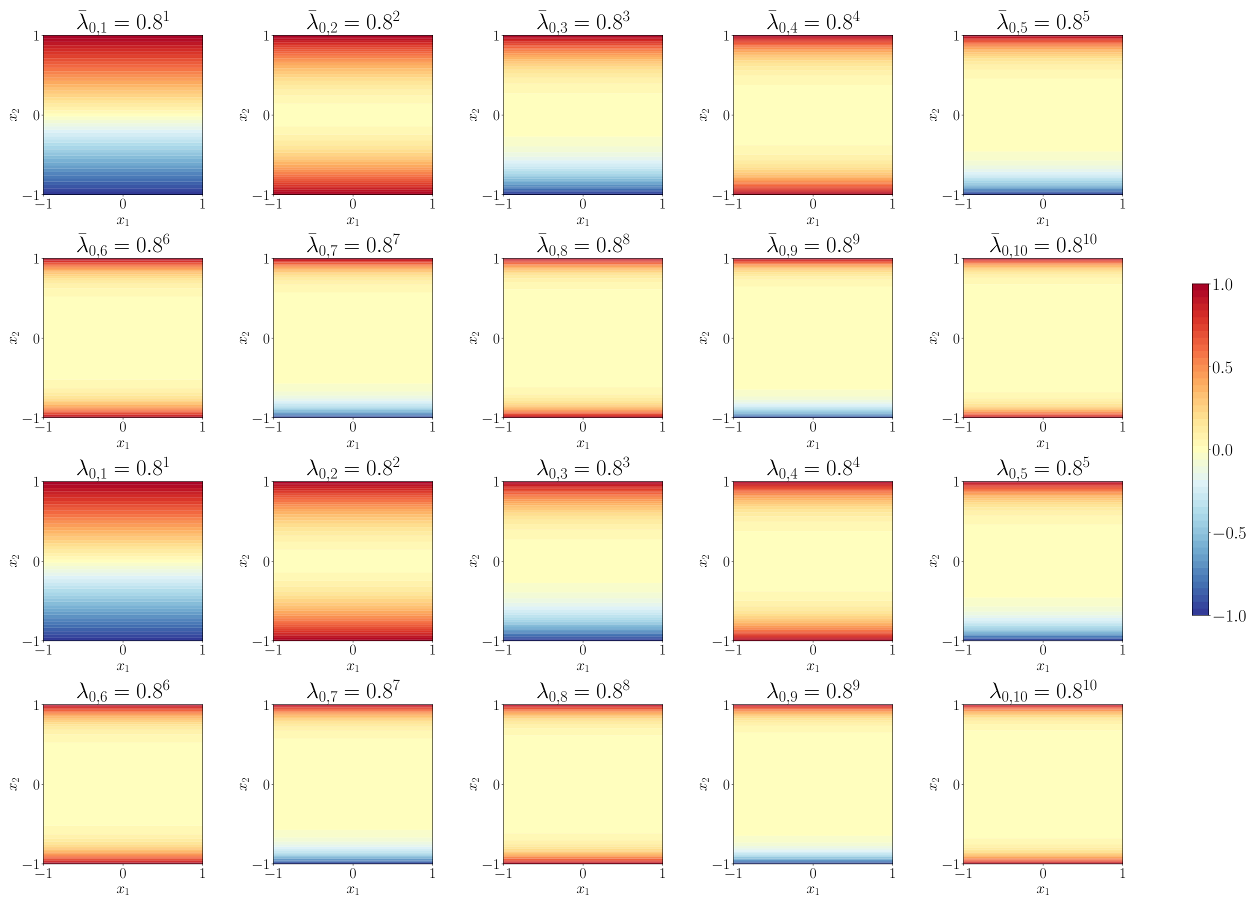

**Figure 3.3** Comparison of explicit eigenfunctions with computed eigenfunctions approximated using DMD eigenvectors for the discrete linear system (3.24). First and second rows show computed eigenfunction contours for powers $1 \leq q \leq 10$, and third and fourth rows show explicit eigenfunction contours for powers $1 \leq q \leq 10$. The eigenfunctions have been normalized such that $\|\phi\|_\infty \leq 1$.

| power p | residual | eig_eq_norm |
|---|---|---|
| 1 | 4.924587e-25 | 2.636780e-16 |
| 2 | 6.938401e+02 | - |
| 3 | 9.603623e+01 | - |
| 4 | 2.406468e+02 | - |
| 5 | 8.279875e+01 | - |
| 6 | 1.237877e+02 | - |
| 7 | 6.167356e+01 | - |
| 8 | 7.656635e+01 | - |
| 9 | 4.639823e+01 | - |
| 10 | 5.272445e+01 | - |

| power q | residual | eig_eq_norm |
|---|---|---|
| 1 | 1.956396e-27 | 1.814610e-18 |
| 2 | 2.081347e+03 | - |
| 3 | 2.424390e+02 | - |
| 4 | 1.202693e+03 | - |
| 5 | 3.272690e+02 | - |
| 6 | 8.655758e+02 | - |
| 7 | 3.393607e+02 | - |
| 8 | 6.877325e+02 | - |
| 9 | 3.310601e+02 | - |
| 10 | 5.781470e+02 | - |

**Figure 3.4** Residual for problem (3.8) for different powers of p and q of DMD eigenfunctions of the discrete linear system (3.24). The eigen equation norm checks if the computed vector in (3.8) is a left eigenvector of the Koopman matrix.

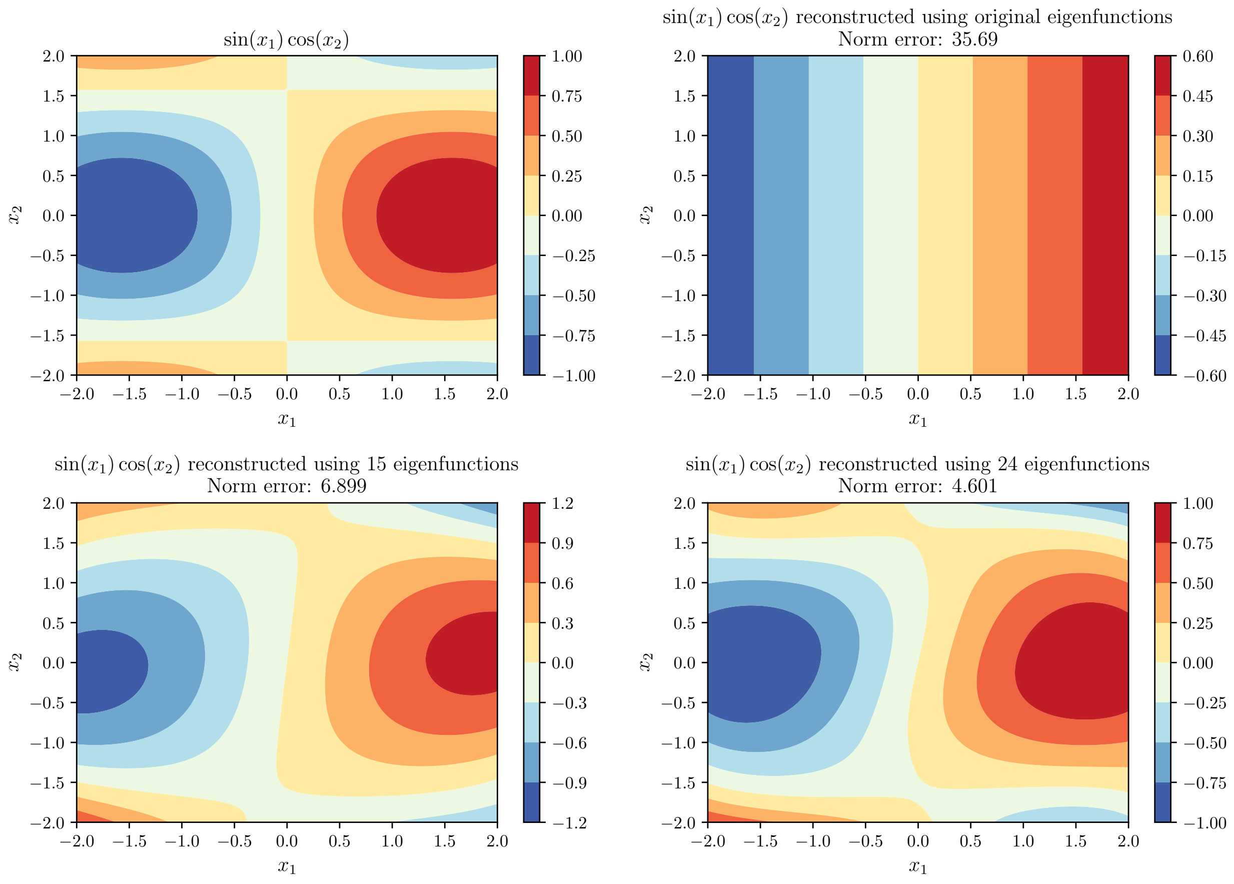

**Figure 3.5** Reconstruction of the observable $\sin(x_1)\cos(x_2)$ using the DMD eigenfunctions of the discrete linear system (3.24). Top left: $\sin(x_1)\cos(x_2)$. Top right: Observable reconstructed using original set of DMD eigenfunctions. Bottom left, bottom right: Observable reconstructed using extended set of eigenfunctions.

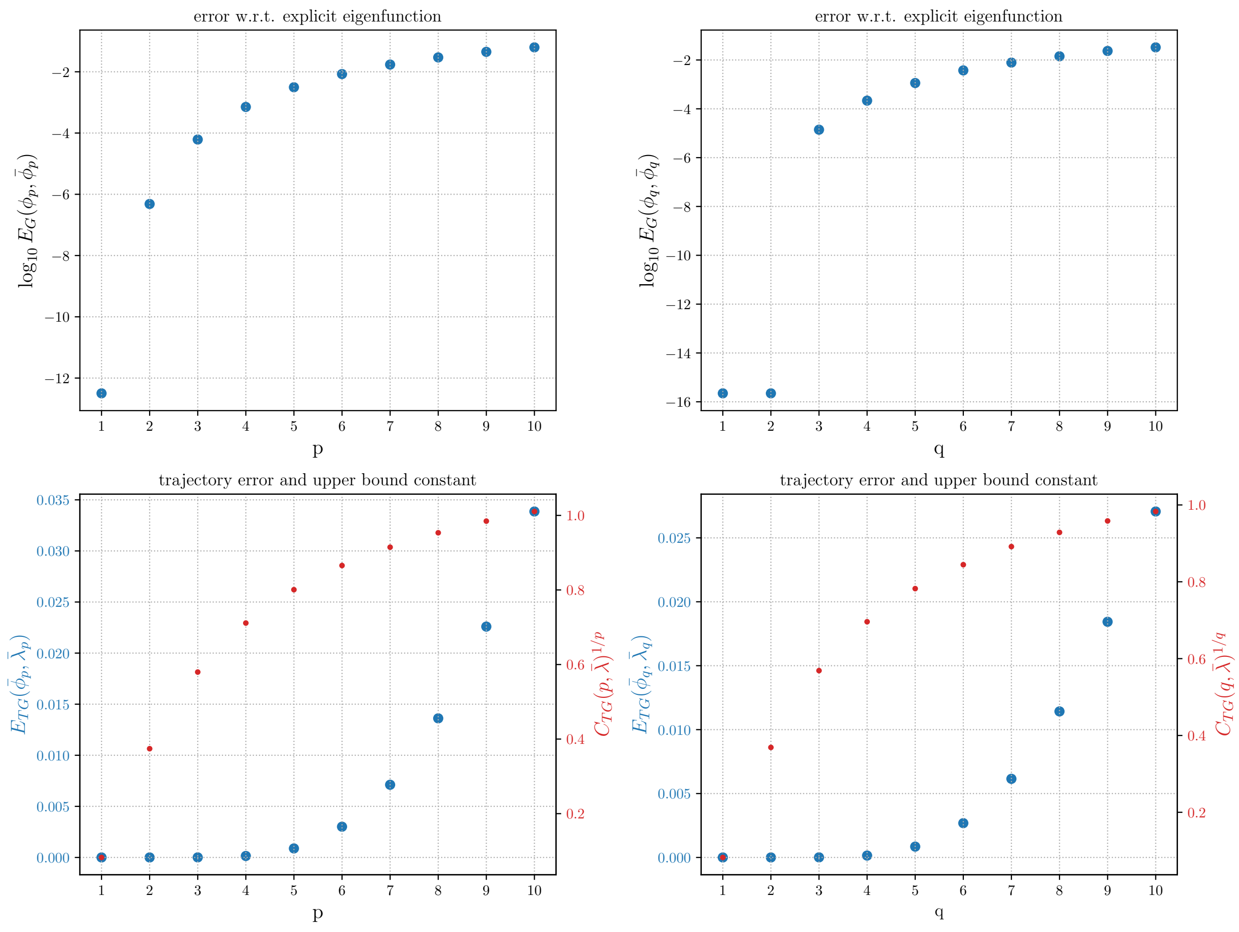

**Figure 3.6** Error analysis for DMD eigenfunctions of discrete linear system (3.24). Top row: $\log_{10}$ error between computed and explicit eigenfunctions on grid G ($E_G$). Bottom row: Trajectory error for computed eigenfunctions on grid G – $E_{TG}$(blue) and constant of the upper bound – $C_{TG}$(red).

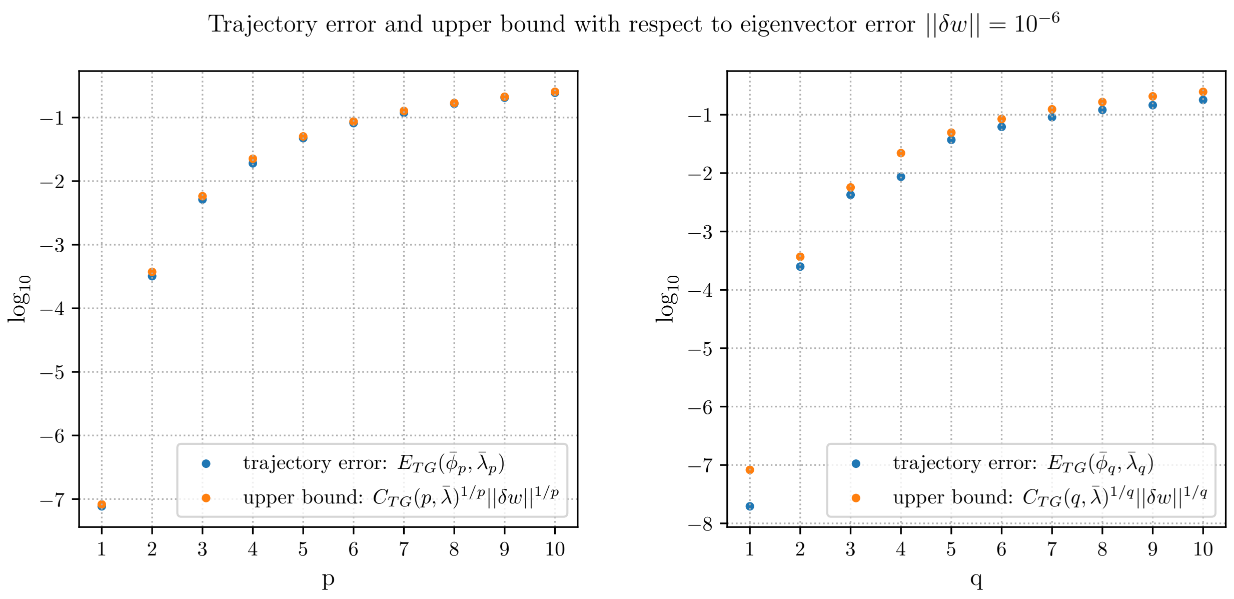

**Figure 3.7** Trajectory error and upper bound calculation for DMD eigenfunctions of discrete linear system (3.24) using $||\delta w|| = 10^{-6}$ error in the eigenvectors.

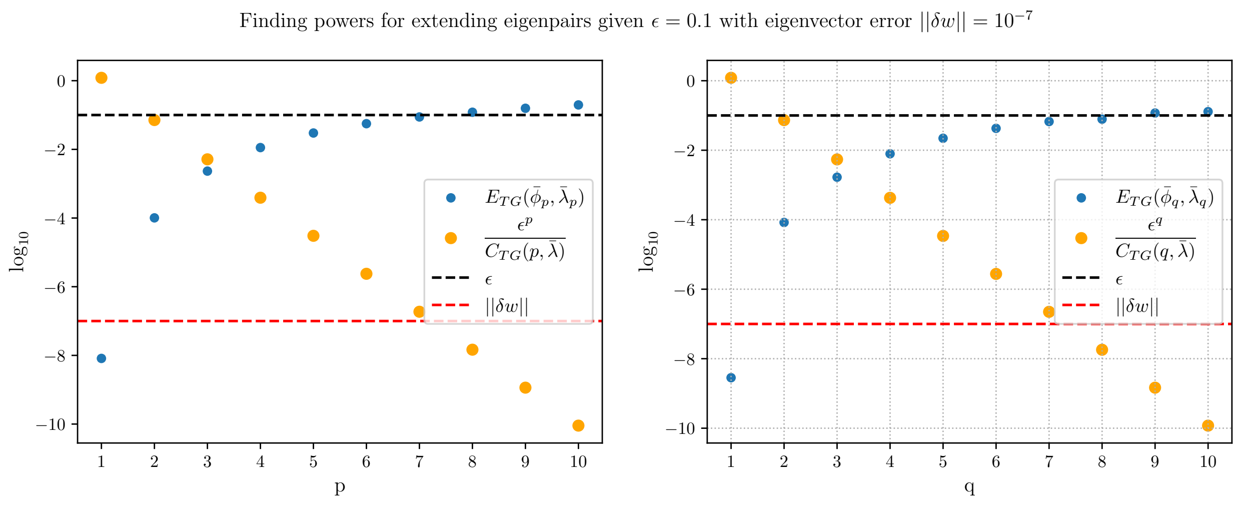

**Figure 3.8** Results of Algorithm 4 applied to the DMD approximation of discrete linear system (3.24) with eigenvector error $||\delta w|| = 10^{-7}$ and desired trajectory error $\epsilon = 0.1$. The value of $p$ and $q$ suggested by the algorithm– where the upper bound for $||\delta w||$ (orange) crosses $||\delta w||$ (red line) is close to actual value of $p$ and $q$ where the trajectory error (blue) crosses the required $\epsilon$ (black).

residual, and the vector in the dictionary space is an eigenvector. The extended eigenfunctions for higher powers have a high residual, indicating they do not lie in the polynomial dictionary space and therefore, there does not exist a vector in $\mathbb{R}^d$ that makes them an eigenfunction in this space.

We reconstruct the observable $g(x) = \sin(x_1)\cos(x_2)$ and calculate the norm error of the reconstruction. Figure 3.13 shows the reconstruction for the original set of EDMD eigenfunctions and sets with an increasing number of eigenfunctions. As the number of extended eigenfunctions used for reconstruction increases, the norm error of reconstruction decreases.

We take the grid $G$ with $a = -1, b = 1, n = 100, h = 0.01$ for the trajectory error analysis. Figure 3.14 shows the trend of the error between explicit eigenfunctions and computed eigenfunctions, the trajectory error for the computed extended eigenfunctions, and the computed upper bound constants ($C_{TG}$). The error between the computed and explicit eigenfunctions, and the trajectory error increase as the powers increase.

### 3.3.2 Example – continuous linear system

Consider the continuous linear system

$$\dot{x} = Ax, \tag{3.38}$$

where $A \in \mathbb{R}^{2X2}$ with left eigenpairs $(\lambda_1, w_1)$ and $(\lambda_2, w_2)$.

Let the system be sampled with sampling interval $\Delta t$. Then the Koopman eigenfunctions and eigenvalues of the system are given by

$$\lambda_p = (e^{\lambda_1 \Delta t})^p \tag{3.39}$$
$$\phi_q(x) = \big(\langle w_1, x\rangle\big)^p \tag{3.40}$$
$$\lambda_q = (e^{\lambda_2 \Delta t})^q \tag{3.41}$$
$$\phi_q(x) = \big(\langle w_2, x\rangle\big)^q. \tag{3.42}$$

If $\bar{\lambda}$ is a computed eigenvalue of the Koopman matrix $K$ and $\lambda$ is an eigenvalue of the Koopman generator of the continuous system, then $\bar{\lambda} \approx e^{\lambda \Delta t}$ where $\Delta t$ is the sampling interval for the system.

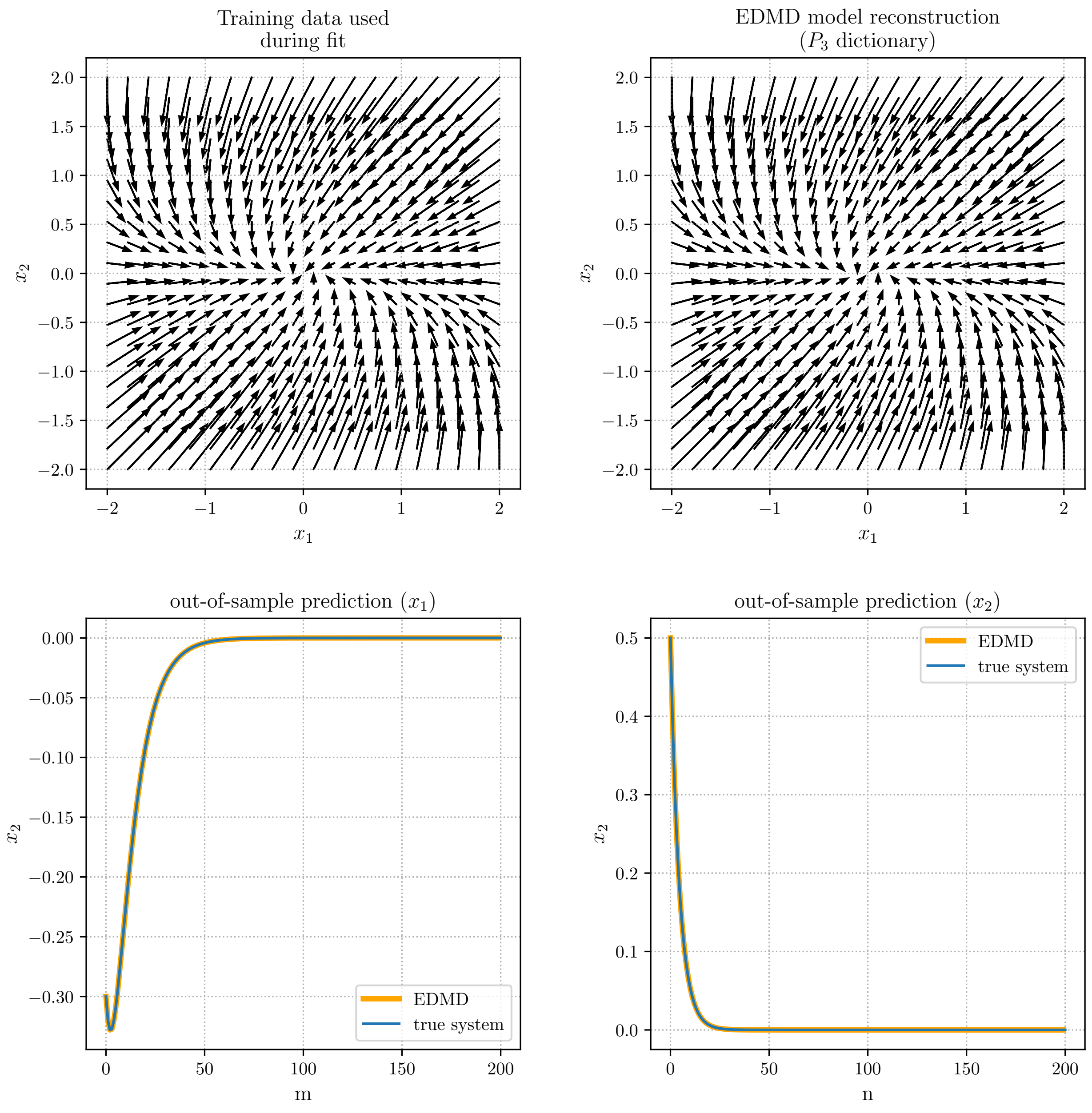

**Figure 3.9** EDMD model for the discrete linear system (3.24). Top left shows the training data used for approximation, top right shows the reconstructed data, bottom shows the comparison between predicted trajectory and actual trajectory for one initial condition.

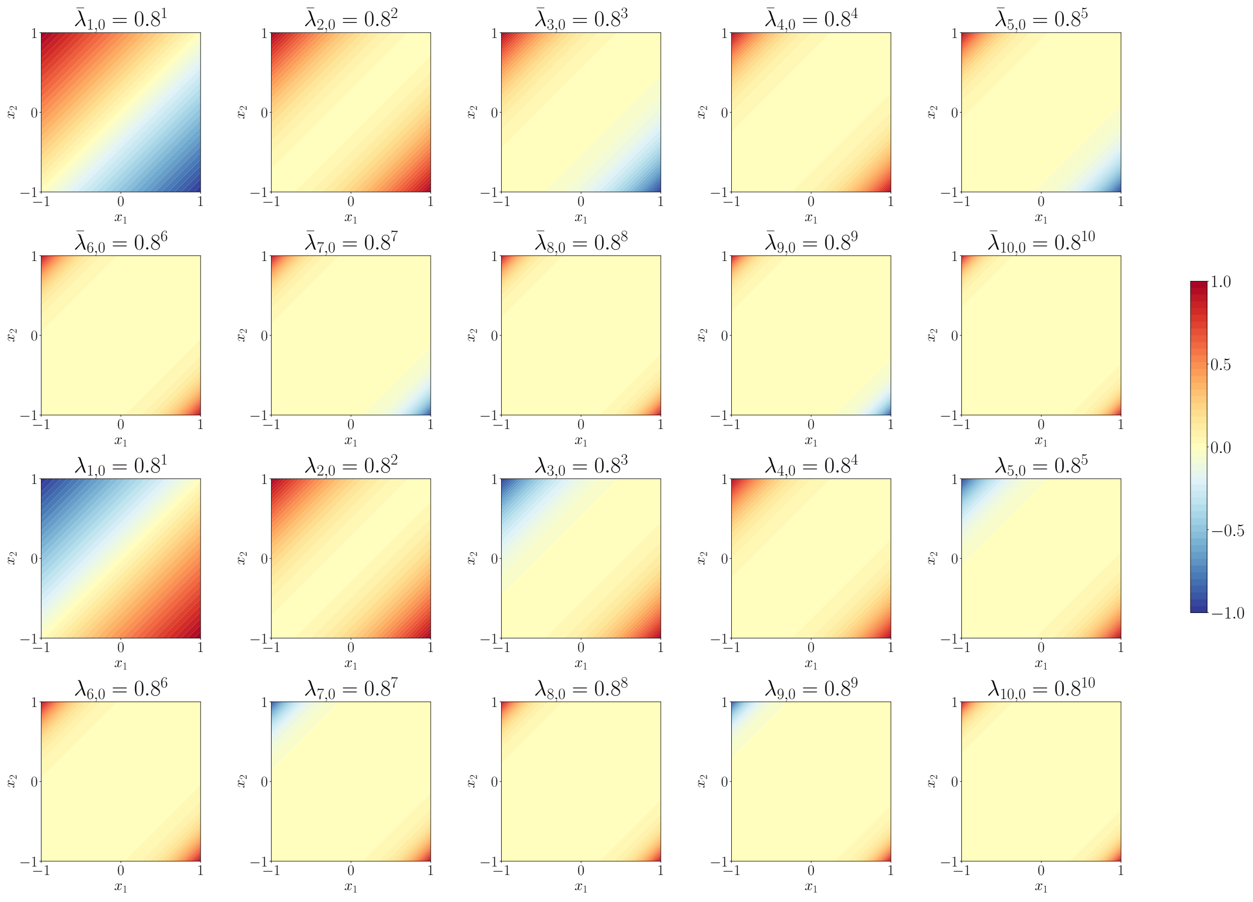

**Figure 3.10** Comparison of explicit eigenfunctions with computed eigenfunctions approximated using EDMD eigenvectors for the discrete linear system (3.24). First and second rows show computed eigenfunction contours for powers $1 \leq p \leq 10$, and third and fourth rows show explicit eigenfunction contours for powers $1 \leq p \leq 10$. The eigenfunctions have been normalized such that $\|\phi\|_\infty \leq 1$.

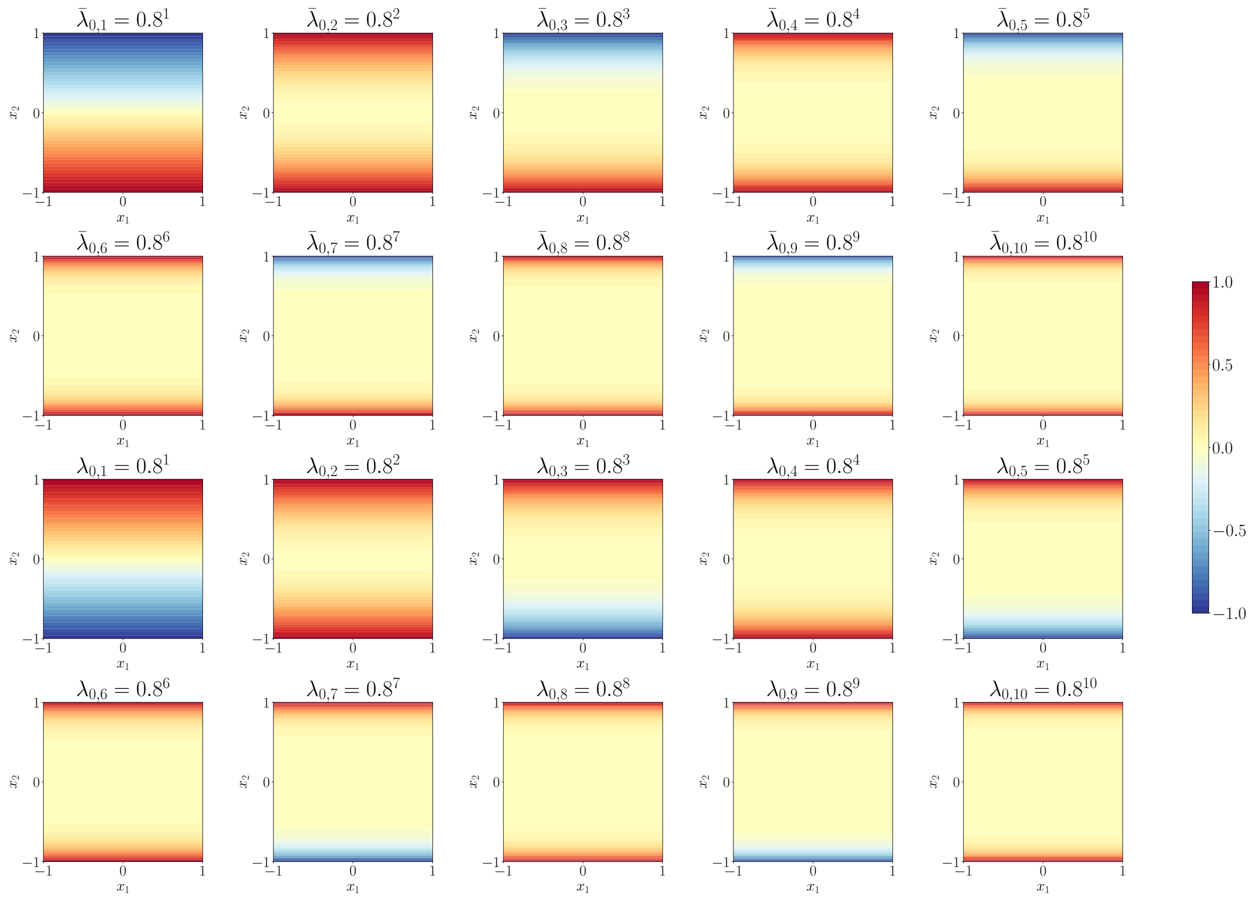

**Figure 3.11** Comparison of explicit eigenfunctions with computed eigenfunctions approximated using EDMD eigenvectors for the discrete linear system (3.24). First and second rows show computed eigenfunction contours for powers $1 \leq q \leq 10$, and third and fourth rows show explicit eigenfunction contours for powers $1 \leq q \leq 10$.. The eigenfunctions have been normalized such that $\|\phi\|_\infty \leq 1$.

| power p | residual | eig_eq_norm |
|---:|---:|---:|
| 1 | 5.141649e-25 | 3.293027e-16 |
| 2 | 5.696217e-24 | 3.052170e-16 |
| 3 | 3.080359e-24 | 2.865091e-15 |
| 4 | 3.131271e+01 | - |
| 5 | 5.452440e+00 | - |
| 6 | 3.885154e+01 | - |
| 7 | 1.109083e+01 | - |
| 8 | 3.533642e+01 | - |
| 9 | 1.352306e+01 | - |
| 10 | 3.011819e+01 | - |

| power q | residual | eig_eq_norm |
|---:|---:|---:|
| 1 | 9.244998e-27 | 9.897038e-17 |
| 2 | 2.614065e-23 | 1.426294e-15 |
| 3 | 1.318227e-22 | 1.230384e-14 |
| 4 | 8.541984e+01 | - |
| 5 | 1.611994e+01 | - |
| 6 | 1.544213e+02 | - |
| 7 | 4.471709e+01 | - |
| 8 | 1.887074e+02 | - |
| 9 | 7.020917e+01 | - |
| 10 | 2.046077e+02 | - |

**Figure 3.12** Residual for problem (3.8) for different powers of p and q of EDMD eigenfunctions of the discrete linear system (3.24). The eigen equation norm checks if the computed vector in (3.8) is a left eigenvector of the Koopman matrix.

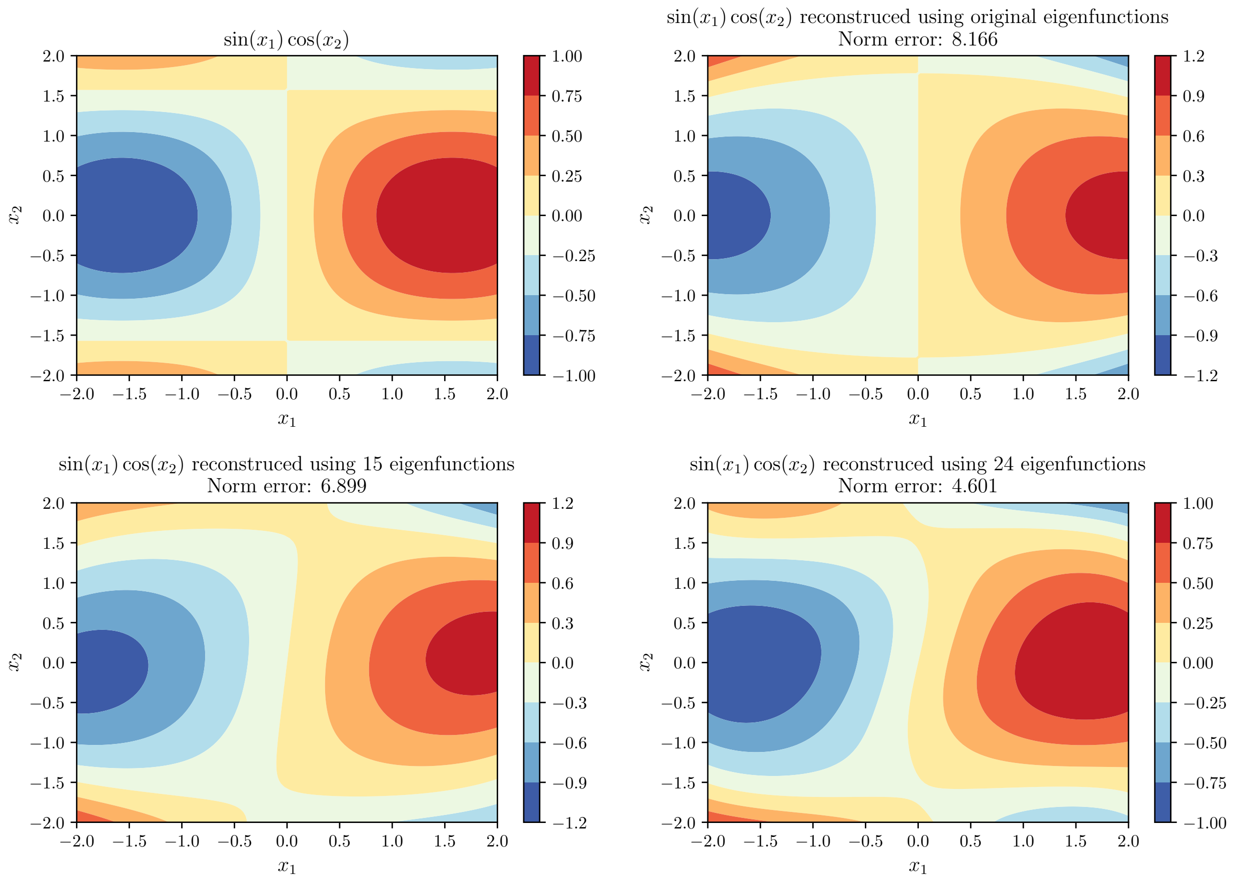

**Figure 3.13** Reconstruction of the observable $\sin(x_1)\cos(x_2)$ using the EDMD eigenfunctions of the discrete linear system (3.24). Top left: $\sin(x_1)\cos(x_2)$. Top right: Observable reconstructed using original set of EDMD eigenfunctions. Bottom left, bottom right: Observable reconstructed using extended set of eigenfunctions.

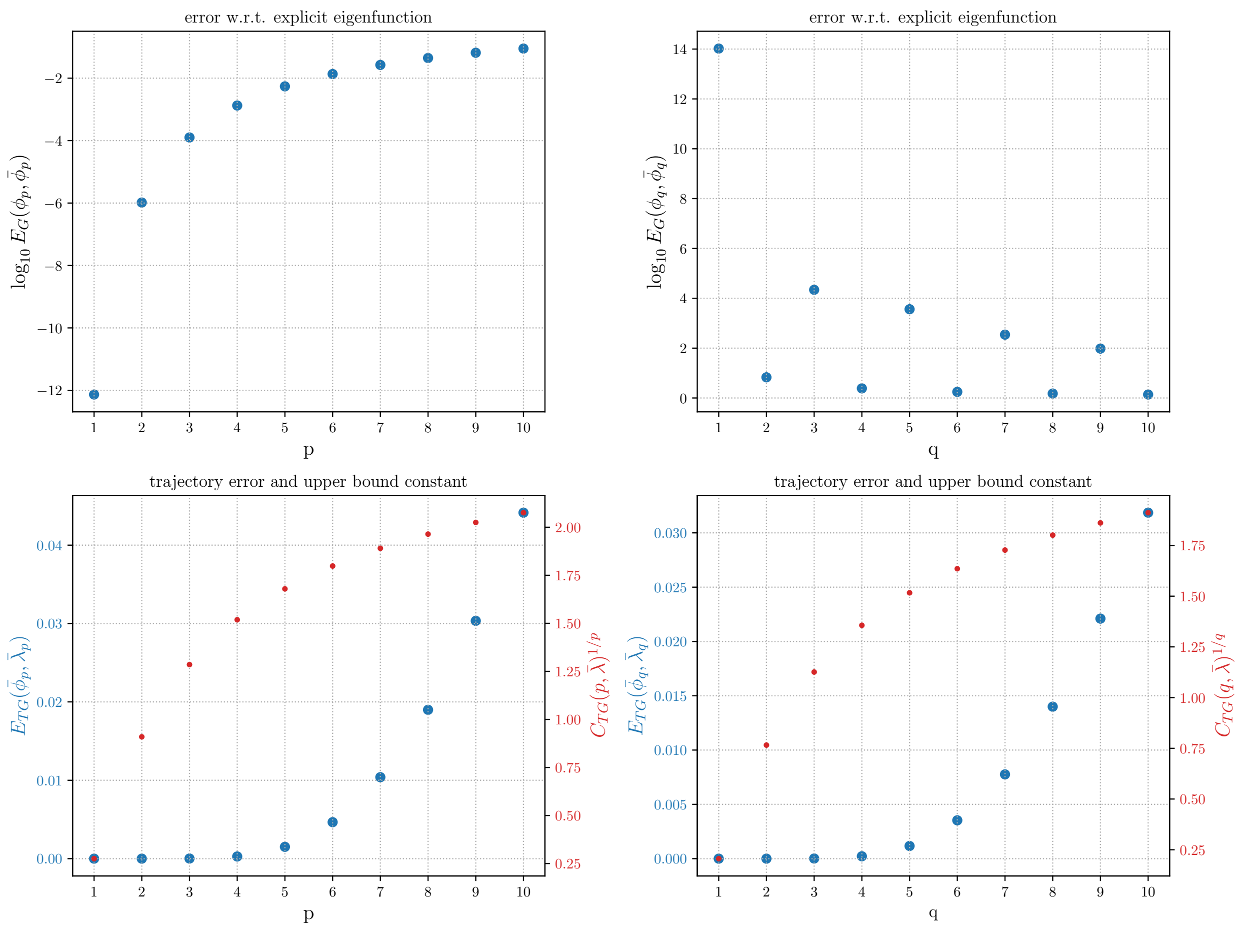

**Figure 3.14** Error analysis for EDMD eigenfunctions of discrete linear system (3.24). Top row: $\log_{10}$ error between computed and explicit eigenfunctions on grid G ($E_G$). Bottom row: Trajectory error for computed eigenfunctions on grid G – $E_{TG}$(blue) and constant of the upper bound – $C_{TG}$(red).

We can compare the extended eigenfunction with the explicit eigenfunctions. Let $\bar{\phi}_p$ be an extended eigenfunction and $\phi_p$ be an explicit eigenfunction. Then on a grid G as defined in (3.4), the error given by $E_G$ is defined similarly to the discrete system:

$$G_0 = \{x \in G|\ |\phi_p| < \varepsilon\} \cup \{x \in G|\ |\bar{\phi}_p| < \varepsilon\} \tag{3.43}$$

$$c_{mode} = mode\left(\frac{\phi_{p|G/G_0}}{\bar{\phi}_{p|G/G_0}}\right) \tag{3.44}$$

$$E_G(\phi_p, \bar{\phi}_p) = \left\|\phi_p - c_{mode}\bar{\phi}_p\right\|_G^{1/p}, \tag{3.45}$$

where $\varepsilon$ is some tolerance.

The trajectory error for the computed eigenpair $(\bar{\lambda}_p, \bar{\phi}_p)$ is given by

$$E_{TG}(\bar{\lambda}_p, \bar{\phi}_p) := \left\|\bar{\phi}_p(T^{\Delta t}(\cdot)) - \bar{\lambda}_p\phi_p(\cdot)\right\|_G^{1/p}. \tag{3.46}$$

.

We use

$$A = \begin{bmatrix} -0.9 & 0.1 \\ 0 & -0.8 \end{bmatrix}. \tag{3.47}$$

The matrix has left eigenpairs $(-0.9, [1, -1]^T)$ and $(0.8, [0, 1]^T)$. The system is sampled with sampling interval $\Delta t$. Therefore, the explicit Koopman eigenfunctions of the system are given by

$$\phi_p(x) = \left(\frac{x_1 - x_2}{\sqrt{2}}\right)^p \tag{3.48}$$

$$\phi_q(x) = x_2^q, \tag{3.49}$$

with eigenvalues

$$\lambda_p = (e^{-0.9\Delta t})^p \tag{3.50}$$

$$\lambda_q = (e^{-0.8\Delta t})^q. \tag{3.51}$$

To approximate the eigenfunctions using DMD we collect 400 snapshot pairs where the initial conditions are uniformly randomly distributed between $[-2, 2] \times [-2, 2]$, and sampling interval $\Delta t = 0.2$

Figure 3.15 shows the results of the DMD approximation. The out-of-sample prediction shows that the DMD approximation is able to predict trajectories accurately.

We take the grid $G$ with $a = -1, b = 1, n = 100, h = 0.01$. On grid G, we calculate the flow approximated by Euler method using step size $h = 0.001$. Then we calculate $\epsilon_G$ using (3.16) using the true solution,

$$x^{\Delta t} = e^{A\Delta t}x, \tag{3.52}$$

and approximated solution using Euler method:

$$\begin{aligned} x^{(0)} &= x \\ N &= \frac{\Delta t}{h} \\ x^{(i+1)} &= x^{(i)} + hAx^{(i)},\ i = 0, \dots, N \\ T^{\Delta t}(x) &= x^{(N)}. \end{aligned}$$

We then compute the trajectory error for increasing powers, $p$ and $q$ and their upper bound. For DMD, as $\Psi = I$, upper bound, L for $\|J_\Psi(x)\|_2 \leq L$ where L = 1. Figure 3.16 shows the trajectory error and upper bound with respect to the euler integration error for extended eigenfunctions $\bar{\phi}_p$ and $\bar{\phi}_q$.

Assuming that flow $T^{\Delta t}(x) = e^{A\Delta t}x$ we can calculate the trajectory error with respect to eigenvector error by adding a random error vector $\delta v$ with $\|\delta v\| = 10^{-6}$. Figure 3.17 shows the trajectory error and the upper bound of the trajectory error due to this error in the eigenvector for extended eigenfunctions $\bar{\phi}_p$ and $\bar{\phi}_q$.

Finally, we try Algorithm 5 described in the last section using the trajectory errors and upper bounds with respect to integration error to extend eigenfunctions $\bar{\phi}_p$ and $\bar{\phi}_q$ up to powers $p$ and $q$ such that the trajectory error stays below a desired upper bound $\epsilon = 0.2$. Figure 3.18 shows the results of Algorithm 5 for a desired trajectory error upper bound $\epsilon = 0.1$. As seen in the figure, the powers $p$ and $q$ suggested by the algorithm are close to the actual powers up to which the eigenfunctions can be extended.

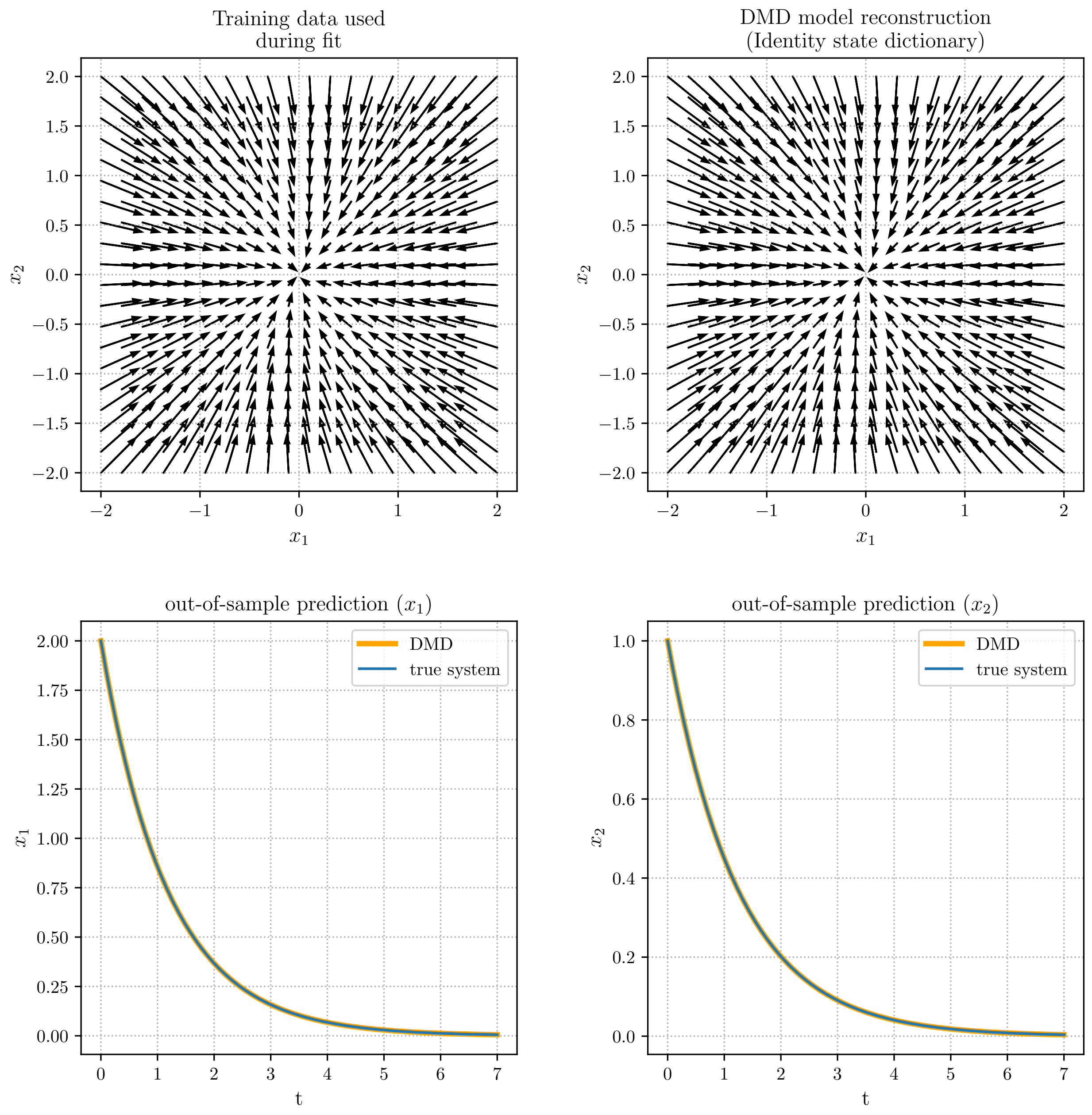

**Figure 3.15** DMD model for the continuous linear system (3.38). Top left shows the training data used for approximation, top right shows the reconstructed data, bottom shows the comparison between predicted trajectory and actual trajectory for one initial condition.

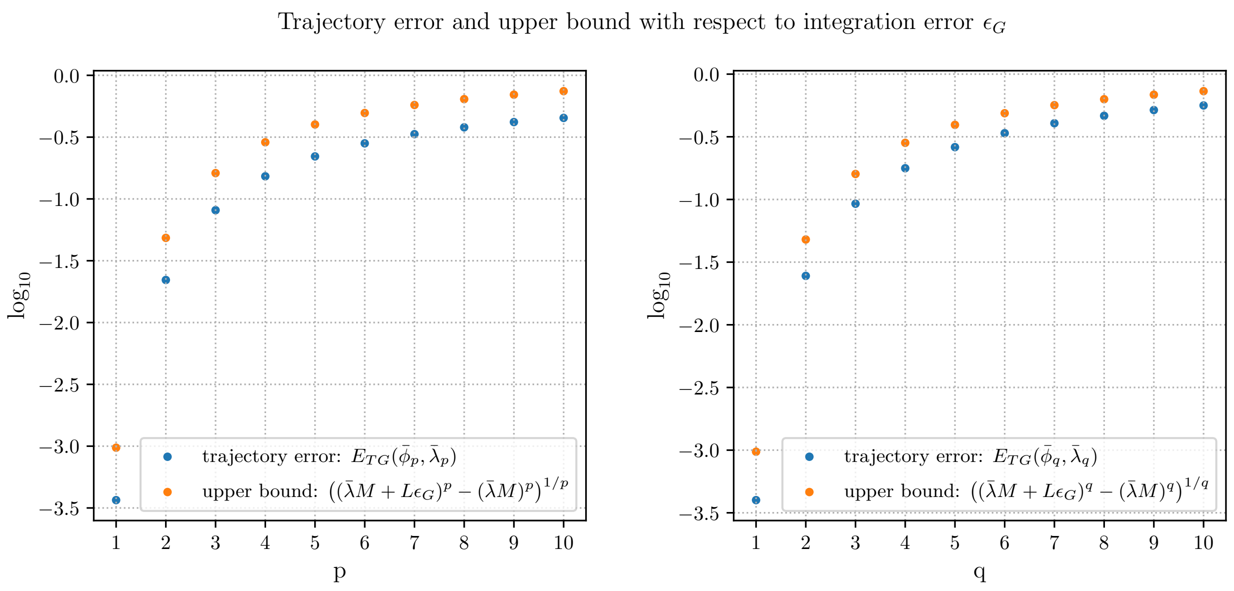

**Figure 3.16** Integration error analysis for DMD eigenfunctions of continuous linear system (3.38). The trajectory error for computed eigenfunctions (blue) and upper bound (orange) with respect to the Euler integration error $\epsilon_G$ for powers $p$ and $q$.

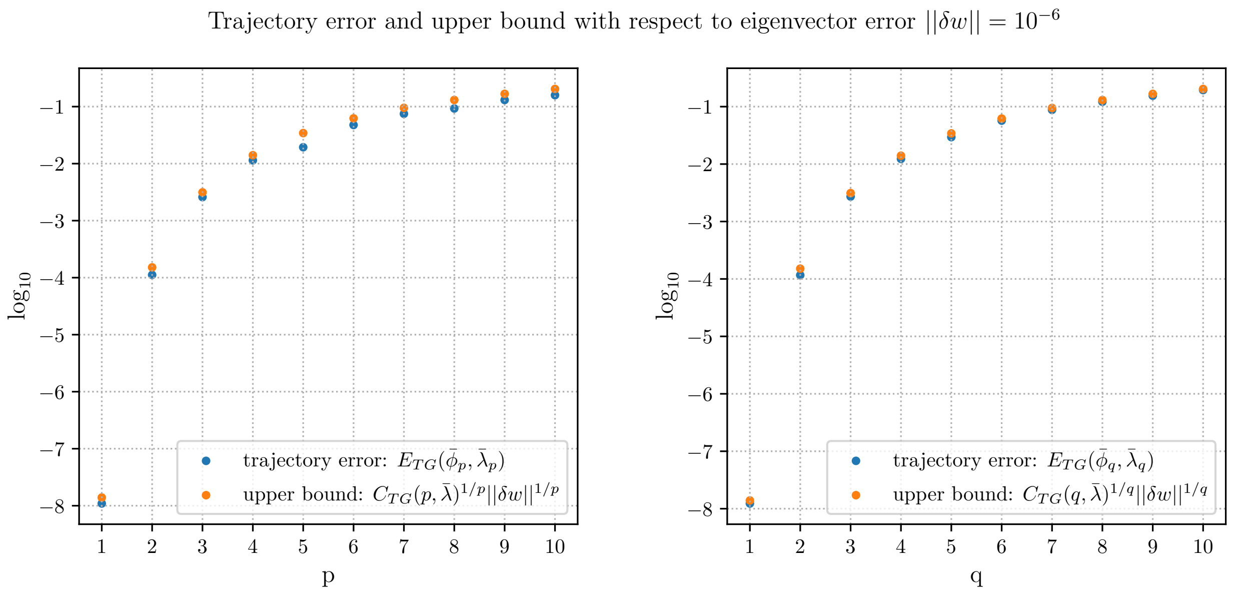

**Figure 3.17** Eigenvector error analysis for DMD eigenfunctions of continuous linear system (3.38). The trajectory error for computed eigenfunctions (blue) and upper bound (orange) with respect to the eigenvector error $\|\delta w\| = 10^{-6}$ for powers $p$ and $q$.

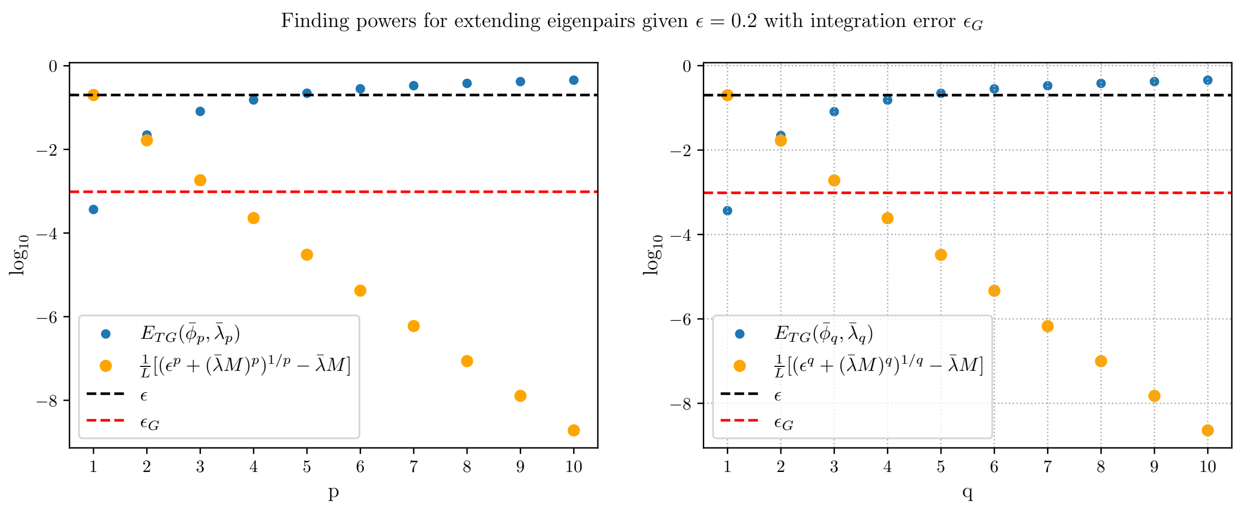

**Figure 3.18** Results of Algorithm 5 applied to the DMD approximation of continuous linear system (3.38) with Euler integration error $\epsilon_G$ and desired trajectory error $\epsilon = 0.1$. The value of $p$ and $q$ suggested by the algorithm– where the upper bound for $\epsilon_G$ (orange) crosses $\epsilon_G$ (red line) is close to actual value of $p$ and $q$ where the trajectory error (blue) crosses the required $\epsilon$ (black).

## 3.4 An iterative Koopman eigensolver algorithm

Now that we have developed an algorithm for deciding powers up to which extended eigenfunctions can be computed given a single eigenvector, we develop an iterative algorithm to find eigenfunctions and extend them iteratively. Since this requires an iterative eigensolver algorithm, we develop a deflation-based algorithm for general non-Hermitian real matrices.

### 3.4.1 Deflation based iterative algorithm for general matrices

We start with the familiar power method and use the deflation technique described in Lemma 1 to find all the eigenvalues and eigenvectors of a matrix. Assuming $A \in \mathbb{R}^{nxn}$ is a non-Hermitian real matrix and it has a complete set of eigenpairs $\{(\lambda_i, v_i)\}_{i=1}^{n}$ with $\lambda_i \in \mathbb{R}$ then Algorithm 6 computes all the eigenpairs.

**Algorithm 6** Deflation based algorithm for real non-Hermitian matrix $A$ with real eigenvalues

```
$i \leftarrow 0$.
while $i < n$ do
    $\lambda_i, v_i \leftarrow$ power iteration$(A)$.
    $\lambda_i, w_i \leftarrow$ power iteration$(A^T)$.
    $w_i \leftarrow \frac{w_i}{w_i^T v_i}$.
    $A \leftarrow A - \lambda_i v_i w_i^T$.
    $i \leftarrow i + 1$.
end while
```

As discussed in Section 2.3.1, the power method does not converge for real matrices with dominant eigenvalues that are complex conjugate pairs. Therefore, we develop an Arnoldi-based method to deal with such matrices.

In every iteration, we use an Arnoldi algorithm with modified Gram-Schmidt orthogonalisation and subspace dimension $m = 2$ to get a 2x2 Hessenberg matrix $H$ and orthonormal matrix $V$. We can then explicitly get the eigenvector corresponding to the largest eigenvalue ($\lambda_m ax$) of $H$. Let this be $h$. Then, $Vh$ is an eigenvector of $A$. We then use this $Vh$ eigenvector of $H$ to restart the Arnoldi algorithm. To make the algorithm work for a matrix with real eigenvalues, we run a few steps of power iteration before the Arnoldi steps. The pseudocode is given in Algorithm 7.

We run a similar algorithm on $A^T$ to get the left eigenvector. We use the same algorithm as before, with the power method replaced by this new algorithm for complex conjugate pair eigenvalues. After we get a complex eigenpair, we run the deflation step with $(\lambda, v)$ and $(\bar{\lambda}, \bar{v})$.

Algorithm 8 describes this deflation algorithm. It is similar to the deflation algorithm described in Algorithm 6, except we use Algorithm 7 instead of the power method.

### 3.4.2 Iterative algorithm for extending Koopman eigenpairs based on integration error

We combine Algorithm 8 with Algorithm 5 to develop an algorithm that extends eigenfunctions based on the integration error bound. The resulting algorithm is given by Algorithm 9, and $\bar{\phi}_p^i$ is the extended $p$th power eigenfunction with eigenvalue $\bar{\lambda}_p^i$ for the $i$th eigenvalue of the Koopman matrix.

**Algorithm 7** (power iteration complex) Arnoldi-based power iteration algorithm for real non-Hermitian matrix $A$ with complex dominant eigenvalues $\lambda_{max}, \bar{\lambda}_{max}$ and eigenvectors $v, \bar{v}$, given tolerance and maximum iterations $N$

```
$x \leftarrow$ power iteration($A$, max iterations = 500).
$\lambda_o \leftarrow \infty$.
$i \leftarrow 0$.
**while** $i < N$ **do**
    $V[:, 0] \leftarrow x$.
    $h, V \leftarrow$ modified Gram-Schmidt($A, V$, degree = 2).
    $\lambda_1, \lambda_2 \leftarrow$ eigenvalue2D($h$).
    $k \leftarrow \arg\max_{1,2}\{|\lambda_1|, |\lambda_2|\}$.
    $\lambda_{max} \leftarrow \lambda_k$
    $w \leftarrow$ eigvector2D($h, \lambda_{max}$).
    $v \leftarrow V[:\ 0 : 2]w$.
    $v \leftarrow \frac{v}{||v||}$.
    **if** $|\lambda_{max} - \lambda_o| <$ tolerance **then**
        **break**.
    **else**
        $\lambda_o \leftarrow \lambda_{max}$.
    **end if**
    $i \leftarrow i + 1$.
**end while**
```

**Algorithm 8** Deflation based algorithm for computing left eigenpairs $\{(\lambda_i, w_i)\}$ of a real non-Hermitian matrix $A$ with complex eigenvalues

```
$i \leftarrow 0$.
**while** $i < n$ **do**
    $\lambda_i, v_i \leftarrow$ power iteration complex($A$).
    $\lambda_i, w_i \leftarrow$ power iteration complex($A^T$).
    $w_i \leftarrow \frac{w_i}{w_i^T v_i}$.
    $A \leftarrow A - \lambda_i v_i w_i^T$.
    **if** imag($\lambda_i$) $> 10^{-6}$ **then**
        $v_{i+1} \leftarrow \bar{v}_i$.
        $w_{i+1} \leftarrow \bar{w}_i$.
        $\lambda_{i+1} \leftarrow \bar{\lambda}_i$.
        $w_{i+1} \leftarrow \frac{w_{i+1}}{w_{i+1}^T . v_{i+1}}$
        $A \leftarrow A - \lambda_{i+1} v_{i+1} w_{i+1}^T$.
        $i \leftarrow i + 1$
    **end if**
    $i \leftarrow i + 1$
**end while**
```

**Algorithm 9** (Iterative Koopman eigensolver) Algorithm for computing extending eigenvalues $\bar{\lambda}_p^i$ and extended eigenfunctions $\bar{\phi}_p^i$ of a continuous system, given integration error $\epsilon_G$, desired trajectory error $\epsilon$, the Koopman matrix $K$, $\Psi$ as the dictionary basis and constant $L$

```
$i \leftarrow 0$.
while $i < n$ do
    $\bar{\lambda}_i, v_i \leftarrow$ power iteration complex$(K)$.
    $\bar{\lambda}_i, w_i \leftarrow$ power iteration complex$(K^T)$.
    while $p \in \mathbb{N}$ do
        if $\frac{1}{L}((\epsilon^p + (\bar{\lambda}_i M)^p)^{1/p} - \bar{\lambda}_i M) > \epsilon_G$ then
            break
        end if
        $\bar{\phi}_p^i \leftarrow (w_i^T \psi)^p$.
        $\bar{\lambda}_p^i = (\bar{\lambda}_i)^p$.
    end while
    $w_i \leftarrow \frac{w_i}{w_i^T v_i}$.
    $A \leftarrow A - \lambda_i v_i w_i^T$.
    if imag$(\lambda_i) > 10^{-6}$ then
        $v_{i+1} \leftarrow \bar{v}_i$.
        $w_{i+1} \leftarrow \bar{w}_i$.
        $\lambda_{i+1} \leftarrow \bar{\lambda}_i$.
        $w_{i+1} \leftarrow \frac{w_{i+1}}{w_{i+1}^T v_{i+1}}$.
        $A \leftarrow A - \lambda_{i+1} v_{i+1} w_{i+1}^T$.
        $i \leftarrow i + 1$.
    end if
    $i \leftarrow i + 1$.
end while
```

### 3.4.3 Example – non-linear system

We apply the algorithm developed in the previous section on a continuous non-linear system.
Consider the continuous non-linear system

$$\begin{aligned}\dot{x_1} &= x_1 \\ \dot{x_2} &= -x_2 + x_1^2.\end{aligned} \tag{3.53}$$

The system has the explicit solution given by

$$\varphi^t(x) = \begin{bmatrix} x_1 e^t \\ (x_2 - \frac{x1^2}{3})e^{-t} + \frac{x1^2}{3}e^{2t} \end{bmatrix}. \tag{3.54}$$

The Koopman eigenfuncions and eigenvalues of the system are given by

$$\phi_1(x) = x_1 \tag{3.55}$$

$$\phi_2(x) = x_2, \tag{3.56}$$

with eigenvalues $(e^{\Delta t}, e^{2\Delta t})$ where $\Delta t$ is the sampling time.
We sample the system using the exact solution with $\Delta t = 0.02$, collecting 400 snapshot pairs with uniformly randomly distributed initial conditions between $[-2, 2] \times [-2, 2]$. Then, we perform EDMD with $P^5$ dictionary basis to get the Koopman matrix $K$. Figure 3.19 shows the result of the EDMD approximation. The out-of-sample prediction shows that the approximation predicts trajectories accurately. We use the grid G with $a = -2, b - 2, n = 100, h = 0.01$. We calculate $\epsilon_G$ using the exact solution and the *RK23* solver [BS89]. The desired upper bound on trajectory error is set as $\epsilon = 0.1$.
Then we run Algorithm 9 to get the first 5 eigenpairs. 3 out of the 5 eigenpairs are powers of the explicit eigenfunctions and generate more explicit eigenfunctions. Figure shows 3.20 the extended eigenfunctions calculated using the algorithm and the corresponding exact eigenfunctions that match them.

### 3.4.4 Example – non-linear system constructed from linear system

Consider the linear continuous system

$$\dot{x} = Ax, \tag{3.57}$$

where $A = \begin{bmatrix} -0.9 & 0.1 \\ 0 & -0.8 \end{bmatrix}$.
We transform using the diffeomorphism $y = h(x) = log(e^x + 1)$ to get the new system

$$\dot{y} = \begin{bmatrix} 1 - e^{-y_1} & 0 \\ 0 & 1 - e^{-y_2} \end{bmatrix} Alog(e^y - 1). \tag{3.58}$$

Using Proposition 2, the explicit eigenfunctions of this non linear system are then given by $(\phi \circ h^{-1})$ with eigenvalue $\lambda$, where $\phi$ is an eigenfunction of the linear system (3.57) with eigenvalue $\lambda$.
As the eigenfunctions of the linear system are given by $\phi_i(x) = w_i^T x$ with eigenvalue $\lambda_i$, $(i = 1, 2)$ where $w_i$ is left eigenvector of $A$ with eigenvalue $\lambda_i$, the eigenfunctions of system (3.58) are given by

$$\phi_i^{nonlin}(y) = \phi_i \circ h^{-1}(y) = w_i^T log(e^y - 1) \tag{3.59}$$

with eigenvalue $\lambda_i$ for $i = 1, 2$.
We sample the linear system using the exact solution with $\Delta t = 0.02$, collecting 400 snapshot pairs with initial conditions uniformly randomly distributed between $[-2, 2] \times [2, 2]$. Then, we transform the sampled data using the diffeomorphism $h$. We use this transformed data to perform EDMD with a radial basis function (RBF) dictionary with 40 RBF Gaussian kernel functions with centers calculated using k-means clustering of the transformed data.

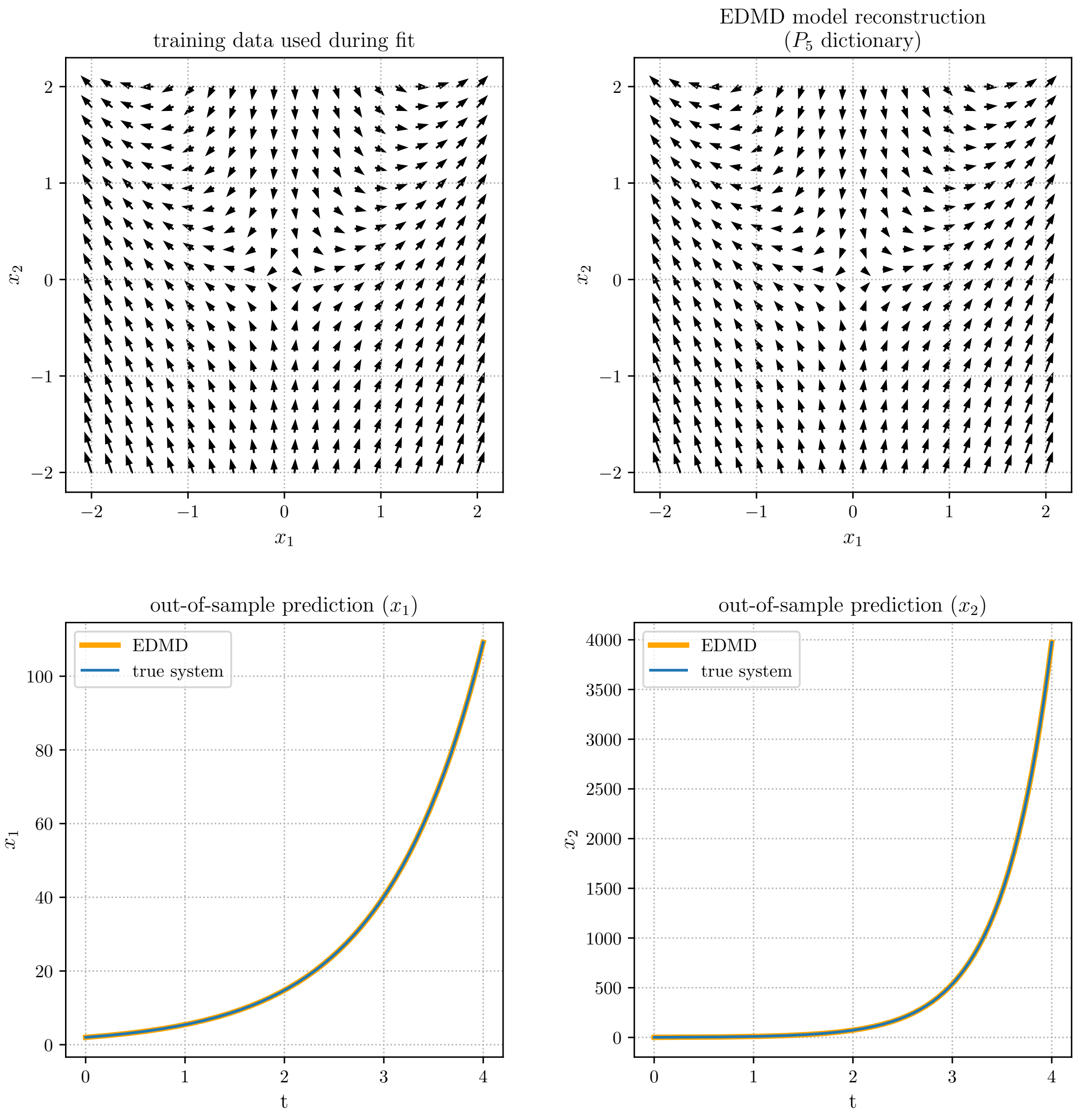

**Figure 3.19** EDMD model for the non-linear linear system (3.53). Top left shows the training data used for approximation, top right shows the reconstructed data, bottom shows the comparison between predicted trajectory and actual trajectory for one initial condition.

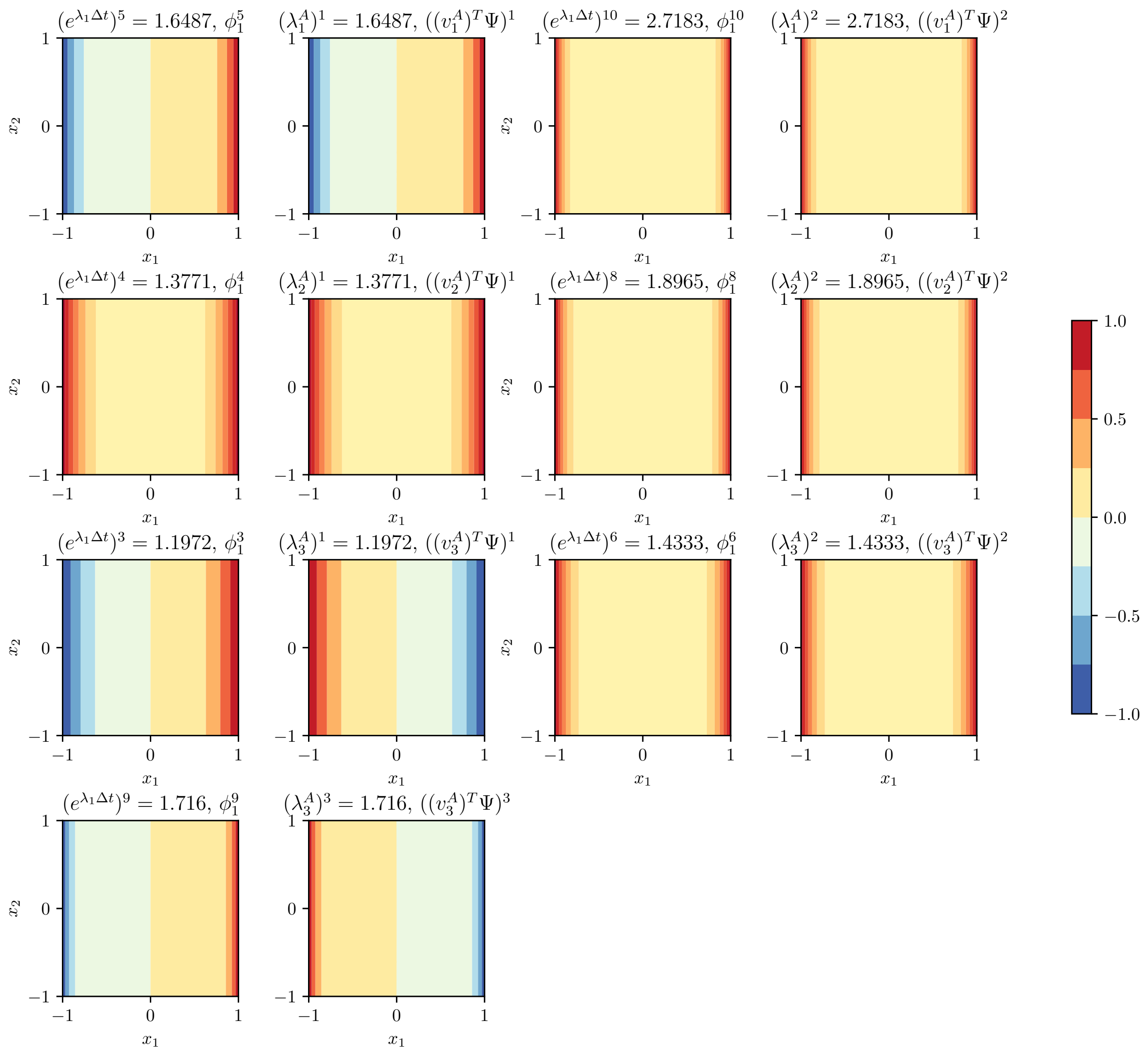

**Figure 3.20** Eigenfunction and eigenvalues for the non-linear system (3.53). The first and third columns show the explicit eigenvalues and eigenfunctions. The second and fourth columns show the extended eigenvalues($\lambda^A$) and eigenfunctions($(v^A)^T\Psi$) calculated using Algorithm 9 that match the explicit eigenfunctions.

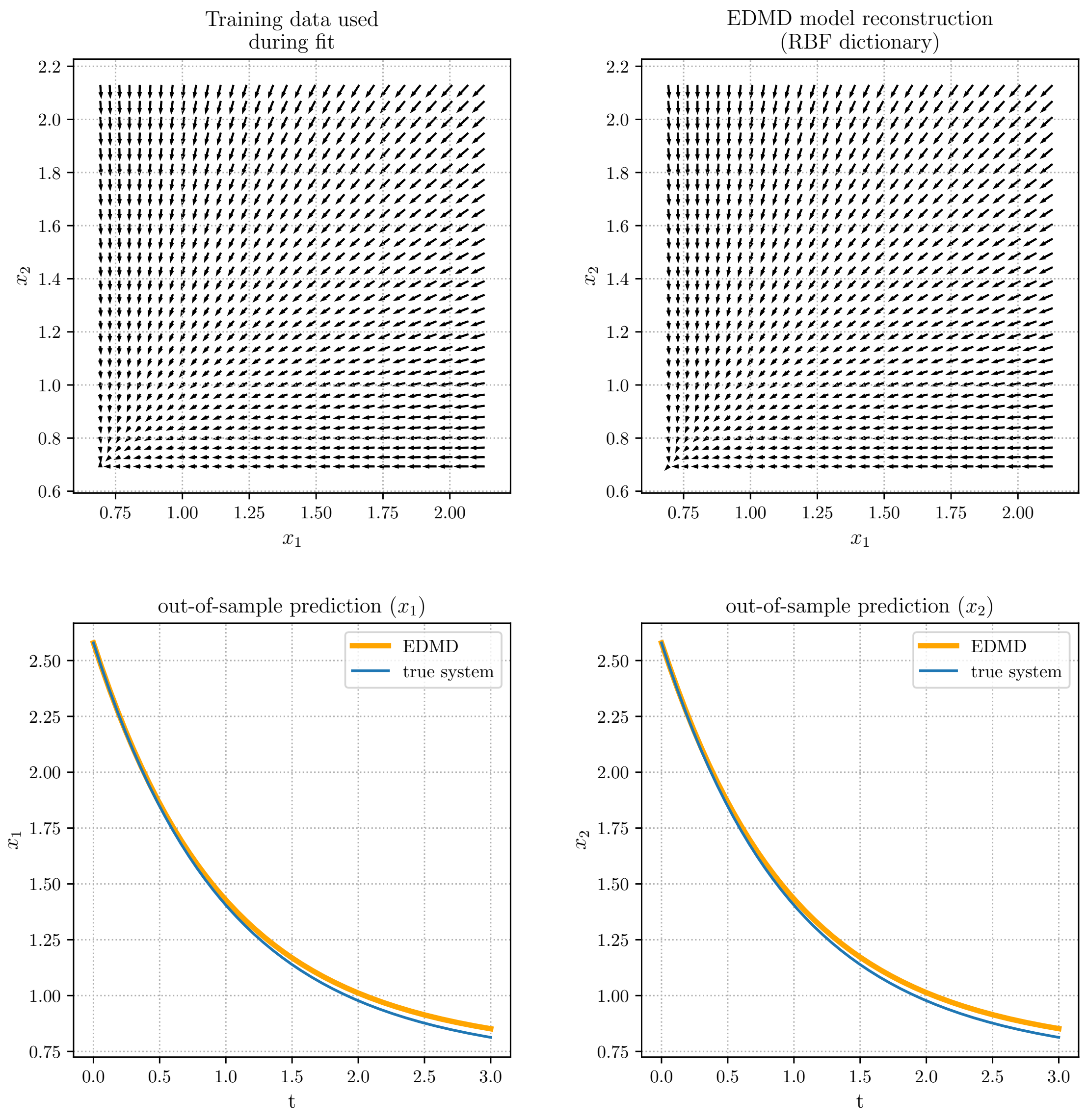

**Figure 3.21** EDMD model for the non-linear system (3.58). Top left shows the training data used for approximation, top right shows the reconstructed data, bottom shows the comparison between predicted trajectory and actual trajectory for one initial condition.

We take the grid $G$ with $a = 1, b = 2, n = 100, h = 0.01$. Some of the explicit eigenfunctions on grid $G$ are shown in Figure 3.22, and the computed EDMD eigenfunctions are shown in Figure 3.23. Then we calculate $\epsilon_G$ by integrating over the grid and using the integration method *RK45* with the explicit system (3.58) [DP80]. The calculation of upper bound, $L$ for the spectral norm of Jacobian for the RBF functions is shown in Appendix A.3.1.

Then, we use the Koopman eigensolver algorithm defined in Algorithm 9 to extend the eigenfunctions of the system. The desired trajectory error is set to $\epsilon = 0.01$. We calculate the first nine eigenpairs of the Koopman matrix. We use the algorithm to get up to $p = 3$ extended eigenfunctions for the first nine eigenfunctions with trajectory error less than $\epsilon$. Figure 3.24 shows the spectrum and the powers up to which each eigenpair can be extended. Figure 3.25 shows some of the extended eigenfunctions. The extended eigenfunctions do not match the explicit eigenfunctions in this case. This is expected as the EDMD eigenfunctions can differ from the explicit eigenfunctions, as the transformed non-linear system can have an infinite number of independent eigenfunctions.

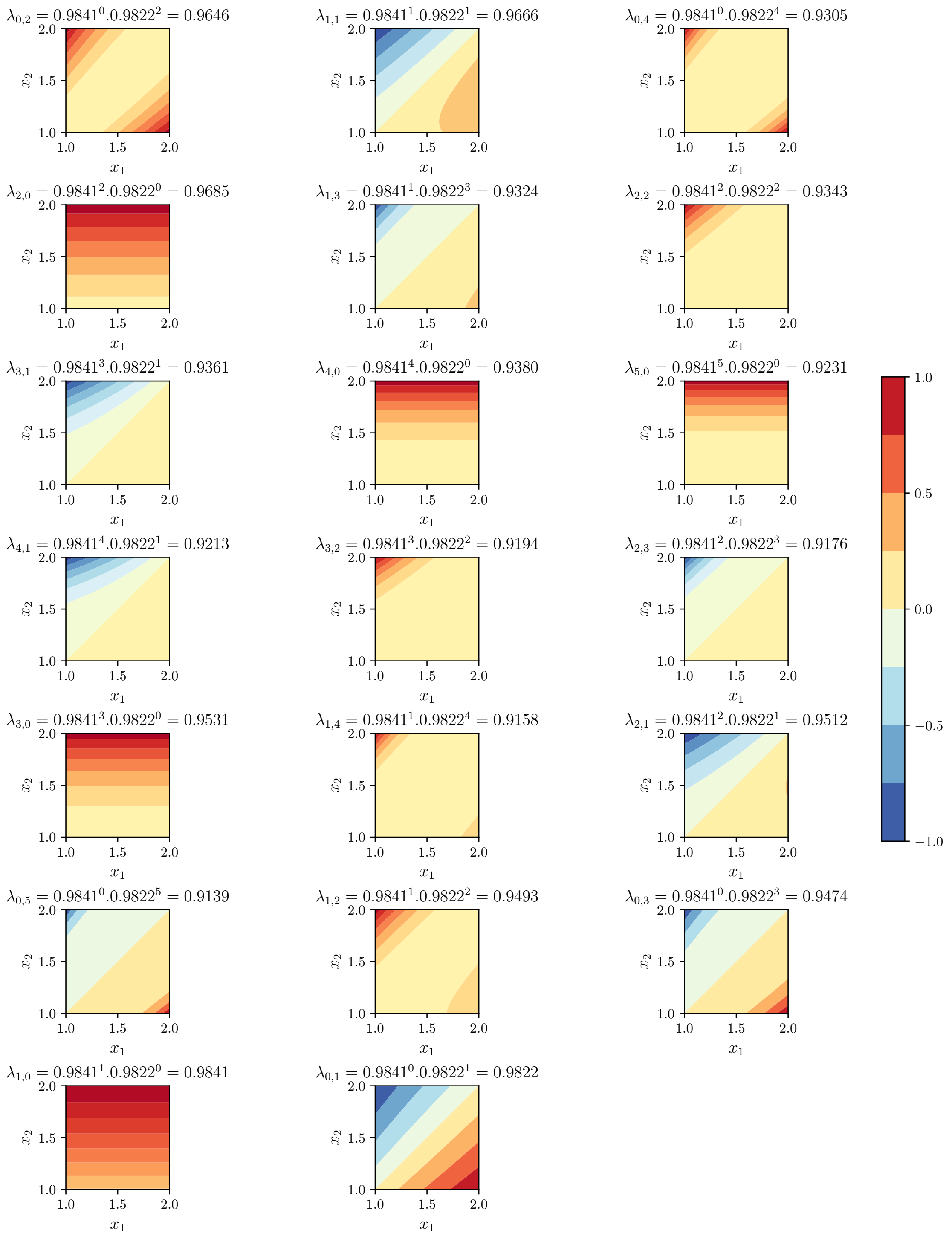

**Figure 3.22** Explicit eigenfunctions for the non-linear system (3.58) computed using $\phi \circ h^{-1}$.

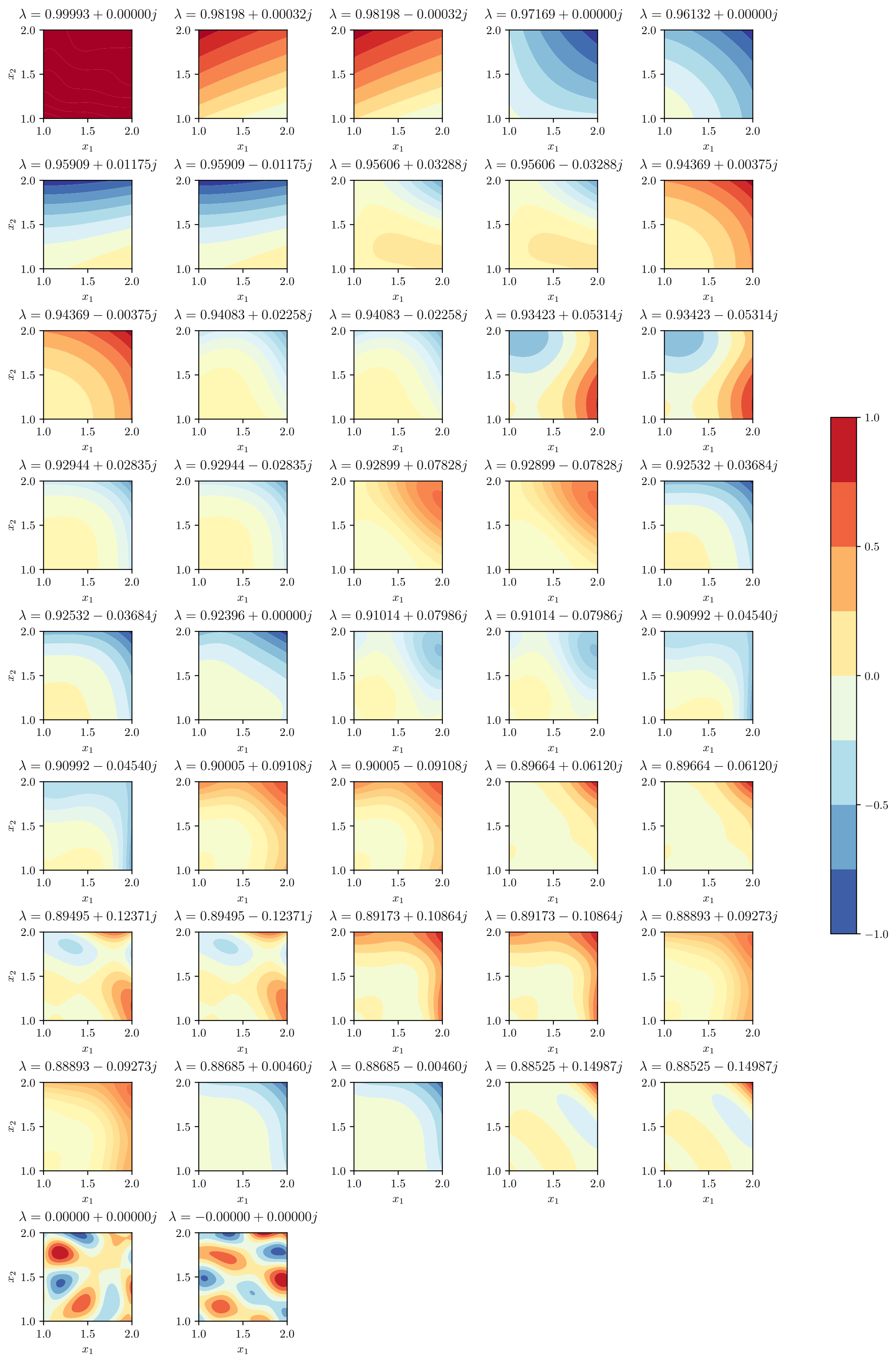

**Figure 3.23** Eigenfunctions for the non-linear system (3.58) approximated using EDMD.

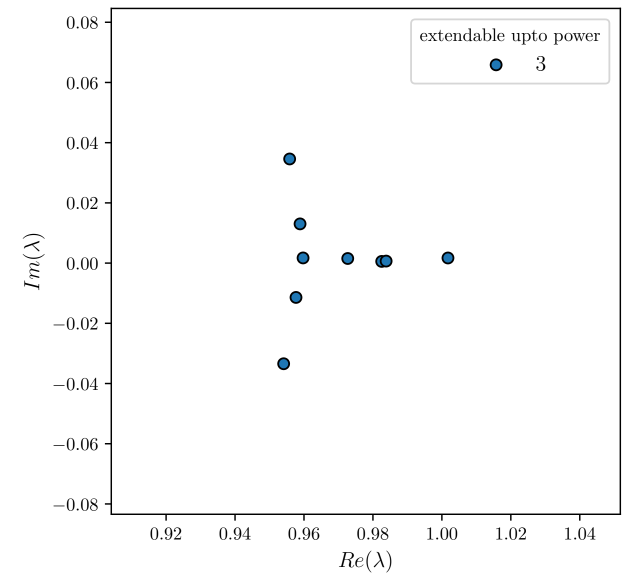

**Figure 3.24** Spectrum computed using Algorithm 9 for the non-linear system (3.58) and powers up to which the eigenfunctions can be extended for first 9 eigenvalues.

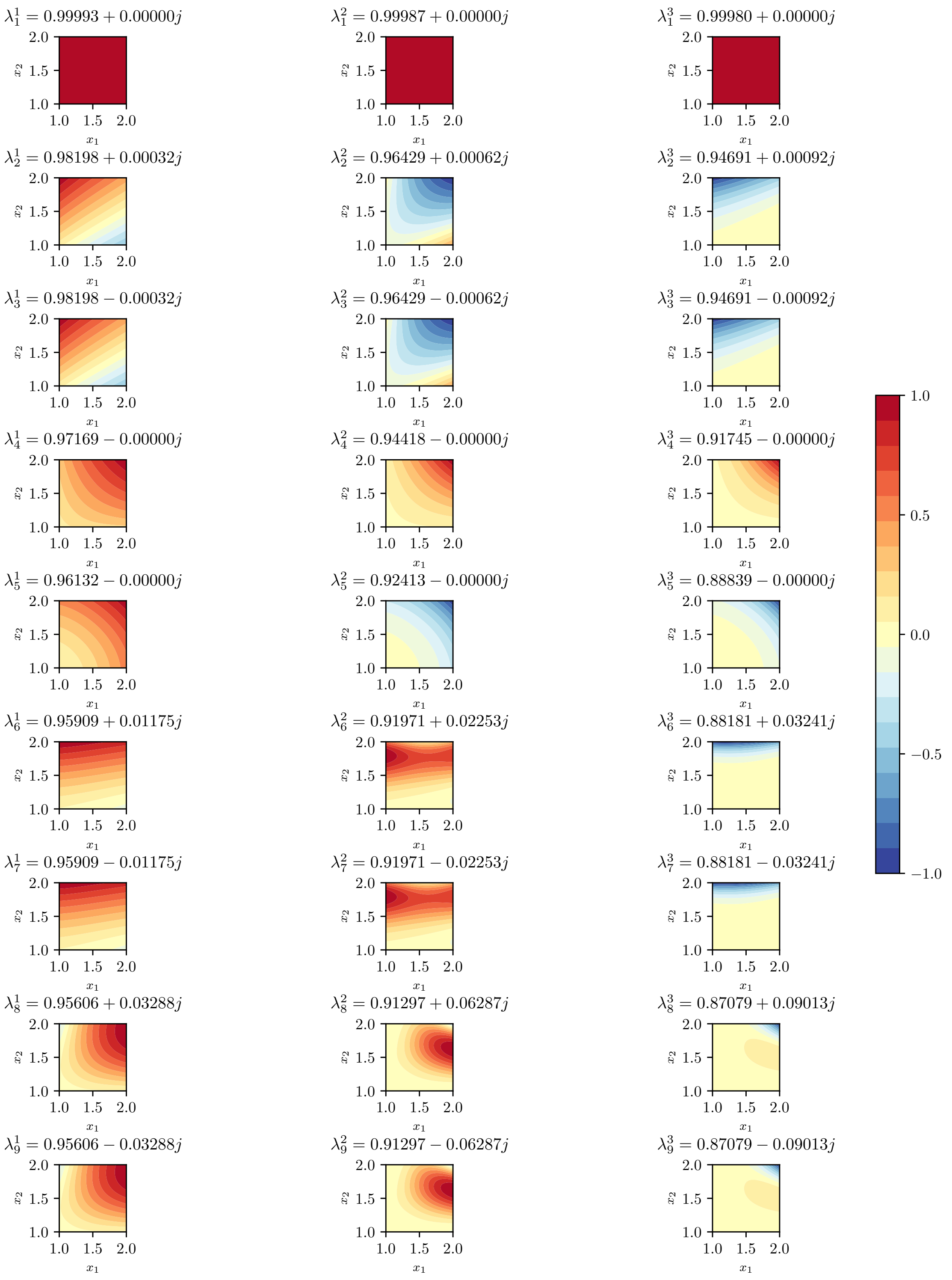

**Figure 3.25** Extended eigenfunctions for the first 9 eigenvalues computed using Algorithm 9 for the non-linear system (3.58).

# 4 Conclusion

This thesis develops methods and analysis for extending eigenfunction approximations of the Koopman operator using the multiplicative property. We presented a measure of error for extended eigenfunctions - *trajectory error* and derived error bounds for it based on eigenvector error and integration error. We then describe an algorithm for extending eigenfunctions based on these error bounds and apply it to discrete and continuous linear systems. We see that as the power of eigenfunctions increases, the trajectory error also increases, and the upper bound stays close to the trajectory error. Moreover, the extended eigenfunctions of the linear system match the explicitly computed eigenfunctions. Additionally, we demonstrated that observables can be reconstructed more accurately using the extended set of eigenfunctions. We also showed that the powers suggested by the algorithm are close to the actual powers up to which the eigenfunction can be extended for a desired error bound. To overcome the limitations of the power method on general non-Hermitian real matrices, we developed an iterative algorithm based on the power method and Arnoldi iteration for computing eigenvalues and eigenvectors. This algorithm is then combined with the upper bound algorithm suggested previously to develop an iterative eigensolver for the Koopman approximation that can be used to extend eigenfunctions. We demonstrated the applicability of this algorithm on non-linear dynamical systems and showed that the algorithm can generate extended eigenfunctions.

We list some limitations and suggest directions for future work.

- The deflation technique used in the Algorithm 9 can be numerically unstable. Deflation techniques, in general, are not recommended for computing more than a few eigenvalues as they accumulate errors over every step and can cause problems if the current eigenvalue is poorly conditioned.
- The Arnoldi-based power method in Algorithm 7 can be analysed for its efficiency and stability. The power method can also suffer from convergence issues if the eigenvalues have similar magnitude. The inverse iteration can be used to remedy this, although inverting a large matrix is expensive.
- The error analysis of trajectory error for extended eigenfunctions in Section 3.2 was performed only for powers of eigenfunctions of the form $\phi^p$. Similar error bounds can also be derived for extended eigenfunctions of the form $\phi_1^p \phi_2^q$.
- The error analysis also ignores the error in eigenvalue approximation and only considers the error in eigenvectors. Eigenvalue error can also be taken into account to derive error bounds.
- The error analysis was performed independently for the eigenvector error and integration error. However, for continuous systems, both errors are present simultaneously.
- The eigensolver algorithm presented in Algorithm 9 can also be improved further by discovering new eigenvectors corresponding to an extended eigenfunction as described in Section 3.1.3, and using these vectors to *deflate* the matrix thereby leading to a smaller matrix in every iteration.
- The eigensolver algorithm presented in Algorithm 9 can also be used to *inflate* the Koopman matrix using new eigenvectors corresponding to extended eigenfunctions, thereby leading to a larger matrix that represents a more informative finite-dimensional approximation of the operator.
- The predictive power of the eigenfunctions generated by Algorithm 9 for the non-linear system in Section 3.4.4 can be evaluated by computing trajectories and comparing them to explicit solutions.
- The applicability of the eigensolver algorithm can be tested on high-dimensional non-linear dynamical systems.

This work demonstrated that the set of approximated eigenfunctions can be expanded by exploiting the multiplicative property of the operator. Combined with iterative algorithms for eigenvalue computation based on the power method and deflation, this method can generate many eigenfunctions and lead to efficient eigensolvers for systems with a high-dimensional Koopman matrix. To our knowledge, no other work exists that explicitly uses the spectral properties of the Koopman operator to design efficient eigensolver algorithms. As the data-driven analysis of dynamical systems gains favour, further developments of eigensolver algorithms for the Koopman operator are expected.

# Bibliography

[BS89] P. Bogacki and L. F. Shampine. “A 3(2) pair of Runge - Kutta formulas”. In: *Applied Mathematics Letters* 2 (1989), pp. 321–325.

[Bol21] E. M. Bollt. “Geometric considerations of a good dictionary for Koopman analysis of dynamical systems: Cardinality, “primary eigenfunction,” and efficient representation”. In: *Communications in Nonlinear Science and Numerical Simulation* 100 (2021), p. 105833. ISSN: 1007-5704.

[BM12] M. Budišić and I. Mezić. “Geometry of the ergodic quotient reveals coherent structures in flows”. In: *Physica D: Nonlinear Phenomena* 241.15 (2012), pp. 1255–1269. ISSN: 0167-2789.

[BMM12] M. Budišić, R. Mohr, and I. Mezić. “Applied Koopmanism”. In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 22.4 (Dec. 2012), p. 047510. ISSN: 1054-1500. eprint: `https://pubs.aip.org/aip/cha/article-pdf/doi/10.1063/1.4772195/13471578/047510\_1\_online.pdf`.

[Col23] M. J. Colbrook. “The mpEDMD Algorithm for Data-Driven Computations of Measure Preserving Dynamical Systems”. In: *SIAM Journal on Numerical Analysis* 61.3 (2023), pp. 1585–1608. eprint: `https://doi.org/10.1137/22M1521407`.

[DTK20] F. Dietrich, T. N. Thiem, and I. G. Kevrekidis. “On the Koopman Operator of Algorithms”. In: *SIAM Journal on Applied Dynamical Systems* 19.2 (2020), pp. 860–885. eprint: `https://doi.org/10.1137/19M1277059`.

[DP80] J. Dormand and P. Prince. “A family of embedded Runge-Kutta formulae”. In: *Journal of Computational and Applied Mathematics* 6.1 (1980), pp. 19–26. ISSN: 0377-0427.

[Eis+10] B. Eisenhower et al. “Decomposing building system data for model validation and analysis using the Koopman operator”. In: *SimBuild 2010* (Jan. 2010).

[Gle94] P. Glendinning. *Stability, Instability and Chaos: An Introduction to the Theory of Nonlinear Differential Equations.* Cambridge Texts in Applied Mathematics. Cambridge University Press, 1994.

[Har+20] C. R. Harris et al. “Array programming with NumPy”. In: *Nature* 585.7825 (Sept. 2020), pp. 357–362.

[KKB17] E. Kaiser, J. N. Kutz, and S. L. Brunton. “Data-driven discovery of Koopman eigenfunctions for control”. In: *Machine Learning: Science and Technology* 2 (2017).

[Koo31] B. O. Koopman. “Hamiltonian Systems and Transformations in Hilbert Space”. In: *Proceedings of the National Academy of Sciences of the United States of America* 17.5 (1931), pp. 315–318. ISSN: 00278424.

[Leh+20] D. Lehmberg et al. “datafold: data-driven models for point clouds and time series on manifolds”. In: *Journal of Open Source Software* 5.51 (2020), p. 2283.

[LKB18] B. Lusch, J. N. Kutz, and S. L. Brunton. “Deep learning for universal linear embeddings of nonlinear dynamics”. In: *Nature Communications* 9.1 (2018), p. 4950.

[Man+20] I. Manojlovic et al. “Applications of Koopman Mode Analysis to Neural Networks”. In: *CoRR* abs/2006.11765 (2020).

[MM13] A. Mauroy and I. Mezic. “A spectral operator-theoretic framework for global stability”. In: Dec. 2013.

[Mez13] I. Mezić. “Analysis of Fluid Flows via Spectral Properties of the Koopman Operator”. In: *Annual Review of Fluid Mechanics* 45.1 (2013), pp. 357–378. eprint: `https://doi.org/10.1146/annurev-fluid-011212-140652`.

[Neu32] J. v. Neumann. “Zur Operatorenmethode In Der Klassischen Mechanik”. In: *Annals of Mathematics* 33.3 (1932), pp. 587–642. ISSN: 0003486X.

[Row] T. Rowland. *Bounded Operator. A Wolfram Web Resource, created by Eric W. Weisstein.* URL: `https://mathworld.wolfram.com/BoundedOperator.html`.

[ROW+09] C. W. ROWLEY et al. “Spectral analysis of nonlinear flows”. In: *Journal of Fluid Mechanics* 641 (2009), 115–127.

[Saa11] Y. Saad. *Numerical Methods for Large Eigenvalue Problems.* Society for Industrial and Applied Mathematics, 2011. eprint: `https://epubs.siam.org/doi/pdf/10.1137/1.9781611970739`.

[SCH10] P. J. SCHMID. “Dynamic mode decomposition of numerical and experimental data”. In: *Journal of Fluid Mechanics* 656 (2010), 5–28.

[TB22] L. N. Trefethen and D. Bau. *Numerical Linear Algebra, Twenty-fifth Anniversary Edition.* Philadelphia, PA: Society for Industrial and Applied Mathematics, 2022. eprint: `https://epubs.siam.org/doi/pdf/10.1137/1.9781611977165`.

[Tu+14] J. H. Tu et al. “On dynamic mode decomposition: Theory and applications”. In: *Journal of Computational Dynamics* 1.2 (2014), pp. 391–421. ISSN: 2158-2491.

[Vir+20] P. Virtanen et al. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. In: *Nature Methods* 17 (2020), pp. 261–272.

[WKR15] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley. “A Data–Driven Approximation of the Koopman Operator: Extending Dynamic Mode Decomposition”. In: *Journal of Nonlinear Science* 25.6 (2015), pp. 1307–1346.

# A Appendix

## A.1 Some code listings

The Python code for the analysis and algorithm is available in the repository – `https://gitlab.lrz.de/00000000014AE04B/koopman-eigensolvers`. The code uses the package DataFold to perform the DMD and EDMD approximation [Leh+20].

The least squares problem is solved using the method `numpy.linalg.lstsq` in the Numpy package [Har+20]. The differential equation integration is done using the method `scipy.integrate.solve_ivp` in the Scipy package [Vir+20]. The code for the figures is in the following Jupyter notebooks:

1. `extend_linear_system_eigenfunctions_DMD.ipynb`: Analysis for discrete linear system (3.24) using DMD.

2. `extend_linear_system_eigenfunctions.ipynb`: Analysis for discrete linear system (3.24) using EDMD .

3. `linear_system_continuous.ipynb`: Analysis for continuous linear system (3.38).

4. `new_nonlln.ipynb`: Analysis for non-linear system (3.53).

5. `non_linear_from_linear_system.ipynb`: Analysis for non-linear system (3.58).

## A.2 Mathematical foundations

### A.2.1 Relation between vector field and Koopman generator

Proof of equation (2.4):

$$[\mathcal{A}_{\mathcal{K}} f](x) = \langle T, \nabla f(x) \rangle, \ \forall f \in \mathcal{F}.$$

Using the definiton of the Koopman generator and Koopman operator, and the flow $T^t$ for the system $\dot{x} = T(x)$, we get

*Proof.*

$$\begin{aligned}
[\mathcal{A}_{\mathcal{K}} f](x) &= \lim_{t\to 0} \frac{\mathcal{K}^t f - f}{t} = \frac{d}{dt}\mathcal{K}^t f(x) \mid_{t=0} \\
&= \frac{d}{dt} f(T^t(x)) \mid_{t=0} = \sum_{i=0}^{D} \frac{\partial}{\partial x_i} f(T^t(x)) \frac{d}{dt} x_i \mid_{t=0} \\
&= \langle \nabla f(T^0(x)), \frac{d}{dt} x \rangle \\
&= \langle \nabla f(x), T \rangle.
\end{aligned}$$

□

### A.2.2 Finding eigenfunctions using right eigenvectors

The EDMD eigenfunction approximations using the right eigenvectors of Koppman matrix $K$ are given by equation (2.27):

$$\begin{bmatrix} \phi_1(X)^T \\ \vdots \\ \phi_d(X)^T \end{bmatrix} = \underset{B \in \mathbb{R}^{d \times r}}{\arg\min} \, \|VB - \Psi(X)\|_2^2 = V^\dagger \Psi(X).$$

*Proof.* Using the relation $KV = V\Lambda$, and (2.23) we get

$$\begin{bmatrix} (\mathcal{K}\phi_1(X))^T \\ \vdots \\ (\mathcal{K}\phi_d(X))^T \end{bmatrix} = V^\dagger \Psi(X) \approx V^\dagger K \Psi(X) = \Lambda V^\dagger \Psi(X) = \begin{bmatrix} (\lambda_1\phi_1(X))^T \\ \vdots \\ (\lambda_d\phi_d(X))^T \end{bmatrix}.$$

where $V = \begin{bmatrix} v_1 \dots v_d \end{bmatrix}$ and $\Lambda = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_d \end{bmatrix}$. □

## A.3 Fast eigensolvers for Koopman operator approximation

### A.3.1 Upper bound calculation for spectral norm of Jacobian for RBF functions with Gaussian kernel

We consider a two dimensional domain $M \in \mathbb{R}^2$. The RBF function with Gaussian kernel centered at $c \in \mathbb{R}^2$ is given by

$$\psi_c(x) = e^{\frac{-1}{2\epsilon}\|x-c\|^2}. \tag{A.1}$$

Therefore the Jacobian matrix of the RBF dictionary basis with Gaussian kernel at a point $x \in M$ is given by

$$J_\Psi(x) = \frac{-1}{\epsilon} \begin{bmatrix} e^{\frac{-1}{2\epsilon}\|x-c^1\|^2}(x_1^1 - c_1^1) & e^{\frac{-1}{2\epsilon}\|x-c^1\|^2}(x_1^2 - c_2^1) \\ \vdots & \vdots \\ e^{\frac{-1}{2\epsilon}\|x-c^M\|^2}(x_1^1 - c_1^M) & e^{\frac{-1}{2\epsilon}\|x-c^1\|^2}(x_1^2 - c_2^M) \end{bmatrix}. \tag{A.2}$$

Now to calculate an upper bound for spectral norm of the Jacbobian, $\left\|J_{\Psi(x)}\right\|_2$, we can calculate the maximum singular value and use it to bound the spectral norm:

$$\lambda_{max}(x) = \max\{\lambda : J_\Psi(x)v = \lambda(x)v\}$$
$$L = \max_{x \in G} \sqrt{\lambda_{max}(x)} \geq \|J_\Psi(x)\|_2 \, .$$

# List of Figures

# List of Algorithms