École Polytechnique Fédérale de Lausanne

Master in Computational Sciences and Engineering

Rigorous data-driven computation of spectral properties of Koopman operators for dynamical systems

Semester Project

Author:
Ivan Bioli

Professor:
Daniel Kressner
Supervisor:
Alice Cortinovis

June 10, 2022

Abstract

When dealing with dynamical systems with discrete time steps, nonlinear and unknown dynamics pose challenges to the standard Poincaré's approach. Koopman operators are infinite-dimensional operators that linearize the dynamics by lifting it to the observables state space. We review spectral properties of Koopman operators, focusing in particular on the case of linear dynamics, as well as numerical methods for computing spectral information of the Koopman operator from snapshot data. We also examine the problem of spectral pollution, recalling Residual Dynamic Mode Decomposition (ResDMD) and proving a proposition that allows reducing ResDMD to a standard matrix pseudospectrum approximation problem. Numerical examples from [9] are reproduced. Finally, a two step procedure for extracting a dictionary of observables directly from snapshot data and a general-purpose kernel is introduced.

Contents

1 Introduction **2**

1.1 Introduction to the problem and paper structure 2

1.1.1 Paper structure 2

1.2 Definition of the Koopman Operator 3

1.3 Koopman eigenvalues, eigenfunctions and eigenmodes 3

1.3.1 Linear systems with simple spectrum 4

1.3.2 Linear systems: general case 4

1.3.3 Koopman Mode Decomposition: nonlinear systems 6

2 Dynamic Mode Decomposition and its variants **8**

2.1 Dynamic Mode Decomposition (DMD) 8

2.2 SVD-based DMD 10

2.3 DMD and the Arnoldi method 11

2.4 DMD for snapshot data 13

3 Extended Dynamic Mode Decomposition and its variants **14**

3.1 Extended Dynamic Mode Decomposition (EDMD) 14

3.1.1 Approximation of $\mathcal{K}$ and its eigenpairs 14

3.1.2 Approximation of the Koopman modes for the full state observable 16

3.2 Spectral pollution and Residual DMD (ResDMD) 16

3.2.1 ResDMD to remove spectral pollution 17

3.2.2 ResDMD to approximate the pseudospectrum 18

3.3 Numerical Examples 20

3.3.1 Gauss iterated map 20

3.3.2 Nonlinear pendulum 21

4 A kernel-based approach for high dimensional data **23**

4.1 Kernelized EDMD (K-EDMD) 23

4.1.1 Computing the Koopman eigenvalues, eigenfunctions and modes 24

4.1.2 The choice of the kernel 25

4.2 Kernelized ResDMD (K-ResDMD) 26

4.2.1 A two step procedure for data-driven dictionaries 26

4.3 A numerical example: a data-driven dictionary for the Gauss iterated map 27

5 Conclusions **29**

1. Introduction

1.1 Introduction to the problem and paper structure

Dynamical systems are mathematical models describing the variability of a state over time, based on a fixed rule. The applications of dynamic systems are innumerable and range from classical mechanics, electrical circuits, fluid flows and thermodynamic systems to neuroscience, finance and epidemiology. Throughout this work we will consider autonomous dynamical systems with finite-dimensional state space $\Omega \subseteq \mathbb{R}^d$ and discrete time steps according to a function F, i.e. dynamical systems whose evolution is characterized by the relation:

$$\mathbf{x}_{n+1} = \mathbf{F}(\mathbf{x}_n) \;\; n \geq 0, \qquad \mathbf{F} : \Omega \to \Omega \tag{1.1}$$

where $\mathbf{x}_0$ is a given initial condition. We are interested in analyzing the system's behaviour from its trajectories $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, ...$, or, more in general, measurements of these trajectories collected through data. Observe that the discrete-time formulation is more natural when considering measurements, which are usually taken at discrete time steps. If we consider the autonomous continuous-time dynamical system described by

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t)),$$

then sampling the trajectories with time steps Δt we are back to the formulation in (1.1) by considering

$$\mathbf{x}(t + \Delta t) = \mathbf{F}_{\Delta t}(\mathbf{x}(t)), \qquad F_{\Delta t}(\mathbf{x}(t)) = \mathbf{x}(t) + \int_t^{t+\Delta t} \mathbf{f}(\mathbf{x}(\tau))d\tau. \tag{1.2}$$

The classical approach to the study of dynamical systems is the geometric one, originated by the work of Poincaré [13]. However, this geometric viewpoint, based on fixed points, local behaviours and invariant manifolds, is ill-suited to many real-world applications [8]. As pointed out in [9], the fundamental challenges of Poincaré's approach in a data-driven perspective are two: non-linear dynamics, for which the geometric viewpoints provides good trajectories approximations only locally and not for all initial conditions, and unknown dynamics. The latter case typically arises when the system's dynamic is too complex to be analytically described or partially unknown, and we have only access to experimental data related to sequences of iterates sampled starting from different initial conditions. Koopman Operator theory [15, 14] provides an alternative viewpoint, which is a general framework for connecting data and measurements to the state space of a dynamical system [1].

1.1.1 Paper structure

In this first chapter, we introduce the Koopman Operator and its spectral properties, analysing in details the case of linear dynamical systems. In Chapter 2 we introduce Dynamic Mode Decomposition (DMD), we link it to the Arnoldi method in the case of linear dynamical systems, and we present how it can be extended to the general case of nonlinear dynamics. In Chapter 3 we present Extended Dynamic Mode Decomposition (EDMD) for computing spectral properties of the Koopman Operator, discussing the spectral pollution problem and introducing the Residual Dynamic Mode Decomposition (ResDMD) algorithm for removing spectral pollution and approximating the pseudospectrum of the Koopman Operator. We also reproduce some of the numerical examples in [9]. Finally, in Chapter 4 we present a kernel-based approach for dynamical systems with high dimensional data, analysing experimentally possible applications also to low dimensional data.

1.2 Definition of the Koopman Operator

When dealing with real-world applications, especially when the dynamics are unknown, the state of the dynamical system is indirectly measured through data, which are variables related to the system's state. We can mathematically formulate this fact by assuming that data are evaluations of functions of the states, leading to the following definition.

Definition 1.1 (Observable). *A function $g : \Omega \to \mathbb{C}$ used to indirectly measure the state of the dynamical system is called an observable.*

Our goal is to be able to approximate as accurately as possible the dynamics of the system. To achieve such goal from experimental data, i.e. from measurements, we need to understand how the observables evolve over time, and here is where the Koopman Operator comes into play. The Koopman Operator advances the measurement forward in time by one-time step: it takes the state, advances the system by one-time step and measures the system again. Given an observable $g : \Omega \to \mathbb{C}$, we define $[\mathcal{K}g](\mathbf{x}) = g(\mathbf{F}(\mathbf{x}))$. It is typical to define the Koopman Operator on (a subset of) the Hilbert space $L^2(\Omega, \omega)$, where ω is a positive measure on the state-space (in our examples we will use only the Lebesgue measure).

Definition 1.2 (Koopman Operator). *Let Ω be the state-space of the dynamical system described by $\mathbf{F} : \Omega \to \Omega$ and let ω be a positive measure on the state-space. Given a suitable domain of observables $\mathcal{D}(\mathcal{K}) \subseteq L^2(\Omega, \omega)$ we define the Koopman Operator as:*

$$\begin{aligned} \mathcal{K} : \mathcal{D}(\mathcal{K}) &\longrightarrow L^2(\Omega, \omega) \\ g &\longmapsto g \circ \mathbf{F} \end{aligned} \tag{1.3}$$

We can think of the Koopman Operator as lifting the dynamics from the state space to the space of observables. From this lifting, we gain that, regardless of the linearity or nonlinearity of $\mathbf{F}$, the Koopman Operator is a linear operator. Therefore, to understand the dynamics of the system, we can analyze the spectral properties of $\mathcal{K}$. However, the disadvantage is that the space of the observables is infinite-dimensional and $\mathcal{K}$ can have a continuous spectrum.

1.3 Koopman eigenvalues, eigenfunctions and eigenmodes

Since the definition of an eigenvalue-eigenfunction pair (abbr. *eigenpair*) may slightly vary according to the context in which the Koopman Operator theory is applied, let us specify the following definition.

Definition 1.3 (Koopman eigenpair). *Let $\phi_j : \Omega \to \Omega$ be an observable of the dynamical system and let $\lambda_j \in \mathbb{C}$. The pair $(\phi_j,\, \lambda_j)$ is called an eigenpair of the Koopman Operator $\mathcal{K}$ if $\mathcal{K}\phi_j = \lambda_j \phi_j$, i.e if $[\mathcal{K}\phi_j](\mathbf{x}) = \phi_j(\mathbf{F}(\mathbf{x})) = \lambda_j \phi_j(\mathbf{x})$.*

Observing that the composition of function is not only linear but also preserves the product, i.e. $(g_1 \cdot g_2) \circ f = (g_1 \circ f) \cdot (g_2 \circ f)$, it is straightforward that $\mathcal{K}(g_1 \cdot g_2) = \mathcal{K}g_1 \cdot \mathcal{K}g_2$. The following lemma is, therefore, trivial.

Lemma 1.1. *Let $(\phi_j,\, \lambda_j)$ and $(\phi_k,\, \lambda_k)$ be two not necessarily distinct eigenpairs of $\mathcal{K}$, such that the supports of ϕ_j and ϕ_k have an intersection with nonzero measure. Then also $(\phi_j \cdot \phi_k,\, \lambda_j \cdot \lambda_k)$ is an eigenpair of $\mathcal{K}$. In particular, given an eigenpair $(\phi,\, \lambda)$, also $(\phi^k,\, \lambda^k)$ is an eigenpair of $\mathcal{K}$ $\forall k \in \mathbb{N},\ k \geq 0$.*

1.3.1 Linear systems with simple spectrum

Let us consider the case in which $\mathbf{F}(\mathbf{x}) = \boldsymbol{A}\mathbf{x}$, $\boldsymbol{A} \in \mathbb{R}^{d\times d}$, i.e. the linear case. Then the eigenvalues of $\boldsymbol{A}$ are eigenvalues of $\mathcal{K}$ and the eigenvectors of $\boldsymbol{A}$ are closely related to the associated eigenfunctions.

Proposition 1.2. *Let λ be an eigenvector of $\boldsymbol{A}$, let $\mathbf{w}$ be an associated left eigenvector, i.e. an eigenvector of $\boldsymbol{A}^*$ associated with $\overline{\lambda}$, and let us define $\phi(\mathbf{x}) = \langle \mathbf{x}, \mathbf{w}\rangle$. Then λ is an eigenvalue of $\mathcal{K}$, with corresponding eigenfunction ϕ.*

Proof.

$$[\mathcal{K}\phi](\mathbf{x}) = \phi(\boldsymbol{A}\mathbf{x}) = \langle \boldsymbol{A}\mathbf{x}, \mathbf{w}\rangle = \langle \mathbf{x}, \boldsymbol{A}^*\mathbf{w}\rangle = \langle \mathbf{x}, \overline{\lambda}\mathbf{w}\rangle = \lambda\langle \mathbf{x}, \mathbf{w}\rangle = \lambda\phi(\mathbf{x})$$

Hence λ is an eigenvalue of $\mathcal{K}$, with corresponding eigenfunction ϕ. □

Let us observe that, even in this very simple setting, from Lemma 1.1 we obtain that if $\boldsymbol{A}$ has at least one nonzero eigenvalue that is not a root of unity, then $\mathcal{K}$ has an infinite number of eigenvalues.

Corollary 1.2.1. *Let us assume that $\boldsymbol{A} \in \mathbb{R}^{d\times d}$ is diagonalizable with a full set of eigenpairs $\{(\lambda_j, \mathbf{v}_j)\}_{j=1}^d$. Let $\{(\overline{\lambda}_j, \mathbf{w}_j)\}_{j=1}^d$ be the eigenpairs of of $\boldsymbol{A}^*$, with $\{\mathbf{w}_j)\}_{j=1}^d$ such that $\langle \mathbf{v}_j, \mathbf{w}_k\rangle = \delta_{k,j}$. Then we can rewrite $\mathbf{x} \in \mathbb{R}^d$:*

$$\mathbf{x} = \sum_{j=1}^{d} \langle \mathbf{x}, \mathbf{w}_j\rangle \mathbf{v}_j = \sum_{j=1}^{d} \phi_j(\mathbf{x})\mathbf{v}_j \tag{1.4}$$

and the evolution of the system reads

$$\mathbf{F}(\mathbf{x}) = \boldsymbol{A}\mathbf{x} = \sum_{j=1}^{d} \phi_j(\mathbf{x})\boldsymbol{A}\mathbf{v}_j = \sum_{j=1}^{d} \lambda_j\phi_j(\mathbf{x})\mathbf{v}_j = \sum_{j=1}^{d} [\mathcal{K}\phi_j](\mathbf{x})\mathbf{v}_j. \tag{1.5}$$

The decomposition in (1.4) is nothing more than an expansion of $\mathbf{x}$ as a linear combination of the vectors $\mathbf{v}_j$, where the $\phi_j(\mathbf{x})$ are the coefficients. However, from the Koopman Operator's viewpoint, it is a linear expansion of the (vector) full state observable as a linear combination of the eigenfunctions of $\mathcal{K}$, where now the role of the coefficients is played by the vectors $\mathbf{v}_j$ [21].

The vectors $\mathbf{v}_j$ are called *Koopman modes* with respect to the observable g, which in this case is the full state (vector) observable $g(\mathbf{x}) = \mathbf{x}$. They are not associated to the Koopman Operator itself, but rather to the action of $\mathcal{K}$ on a particular observable. For instance, if we consider another linear observable $g(\mathbf{x}) = \boldsymbol{C}\mathbf{x}$ where $\boldsymbol{C} \in \mathbb{R}^{m\times d}$ then we obtain

$$[\mathcal{K}g](\mathbf{x}) = g(\boldsymbol{A}\mathbf{x}) = \sum_{j=1}^{d} \lambda_j\phi_j(\mathbf{x})\boldsymbol{C}\mathbf{v}_j.$$

i.e. $\{\boldsymbol{C}\mathbf{v}_j\}_{j=1}^d$ are the new Koopman modes.

We can therefore conclude that for linear dynamical systems with simple spectrum, the two formulations, namely the one using $\mathcal{K}$ and the one using $\boldsymbol{F}$, are strictly linked. In particular, the full state observable can be expressed as a combination of eigenfunctions derived from the eigenvectors of $\boldsymbol{A}^*$, and the Koopman modes coincide with the eigenvectors of $\boldsymbol{A}$.

1.3.2 Linear systems: general case

The previous discussion can be generalized to the case of non-simple spectrum [17], i.e. $\boldsymbol{A}$ non-diagonalizable, defining generalized eigenfunctions strictly linked to the generalized eigenvectors of $\boldsymbol{A}$.

Definition 1.4. *Let $\mathbf{A} \in \mathbb{R}^{d\times d}$ be a matrix with canonical Jordan form:*

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{J}_{\lambda_1} & & & \\ & \boldsymbol{J}_{\lambda_2} & & \\ & & \ddots & \\ & & & \boldsymbol{J}_{\lambda_s} \end{bmatrix}, \qquad \boldsymbol{J}_{\lambda_h} = \begin{bmatrix} \lambda_h & 1 & & & \\ & \lambda_h & 1 & & \\ & & \ddots & \ddots & \\ & & & \lambda_h & 1 \\ & & & & \lambda_i \end{bmatrix} \in \mathbb{C}^{m_h}$$

and let $\mathbf{v}_h^1, \dots, \mathbf{v}_h^{m_h}$ *be the generalized eigenvectors associated with the Jordan block* $\boldsymbol{J}_{\lambda_h}$*, i.e.*

$$\begin{aligned}\boldsymbol{A}\mathbf{v}_h^1 &= \lambda_h \mathbf{v}_h^1 \\ \boldsymbol{A}\mathbf{v}_h^i &= \lambda_h \mathbf{v}_h^i + \mathbf{v}_h^{i-1} \qquad i = 2, \dots, m_h.\end{aligned}$$

Let $\mathbf{w}_h^i$ *be the dual basis vector to* $\mathbf{v}_h^i$*. We define the generalized eigenfunction as*

$$\phi_h^i(\mathbf{x}) = \langle \mathbf{x}, \mathbf{w}_h^i \rangle. \tag{1.6}$$

Lemma 1.3. *Let us consider the generalized eigenfunctions as in Definition 1.4. Then:*

$$\begin{aligned}[][\mathcal{K}\phi_h^i](\mathbf{x}) &= \lambda_h \phi_h^i(\mathbf{x}) + \phi_h^{i+1}(\mathbf{x}) \qquad i = 1, \dots, m_h - 1 \\ [\mathcal{K}\phi_h^{m_h}](\mathbf{x}) &= \lambda_h \phi_h^{m_h}(\mathbf{x})\end{aligned}$$

Proof. The dual basis satisfies

$$\begin{aligned}\boldsymbol{A}^*\mathbf{w}_h^i &= \overline{\lambda_h}\mathbf{w}_h^i + \mathbf{w}_h^{i+1} \qquad i = 1, \dots, m_h - 1 \\ \boldsymbol{A}^*\mathbf{w}_h^{m_h} &= \overline{\lambda_h}\mathbf{w}_h^{m_h}\end{aligned}$$

thus it holds for $i = 1, \dots, m_h - 1$:

$$\begin{aligned}[][\mathcal{K}\phi_h^i](\mathbf{x}) = \phi_h^i(\boldsymbol{A}\mathbf{x}) = \langle \boldsymbol{A}\mathbf{x}, \mathbf{w}_h^i \rangle = \langle \mathbf{x}, \boldsymbol{A}^*\mathbf{w}_h^i \rangle = \langle \mathbf{x}, \overline{\lambda_h}\mathbf{w}_h^i + \mathbf{w}_h^{i+1} \rangle = \\ = \lambda_h \langle \mathbf{x}, \mathbf{w}_h^i \rangle + \langle \mathbf{x}, \mathbf{w}_h^{i+1} \rangle = \lambda_h \phi_h^i(\mathbf{x}) + \phi_h^{i+1}(\mathbf{x})\end{aligned}$$

and similarly $[\mathcal{K}\phi_h^{m_h}](\mathbf{x}) = \lambda_h \phi_h^{m_h}(\mathbf{x})$. □

Similarly to what has been done to the case of linear systems with simple spectrum, we can now recover the evolution of the full state observable and any linear observable from the above defined generalized eigenvalues

Proposition 1.4. *Let* $\mathbf{g}$ *be the full state observable, i.e.* $\mathbf{g}(\mathbf{x}) = \mathbf{x}$*. Then*

$$\mathbf{g}(\mathbf{x}) = \mathbf{x} = \sum_{h=1}^{s}\sum_{i=1}^{m_h} \langle \mathbf{x}, \mathbf{w}_h^i \rangle \mathbf{v}_h^i = \sum_{h=1}^{s}\sum_{i=1}^{m_h} \phi_h^i(\mathbf{x})\mathbf{v}_h^i$$

and the evolution of the system reads:

$$\begin{aligned}[][\mathcal{K}\mathbf{g}](\mathbf{x}) &= \sum_{h=1}^{s}\sum_{i=1}^{m_h}[\mathcal{K}\phi_h^i](\mathbf{x})\mathbf{v}_h^i = \sum_{h=1}^{s}\left(\lambda_h \sum_{i=1}^{m_h}\phi_h^i(\mathbf{x})\mathbf{v}_h^i + \sum_{i=1}^{m_h-1}\phi_h^{i+1}(\mathbf{x})\mathbf{v}_h^i\right) = \\ &= \sum_{h=1}^{s}(\lambda_h P_h(\mathbf{x}) + D_h(\mathbf{x}))\end{aligned} \tag{1.7}$$

where P_h *is the projection onto* $\operatorname{Span}(\mathbf{v}_h^1, \dots, \mathbf{v}_h^{m_h})$ *and* D_h *is a nilpotent operator.*

Proof. We just need to show that D_h is a nilpotent operator. It suffices to observe that D_h is defined by

$$D_h(\mathbf{x}) = D_h\left(\sum_{h=1}^{s}\sum_{i=1}^{m_h}\phi_h^i(\mathbf{x})\mathbf{v}_h^i\right) = \sum_{i=1}^{m_h-1}\phi_h^{i+1}(\mathbf{x})\mathbf{v}_h^i = \sum_{i=2}^{m_h}\phi_h^i(\mathbf{x})\mathbf{v}_h^{i-1} \tag{1.8}$$

hence by induction:

$$D_h^k(\mathbf{x}) = \sum_{i=k+1}^{m_h}\phi_h^i(\mathbf{x})\mathbf{v}_h^{i-k} \tag{1.9}$$

where we use the convention that the sum is equal to zero if $k + 1 > m_h$, i.e. if $k \geq m_h$. □

Corollary 1.4.1. *Let $\boldsymbol{A} \in \mathbb{R}^{d\times d}$ be a matrix and let $\mu_1, \dots, \mu_p$ be its eigenvalues, with algebraic multiplicity* $\mathrm{am}_1, \dots, \mathrm{am}_p$. *Let $\boldsymbol{g}$ be the full state observable of the dynamical system described by $\boldsymbol{A}$, then*

$$[\mathcal{K}\mathbf{g}](\mathbf{x}) = \sum_{j=1}^{p} (\mu_j P_j(\mathbf{x}) + D_j(\mathbf{x}))$$

where P_j is the projection onto the algebraic eigenspace associated to μ_j, which is defined as the null space of $(\boldsymbol{A} - \mu_j \boldsymbol{I})^{\mathrm{ma}_j}$, and D_j is a nilpotent operator of order equal to the maximum size of the Jordan blocks corresponding to the eigenvalue μ_j.

Proof. From Proposition 1.4, supposing that $\mu_j = \lambda_{h_1} = \dots = \lambda_{h_r}$ and $\mu_j \neq \lambda_h$ for $h \notin \{h_1, \dots, h_r\}$, we can define:

$$P_j = \sum_{i=1}^{r} P_{h_i}$$
$$D_j = \sum_{i=1}^{r} D_{h_i}.$$

Then P_j is the projection onto the algebraic eigenspace associated to μ_j, and D_{h_i} is a nilpotent operator with maximum order equal to the maximum size of the Jordan blocks corresponding to the eigenvalue μ_j. Since $D_{h_i} D_{h_k} = 0$ if $i \neq k$, it holds

$$D_j^k = \sum_{i=1}^{r} D_{h_i}^k$$

and the thesis follows. □

Hence the action of the Koopman Operator on the full state observable can be obtained from the standard spectral analysis of $\boldsymbol{A}$ and $\boldsymbol{A}^*$, and the same is true for all linear observables $g(\mathbf{x}) = \langle \mathbf{c}, \mathbf{x} \rangle$.

1.3.3 Koopman Mode Decomposition: nonlinear systems

So far, the Koopman Operator theory applied to the linear case did not provide any new insight in the analysis of the dynamical systems. Let us now consider the more general non-linear case, in which the linearization introduced by the Koopman Operator (at the price of an infinite-dimensional space) can significantly help us.

We are interested in understanding the evolution of observables over time. This is particularly simple if the observable can be written as a linear combination of Koopman eigenfunctions. For this reason, the following definition was introduced in [21].

Definition 1.5 (Koopman Mode Decomposition)**.** *Let $g : \Omega \to \mathbb{C}^p$ be a vector valued observable. Let us assume that each of its component g_i lies in the closure of the Span of J Koopman eigenfunctions, where the case $J = +\infty$ is possible (and often occurs). Then we can write*

$$g_i = \sum_{j=1}^{J} v_{ij}\phi_j, \qquad v_{ij} \in \mathbb{C}$$

and staking the weights into the vectors $\mathbf{v}_j = [v_{1j}, \dots, v_{pj}]^T \in \mathbb{C}^p$

$$g(\mathbf{x}) = \sum_{j=1}^{J} \phi_j(\mathbf{x})\mathbf{v}_j. \tag{1.10}$$

This type of decomposition based on the Koopman eigenfunctions is named Koopman Mode Decomposition (KMD). The vectors $\mathbf{v}_j$ are called the Koopman modes of the map $\mathbf{F}$ corresponding to the observable g.

We can think of a vector valued observable $\mathbf{g}$ as a vector gathering different measurements of the system state. By linearity of the Koopman operator, the evolution of the observable is

$$g(\mathbf{x}_n) = [\mathcal{K}^n g](\mathbf{x}_0) = \sum_{j=1}^{J} \lambda_j^n \phi_j(\mathbf{x}_0)\mathbf{v}_j. \tag{1.11}$$

From (1.11) we can understand that the Koopman eigenvalue λ_j characterizes the contribution of the corresponding Koopman mode $\mathbf{v}_j$ to the evolution of the observable over time: the phase of λ_j determines its frequency, while the magnitude determines the growth rate.

As already discussed, the dynamical system defined by $\mathbf{F}$ and the one defined by $\mathcal{K}$ are two different ways of describing the same underlying phenomenon. The former is finite dimensional but in general non-linear, the latter is linear but infinite dimensional. The idea to link these two formulations is the full state observable $g(\mathbf{x}) = \mathbf{x}$ and the set $\{(\lambda_j, \phi_j, \mathbf{v}_j)\}_{j=1}^{J}$ of J tuples of Koopman eigenvalues, eigenfunctions and eigenmodes corresponding to the full state observable. Indeed, if the components of the full state observable lie in the Span of $\{(\lambda_j, \phi_j, \mathbf{v}_j)\}_{j=1}^{J}$, the system evolution can be obtained either applying the complex and non-linear $\mathbf{F}$ to $\mathbf{x}$ or evolving g linearly using $\mathcal{K}$ through

$$\mathbf{F}(\mathbf{x}) = [\mathcal{K}g](\mathbf{x}) = \sum_{j=1}^{J} [\mathcal{K}\phi_j](\mathbf{x})\mathbf{v}_j = \sum_{j=1}^{J} \lambda_j \phi_j(\mathbf{x})\mathbf{v}_j$$

with the behaviour of the system along each eigenfunction that is determined by the corresponding eigenvalue.

2. Dynamic Mode Decomposition and its variants

In this second chapter, we introduce Dynamic Mode Decomposition (DMD). Firstly, we consider the case of linear dynamical systems and we present two versions of the DMD algorithm: one more similar to the Arnoldi algorithm and one based on the Singular Value Decomposition and more numerically stable. We then analyse in Section 2.3 the link between the DMD and the Arnoldi algorithm, comparing the three algorithms theoretically and through a toy numerical example. Finally, in Section 2.4 we extend DMD to nonlinear dynamics.

2.1 Dynamic Mode Decomposition (DMD)

Let us go back to the case of a linear dynamical system

$$\mathbf{x}_{n+1} = \boldsymbol{A}\mathbf{x}_n, \qquad \boldsymbol{A} \in \mathbb{R}^{n\times n}.$$

As already discussed, in order to extract the dynamic characteristics we need to analyze the spectral properties of $\mathbf{A}$. However, in a data-driven perspective, we cannot assume that we have access to $\boldsymbol{A}$, but only to a sequence of snapshots. Therefore, we are interested in computing the eigenpairs of $\mathbf{A}$, or at least the dominant ones, from a sequence of snapshots

$$\boldsymbol{V}_1^N = [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_N] = \left[\mathbf{v}_1, \boldsymbol{A}\mathbf{v}_1, \dots, \boldsymbol{A}^N\mathbf{v}_1\right].$$

Since the columns of $\boldsymbol{V}_1^N$ span a Krylov subspace, it is natural to apply a Krylov subspace method. As we do not have access to the matrix $\boldsymbol{A}$ and we cannot compute products of vectors by this matrix, we cannot directly apply numerically stable algorithms such as the Arnoldi method, and we can rely only on the sequence of snapshots we are given in input. *Dynamic Mode Decomposition* (DMD) [22] is a variant of the Arnoldi method that does not require computing multiplications by $\boldsymbol{A}$, nor any knowledge of the matrix.

Let us first suppose that, after a certain number of iterates, we found an $\boldsymbol{A}$-invariant Krylov subspace, and that $\mathbf{v}_N$ can be expressed as a linear combination of the previous linearly independent snapshots, i.e.

$$\mathbf{v}_N = a_1\mathbf{v}_1 + \dots + a_{N-1}\mathbf{v}_{N-1} = \mathbf{V}_1^{N-1}\mathbf{a}, \qquad \mathbf{a} \in \mathbb{R}^{N-1}. \tag{2.1}$$

We can rewrite the relation in (2.1) as:

$$\boldsymbol{A}\boldsymbol{V}_1^{N-1} = \boldsymbol{V}_2^N = \boldsymbol{V}_1^{N-1}\boldsymbol{S}$$

where $\boldsymbol{S}$ is the companion matrix associated with $\mathbf{a}$, defined by:

$$\boldsymbol{S} := \begin{bmatrix} 0 & & & & a_1 \\ 1 & 0 & & & a_2 \\ & \ddots & \ddots & & \vdots \\ & & 1 & 0 & a_{N-2} \\ & & & 1 & a_{N-1} \end{bmatrix}. \tag{2.2}$$

The eigenvalues of $\boldsymbol{S}$ are also eigenvalues of $\boldsymbol{A}$. Indeed, if $\boldsymbol{S}\mathbf{x} = \lambda\mathbf{x},\ \mathbf{x} \neq 0$ then $\mathbf{y} = \mathbf{V}_1^{N-1}\mathbf{x} \neq 0$ (we assumed that the first $N-1$ snapshots are linearly independent) and $\boldsymbol{A}\mathbf{y} = \boldsymbol{A}\mathbf{V}_1^{N-1}\mathbf{x} = \mathbf{V}_1^{N-1}\boldsymbol{S}\mathbf{x} = \lambda\mathbf{V}_1^{N-1}\mathbf{x} = \lambda\mathbf{y}$.

In general, $\mathbf{v}_N$ does not necessarily belong to the Span of the columns of $\boldsymbol{V}_1^{N-1}$, i.e. in general $\boldsymbol{V}_1^N$ is not rank deficient. However, as N increases, the snapshots tend to lie in the same direction (or at least in the same subspace), and we can expect that, after a critical number of snapshots, $\mathbf{v}_N$ *almost* lies in the Span($\boldsymbol{V}_1^{N-1}$), i.e. the angle between $\mathbf{v}_N$ and Span($\boldsymbol{V}_1^{N-1}$) is close to zero. Therefore, again assuming that the previous vectors are linearly independent, we we want to write

$$\begin{aligned} &\mathbf{v}_N = a_1\mathbf{v}_1 + \cdots + a_{N-1}\mathbf{v}_{N-1} + \mathbf{r}, \qquad \text{i.e.} \\ &\boldsymbol{A}\boldsymbol{V}_1^{N-1} = \boldsymbol{V}_1^{N-1}\boldsymbol{S} + \mathbf{r}\mathbf{e}_{N-1}^T \end{aligned} \tag{2.3}$$

so that we minimize the norm of the residual $\mathbf{r} = \mathbf{v}_N - \boldsymbol{V}_1^{N-1}\mathbf{a}$. To solve the corresponding least square problem, we compute the thin QR-decomposition of $\boldsymbol{V}_1^{N-1} = \boldsymbol{Q}\boldsymbol{R}$ and then we obtain

$$\mathbf{a} = \boldsymbol{R}^{-1}\boldsymbol{Q}^*\mathbf{v}_N = (\boldsymbol{V}_1^{N-1})^\dagger\mathbf{v}_N. \tag{2.4}$$

The eigenvalues $\{\lambda_j\}_{j=1}^{N-1}$ of the companion matrix $\boldsymbol{S}$ are now approximations of the eigenvalues of $\boldsymbol{A}$ and are called *Ritz values*. The corresponding approximate eigenvectors $\mathbf{y}_j = \boldsymbol{V}_1^{N-1}\mathbf{x}_j$, where $\mathbf{x}_j$ is an eigenvector of $\boldsymbol{S}$ are the so called *Ritz vectors*. The following proposition summarizes the properties of the approximations provided by the above described DMD algorithm.

Proposition 2.1 ([10]). *Suppose that $\boldsymbol{V}_1^{N-1}$ is of full column rank and let us define the Krylov subspace $\mathcal{V}_k = \mathrm{range}(V_1^k)$ for $k = 1, \ldots, N$. Let $\mathbf{a}$ be computed solving the least square problem*

$$\mathbf{a} = \operatorname*{arg\,min}_{\mathbf{x}\in\mathbb{R}^{N-1}} \left\|\mathbf{v}_N - \boldsymbol{V}_1^{N-1}\mathbf{x}\right\| = (\boldsymbol{V}_1^{N-1})^\dagger\mathbf{v}_N$$

and let $\boldsymbol{S}$ be the companion matrix associated with $\mathbf{a}$ as in (2.2). Then:

(i) $\boldsymbol{S} = (\boldsymbol{V}_1^{N-1})^\dagger\boldsymbol{A}\boldsymbol{V}_1^{N-1} = (\boldsymbol{V}_1^{N-1})^\dagger\boldsymbol{V}_2^N$ *is the matrix representation of the projection onto $\mathcal{V}_{N-1}$ of the linear operator defined by $\boldsymbol{A}$ restricted to $\mathcal{V}_{N-1}$, i.e. $P_{\mathcal{V}_{N-1}}\boldsymbol{A}|_{\mathcal{V}_{N-1}}$, in the Krylov basis $\mathbf{V}_1^{N-1}$ of $\mathcal{V}_{N-1}$.*

(ii) if $\mathbf{r} = \mathbf{v}_N - \boldsymbol{V}_1^{N-1}\mathbf{a} = 0$ *then* $\boldsymbol{A}\boldsymbol{V}_1^{N-1} = \boldsymbol{V}_1^{N-1}\boldsymbol{S}$*, and each eigenpair $(\lambda, \mathbf{x})$ of $\boldsymbol{S}$ yields an eigenpair of $\boldsymbol{A}$, $(\lambda, \mathbf{y} = \boldsymbol{V}_1^{N-1}\mathbf{x})$*

(iii) if $\mathbf{r} = \mathbf{v}_N - \boldsymbol{V}_1^{N-1}\mathbf{a} \neq 0$ *then* $\boldsymbol{A}\boldsymbol{V}_1^{N-1} = \boldsymbol{V}_1^{N-1}\boldsymbol{S} + \mathbf{r}\mathbf{e}_{N-1}^T$ *and the residual $\mathbf{r}$ is orthogonal to* $\mathrm{range}(\boldsymbol{V}_1^{N-1})$*. Moreover each eigenpair $(\lambda, \mathbf{x})$ of $\boldsymbol{S}$ yields an approximate eigenpair of $\boldsymbol{A}$, $(\lambda, \mathbf{y} = \boldsymbol{V}_1^{N-1}\mathbf{x})$ called Ritz pair.*

(iv) The Ritz pair is an exact eigenpair of the perturbed matrix

$$\boldsymbol{A} + \boldsymbol{\Delta A}, \qquad \boldsymbol{\Delta A} = -\mathbf{r}\mathbf{e}_{N-1}^T(\boldsymbol{V}_1^{N-1})^\dagger \tag{2.5}$$

(v) if $\boldsymbol{A}$ is diagonalizable with eigenvalues $\lambda_1, \ldots, \lambda_d$ and eigenvector matrix $\boldsymbol{M}$, then for each eigenvalue λ of $\boldsymbol{S}$ it holds

$$\begin{aligned} &\min_{\lambda_i} |\lambda - \lambda_i| \leq \kappa_2(\boldsymbol{M})\|\boldsymbol{\Delta A}\|_2 \\ &\min_{\lambda_i} \frac{|\lambda - \lambda_i|}{|\lambda_i|} \leq \kappa_2(\boldsymbol{M})\left\|\boldsymbol{A}^{-1}\boldsymbol{\Delta A}\right\|_2 \textit{if } \boldsymbol{A} \textit{ is non-singular} \end{aligned}$$

where $\kappa_2(\boldsymbol{M}) = \|\boldsymbol{M}\|_2\left\|\boldsymbol{M}^{-1}\right\|_2$ is the condition number in 2-norm of the matrix $\boldsymbol{M}$.

Proof. Since $\boldsymbol{V}_1^{N-1}$ is of full column rank, then $(\boldsymbol{V}_1^{N-1})^\dagger = ((\boldsymbol{V}_1^{N-1})^*\boldsymbol{V}_1^{N-1})^{-1}(\boldsymbol{V}_1^{N-1})^*$ is the matrix representation of the projection onto $\mathcal{V}_{N-1}$ with input basis the canonical basis and output basis $\boldsymbol{V}_1^{N-1}$. Moreover $\boldsymbol{A}\boldsymbol{V}_1^{N-1} = \boldsymbol{V}_2^N$ is the matrix representation of the linear operator defined by $\boldsymbol{A}$ restricted to $\mathcal{V}_{N-1}$, whit input basis $\mathcal{V}_{N-1}$ and output basis the canonical basis. Statement (i) follows by composition.

We have already discussed $(ii)-(iii)$ and we know that $\boldsymbol{A}\boldsymbol{V}_1^{N-1} = \boldsymbol{V}_1^{N-1}\boldsymbol{S}+\mathbf{r}\mathbf{e}_{N-1}^T$. Moreover, since $\boldsymbol{V}_1^{N-1}$ is of full column rank, $(\boldsymbol{V}_1^{N-1})^\dagger\boldsymbol{V}_1^{N-1} = \boldsymbol{I}_{N-1}$, hence:

$$\boldsymbol{V}_1^{N-1}\boldsymbol{S} = \boldsymbol{A}\boldsymbol{V}_1^{N-1} - \mathbf{r}\mathbf{e}_{N-1}^T = \boldsymbol{A}\boldsymbol{V}_1^{N-1} - \mathbf{r}\mathbf{e}_{N-1}^T(\boldsymbol{V}_1^{N-1})^\dagger\boldsymbol{V}_1^{N-1} = (\boldsymbol{A}+\boldsymbol{\Delta A})\boldsymbol{V}_1^{N-1}$$

and statement (iv) follows. Finally, applying the Bauer-Fike theorem [12] the first bound in (v) follows immediately. For the second bound, let us observe that, if $\boldsymbol{A}$ is non singular, we can multiply by $\boldsymbol{A}^{-1}$ the equality $(\boldsymbol{A}+\boldsymbol{\Delta A})\mathbf{x} = \lambda\mathbf{x}$, obtaining $(\lambda\boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{\Delta A})\mathbf{x} = \mathbf{x}$. Hence from the Bauer-Fike theorem:

$$\min_{\lambda_i}\left|1-\frac{\lambda}{\lambda_i}\right| = \min_{\lambda_i}\frac{|\lambda-\lambda_i|}{|\lambda_i|} \leq \kappa_2(\boldsymbol{M}^{-1})\left\|\boldsymbol{A}^{-1}\boldsymbol{\Delta A}\right\|_2 = \kappa_2(\boldsymbol{M})\left\|\boldsymbol{A}^{-1}\boldsymbol{\Delta A}\right\|_2$$

□

2.2 SVD-based DMD

The above mentioned method was the original version of the DMD [22]. However, even if mathematically correct and equivalent to the Arnoldi method in exact arithmetic (see Section 2.3), an implementation using the companion matrix $\boldsymbol{S}$ gives rise to a numerically unstable algorithm that is usually not capable to extract more than one or two dominant eigenpairs. Indeed, as already pointed out, the matrix $\boldsymbol{V}_1^{N-1}$ might be numerically rank deficient, hence ill conditioned. Therefore, even when a QR-factorization $\boldsymbol{V}_1^{N-1} = \boldsymbol{Q}\boldsymbol{R}$ is available, it is not advised to compute $\mathbf{a}$ as $\mathbf{a} = \boldsymbol{R}^{-1}\boldsymbol{Q}\mathbf{v}_N$ or $\boldsymbol{S}$ as $\boldsymbol{S} = \boldsymbol{R}^{-1}\boldsymbol{Q}\boldsymbol{A}\boldsymbol{V}_1^{N-1}$.

Using the Singular Value Decomposition (SVD) it is possible to obtain a mathematically equivalent but better-conditioned algorithm, which is particularly useful when $\boldsymbol{V}_1^{N-1}$ is close to singular (it is very likely to happen for large N). This second version of the DMD is what nowadays is referred as DMD. Suppose that we computed a SVD of the snapshots matrix

$$\boldsymbol{V}_1^{N-1} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{W}^* \qquad \boldsymbol{\Sigma} \in \mathbb{R}^{N-1\times N-1},$$

we can now rewrite (2.3) as

$$\boldsymbol{A}\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{W}^* = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{W}^*\boldsymbol{S} + \mathbf{r}\mathbf{e}_{N-1}^T = \boldsymbol{V}_2^N \tag{2.6}$$

and rearranging

$$\boldsymbol{U}^*\boldsymbol{A}\boldsymbol{U} = \boldsymbol{U}^*\boldsymbol{V}_2^N\boldsymbol{W}\boldsymbol{\Sigma}^{-1} =: \widetilde{\boldsymbol{S}}. \tag{2.7}$$

Once we have computed the eigenpairs $\{(\lambda_j, \mathbf{x}_j)\}_j$ of the matrix $\widetilde{\boldsymbol{S}}$, which in this case is a full matrix and not of companion type, we obtain the Ritz values and vectors as $\{(\lambda_j, \boldsymbol{U}\mathbf{x}_j)\}_j$.

Besides the better conditioning, a great advantage of the SVD-based approach over the Arnoldi-based one is the opportunity to account for rank deficiency in $\boldsymbol{V}_1^{N-1}$ and noise in the data by truncating the SVD of $\boldsymbol{V}_1^{N-1}$. Indeed, we might include a first step in which the matrix $\boldsymbol{V}_1^{N-1}$ is approximated by a low-rank matrix with approximately the same *numerical* rank. This is achieved in a standard way, that is truncating the SVD of the matrix $\boldsymbol{V}_1^{N-1}$, such as suggested by the Eckhart–Young-Mirsky theorem [12]. The use of TSVD can be particularly useful when using data from experiments, as the problem might be ill-conditioned.

As previously mentioned, the two approaches are equivalent since $\boldsymbol{S}$ and $\widetilde{\boldsymbol{S}}$ are linked through a similarity transformation, as long as $\boldsymbol{V}_1^{N-1}$ is of full column rank and we do not truncate the SVD. This is summarized in the following proposition

Proposition 2.2. *Suppose that $\boldsymbol{V}_1^{N-1}$ is of full column rank and let $\boldsymbol{V}_1^{N-1} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{W}^*$ be its SVD. Given $k \in \{1, 2, \ldots, N-1\}$, let us define $\boldsymbol{U}_k = \boldsymbol{U}(:, 1:k)$, $\boldsymbol{W}_k = \boldsymbol{W}(:, 1:k)$, $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}(1:k, 1:k)$ and finally $\widetilde{\boldsymbol{S}} = \boldsymbol{U}_k^*\boldsymbol{V}_2^N\boldsymbol{W}_k\boldsymbol{\Sigma}_k^{-1}$. Then:*

(i) if $k = N-1$, i.e. if we do not truncate the SVD, then $\widetilde{\boldsymbol{S}}$ is similar to $\boldsymbol{S}$, where $\boldsymbol{S}$ is defined as in (2.2) *and the similarity is realized by the matrix $\boldsymbol{W}\boldsymbol{\Sigma}^{-1}$.*

(ii) if $k < N-1$, then $\widetilde{\boldsymbol{S}}$ is the matrix representation in the basis $\boldsymbol{W}_k\boldsymbol{\Sigma}_k^{-1}$ of the projection onto $\operatorname{range}(\boldsymbol{W}_k\boldsymbol{\Sigma}_k^{-1})$ *of the linear operator defined by $\boldsymbol{S}$ restricted to* $\operatorname{range}(\boldsymbol{W}_k\boldsymbol{\Sigma}_k^{-1})$.

(iii) $\widetilde{\boldsymbol{S}}$ *is the matrix representation in the basis* $\boldsymbol{U}_k$ *of the projection onto* $\mathrm{range}(\boldsymbol{U}_k)$ *of the linear operator defined by* $\boldsymbol{A}$ *restricted to* $\mathrm{range}(\boldsymbol{U}_k)$.

Proof. Let us first suppose that $k = m$, i.e. that the SVD is not truncated. Then the residual of the least squares solution is orthogonal to the columns of $\boldsymbol{U}$, hence from (2.6) and (2.7):

$$\widetilde{\boldsymbol{S}} = \boldsymbol{U}^*\boldsymbol{V}_2^N\boldsymbol{W}\boldsymbol{\Sigma}^{-1} = \boldsymbol{U}^*(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{W}^*\boldsymbol{S} + \mathbf{r}\mathbf{e}_{N-1}^T)\boldsymbol{W}\boldsymbol{\Sigma}^{-1} = \\ = (\boldsymbol{\Sigma}\boldsymbol{W}^*)\boldsymbol{S}(\boldsymbol{W}\boldsymbol{\Sigma}^{-1}) = (\boldsymbol{W}\boldsymbol{\Sigma}^{-1})^{-1}\boldsymbol{S}(\boldsymbol{W}\boldsymbol{\Sigma}^{-1})$$

and the matrices $\widetilde{\boldsymbol{S}}$ and $\boldsymbol{S}$ are liked by the stated similarity transformation.

If $k < m$ we can write $\boldsymbol{V}_1^{N-1} = \boldsymbol{U}_k\boldsymbol{\Sigma}_k\boldsymbol{W}_k^* + \boldsymbol{\delta V}$, where $\boldsymbol{\delta V} = \sum_{i=k+1}^{N-1}\sigma_i\boldsymbol{U}(:,i)\boldsymbol{W}(:,i)^*$ is such that $\boldsymbol{U}_k^*\boldsymbol{\delta V} = \boldsymbol{\delta V}\boldsymbol{W}_k = \boldsymbol{0}$. Hence we can write (2.3) as

$$\boldsymbol{A}(\boldsymbol{U}_k\boldsymbol{\Sigma}_k\boldsymbol{W}_k^* + \boldsymbol{\delta V}) = (\boldsymbol{U}_k\boldsymbol{\Sigma}_k\boldsymbol{W}_k^* + \boldsymbol{\delta V})\boldsymbol{S} + \mathbf{r}\mathbf{e}_{N-1}^T = \boldsymbol{V}_2^N$$

Multiplying by $\boldsymbol{U}_k^*$ on the left and by $\boldsymbol{W}_k\boldsymbol{\Sigma}_k^{-1}$ on the right, we get

$$\boldsymbol{U}_k^*\boldsymbol{A}\boldsymbol{U}_k = (\boldsymbol{W}_k\boldsymbol{\Sigma}_k^{-1})^{-1}\boldsymbol{S}(\boldsymbol{W}_k\boldsymbol{\Sigma}_k^{-1}) = \widetilde{\boldsymbol{S}}$$

and the thesis follows. □

The SVD-based version of DMD is summarized in Algorithm 1.

Algorithm 1 : SVD-based DMD

```
Input: $\mathbf{v}_1, \dots, \mathbf{v}_N$
1: Define the matrices $\boldsymbol{V}_1^{N-1} = [\mathbf{v}_1, \dots, \mathbf{v}_{N-1}]$ and $\boldsymbol{V}_2^N = [\mathbf{v}_2, \dots, \mathbf{v}_N]$.
2: Compute the (possibly truncated) SVD of $\boldsymbol{V}_1^{N-1} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{W}^*$, $\boldsymbol{\Sigma} \in \mathbb{R}^{p\times p}$.
3: Define the matrix $\widetilde{\boldsymbol{S}} := \boldsymbol{U}^*\boldsymbol{V}_2^N\boldsymbol{W}\boldsymbol{\Sigma}^{-1}$.
4: Compute the eigenpairs $\{(\lambda_j, \mathbf{x}_j)\}_j$ of $\widetilde{\boldsymbol{S}}$.
5: Compute the Ritz vector corresponding to the Ritz value $\lambda_j$ as $\mathbf{y}_j = \boldsymbol{U}\mathbf{x}_j$.
Output: Ritz values $\lambda_1, \dots, \lambda_p$ and corresponding Ritz vectors $\mathbf{y}_1, \dots, \mathbf{y}_p$
```

2.3 DMD and the Arnoldi method

The Arnoldi method and the DMD are equivalent in exact arithmetic. In the following section, we will discuss and prove this statement, by emphasizing why the Arnoldi algorithm is generally more stable.

The Arnoldi method computes an orthonormal basis of the Krylov subspace spanned by the columns of $\boldsymbol{V}_1^{N-1}$ computing the snapshots implicitly. In particular, if $\boldsymbol{V}_1^{N-1}$ is of full colum rank and therefore the algorithms does not stop, it computes a matrix $\boldsymbol{Q}_{N-1} = [\mathbf{q}_1, \dots, \mathbf{q}_{N-1}]$ with columns that form an orthonormal basis of $\mathrm{Span}(\boldsymbol{V}_1^{N-1})$ and:

(i) $\boldsymbol{V}_1^{N-1} = \boldsymbol{Q}_{N-1}\boldsymbol{R}_{N-1}$ is a thin QR-decomposition (with $\boldsymbol{R}_{N-1}$ that is computed implicitly);

(ii) $\boldsymbol{H}_{N-1} = \boldsymbol{Q}_{N-1}^*\boldsymbol{A}\boldsymbol{Q}_{N-1}$ is a Hessenberg matrix.

Then the eigenpairs approximations are obtained from the eigenpairs of $\boldsymbol{H}_{N-1}$. This is mathematically equivalent to the DMD algorithm, as specified in the following proposition.

Proposition 2.3. *Suppose that* $\boldsymbol{V}_1^{N-1}$ *is of full column rank. Then the Arnoldi method and the DMD algorithms are equivalent in exact arithmetic, i.e. the Ritz pairs returned by the two algorithms coincide.*

Proof. From (2.3) and (2.4) the residual $\mathbf{r}$ can be written as

$$\mathbf{r} = \mathbf{v}_N - \boldsymbol{V}_1^{N-1}\boldsymbol{R}_{N-1}^{-1}\boldsymbol{Q}_{N-1}^*\mathbf{v}_N = \mathbf{v}_N - \boldsymbol{Q}_{N-1}\boldsymbol{Q}_{N-1}^*\mathbf{v}_N$$

and therefore $\boldsymbol{Q}_{N-1}^* \mathbf{r} = \mathbf{0}$.Plugging the QR-decomposition into (2.3)

$$\boldsymbol{A}\boldsymbol{Q}_{N-1}\boldsymbol{R}_{N-1} = \boldsymbol{Q}_{N-1}\boldsymbol{R}_{N-1}\boldsymbol{S} + \mathbf{r}\mathbf{e}_{N-1}^T$$

and multiplying by $\boldsymbol{Q}_{N-1}^*$ on the left and by $\boldsymbol{R}_{N-1}^{-1}$ on the right

$$\boldsymbol{H}_{N-1} = \boldsymbol{Q}_{N-1}^*\boldsymbol{A}\boldsymbol{Q}_{N-1} = \boldsymbol{R}_{N-1}\boldsymbol{S}\boldsymbol{R}_{N-1}^{-1}. \tag{2.8}$$

Hence $\boldsymbol{H}_{N-1}$ and $\boldsymbol{S}$ are linked through a similarity transformation and, in particular, have the same eigenvalues. As a consequence of Proposition 2.2, this is true also if we apply the SVD-based DMD without truncation.

Let $\boldsymbol{M}_{N-1}$ be the eigenvector matrix of $\boldsymbol{H}_{N-1}$, then the Ritz vectors returned by the Arnoldi method are the columns of $\boldsymbol{Q}_{N-1}\boldsymbol{M}_{N-1}$. From (2.8), the eigenvector matrix of $\boldsymbol{S}$ is $\boldsymbol{R}_{N-1}^{-1}\boldsymbol{M}_{N-1}$, thus the Ritz vectors returned by the DMD algorithm are the columns of $\boldsymbol{V}_1^{N-1}\boldsymbol{R}_{N-1}^{-1}\boldsymbol{M}_{N-1} = \boldsymbol{Q}_{N-1}\boldsymbol{R}_{N-1}\boldsymbol{R}_{N-1}^{-1}\boldsymbol{M}_{N-1} = \boldsymbol{Q}_{N-1}\boldsymbol{M}_{N-1}$ and also the Ritz vectors coincide. □

Although the two algorithms are equivalent mathematically and in exact arithmetic, the Arnoldi algorithm is numerically more stable. Indeed, as the number of snapshots increases, the vectors tend to become linearly dependent and, therefore, the matrix V_1^N becomes close to singular, hence ill-conditioned. Even if the problem can be partially solved using the SVD-based DMD and performing a truncated SVD, another major problem is intrinsic to the data-driven approach. One of the major strengths of the Arnoldi algorithm is that it does not even compute the vectors $\mathbf{x}_j = \boldsymbol{A}^j\mathbf{x}_0$, as errors are already made in these computations. However, in a data-driven perspective, this is not possible since what we are given is a sequence of snapshots and not the matrix $\boldsymbol{A}$. Hence, we trade-off better stability and convergence properties for an algorithm that only relies on V_1^N and is therefore applicable to snapshots experimentally collected.

In order to compare the algorithms from S numerical point of view, we performed 60 iterations with the matrix `A` of size `n = 400` defined as `A = Q'*diag([1, 0.9, 0.8.^(1:n-2)])*Q`, where `Q` is an orthogonal matrix. and a normalised random starting vector `x = rand(n,1)`. Figure 2.1 shows the convergence of the approximations of the five dominant eigenvalues. It can be seen that the Arnoldi method is able to approximate all of them up to the epsilon-machine-precision. The Arnoldi-based DMD approximates the first two dominant eigenvalues with an accuracy that is eight orders of magnitude lower than the one obtained with the SVD-based DMD. Moreover, it fails in approximating the remaining 3 eigenvalues. The SVD-based DMD is able to approximate all five dominant eigenvalues, with an accuracy that is lower than the one of the Arnoldi algorithm, but still acceptable.

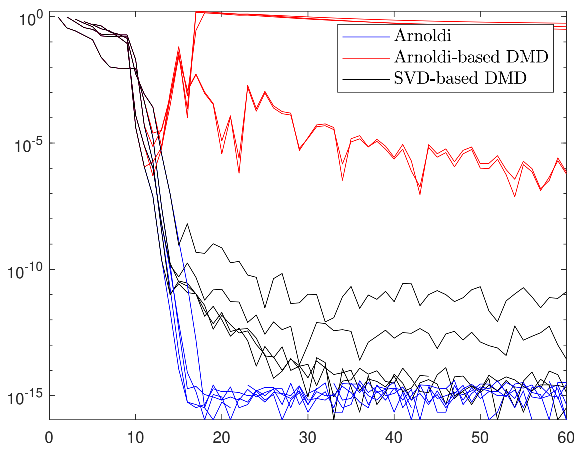


Figure 2.1: Convergence of the five dominant eigenvalues for the Arnoldi method, the Arnoldi-based DMD algorithm and the SVD-based DMD algorithm. 60 iterations are performed with the matrix `A` of size `n = 400` defined as `A = Q'*diag([1, 0.9, 0.8.^(1:n-2)])*Q`, where `Q` is an orthogonal matrix.

2.4 DMD for snapshot data

The original version of the DMD algorithm [22] assumes that data consist of a trajectory of snapshots $\mathbf{v}_1, \dots, \mathbf{v}_N$ where $\mathbf{v}_{i+1} = \boldsymbol{A}\mathbf{v}_i$. While this is required to write Equation (2.3) with $\mathbf{S}$ a companion matrix, the SVD-based approach does not exploit this structure. Indeed, as observed in [25], Algorithm 1 can be carried out for general matrices $\boldsymbol{X}_0$ and $\boldsymbol{X}_1$ replacing $\boldsymbol{V}_1^{N-1}$ and $\boldsymbol{V}_2^N$ respectively, where we may assume that

$$\boldsymbol{X}_0 = \left[\mathbf{x}_0^{(1)}, \dots, \mathbf{x}_0^{(M)}\right] \qquad \boldsymbol{X}_1 = \left[\mathbf{x}_1^{(1)}, \dots, \mathbf{x}_1^{(M)}\right] \tag{2.9}$$

are such that $\mathbf{x}_1^{(m)} = \mathbb{A}\mathbf{x}_0^{(m)}$ for some linear operator $\mathbb{A}$. This allows to generalise DMD to the case of nonlinear dynamics.

Again, let us start with the linear case, and suppose that, instead of snapshots along a single trajectory $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_N$, we are given snapshot pairs of the system state $\{(\mathbf{x}_0^{(m)}, \mathbf{x}_1^{(m)})\}_{m=1}^M$ with $\mathbf{x}_1^{(m)} = \mathbf{F}(\mathbf{x}_0^{(m)}) = \boldsymbol{A}\mathbf{x}_0^{(m)}$. The first setting is just a particular case of the latter one, with $\mathbf{x}_0^{(m)} = \mathbf{v}_m$, $\mathbf{x}_1^{(m)} = \mathbf{v}_{m+1}$. However, even in the linear case, requiring in input only snapshot pairs instead of one trajectory allows us to explore as many different initial states as we want. This is of particular help if the trajectories tend to converge to an equilibrium. Let us consider $\boldsymbol{X}_0$ and $\boldsymbol{X}_1$ defined as in (2.9), then the best approximation of A that we can get from data is the least-squares solution is $\hat{\boldsymbol{A}} = \boldsymbol{X}_1\boldsymbol{X}_0^\dagger$. The eigenvalues of $\boldsymbol{A}$ can now be approximated by the eigenvalues of $\hat{\boldsymbol{A}}$. The DMD eigenvalues and modes are defined as the nonzero eigenvalues and corresponding eigenvectors of $\hat{\boldsymbol{A}}$, respectively.

If the system is nonlinear, then $\hat{\boldsymbol{A}}$ only provides the best-fit linear approximation of $\mathbf{F}$, but there is no apparent reason why $\mathbf{F}$ should be well approximated by a linear finite-dimensional operator. However, even if the dynamics is nonlinear, the Koopman Operator is linear (but infinite dimensional). Therefore, we could apply the DMD algorithm to observables to obtain an approximation of the Koopman eigenvalues and modes. Let us consider a vector-valued observable $\mathbf{g} = \mathbf{g}(\mathbf{x})$ and its measurements over a set of arbitrary initial states $\{\mathbf{x}_1, \dots, \mathbf{x}_M\}$ and one step after. Let us define:

$$\begin{aligned} \boldsymbol{G}_0 &= [\mathbf{g}(\mathbf{x}_1), \dots, \mathbf{g}(\mathbf{x}_M)] \\ \boldsymbol{G}_1 &= [\mathbf{g}(\mathbf{F}(\mathbf{x}_1)), \dots, \mathbf{g}(\mathbf{F}(\mathbf{x}_M))] = [\mathcal{K}\mathbf{g}(\mathbf{x}_1), \dots, \mathcal{K}\mathbf{g}(\mathbf{x}_M)] \,. \end{aligned} \tag{2.10}$$

In analogy to what has been done in the linear case, $\boldsymbol{K} = \boldsymbol{G}_1\boldsymbol{G}_0^\dagger$ provides the best finite-dimensional approximation of the Koopman operator that can be achieved using the available data. Therefore, we can approximate the Koopman eigenvalues and modes from the eigendecomposition of $\boldsymbol{K}$, as summarized in Algorithm 2. In Algorithm 2 we actually transform $\boldsymbol{K}$ by multiplying it by $\boldsymbol{U}^*$ on the left and $\boldsymbol{U}$ on the right to highlight the similarities with algorithm 1. In our setting, where the number of components of the observable is much lower than the number of snapshots and $\boldsymbol{U}$ is a unitary matrix, this does not make any difference, as it is simply a similarity transformation.

Algorithm 2 : DMD for Koopman Operator

Input: Measurements of a vector values observable, collected in data matrices $\boldsymbol{G}_0$, $\boldsymbol{G}_1$ defined as in (2.10).

```
1: Compute the (possibly truncated) SVD of $\boldsymbol{G}_0 = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{W}^*$.
2: Define the matrix $\boldsymbol{K} := \boldsymbol{U}^*\boldsymbol{G}_1\boldsymbol{W}\boldsymbol{\Sigma}^\dagger$.
3: Compute the eigenpairs $\{(\lambda_j, \mathbf{x}_j)\}_j$ of $\boldsymbol{K}$. Each nonzero eigenvalue is a DMD eigenvalue.
4: Compute the modes corresponding to the nonzero eigenvalue $\lambda_j$ as $\mathbf{v}_j = \boldsymbol{U}\mathbf{x}_j$.
```

Output: Approximation of the Koopman eigenvalues (λ_j) and modes $(\mathbf{v}_j)$.

The interpretation of $\boldsymbol{K}$ as the best finite-dimensional approximation of the Koopman Operator gives an idea of why Algorithm 2 might work. However, this does not provide a mathematical foundation for the method, and indeed in [25, 3] it is argued that in the non-linear case Algorithm 2 might not perform well in general. However, under fairly restrictive hypothesis, Algorithm 2 outputs approximations of the Koopman modes and associated eigenvalues that are indistinguishable from the exact ones using the available data. We refer the reader to [21, 25] for a more precise statement.

3. Extended Dynamic Mode Decomposition and its variants

In this third chapter, we introduce Extended Dynamic Mode Decomposition (EDMD) as a method for approximating eigenpairs from a dictionary of observables. In Section 3.2 we introduce the problem of spectral pollution, and we explain how Residual DMD (ResDMD) can be used to remove it. Then we present pseudospectra as an effective method to individuate spectral pollution, as well as the use of ResDMD for approximating the pseudospectrum of the Koopman Operator. Finally, in Section 3.3 we reproduce some of the numerical examples presented in [9].

3.1 Extended Dynamic Mode Decomposition (EDMD)

The *Extended Dynamic Mode Decomposition* (EDMD), originally presented in [26], is a method that seeks at approximating the Koopman Operator as a finite dimensional operator and then approximate the Koopman (eigenvalue, eigenfunction, mode) tuples from this finite dimensional approximation. The algorithm requires in input:

- a dataset of snapshot pairs of the system state, $\{(\mathbf{x}_0^{(m)}, \mathbf{x}_1^{(m)})\}_{m=1}^{M}$ with $\mathbf{x}_1^{(m)} = \mathbf{F}(\mathbf{x}_0^{(m)})$;
- a dictionary of observables $\mathcal{D} = \{\psi_1, \dots, \psi_K\} \subseteq \mathcal{D}(\mathcal{K})$. Let us define the vector-valued observable $\Psi : \Omega \to \mathbb{C}^{1\times K}$ as $\Psi(x) = [\psi_1(x), \dots, \psi_K(x)]$. The choice of the dictionary $\mathcal{D}$ is not easy and depends on the problem at hand. For the moment, let us assume that $\mathcal{D}$ is rich enough to approximate at least a few of the dominant Koopman eigenfunctions.

3.1.1 Approximation of $\mathcal{K}$ and its eigenpairs

Let $\phi \in \text{Span}(\mathcal{D})$, then we can write:

$$\phi = \sum_{k=1}^{K} a_k \psi_k = \Psi \mathbf{a}, \qquad \mathbf{a} \in \mathbb{C}^K.$$

We aim at generating $\boldsymbol{K} \in \mathbb{C}^{k\times k}$ finite dimensional approximation of $\mathcal{K}$ such that

$$\mathcal{K}\phi = (\Psi \circ \mathbf{F})\mathbf{a} = \Psi \boldsymbol{K} \mathbf{a} + r \tag{3.1}$$

where the residual term $r = r(\mathbf{a}, \mathbf{x})$ is due to the fact that, in general, Span($\mathcal{D}$) is not $\mathcal{K}$-invariant. To obtain the "best" $\boldsymbol{K} \in \mathbb{C}^{K\times K}$ it is natural to minimize the norm of the point-wise maximum of the residual of all possible $\phi \in \text{Span}(\mathcal{D})$, with some kind of normalisation. Hence we aim at solving the following problem [9]:

$$\underset{\boldsymbol{K}\in\mathbb{C}^{K\times K}}{\arg\min} \int_\Omega \max_{\mathbf{a}\in\mathbb{C}^k \|\mathbf{a}\|=1} |r(\mathbf{a}, \mathbf{x})|^2 d\omega(\mathbf{x}) = \underset{\boldsymbol{K}\in\mathbb{C}^{K\times K}}{\arg\min} \int_\Omega \|\Psi(\mathbf{F}(\mathbf{x})) - \Psi(\mathbf{x})\boldsymbol{K}\|^2 d\omega(\mathbf{x}). \tag{3.2}$$

We cannot directly compute the integral, thus we need to use a quadrature rule. We take as quadrature nodes our snapshot data $\{\mathbf{x}_0^{(m)}\}_{m=1}^M$ with weights $\{w_m\}_{m=1}^M$. The discretized problem reads:

$$\begin{aligned}
&\underset{\boldsymbol{K}\in\mathbb{C}^{k\times k}}{\arg\min} \sum_{j=1}^{M} w_j \left\| \Psi(\mathbf{F}(\mathbf{x}_0^{(j)})) - \Psi(\mathbf{x}_0^{(j)})\boldsymbol{K} \right\|^2 = \\
=&\underset{\boldsymbol{K}\in\mathbb{C}^{k\times k}}{\arg\min} \sum_{j=1}^{M} w_j \left\| \Psi(\mathbf{x}_1^{(j)}) - \Psi(\mathbf{x}_0^{(j)})\boldsymbol{K} \right\|^2 .
\end{aligned} \tag{3.3}$$

If we define the matrices

$$\boldsymbol{W} = \operatorname{diag}(w_1, \dots, w_M) \in \mathbb{R}_+^{M\times M} \tag{3.4}$$

$$\Psi_0 = \left[\Psi(\mathbf{x}_0^{(1)})^T, \dots, \Psi(\mathbf{x}_0^{(M)})^T\right]^T \in \mathbb{C}^{M\times K} \tag{3.5}$$

$$\Psi_1 = \left[\Psi(\mathbf{x}_1^{(1)})^T, \dots, \Psi(\mathbf{x}_1^{(M)})^T\right]^T \in \mathbb{C}^{M\times K} \tag{3.6}$$

we can write the weighted least-square problem in (3.3) as

$$\underset{\boldsymbol{K}\in\mathbb{C}^{k\times k}}{\arg\min} \left\| \sqrt{\boldsymbol{W}}(\Psi_1 - \Psi_0\boldsymbol{K}) \right\|_F^2 . \tag{3.7}$$

The solution of the weighted least squares problem is

$$(\Psi_0^*\boldsymbol{W}\Psi_0)\boldsymbol{K} = \Psi_0^*\boldsymbol{W}\Psi_1 \implies \boldsymbol{K} = (\Psi_0^*\boldsymbol{W}\Psi_0)^\dagger(\Psi_0^*\boldsymbol{W}\Psi_1), \tag{3.8}$$

where by $\implies$ we mean that $\boldsymbol{K}$ is the solution with minimal norm, but to have uniqueness of the solution one might use regularization techniques. If the weights are the same for all quadrature points, then $\mathbf{K} = (\Psi_0^*\Psi_0)^\dagger\Psi_0^*\Psi_1 = \Psi_0^\dagger\Psi_1$ and we are back to the transpose of the matrix that we would obtain applying the DMD algorithm with vector-valued observable Ψ^T.

Observe that the matrices involved in the computations are

$$\begin{aligned}
\Psi_0^*\boldsymbol{W}\Psi_0 &= \sum_{j=1}^{M} w_j \Psi(\mathbf{x}_0^{(j)})^*\Psi(\mathbf{x}_0^{(j)}) \\
\Psi_0^*\boldsymbol{W}\Psi_1 &= \sum_{j=1}^{M} w_j \Psi(\mathbf{x}_0^{(j)})^*\Psi(\mathbf{x}_1^{(j)}).
\end{aligned}$$

This rewriting is particularly advantageous when the number of snapshots is much larger than the size of the dictionary, i.e. $M >> K$, because each term of the sum can be computed individually, and the products in (3.1.1) can be built iteratively, without the need of explicitly storing the matrices Ψ_0, $\Psi_1 \in \mathbb{C}^{M\times K}$.

To approximate the eigenvalue-eigenfunction pair of the Koopman Operator, we use the eigenpairs of its finite dimensional approximation $\boldsymbol{K}$. If λ_j is an eigenvalue of $\boldsymbol{K}$ with eigenvector $\boldsymbol{\xi}_j$, an approximation of an eigenvalue-eigenfunction pair of $\mathcal{K}$ is $(\lambda_j, \phi_j = \Psi\boldsymbol{\xi}_j)$.

A few final remarks for practical implementation of EDMD:

- by reducing the size of the dictionary, we may assume without loss of generality that the matrix $(\Psi_0^*\boldsymbol{W}\Psi_0)$ is non-singular. However, we might also consider using TSVD when computing its pseudo-inverse in practice.
- Instead of computing the matrix $\boldsymbol{K} = (\Psi_0^*\boldsymbol{W}\Psi_0)^\dagger(\Psi_0^*\boldsymbol{W}\Psi_1)$ and then its eigenpairs, it is often numerically more stable to solve the generalized eigenvalue problem $(\Psi_0^*\boldsymbol{W}\Psi_0)\boldsymbol{\xi} = \lambda(\Psi_0^*\boldsymbol{W}\Psi_1)\boldsymbol{\xi}$.

3.1.2 Approximation of the Koopman modes for the full state observable

Let us consider the full state observable

$$\mathbf{g}(\mathbf{x}) = \begin{bmatrix} g_1(\mathbf{x}) \\ \vdots \\ g_d(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \mathbf{e}_1^*\mathbf{x} \\ \vdots \\ \mathbf{e}_d^*\mathbf{x} \end{bmatrix} = \mathbf{x}. \tag{3.9}$$

Let us assume that $g_i \in \operatorname{Span}(\mathcal{D})$ for all $i = 1, \dots, d$. If this is not the case, an intermediate step to project g_i onto $\operatorname{Span}(\mathcal{D})$ is required, with an accuracy strongly dependent on the choice of the dictionary. If $g_i \in \operatorname{Span}(\mathcal{D})$ we can write

$$g_i = \sum_{k=1}^{K} b_{ki}\psi_k = \Psi\mathbf{b}_i \tag{3.10}$$

and in matrix form

$$\mathbf{g} = \boldsymbol{B}^T\Psi^T = (\Psi\boldsymbol{B})^T, \qquad \boldsymbol{B} = [\mathbf{b}_1, \dots, \mathbf{b}_d] \in \mathbb{C}^{k\times d}.$$

If we choose an orthonormal dictionary this step can be accomplished through the computation of the inner products $b_{ki} = \langle g_i, \psi_k\rangle$. This also accounts for the projection onto $\operatorname{Span}(\mathcal{D})$, if required.

Let $\boldsymbol{\Xi} = [\boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_K]$ be the matrix of the eigenvectors of $\boldsymbol{K}$, the vector of approximate Koopman eigenfunctions $\Phi(\mathbf{x}) = [\phi_1(\mathbf{x}), \dots, \phi_K(\mathbf{x})]$ can be written as $\Phi = \Psi\boldsymbol{\Xi}$. Thus

$$\mathbf{g} = \boldsymbol{B}^T\Psi^T = \boldsymbol{B}^T(\boldsymbol{\Xi}^T)^{-1}\Phi^T = (\boldsymbol{\Xi}^{-1}\boldsymbol{B})^T\Phi^T. \tag{3.11}$$

Since $\boldsymbol{\Xi}$ is the matrix of the eigenvectors of $\boldsymbol{K}$, its inverse is $\boldsymbol{\Xi}^{-1} = [\mathbf{u}_1, \dots, \mathbf{u}_K]$, where $\mathbf{u}_i$ is the left eigenvector of $\boldsymbol{K}$ also associated with λ_i and appropriately scaled so that $\mathbf{u}_i^*\boldsymbol{\xi}_j = \delta_{ij}$. We can therefore compute the left eigenvectors of $\boldsymbol{K}$ and letting $\boldsymbol{V} = (\boldsymbol{U}^*\boldsymbol{B})^T$, then

$$\mathbf{g} = \boldsymbol{V}\Psi^T = \sum_{k=1}^{K} \mathbf{v}_k\psi_k, \tag{3.12}$$

i.e. $\mathbf{v}_j = (\mathbf{u}_j^*\boldsymbol{B})^T$ is the j-th Koopman mode.

In conclusion, as summarized in Algorithm 3, once we have computed the finite-dimensional approximation $\boldsymbol{K}$ of the Koopman Operator:

- the eigenvalues of $\boldsymbol{K}$ are the EDMD approximation of the Koopman eigenvalues;
- the right eigenvectors of $\boldsymbol{K}$ generate the approximation of the eigenfunctions;
- the left eigenvectors of $\boldsymbol{K}$ generate the approximation of the Koopman modes.

If we are interested in an approximation of only a leading subset of eigenvalues, instead of a complete eigendecomposition of the matrix $\boldsymbol{K}$, Krylov methods or other iterative methods might be considered to efficiently compute only the leading eigenpairs of $\boldsymbol{K}$.

3.2 Spectral pollution and Residual DMD (ResDMD)

In order to compute the eigenvalues of the Koopman Operator, we approximate the infinite-dimensional operator $\mathcal{K}$ by a finite matrix. Therefore, even if some of the eigenvalues produced by the EDMD algorithm are reliable, most of them are not. *Pseudospectra* [24] are a tool that allow us to detect this so-called *spectral pollution* [9], i.e. spurious eigenvalues of the matrix $\boldsymbol{K}$ that are only due to the discretization and have no connection with the latent Koopman Operator.

Definition 3.1 (Pseudospectrum of a square matrix)**.** *Given a matrix $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ and $\varepsilon > 0$, the ε-pseudospectrum of $\boldsymbol{A}$ is defined as*

$$\sigma_\varepsilon(\boldsymbol{A}) = \left\{\lambda \in \mathbb{C} : \left\|(\boldsymbol{A} - \lambda\boldsymbol{I})^{-1}\right\|_2 \geq \frac{1}{\varepsilon}\right\} = \{\lambda \in \mathbb{C} : \sigma_{\min}(\boldsymbol{A} - \lambda\boldsymbol{I}) \leq \varepsilon\} \tag{3.13}$$

where $\sigma_{\min}(\boldsymbol{M})$ indicates the minimum singular value of the matrix $\boldsymbol{M}$.

Algorithm 3 : Extended Dynamic Mode Decomposition (EDMD)

Input: Snapshot pairs of the system state, $\{(\mathbf{x}_0^{(m)}, \mathbf{x}_1^{(m)})\}_{m=1}^M$, quadrature weights $\{w_m\}_{m=1}^M$, dictionary of observables $\mathcal{D} = \{\psi_1, \ldots, \psi_K\} \subseteq \mathcal{D}(\mathcal{K})$

```
1: Define the matrices $\boldsymbol{W}$ (3.4), $\Psi_0$ (3.5), $\Psi_1$ (3.6).
2: Compute the matrices $(\Psi_0^*\boldsymbol{W}\Psi_0)$ and $(\Psi_0^*\boldsymbol{W}\Psi_1)$ .
3: Solve the generalized eigenvalue problem $(\Psi_0^*\boldsymbol{W}\Psi_0)\boldsymbol{\xi} = \lambda(\Psi_0^*\boldsymbol{W}\Psi_1)\boldsymbol{\xi}$.
4: Obtain the eigenfunctions as $\Phi = \Psi\Xi$.
5: if want to compute eigenmodes then
6:     Write the (projection onto Span($\mathcal{D}$) of the) full observable $\mathbf{g}$ as $\mathbf{g} = \boldsymbol{B}^T\Psi^T$.
7:     Compute the left eigenvectors $\mathbf{u}_j$ of $\boldsymbol{K} = (\Psi_0^*\boldsymbol{W}\Psi_0)^\dagger(\Psi_0^*\boldsymbol{W}\Psi_1)$.
8:     Compute $\boldsymbol{V} = (\boldsymbol{U}^*\boldsymbol{B})^T$, i.e $\mathbf{v}_j = (\mathbf{u}_j^*\boldsymbol{B})^T$.
9: end if
```

Output: Approximation of the Koopman eigenvalues (λ_j), eigenfunctions (ϕ_j) and, possibly, eigemodes $(\mathbf{v}_j)$.

In analogy with the second form of previous definition, we can define the pseudospectrum of a rectangular matrix pencil as follows.

Definition 3.2 (Pseudospectrum of a rectangular matrix pencil)**.** *Given two rectangular matrices* $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{C}^{m\times n}$ *and* $\varepsilon > 0$*, the* ε*-pseudospectrum of the matrix pencil* $\boldsymbol{A} - \lambda\boldsymbol{B}$ *is defined as*

$$\sigma_\varepsilon(\boldsymbol{A}, \boldsymbol{B}) = \{\lambda \in \mathbb{C} : \sigma_{\min}(\boldsymbol{A} - \lambda\boldsymbol{B}) \le \varepsilon\} \tag{3.14}$$

The pseudospectrum of a rectangular matrix $\boldsymbol{A} \in \mathbb{C}^{m\times n}$ *is defined as* $\sigma_\varepsilon(\boldsymbol{A}, \boldsymbol{I})$ *where*

$$\boldsymbol{I} = \begin{bmatrix} \boldsymbol{I}_n \\ \mathbf{0}_{m-n,n} \end{bmatrix} \text{ if } n < m, \qquad \boldsymbol{I} = \begin{bmatrix} \boldsymbol{I}_m & \mathbf{0}_{m,n-m} \end{bmatrix} \text{ if } m < n. \tag{3.15}$$

However, $\mathcal{K}$ is an infinite dimensional operator and might be unbounded, hence following [9] we can define

$$\sigma_\varepsilon(\mathcal{K}) = \mathrm{cl}\left(\left\{\lambda \in \mathbb{C} : \left\|(\mathcal{K} - \lambda \cdot \mathrm{id})^{-1}\right\| > \frac{1}{\varepsilon}\right\}\right) = \mathrm{cl}\left(\bigcup_{\|\boldsymbol{B}\|<\varepsilon} \sigma(\mathcal{K} + \boldsymbol{B})\right) \tag{3.16}$$

where id is the identity operator and cl is the closure of a set.

3.2.1 ResDMD to remove spectral pollution

The *Residual DMD* (ResDMD) algorithm performs the same steps of EDMD, but then discards the approximate eigenpairs $\{(\lambda_j, \phi_j)\}_j$ that have a residual above a certain prescribed tolerance ε. Given a candidate eigenpair (λ, ϕ) of $\mathcal{K}$, where $\phi = \boldsymbol{\Psi\xi}$, to measure its accuracy we can consider the relative residual

$$\begin{aligned} \mathrm{res}(\lambda, \phi)^2 &= \frac{\int_\Omega |[\mathcal{K}\phi](x) - \lambda\phi(x)|^2 \, d\omega(x)}{\int_\Omega |\phi(x)|^2 \, d\omega(x)} = \frac{\langle(\mathcal{K} - \lambda \cdot \mathrm{id})\phi, (\mathcal{K} - \lambda \cdot \mathrm{id})\phi\rangle}{\langle\phi, \phi\rangle} = \\ &= \frac{\sum_{j,k=1}^K \overline{\xi}_j\xi_k \left(\langle\mathcal{K}\psi_k, \mathcal{K}\psi_j\rangle - \lambda\langle\psi_k, \mathcal{K}\psi_j\rangle - \overline{\lambda}\langle\mathcal{K}\psi_k, \psi_j\rangle + |\lambda|^2\langle\psi_k, \psi_j\rangle\right)}{\sum_{j,k=1}^K \overline{\xi}_j\xi_k\langle\psi_k, \psi_j\rangle}. \end{aligned} \tag{3.17}$$

Once again, this integral cannot be computed exactly, and to approximate it we use the same quadrature rule used to approximate (3.2). Hence:

$$\begin{aligned} \mathrm{res}(\lambda, \phi)^2 &\approx \frac{\sum_{j,k=1}^K \overline{\xi}_j\xi_k \left((\Psi_1^*\boldsymbol{W}\Psi_1)_{jk} - \lambda(\Psi_1^*\boldsymbol{W}\Psi_0)_{jk} - \overline{\lambda}(\Psi_0^*\boldsymbol{W}\Psi_1)_{jk} + |\lambda|^2(\Psi_0^*\boldsymbol{W}\Psi_0)_{jk}\right)}{\sum_{j,k=1}^K \overline{\xi}_j\xi_k(\Psi_0^*\boldsymbol{W}\Psi_0)_{jk}} = \\ &= \frac{\boldsymbol{\xi}^* \left((\Psi_1^*\boldsymbol{W}\Psi_1) - \lambda(\Psi_1^*\boldsymbol{W}\Psi_0) - \overline{\lambda}(\Psi_0^*\boldsymbol{W}\Psi_1) + |\lambda|^2(\Psi_0^*\boldsymbol{W}\Psi_0)\right)\boldsymbol{\xi}}{\boldsymbol{\xi}^*(\Psi_0^*\boldsymbol{W}\Psi_0)\boldsymbol{\xi}} =: \widetilde{\mathrm{res}}^2(\lambda, \phi). \end{aligned} \tag{3.18}$$

If $\widetilde{\text{res}}(\lambda, \phi) > \varepsilon$, the eigenpair (λ, ϕ) is discarded. This algorithm is summarized in Algorithm 4. Observe that to compute the approximation of the residual we only need to compute $(\Psi_1^* \boldsymbol{W} \Psi_1)$, because the other matrices are already computed in the standard EDMD algorithm.

Algorithm 4 : Residual Dynamic Mode Decomposition (ResDMD)

Input: Snapshot pairs of the system state, $\{(\mathbf{x}_0^{(m)}, \mathbf{x}_1^{(m)})\}_{m=1}^M$, quadrature weights $\{w_m\}_{m=1}^M$, dictionary of observables $\mathcal{D} = \{\psi_1, \ldots, \psi_K\} \subseteq \mathcal{D}(\mathcal{K})$, tolerance $\varepsilon > 0$.

```
1: Execute Algorithm 3.
2: Compute the matrix $(\Psi_1^* \boldsymbol{W} \Psi_1)$.
3: for each approximate eigenpair $(\lambda_j, \phi_j)$ do
4:     Compute $\widetilde{\text{res}}(\lambda_j, \phi)_j$ as in (3.18).
5:     if $\widetilde{\text{res}}(\lambda_j, \phi)_j \geq \varepsilon$ then
6:         Discard the approximate eigenpair $(\lambda_j, \phi_j)$.
7:     end if
8: end for
```

Output: Approximation of the eigenvalues (λ_j), eigenfunctions (ϕ_j) with $\widetilde{\text{res}}(\lambda_j, \phi)_j < \varepsilon$

The above-described cleanup procedure avoids spectral pollution and, if the quadrature converges, in the large data limit, removes eigenpairs with a residual above the prescribed tolerance ε, and retains only eigenpairs that are inside the ε-pseudospectrum of $\mathcal{K}$. This is precised in the following proposition.

Proposition 3.1 ([9], Theorem 4.1)**.** *Assume that the quadrature rule converges, i.e. that*

$$\lim_{M\to+\infty} (\Psi_0^* \boldsymbol{W} \Psi_0)_{jk} = \langle \psi_j, \psi_k \rangle, \qquad \lim_{M\to+\infty} (\Psi_0^* \boldsymbol{W} \Psi_1)_{jk} = \langle \psi_j, \mathcal{K}\psi_k \rangle, \qquad \lim_{M\to+\infty} (\Psi_1^* \boldsymbol{W} \Psi_1)_{jk} = \langle \mathcal{K}\psi_j, \mathcal{K}\psi_k \rangle.$$

Let us denote by Λ the eigenvalues in the output of Algorithm 4, then

$$\limsup_{M\to+\infty} \max_{\lambda\in\Lambda} \left\|(\mathcal{K} - \lambda)^{-1}\right\|^{-1} \leq \epsilon. \tag{3.19}$$

3.2.2 ResDMD to approximate the pseudospectrum

In spite of the fact that the eigenvalues returned by Algorithm 4 lie in the ε-pseudospectrum (in the large data limit), the eigenvalues of $\boldsymbol{K}$ might not approximate the whole spectrum of $\mathcal{K}$ and, therefore, it might not be possible to approximate the whole ε-pseudospectrum of $\mathcal{K}$ using Algorithm 4. A simple modification of Algorithm 4 allows drawing an approximation of the ε-pseudospectrum of $\mathcal{K}$ starting from a grid of points in the complex plane. Given a point $z_j \in \mathbb{C}$ on a grid, we search the function ψ_j in $\text{Span}(\mathcal{D})$ that minimizes $\text{res}(z_j, \psi_j)$, i.e. we solve

$$\tau_j = \min_{\mathbf{g}\in\mathbb{C}^K} \text{res}(z_j, \Psi\mathbf{g}), \qquad \mathbf{g}_j = \arg\min_{\mathbf{g}\in\mathbb{C}^K} \text{res}(z_j, \Psi\mathbf{g}). \tag{3.20}$$

Following (3.18), the discretized problem reads:

$$\tau_j = \min_{\mathbf{g}\in\mathbb{C}^K} \frac{\mathbf{g}^* \boldsymbol{D}(z_j)\mathbf{g}}{\mathbf{g}^*(\Psi_0^* \boldsymbol{W} \Psi_0)\mathbf{g}}\ , \qquad \mathbf{g}_j = \arg\min_{\mathbf{g}\in\mathbb{C}^K} \frac{\mathbf{g}^* \boldsymbol{D}(z_j)\mathbf{g}}{\mathbf{g}^*(\Psi_0^* \boldsymbol{W} \Psi_0)\mathbf{g}} \tag{3.21}$$

$$\begin{aligned} \boldsymbol{D}(z_j) &= (\Psi_1^* \boldsymbol{W} \Psi_1) - z_j(\Psi_1^* \boldsymbol{W} \Psi_0) - \overline{z}_j(\Psi_0^* \boldsymbol{W} \Psi_1) + |z_j|^2(\Psi_0^* \boldsymbol{W} \Psi_0) = \\ &= (\Psi_1 - z_j\Psi_0)^* \boldsymbol{W} (\Psi_1 - z_j\Psi_0). \end{aligned} \tag{3.22}$$

The matrices $(\Psi_0^* \boldsymbol{W} \Psi_0)$ and $\boldsymbol{D}(z_j)$ are hermitian positive semidefinite. If we also assume that Ψ_0 has full column rank, from the properties of the Rayleigh quotient solving (3.21) is equivalent to finding the smallest eigenvalue and the associated eigenvector of the generalized eigenvalue problem $\boldsymbol{D}(z_j)\mathbf{g} = \tau(\Psi_0^* \boldsymbol{W} \Psi_0)\mathbf{g}$, which allows for a more efficient computation. This algorithm, proposed in [9], is summarized in Algorithm 5. Convergence guarantees for the approximation of the ε-pseudospectrum can be found in [9].

The following proposition allows rewriting the pseudospectrum approximation provided by Algorithm 5 as the pseudospectrum either of a rectangular matrix pencil or of a rectangular matrix.

Algorithm 5 : ResDMD for pseudospectrum approximation

Input: Snapshot pairs of the system state, $\{(\mathbf{x}_0^{(m)}, \mathbf{x}_1^{(m)})\}_{m=1}^M$, quadrature weights $\{w_m\}_{m=1}^M$, dictionary of observables $\mathcal{D} = \{\psi_1, \ldots, \psi_K\} \subseteq \mathcal{D}(\mathcal{K})$, tolerance $\varepsilon > 0$, grid $z_1, \ldots, z_k \in \mathbb{C}$.

```
1: Define the matrices $\boldsymbol{W}$ (3.4), $\Psi_0$ (3.5), $\Psi_1$ (3.6).
2: Compute the matrices $(\Psi_0^* \boldsymbol{W} \Psi_0)$, $(\Psi_0^* \boldsymbol{W} \Psi_1)$ and $(\Psi_1^* \boldsymbol{W} \Psi_1)$.
3: for each $z_j$ in the grid do
4:     Solve $\tau_j = \min_{\mathbf{g} \in \mathbb{C}^K} \operatorname{res}(z_j, \Psi\mathbf{g})$, $\mathbf{g}_j = \arg\min_{\mathbf{g} \in \mathbb{C}^K} \operatorname{res}(z_j, \Psi\mathbf{g})$.
5: end for
```

Output: Estimate of the ε-pseudospectrum $\{z_j : \tau_j < \varepsilon\}$ and corresponding approximate eigenfunctions $\{\mathbf{g}_j : \tau_j < \varepsilon\}$.

Proposition 3.2. *Let us assume that Ψ_0 has full column rank and let us consider the singular value decomposition $\sqrt{\boldsymbol{W}}\Psi_0 = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}$, where $\boldsymbol{U} \in \mathbb{C}^{M\times K}$ and $\boldsymbol{\Sigma}, \boldsymbol{V} \in \mathbb{C}^{K\times K}$. Then the approximation of the pseudospectrum of the Koopman operator provided by Algorithm 5 coincides with the pseudospectrum of the rectangular matrix pencil*

$$\sqrt{\boldsymbol{W}}\Psi_1 \boldsymbol{V}\boldsymbol{\Sigma}^{-1} - \lambda \boldsymbol{U}. \tag{3.23}$$

Moreover, if the columns of $\boldsymbol{U}_\perp$ complete the columns of $\boldsymbol{U}$ to an orthonormal basis of $\mathbb{C}^M$, then the pseudospectrum approximation coincides also with the pseudospectrum of the rectangular matrix

$$\begin{bmatrix} \boldsymbol{U}^* \sqrt{\boldsymbol{W}}\Psi_1 \boldsymbol{V}\boldsymbol{\Sigma}^{-1} \\ \boldsymbol{U}_\perp^* \sqrt{\boldsymbol{W}}\Psi_1 \boldsymbol{V}\boldsymbol{\Sigma}^{-1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Sigma}\mathbf{V}^*(\Psi_0^\dagger \Psi_1)\mathbf{V}\boldsymbol{\Sigma}^{-1} \\ \boldsymbol{U}_\perp^* \sqrt{\boldsymbol{W}}\Psi_1 \boldsymbol{V}\boldsymbol{\Sigma}^{-1} \end{bmatrix}. \tag{3.24}$$

Proof. Let $z \in \mathbb{C}$ and let us consider the corresponding residual:

$$\begin{aligned} \tau(z) = \min_{\mathbf{g}\in\mathbb{C}^K} \frac{\mathbf{g}^* \boldsymbol{D}(z)\mathbf{g}}{\mathbf{g}^*(\Psi_0^* \boldsymbol{W}\Psi_0)\mathbf{g}} &= \min_{\mathbf{g}\in\mathbb{C}^K} \frac{\mathbf{g}^* \boldsymbol{D}(z)\mathbf{g}}{\mathbf{g}^*(\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{V}^*)^2\mathbf{g}} = \min_{\mathbf{v}\in\mathbb{C}^K} \frac{\mathbf{v}^*(\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{V}^*)\boldsymbol{D}(z)(\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{V}^*)\mathbf{v}}{\mathbf{v}^*\mathbf{v}} = \\ &= \lambda_{\min}\left((\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{V}^*)\boldsymbol{D}(z)(\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{V}^*)\right) = \lambda_{\min}\left(\boldsymbol{C}(z)^*\boldsymbol{C}(z)\right) = \sigma_{\min}(\boldsymbol{C}(z))^2 \end{aligned}$$

where from (3.22) we can write $\boldsymbol{C}(z) = \sqrt{\boldsymbol{W}}(\Psi_1 - z\Psi_0)\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{V}^*$. Hence $\tau(z) = \sigma_{\min}(\boldsymbol{C}(z))^2 \le \varepsilon^2$ if and only if $\sigma_{\min}(\boldsymbol{C}(z)) \le \varepsilon$. Now observe that

$$\begin{aligned} \sigma_{\min}(\boldsymbol{C}(z)) = \sigma_{\min}(\sqrt{\boldsymbol{W}}(\Psi_1 - z\Psi_0)\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{V}^*) &= \sigma_{\min}(\sqrt{\boldsymbol{W}}(\Psi_1 - z\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^*)\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{V}^*) = \\ = \sigma_{\min}(\sqrt{\boldsymbol{W}}(\Psi_1 - z\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^*)\boldsymbol{V}\boldsymbol{\Sigma}^{-1}) &= \sigma_{\min}(\sqrt{\boldsymbol{W}}\Psi_1\boldsymbol{V}\boldsymbol{\Sigma}^{-1} - z\boldsymbol{U}) \end{aligned}$$

from which the first statement follows.

Moreover $\boldsymbol{Q} = [\boldsymbol{U} \,|\, \boldsymbol{U}_\perp]$ is a unitary matrix, hence multiplying by Q on the left the singular values are preserved, yielding:

$$\begin{aligned} \sigma_{\min}(\boldsymbol{C}(z)) &= \sigma_{\min}(\boldsymbol{Q}^*\sqrt{\boldsymbol{W}}\Psi_1\boldsymbol{V}\boldsymbol{\Sigma}^{-1} - z\boldsymbol{Q}^*\boldsymbol{U}) = \\ &= \sigma_{\min}\left(\begin{bmatrix} \boldsymbol{U}^*\sqrt{\boldsymbol{W}}\Psi_1\boldsymbol{V}\boldsymbol{\Sigma}^{-1} \\ \boldsymbol{U}_\perp^*\sqrt{\boldsymbol{W}}\Psi_1\boldsymbol{V}\boldsymbol{\Sigma}^{-1} \end{bmatrix} - z\begin{bmatrix} \boldsymbol{I}_K \\ \mathbf{0}_{M-K,K} \end{bmatrix}\right). \end{aligned}$$

Finally, one can write the upper part of the matrix as:

$$\boldsymbol{U}^*\sqrt{\boldsymbol{W}}\Psi_1\boldsymbol{V}\boldsymbol{\Sigma}^{-1} = \boldsymbol{\Sigma}\boldsymbol{V}^*\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^*\sqrt{\boldsymbol{W}}\Psi_1\boldsymbol{V}\boldsymbol{\Sigma}^{-1} = \boldsymbol{\Sigma}\mathbf{V}^*(\Psi_0^\dagger\Psi_1)\mathbf{V}\boldsymbol{\Sigma}^{-1}.$$

□

From the previous rewriting, one can use standard techniques for the computation of pseudospectra of rectangular matrices [28], as long as M is not too big. Observe indeed that the matrix pencil in Proposition 3.2 is of size $M \times K$, while the generalized eigenvalue problems in Algorithm 5 involve matrices of size $K \times K$.

3.3 Numerical Examples

In this section we reproduce two numerical examples from Section 4 of [9].

3.3.1 Gauss iterated map

The Gauss iterated map is a function $F : \mathbb{R} \to \mathbb{R}$ defined by $F(x) = \exp\big(\alpha x^2\big) + \beta$, where α and β are parameters. We consider the case $\alpha = 2$ and $\beta = 1 - \exp(-\alpha) = 1 - e^{-2}$, with state space $\Omega = [-1, 0]$ and the Lebesgue measure. Observe that with such a choice of the parameters, F maps Ω onto itself, as necessary to have a well defined dynamical system on Ω. As a dictionary of observable $\mathcal{D}$ we consider the first K normalized Legendre polynomials transplanted to the interval Ω. To understand the influence on the accuracy of the algorithm of the choice of the snapshot pairs $\{(\mathbf{x}_0^{(m)}, \mathbf{x}_1^{(m)})\}_{m=1}^M$, which are the nodes of the quadrature rule, we consider four different types of quadrature rule:

- Gauss-Legendre: $\{\mathbf{x}_0^{(m)}\}_{m=1}^M$ are the Gauss-Legendre quadrature nodes transplanted to the interval Ω, with $\{w_m\}_{m=1}^M$ the corresponding quadrature weights.
- Trapezoidal: $\mathbf{x}_0^{(m)} = -1 + \frac{m-1}{M-1}$, with $w_m = \frac{1}{M-1},\ m = 2, \ldots, M-1$ and $w_1 = w_M = \frac{1}{2(M-1)}$.
- Riemann sum: $\mathbf{x}_0^{(m)} = -1 + \frac{m-1}{M-1}$, with $w_m = \frac{1}{M-1},\ m = 1, \ldots, M$.
- Montecarlo: $\{\mathbf{x}_0^{(m)}\}_{m=1}^M$ are chosen uniformly at random over the interval Ω, with $w_m = \frac{1}{M-1},\ m = 1, \ldots, M$.

Figure 3.1 shows the approximations of the eigenvalues of the Koopman Operator computed using DMD, EDMD and ResDMD for each of the four quadrature rules. Figure 3.2 shows the approximation of the ε-pseudospectrum for $\varepsilon = 0.3,\ 0.1,\ 0.01,\ 0.001$ computed using Algorithm 5, and the convergence of the different quadrature rules measured as $\max_{1\le j,k\le K} |(\Psi_0^* W \Psi_1)_{jk} - \langle \mathcal{K}\psi_k, \psi_j\rangle|$.

It can be seen that, for the Montecarlo and the Riemann Sums quadrature rules, which use the same weights for all quadrature points, EDMD and DMD compute the same eigenvalues (magenta dots are the centres of the green circles in Figure 3.1). The outputs of the two algorithms are almost equal also for the Trapezoidal rule, which changes the weights only at the extremes, while they significantly differ for the Gauss-Legendre rule.

The convergence rates are in agreement with standard theoretical results on numerical integration [20], with the Gauss-Legendre quadrature converging exponentially and the other quadrature rules polynomially. It can be understood that, if one has the possibility of choosing the initial condition of the snapshot data, selecting them wisely can make the discretized problem significantly closer to the continuous one. This has a strong impact also on the accuracy of the eigenvalues computed using Algorithm 4. Indeed, the more accurate is the computation of the residual's integral, the more eigenvalues can be removed due to a too high residual, as shown in Figure 3.1.

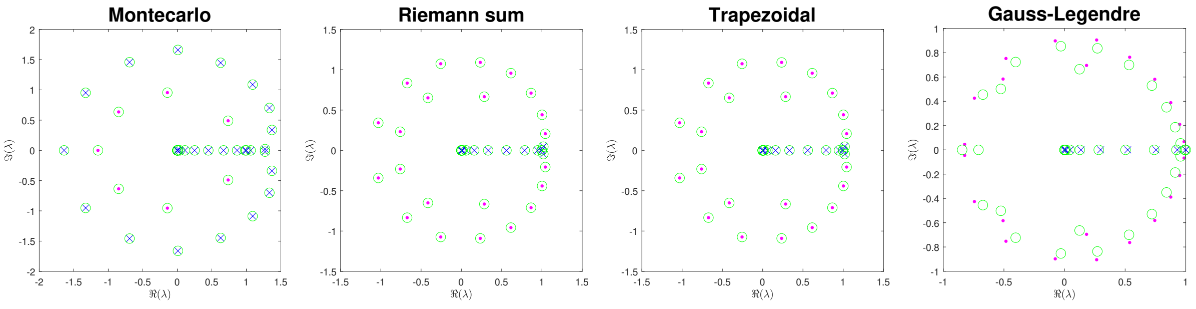
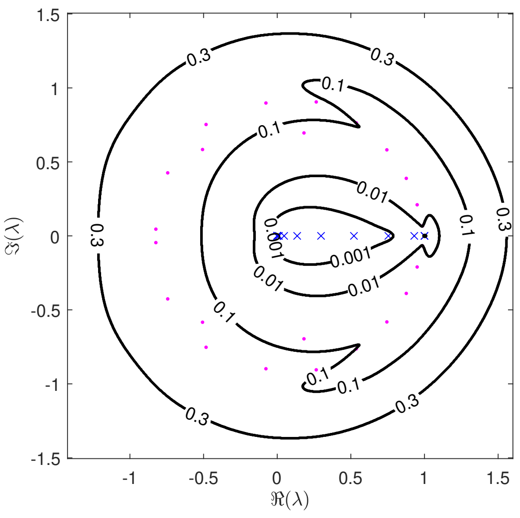
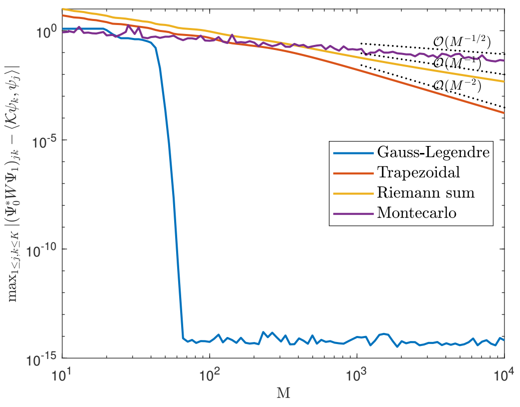


Figure 3.1: Approximations of the eigenvalues of the Koopman Operator associated with the Gauss iterated map computed with the DMD algorithm (green circles) and the EDMD algorithm (magenta dots and blue crosses), using different quadrature rules. The approximate eigenvalues are computed using $K = 40$ observables and $M = 100$ quadrature points. The blue crosses indicate the eigenvalues retained by ResDMD, with tolerance $\varepsilon = 0.01$.



(a) Pseudospectral contours



(b) Convergence of the quadrature rules

Figure 3.2: Contour lines of the ε-pseudospectrum for $\varepsilon = 0.3, 0.1, 0.01, 0.001$ computed using Algorithm 5 with dictionary of $K = 40$ observables (Figure 3.2a) and convergence of one of the approximate integrals for the different quadrature rules measured as $\max_{1\le j,k\le K} |(\Psi_0^* W \Psi_1)_{jk} - \langle \mathcal{K}\psi_k, \psi_j\rangle|$ (Figure 3.2b).

3.3.2 Nonlinear pendulum

Let us consider the continous-time dynamical system of the nonlinear pendulum. The state variable is $\mathbf{x} = (x_1, x_2) = (\theta, \dot\theta) \in \Omega = [-\pi, \pi]_{\text{per}} \times \mathbb{R}$ and the differential equation governing the motion is

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot x_1 \\ \dot x_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\sin(x_1) \end{bmatrix}. \tag{3.25}$$

We consider the corresponding discrete-time dynamical system with time-step $\Delta_t = 0.5$ and the associated Koopman Operator. The system is Hamiltonian and hence the Koopman Operator is unitary [14].

The choice of the dictionary of observables is now more complex, because the function governing the motion is vector-valued (with two components) and periodic in the first component. Since the motion is periodic in $x_1 \in [-\pi, \pi]_{\text{per}}$, it is natural to use the Fourier basis $\{e_n\}_{n\in\mathbb{Z}}$ in x_1 to describe the motion in this direction. Similarly, to approximate the motion in the direction of x_2 one might decide to use the Hermite functions $\{h_n\}_{n\in\mathbb{N}}$, which form a basis of the Hilbert space $L^2(\mathbb{R})$. Therefore, we would ideally consider observables of the form $\phi(x_1, x_2) = \sum_{n\in\mathbb{Z},\, m\in\mathbb{N}} \alpha_{n,m} e_n(x_1) \cdot h_m(x_2)$ for some coefficients $\{\alpha_{n,m}\} \in \mathbb{C}$. However, our dictionary of observables must be finite and we need to decide a criterion to truncate the sum. Observe that there is not a natural ordering, such as if the indices were in $\mathbb{N}$ or $\mathbb{Z}$. Since we want to be able to approximate the (components of the) full state observable, it is important to have the indices $\{(n,0)\}_{n=0}^{n_{\max}}$ and $\{(0,m)\}_{m=-m_{\max}}^{m_{\max}}$ with $n_{\max}, m_{\max}$ as high as possible. Selecting the indices in the rectangle $\{(n,m) : -n_{\max} \le n \le n_{\max},\ 0 \le m \le m_{\max}\}$ would make the size of the dictionary grow too fast in $n_{\max}$ and $m_{\max}$. Therefore, we use an hyperbolic cross approximation [11], i.e. we select the indices in the hyperbolic cross $\{(n,m) : \max(n,1) \cdot \max(m,1) \le N\}$, where N is called the order of the hyperbolic cross approximation.

Using Algorithm 5 we compute an approximation of the ε-pseudospectrum for $\varepsilon = 0.25$, first using $K = 193$ observables and $M = 10^4$ data points, then employing $K = 1265$ and $M = 9 \times 10^4$. As the size K of the dictionary increases, the approximation of the ε-pseudospectrum approaches the annulus with internal radius 0.75 and external radius 1.25(see Figure 3.3). We also compute approximation of the eigenfunctions associated with the eigenvalues $\lambda = \exp(0.4932i)$, $\lambda = \exp(0.9765i)$, $\lambda = \exp(1.4452i)$, $\lambda = \exp(1.8951i)$, (see Figure 3.4).

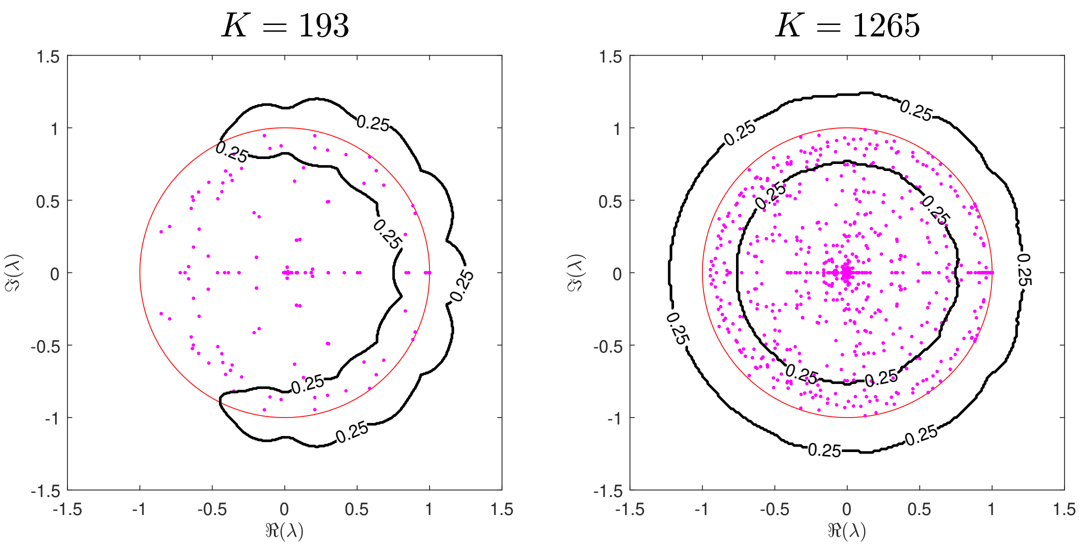


Figure 3.3: Contour lines of the ε-pseudospectrum for $\varepsilon = 0.25$ computed using Algorithm 5 with hyperbolic cross approximation. On the left the approximation obtained using a dictionary of $K = 193$ observables and $M = 10^4$ data points, on the right the one obtained with $K = 1265$ and $M = 9 \times 10^4$.

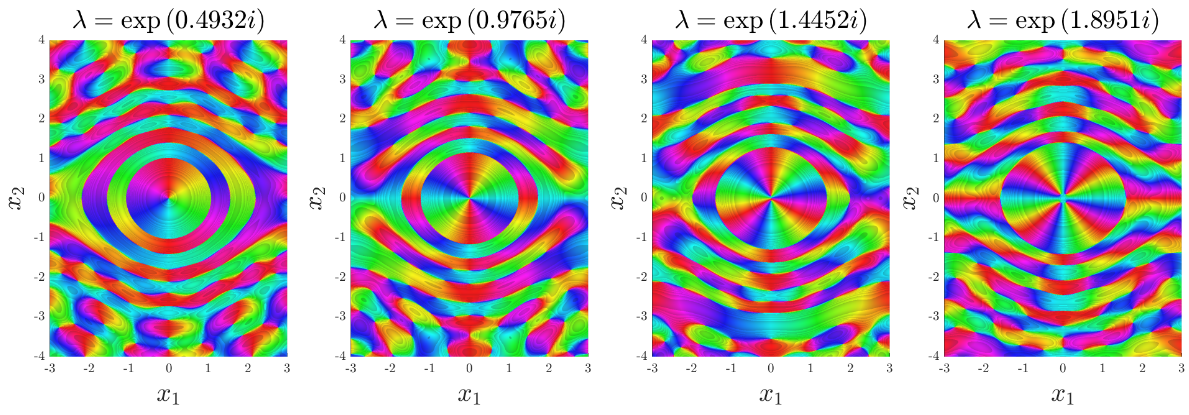


Figure 3.4: Phase portraits of the approximate eigenfunctions of the nonlinear pendulum, where the color illustrates the complex argument of the eigenfunction. Lines of constant modulus are plotted as shadowed steps.

4. A kernel-based approach for high dimensional data

Until now, we made the assumption that the number of snapshots is large compared to the size of the dictionary, which also controls the number of eigenpairs approximated. However, when dealing with high dimensional data, we are usually in the opposite regime, where $K \gg M$. Indeed, as the dimension of the state space increases, the number of functions needed to have a rich enough dictionary overgrows. This blowup of the size K of the dictionary frequently occurs in data-driven and Machine Learning related applications [8, 21, 22], and it can be regarded as a facet of the so-called *curse of dimensionality* [4]. In this final chapter, we present Kernelized EDMD (K-EDMD), an algorithm that uses the kernel trick for a more efficient computation of the EDMD approximations. Unfortunately, having a dictionary size larger than the number of quadrature points makes ResDMD not applicable. This can be solved by using the two-steps procedure presented in Section 4.2. Finally, we apply the previously mentioned two-steps procedure to the Gauss iterated map problem to obtain approximations of the Koopman eigenvalues without the need of a hand-crafted dictionary.

4.1 Kernelized EDMD (K-EDMD)

The EDMD algorithm necessitates the computation of the matrices $(\Psi_0^* \boldsymbol{W} \Psi_0)$ and $(\Psi_0^* \boldsymbol{W} \Psi_1)$, which requires $\mathcal{O}(MK^2)$ flops, and solving an eigenvalue problem associated to the matrix $\boldsymbol{K} \in \mathbb{C}^{K\times K}$, which costs $\mathcal{O}(K^3)$ flops. For large values of K, such computations are impractical. *Kernelized EDMD* (K-EDMD) [27] uses the kernel trick [4] to efficiently compute a matrix $\hat{\boldsymbol{K}} \in \mathbb{C}^{M\times M}$ that has the same non-zero eigenvalues as $\boldsymbol{K}$. The computation of such a matrix relies on the following proposition.

Proposition 4.1 ([9]). *Let $\sqrt{\boldsymbol{W}}\Psi_0 = \boldsymbol{Q}\boldsymbol{\Sigma}\boldsymbol{Z}^*$ be a SVD, where $\boldsymbol{\Sigma} \in \mathbb{C}^{M\times M}$ and let us define*

$$\hat{\boldsymbol{K}} = (\boldsymbol{\Sigma}^\dagger \boldsymbol{Q}^*)(\sqrt{\boldsymbol{W}}\Psi_1 \Psi_0 \sqrt{\boldsymbol{W}})(\boldsymbol{Q}\Sigma^\dagger) \in \mathbb{C}^{M\times M}. \tag{4.1}$$

Then $(\lambda, \mathbf{v})$ with $\lambda \neq 0$ is an eigenpair of $\hat{\boldsymbol{K}}$ if and only if $(\lambda, \boldsymbol{Z}\mathbf{v})$ is an eigenpair of $\boldsymbol{K}$, where $\boldsymbol{K}$ is defined in (3.8).

Proof. Let us recall that the matrix $\boldsymbol{K}$ is defined in (3.8) as $\boldsymbol{K} = (\Psi_0^* \boldsymbol{W} \Psi_0)^\dagger (\Psi_0^* \boldsymbol{W} \Psi_1)$. For a generic matrix $\boldsymbol{A}$, the following two properties of the Moore-Penrose inverse hold: $(\boldsymbol{A}^*\boldsymbol{A})^\dagger \boldsymbol{A}^* = \boldsymbol{A}^\dagger$ and $\operatorname{range}(\boldsymbol{A}^\dagger) = \operatorname{range}(\boldsymbol{A}^*)$, where $\operatorname{range}(\boldsymbol{A})$ denotes the Span of the columns (or range) of $\boldsymbol{A}$. Applying these two properties to the matrix $\boldsymbol{A} = \sqrt{\boldsymbol{W}}\Psi_0$ it follows:

$$\boldsymbol{K} = (\Psi_0^* \boldsymbol{W} \Psi_0)^\dagger (\Psi_0^* \boldsymbol{W} \Psi_1) = (\sqrt{\boldsymbol{W}}\Psi_0)^\dagger \sqrt{\boldsymbol{W}}\Psi_1$$
$$\operatorname{range}(\boldsymbol{K}) \subseteq \operatorname{range}((\sqrt{\boldsymbol{W}}\Psi_0)^\dagger) = \operatorname{range}(\sqrt{\boldsymbol{W}}\Psi_0)^*)$$

Hence any eigenvector $\mathbf{v}$ such that $\boldsymbol{K}\mathbf{v} = \lambda\mathbf{v}$ with $\lambda \neq 0$ can be written as $\mathbf{v} = (\sqrt{\boldsymbol{W}}\Psi_0)^* \tilde{\mathbf{v}} = \boldsymbol{Z}\boldsymbol{\Sigma}\boldsymbol{Q}^* \tilde{\mathbf{v}} = \boldsymbol{Z}\hat{\mathbf{v}}$ for some $\hat{\mathbf{v}}$. Thus:

$$\lambda \boldsymbol{Z}\hat{\mathbf{v}} = \boldsymbol{K}\boldsymbol{Z}\hat{\mathbf{v}} = (\sqrt{\boldsymbol{W}}\Psi_0)^\dagger \sqrt{\boldsymbol{W}}\Psi_1 \boldsymbol{Z}\hat{\mathbf{v}} = \boldsymbol{Z}\left[(\boldsymbol{\Sigma}^\dagger \boldsymbol{Q}^*)(\sqrt{\boldsymbol{W}}\Psi_1 \Psi_0 \sqrt{\boldsymbol{W}})(\boldsymbol{Q}\Sigma^\dagger)\right]\hat{\mathbf{v}}$$

and multiplying by $\boldsymbol{Z}^*$ on the right we obtain that if $(\lambda,\ \mathbf{v})$ is an eigenpair of $\boldsymbol{K}$ with $\lambda \neq 0$, then $(\lambda,\ \hat{\mathbf{v}} = \boldsymbol{Z}^*\mathbf{v})$ is an eigenpair of $\hat{\boldsymbol{K}}$.

Vice versa, if $(\lambda,\ \mathbf{v})$ is an eigenpair of $\hat{\boldsymbol{K}}$, then:

$$\boldsymbol{K}\boldsymbol{Z}\mathbf{v} = \boldsymbol{Z}\hat{\boldsymbol{K}}\mathbf{v} = \lambda\boldsymbol{Z}\mathbf{v}$$

i.e. $(\lambda,\ \boldsymbol{Z}\mathbf{v})$ is an eigenpair of $\boldsymbol{K}$. □

Now observe that to build the matrix $\hat{\boldsymbol{K}}$ we just need to compute inner products of evaluation of our dictionary. The matrices $\boldsymbol{Q}$ and $\boldsymbol{\Sigma}$ can be obtained from the eigenvalue decomposition of $\sqrt{\boldsymbol{W}}\Psi_0\Psi_0^*\sqrt{\boldsymbol{W}} = \boldsymbol{Q}\boldsymbol{\Sigma}^2\boldsymbol{Q}^*$, and both $(\sqrt{\boldsymbol{W}}\Psi_0\Psi_0^*\sqrt{\boldsymbol{W}})$ and $(\sqrt{\boldsymbol{W}}\Psi_1\Psi_0^*\sqrt{\boldsymbol{W}})$ can be computed using inner products only. Indeed, recalling that $\Psi(\mathbf{x})$ is a row vector:

$$\begin{aligned}(\sqrt{\boldsymbol{W}}\Psi_0\Psi_0^*\sqrt{\boldsymbol{W}})_{ij} &= \sqrt{w_i}\Psi(\mathbf{x}_0^{(i)})\Psi(\mathbf{x}_0^{(j)})^*\sqrt{w_j}\\ (\sqrt{\boldsymbol{W}}\Psi_1\Psi_0^*\sqrt{\boldsymbol{W}})_{ij} &= \sqrt{w_i}\Psi(\mathbf{x}_1^{(i)})\Psi(\mathbf{x}_0^{(j)})^*\sqrt{w_j}.\end{aligned} \tag{4.2}$$

The kernel trick [4] is a common technique used to compute these inner products implicitly. Instead of firstly evaluating the dictionary $\Psi(\mathbf{x})$ and afterwards computing the inner product, that would require $\mathcal{O}(K)$ operations, we compute $\kappa(\mathbf{x},\mathbf{y}) = \Psi(\mathbf{x})\Psi(\mathbf{y})^*$ for a suitably defined kernel κ. It is crucial that the kernel κ does not compute the inner product directly, but it does it implicitly in $\mathcal{O}(d)$ operations, where d is the data dimension. Therefore, the matrices $(\sqrt{\boldsymbol{W}}\Psi_0\Psi_0^*\sqrt{\boldsymbol{W}})$ and $(\sqrt{\boldsymbol{W}}\Psi_1\Psi_0^*\sqrt{\boldsymbol{W}})$ can be calculated in $\mathcal{O}(dM^2)$ operations, without suffering from the large size of the dictionary. The total complexity of computing $\hat{\boldsymbol{K}}$ and solving the associated eigenvalue problem is $\mathcal{O}(dM^2 + M^3)$, significantly lower than the $\mathcal{O}(MK^2 + K^3)$ cost of applying EDMD directly.

4.1.1 Computing the Koopman eigenvalues, eigenfunctions and modes

Once we have computed the matrix $\hat{\boldsymbol{K}}$ and its eigendecomposition, we still need to understand how to retrieve from it an approximation of the Koopman eigenvalues, eigenfunctions and modes. The eigenvalues of $\hat{\boldsymbol{K}}$ can be directly used as approximation of the Koopman eigenvalues. Let $\hat{\boldsymbol{\Xi}} = \left[\hat{\boldsymbol{\xi}}_1, \ldots, \hat{\boldsymbol{\xi}}_M\right]$ be the matrix of the eigenvectors of $\hat{\boldsymbol{K}}$, the vector of approximate Koopman eigenfunctions $\Phi(\mathbf{x}) = [\phi_1(\mathbf{x}), \ldots, \phi_M(\mathbf{x})]$ can be written as $\Phi = \Psi\boldsymbol{Z}\hat{\boldsymbol{\Xi}} = \Psi(\Psi_0^*\sqrt{\boldsymbol{W}})(\boldsymbol{Q}\Sigma^\dagger)\hat{\boldsymbol{\Xi}}$. All we have done so far would not be useful if we suffered from the size of the dictionary when approximating the Koopman eigenfunctions. Fortunately, it is possible to express them using kernels only. Indeed:

$$\begin{aligned}\phi_k(\mathbf{x}) &= \Psi(\mathbf{x})(\Psi_0^*\sqrt{\boldsymbol{W}})(\boldsymbol{Q}\Sigma^\dagger)\hat{\boldsymbol{\xi}}_k =\\ &= \left[\sqrt{w_1}\kappa(\mathbf{x},\mathbf{x}_0^{(1)}), \ldots, \sqrt{w_M}\kappa(\mathbf{x},\mathbf{x}_0^{(M)})\right](\boldsymbol{Q}\Sigma^\dagger)\hat{\boldsymbol{\xi}}_k\end{aligned} \tag{4.3}$$

so that the size K of the dictionary does not enter any of the computations.

To compute the Koopman modes of the full state observable, we consider equation (1.10) and we evaluate it at each of the data points using the approximate eigenfunctions we just obtained. When evaluated at $\mathbf{x}_0^{(i)}$ the equation reads:

$$\mathbf{x}_0^{(i)} = \sum_{j=1}^{M}\phi_j(\mathbf{x}_0^{(i)})\hat{\mathbf{v}}_j + \mathbf{r}_i \tag{4.4}$$

where $\{\hat{\mathbf{v}}_j\}_{j=1}^M$ are the Koopman modes that we want to retrieve, and $\mathbf{r}_i$ is a residual that must be minimized. If we introduce the matrices

$$
\begin{aligned}
\boldsymbol{X} &= \begin{bmatrix} (\mathbf{x}_0^{(1)})^T \\ \vdots \\ (\mathbf{x}_0^{(M)})^T \end{bmatrix}, \\
\Phi_0 &= \begin{bmatrix} \Phi(\mathbf{x}_0^{(1)}) \\ \vdots \\ \Phi(\mathbf{x}_0^{(M)}) \end{bmatrix} = \Psi_0 \boldsymbol{Z}\hat{\boldsymbol{\Xi}} = \Psi_0(\Psi_0^* \sqrt{\boldsymbol{W}})(\boldsymbol{Q}\Sigma^\dagger)\hat{\boldsymbol{\Xi}} = \\
&= \sqrt{\boldsymbol{W}^{-1}}\boldsymbol{Q}\boldsymbol{\Sigma}^2\boldsymbol{Q}^*\boldsymbol{Q}\Sigma^\dagger\hat{\boldsymbol{\Xi}} = \sqrt{\boldsymbol{W}^{-1}}\boldsymbol{Q}\Sigma\hat{\boldsymbol{\Xi}}
\end{aligned}
\tag{4.5}
$$

i.e. the data-points matrix and the matrix of the eigenfunctions values at those points, (4.4) can be rewritten as

$$\boldsymbol{X} = \Psi_0 \hat{\boldsymbol{V}} + \boldsymbol{R}. \tag{4.6}$$

The solution to the least squares problem is

$$\hat{\boldsymbol{V}} = \begin{bmatrix} \hat{\mathbf{v}}_1^T \\ \vdots \\ \hat{\mathbf{v}}_M^T \end{bmatrix} = \Psi_0^\dagger \boldsymbol{X} = \hat{\boldsymbol{\Xi}}^{-1}\Sigma^\dagger \boldsymbol{Q}^* \sqrt{\boldsymbol{W}}\boldsymbol{X}. \tag{4.7}$$

Hence the approximation of the j-th Koopman mode is

$$\hat{\mathbf{v}}_j = (\hat{\mathbf{u}}_j^* \Sigma^\dagger \boldsymbol{Q}^* \sqrt{\boldsymbol{W}}\boldsymbol{X})^T \tag{4.8}$$

where $\hat{\mathbf{u}}_j$ is a left eigenvector of $\hat{\boldsymbol{K}}$ appropriately scaled so that $\hat{\mathbf{u}}_j^* \hat{\boldsymbol{\xi}}_i = \delta_{ij}$.

4.1.2 The choice of the kernel

Even if in the previous discussion we assumed that, given a dictionary $\Psi(\mathbf{x})$, there exists a kernel κ that computes $\kappa(\mathbf{x}, \mathbf{y}) = \Psi(\mathbf{x})\Psi(\mathbf{y})^*$ implicitly, in effects usually it is the choice of κ that defines the dictionary. One of the most simple kernels is the polynomial kernel $\kappa(\mathbf{x}, \mathbf{y}) = (1 + \mathbf{x}^*\mathbf{y})^n$, which is equivalent to a dictionary that can represent all polynomials up to degree n. Increasing n the implicitly defined dictionary becomes larger, at the cost of a worst conditioning of the resulting matrices. For this reason, taking inspiration from Machine Learning techniques, other kernels could be considered, such as the widely used radial basis function kernel (RBF kernel), $\kappa(\mathbf{x}, \mathbf{y}) = \exp\left(-\|\mathbf{x} - \mathbf{y}\|^2/\sigma^2\right)$.

Algorithm 6 : Kernelized EDMD (K-EDMD)

Input: Snapshot pairs of the system state, $\{(\mathbf{x}_0^{(m)}, \mathbf{x}_1^{(m)})\}_{m=1}^M$, quadrature weights $\{w_m\}_{m=1}^M$, kernel $\kappa = \kappa(\cdot\,,\cdot)$.

```
1:  Compute the matrices $\hat{\boldsymbol{G}} = \sqrt{\boldsymbol{W}}\Psi_0\Psi_0^*\sqrt{\boldsymbol{W}}$ and $\hat{\boldsymbol{A}} = \sqrt{\boldsymbol{W}}\Psi_1\Psi_0^*\sqrt{\boldsymbol{W}}$, exploiting that $\hat{\boldsymbol{G}}_{ij} = \sqrt{w_i}\kappa(\mathbf{x}_0^{(i)}, \mathbf{x}_0^{(j)})\sqrt{w_j}$ and $\hat{\boldsymbol{A}}_{ij} = \sqrt{w_i}\kappa(\mathbf{x}_1^{(i)}, \mathbf{x}_0^{(j)})\sqrt{w_j}$.
2:  Compute the eigenvalue decomposition of $\hat{\boldsymbol{G}} = \boldsymbol{Q}\boldsymbol{\Sigma}^2\boldsymbol{Q}^*$.
3:  Compute $\hat{\boldsymbol{K}} = (\boldsymbol{\Sigma}^\dagger\boldsymbol{Q}^*)\hat{\boldsymbol{A}}(\boldsymbol{Q}\Sigma^\dagger)$.
4:  Solve the eigenvalue problem of $\hat{\boldsymbol{K}}\hat{\boldsymbol{\xi}} = \lambda\hat{\boldsymbol{\xi}}$.
5:  Obtain the eigenfunctions as $\phi_k(\mathbf{x}) = \left[\sqrt{w_1}\kappa(\mathbf{x}, \mathbf{x}_0^{(1)}), \dots, \sqrt{w_M}\kappa(\mathbf{x}, \mathbf{x}_0^{(M)})\right](\boldsymbol{Q}\Sigma^\dagger)\hat{\boldsymbol{\xi}}_k$.
6:  if want to compute eigenmodes then
7:      Compute the left eigenvectors $\mathbf{u}_j$ of $\hat{\boldsymbol{K}}$ appropriately scaled so that $\hat{\mathbf{u}}_j^*\hat{\boldsymbol{\xi}}_i = \delta_{ij}$.
8:      Compute $\hat{\boldsymbol{B}} = \Sigma^\dagger\boldsymbol{Q}^*\sqrt{\boldsymbol{W}}\boldsymbol{X}$.
9:      Obtain the modes as $\hat{\mathbf{v}}_j = (\hat{\mathbf{u}}_j^*\hat{\boldsymbol{B}})^T$.
10: end if
```

Output: Approximation of the eigenvalues (λ_j), eigenfunctions (ϕ_j) and, possibly, eigemodes $(\mathbf{v}_j)$.

4.2 Kernelized ResDMD (K-ResDMD)

At this point, a natural question is how one could extend ResDMD with a kernelized approach. Unfortunately, applying Algorithm 4 with K-EDMD instead of EDMD makes the approximate residual always vanish (see [9], Proposition 6.2). Basically, since the size of our function space is larger than the number of snapshots, we can "interpolate", and we are actually overfitting our dataset. Similarly to what is done when training and testing a Machine Learning model, to have a reliable estimate of the residual for approximate eigenpairs, we need to compute it using quadrature points that have not been used at the "interpolation stage" and that, at least ideally, are independent on the previously employed ones. This idea leads to the following multi-step approach:

1. Split the snapshot data into two (ideally independent) sets $\{(\mathbf{x}_0^{(j)}, \mathbf{x}_1^{(j)})\}_{j=1}^{M'}$ and $\{(\mathbf{y}_0^{(j)}, \mathbf{y}_1^{(j)})\}_{j=1}^{M''}$, with $M'' \geq M'$.
2. Apply K-EDMD to the first data set $\{(\mathbf{x}_0^{(j)}, \mathbf{x}_1^{(j)})\}_{j=1}^{M'}$, and compute a dictionary that Spans the function space generate by the approximations of dominant $K' < M'$ eigenfunctions (see Algorithm 7 for the details). Let us denote this dictionary as $\Psi(x) = [\psi_1(\mathbf{x}), \ldots, \psi_{K'}(\mathbf{x})]$.
3. Apply Algorithm 4 or Algorithm 5 with snapshot data $\{(\mathbf{y}_0^{(j)}, \mathbf{y}_1^{(j)})\}_{j=1}^{M''}$ and dictionary $\Psi(x)$.

As summarized in Algorithm 7 in the first stage, we reduce the size of the dictionary implicitly defined by the kernel to K', hoping to obtain a good dictionary that already captures the essential dynamics. Then this dictionary is used as a basis for the ResDMD algorithm, which computes reliable approximations of the eigenpairs of the Koopman operator. This also provides an a posteriori check of the quality of the dictionary obtained in the first stage.

Algorithm 7 : Kernelized ResDMD (K-ResDMD)

Input: Snapshot pairs of the system state, $\{(\mathbf{x}_0^{(m)}, \mathbf{x}_1^{(m)})\}_{m=1}^{M}$, quadrature weights $\{w_m\}_{m=1}^{M}$, kernel $\kappa = \kappa(\cdot\,,\cdot)$, hyperparameters M', M'', K'.

```
1: Split the snapshot data into $\{(\mathbf{x}_0^{(j)}, \mathbf{x}_1^{(j)})\}_{j=1}^{M'}$ and $\{(\mathbf{y}_0^{(j)}, \mathbf{y}_1^{(j)})\}_{j=1}^{M''}$, with $M = M' + M''$ and $M'' \geq M' > K'$.
2: Apply K-EDMD (Algorithm 6) with snapshot pairs $\{(\mathbf{x}_0^{(j)}, \mathbf{x}_1^{(j)})\}_{j=1}^{M'}$: compute the dominant $K' < M'$ eigenvectors of $\hat{\boldsymbol{K}}$ and stack them into $\hat{\Xi} = \left[\hat{\boldsymbol{\xi}}_1, \ldots, \hat{\boldsymbol{\xi}}_{K'}\right]$.
3: Orthogonalize the colums of $\hat{\Xi}$ to $\boldsymbol{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_{K'}]$.
4: Apply Algorithm 4 or Algorithm 5 with snapshot data $\{(\mathbf{y}_0^{(j)}, \mathbf{y}_1^{(j)})\}_{j=1}^{M''}$ and dictionary $\{\psi_j\}_{j=1}^{K'}$ defined by
       $$\psi_j(\mathbf{x}) = \left[\sqrt{w_1}\kappa(\mathbf{x}, \mathbf{x}_0^{(1)}), \ldots, \sqrt{w_M}\kappa(\mathbf{x}, \mathbf{x}_0^{(M')})\right] (\boldsymbol{Q}\Sigma^{\dagger})\, \mathbf{p}_j$$
```

Output: According to the algorithm chosen in the final step.

4.2.1 A two step procedure for data-driven dictionaries

The two-steps procedure that has been presented in the previous subsection is applied in [9] to deal with high dimensional data and remove spectral pollution. However, it might also be applied to low dimensional data in order to extract a dictionary of observables from data, without the need of hand-crafting it. Indeed one of the limitations of EDMD and ResDMD is the need to define a suitable dictionary for each problem, which is in general not a straightforward task.

We decided to take a different approach to the one in [9] and tried to employ KEDMD with low dimensional data, in order to apply EDMD and ResDMD to different problems without the need of manually designing a different dictionary according to the properties of each problem. In Section 4.3, we show the effects on the Gauss iterated map problem introduced in Section 3.3.1.

4.3 A numerical example: a data-driven dictionary for the Gauss iterated map

To experimentally show the effects of having a dictionary of observables with a size larger or equal than the the number of quadrature points, we first considered the ResDMD algorithm and the K-EDMD algorithm applied with the same dictionary of observables. To this aim, we considered as a dictionary $\Psi(x)$ for the ResDMD algorithm the first $K = 40$ Legendre polynomials transferred to the interval $\Omega = [-1, 0]$ and as a kernel for the K-EDMD algorithm the kernel defined by Ψ, i.e. $\kappa(x, y) = \Psi(x)\Psi(y)^{\star}$. As snapshot pairs we considered $\{(\mathbf{x}_0^{(m)}, \mathbf{x}_1^{(m)})\}_{m=1}^{M}$ with $M = K = 40$, where $\{(\mathbf{x}_0^{(m)}\}_{m=1}^{M}$ are the Gauss-Legendre quadrature nodes transplanted to Ω and $\mathbf{x}_1^{(m)} = F(\mathbf{x}_0^{(m)})$. For the ResDMD algorithm, we discarded approximate eigenvalues with an estimated residual lower than $\varepsilon = 0.1$. Figure 4.1 shows the output of the ResDMD and K-EDMD algorithms. As expected from Proposition 4.1 the output of the two algorithms coincide. However, since the number of snapshot pairs M is equal to the size K of the dictionary, the ResDMD algorithm does not discard any of the approximated eigenvalues in output of the EDMD algorithm.

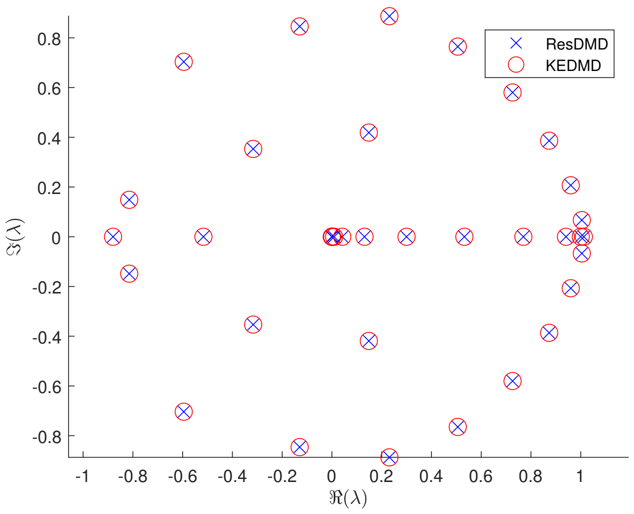


Figure 4.1: Output of the ResDMD ad K-EDMD algorithm for the Gauss iterated map problem, when using a number of snapshots pairs $M = 40$ equal to the size K of the dictionary, and the kernel defined by the dictionary. The firs K Legendre polynomials are used as a dictionary and the Gauss-Legendre quadrature points as initial snapshots.

Finally, to analyse the effectiveness of the procedure for extracting a dictionary from a general-purpose kernel, we applied the two-stages Kernelized ResDMD algorithm with a RBF kernel. Let us observe that the RBF kernel can be seen as the inner product of an infinite polynomial dictionary, hence we do not have any constraint on the number of snapshot datapoints. As snapshot data for the first stage, we considered $M' = 1000$ Montecarlo quadrature nodes and we extracted the $K' = 40$ dominant eigenpairs using the K-EDMD algorithm. In the second stage, we applied the ResDMD algorithm using the dictionary in output of the K-EDMD algorithm and $M'' = 1000$ Gauss-Legendre quadrature points as snapshot data. The tolerance employed in K-ResDMD was $\varepsilon = 0.01$. As suggested in [9], we also employed the following common practices:

- the constant σ of the RBF kernel $\kappa(\mathbf{x}, \mathbf{y}) = \exp\left(-\|\mathbf{x} - \mathbf{y}\|^2/\sigma^2\right)$ was set as the average ℓ^2-norm of the snapshot data after it is shifted to have mean zero.
- the gramian matrices of the first stage $\hat{\boldsymbol{G}} = \sqrt{\boldsymbol{W}}\Psi_0\Psi_0^*\sqrt{\boldsymbol{W}}$ and $\hat{\boldsymbol{A}} = \sqrt{\boldsymbol{W}}\Psi_1\Psi_0^*\sqrt{\boldsymbol{W}}$ are both ill conditioned, with a conditioning number of the order 10^{20} and a numerical rank between 40 and 50 (their size is $M' = 1000$). Therefore, we added regularization in the form $\hat{\boldsymbol{G}} = \hat{\boldsymbol{G}} + \eta\|\hat{\boldsymbol{G}}\|_F \boldsymbol{I}$ where $\eta = 10^{-16}$.

Figure 4.2 shows the results of the above described procedure. The pseudospectral contour lines computed in Section 3.3.1 are plotted as a reference to evaluate the quality of the approximated eigenvalues. It can be observed that the approximated eigenvalues returned by the K-ResDMD via the two-step extraction procedure are very similar to the ones shown in Figure 3.2, obtained using the ResDMD, and they all lie inside the previously computed 0.01-pseudospectrum approximation. However, the eigenvalues shown in Figure 4.2 did not require a hand-crafted dictionary as input of the algorithm, but employed the general-purpose RBF kernel with the scaling factor σ determined using the data.

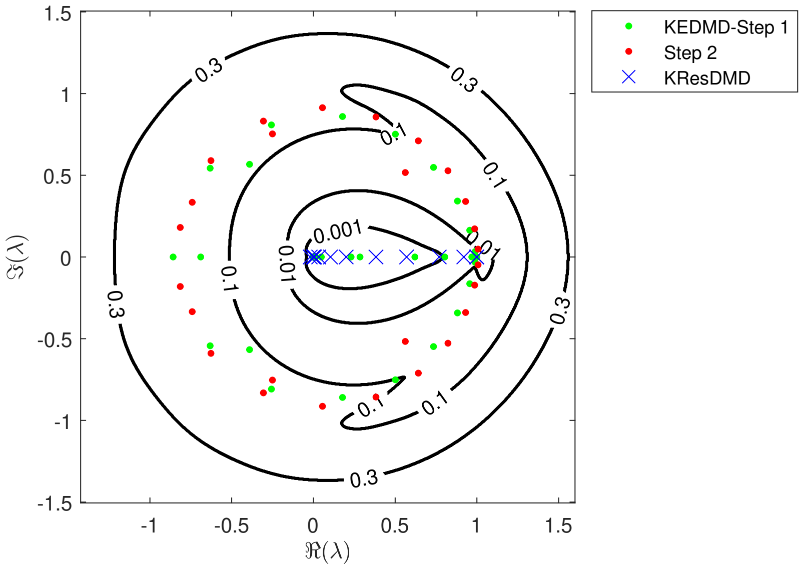


Figure 4.2: Eigenvalues approximation for the Gauss iterated map problem, obtained using the two-step procedure described in Algorithm 7. Green dots show the $K' = 40$ dominant eigenvalues computed in the first step. Eigenvalues returned in the second steps are represented as red dots and blue crosses, where blue crosses show the eigenvalues retained by K-ResDMD with tolerance $\epsilon = 0.01$. The pseudospectral contour lines computed in Section 3.3.1 are also plotted as a reference.

5. Conclusions

Starting from the article [9] by J. Colbrook and A. Townsend, we analysed spectral properties of the Koopman Operator and discussed methods to numerically compute spectral approximations. Dynamic Mode Decomposition was presented, with a particular focus on the case of linear dynamical systems, where DMD is mathematically equivalent to the Arnoldi algorithm, but less stable from a numerical point of view. For nonlinear dynamics, the EDMD algorithm was proposed, and its variant ResDMD was used to both deal with the problem of spectral pollution and approximate the pseudospectrum of the Koopman Operator. Furthermore, a proposition linking the pseudospectral approximation returned by ResDMD and the pseudospectrum of a particular matrix pencil was proven in this work. Two numerical examples from Section 4 of [9] were reproduced in details: the Gauss iterated map and the nonlinear pendulum.

Finally, as an original contribution of this work, the kernelized approach proposed in [9] for high dimensional data was used to create a data-driven dictionary for the Gauss iterated map problem, by providing a viable way for avoiding the step of hand-crafting a problem-dependent dictionary for the EDMD and ResDMD algorithms. This novel approach provided satisfactory results for the Gauss iterated map problem, and might be worth analysing it in more details in future works.

Bibliography

[1] Hassan Arbabi. *Introduction to Koopman operator theory of dynamical systems.* en. 2018.

[2] Hassan Arbabi and Igor Mezić. "Study of dynamics in post-transient flows using Koopman mode decomposition". In: *Physical Review Fluids* 2.12 (Dec. 2017), p. 124402. ISSN: 2469-990X. DOI: `10.1103/PhysRevFluids.2.124402`.

[3] Shervin Bagheri. "Koopman-mode decomposition of the cylinder wake". en. In: *Journal of Fluid Mechanics* 726 (July 2013), pp. 596–623. ISSN: 0022-1120, 1469-7645. DOI: `10.1017/jfm.2013.249`.

[4] Christopher M. Bishop. *Pattern recognition and machine learning.* en. Information science and statistics. New York: Springer, 2006. ISBN: 978-0-387-31073-2.

[5] S. Brunton. *Notes on Koopman Operator Theory.* en. 2019.

[6] Steven L. Brunton et al. "Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control". In: *PLOS ONE* 11.2 (Feb. 2016), e0150171. ISSN: 1932-6203. DOI: `10.1371/journal.pone.0150171`.

[7] Steven L. Brunton et al. "Modern Koopman Theory for Dynamical Systems". In: *SIAM Review* 64.2 (May 2022), pp. 229–340. ISSN: 0036-1445. DOI: `10.1137/21M1401243`.

[8] Marko Budišić, Ryan M. Mohr, and Igor Mezić. "Applied Koopmanism". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 22.4 (Dec. 2012), p. 047510. ISSN: 1054-1500, 1089-7682. DOI: `10.1063/1.4772195`.

[9] Matthew J. Colbrook and Alex Townsend. *Rigorous data-driven computation of spectral properties of Koopman operators for dynamical systems.* Tech. rep. arXiv:2111.14889. arXiv, Nov. 2021. DOI: `10.48550/arXiv.2111.14889`.

[10] Zlatko Drmač, Igor Mezić, and Ryan Mohr. "Data Driven Modal Decompositions: Analysis and Enhancements". In: *SIAM Journal on Scientific Computing* 40.4 (Jan. 2018), A2253–A2285. ISSN: 1064-8275. DOI: `10.1137/17M1144155`.

[11] Dinh Dũng, Vladimir N. Temlyakov, and Tino Ullrich. *Hyperbolic Cross Approximation.* Tech. rep. arXiv:1601.03978. arXiv, Apr. 2017.

[12] Gene H. Golub and Charles F. Van Loan. *Matrix Computations.* en. JHU Press, 2013. ISBN: 978-1-4214-0794-4.

[13] Henri Poincaré. *Les méthodes nouvelles de la mécanique céleste.* French. Gauthier-Villars et fils, 1899.

[14] B. O. Koopman. "Hamiltonian Systems and Transformation in Hilbert Space". In: *Proceedings of the National Academy of Sciences of the United States of America* 17.5 (May 1931), pp. 315–318. ISSN: 0027-8424.

[15] B. O. Koopman and J. v. Neumann. "Dynamical Systems of Continuous Spectra". In: *Proceedings of the National Academy of Sciences of the United States of America* 18.3 (Mar. 1932), pp. 255–263. ISSN: 0027-8424.

[16] Daniel Kressner and Bart Vandereycken. "Subspace Methods for Computing the Pseudospectral Abscissa and the Stability Radius". In: *SIAM Journal on Matrix Analysis and Applications* 35.1 (Jan. 2014), pp. 292–313. ISSN: 0895-4798. DOI: `10.1137/120869432`.

[17] Igor Mezic. *Spectrum of the Koopman Operator, Spectral Expansions in Functional Spaces, and State Space Geometry.* Tech. rep. arXiv:1702.07597. arXiv, Oct. 2019. DOI: `10.48550/arXiv.1702.07597`.

[18] Igor Mezic. *Koopman Operator, Geometry, and Learning.* Tech. rep. arXiv:2010.05377. arXiv, Oct. 2020. DOI: `10.48550/arXiv.2010.05377`.

[19] Igor Mezić. "Spectral Properties of Dynamical Systems, Model Reduction and Decompositions". en. In: *Nonlinear Dynamics* 41.1 (Aug. 2005), pp. 309–325. ISSN: 1573-269X. DOI: `10.1007/s11071-005-2824-x`.

[20] Alfio Quarteroni, Riccardo Sacco, and Fausto Saleri. "Numerical Integration". en. In: *Numerical Mathematics.* Ed. by Alfio Quarteroni, Riccardo Sacco, and Fausto Saleri. New York, NY: Springer, 2007, pp. 371–414. ISBN: 978-0-387-22750-4. DOI: `10.1007/978-0-387-22750-4_9`.

[21] Clarence W. Rowley et al. "Spectral analysis of nonlinear flows". en. In: *Journal of Fluid Mechanics* 641 (Dec. 2009), pp. 115–127. ISSN: 1469-7645, 0022-1120. DOI: `10.1017/S0022112009992059`.

[22] Peter J. Schmid. "Dynamic mode decomposition of numerical and experimental data". en. In: *Journal of Fluid Mechanics* 656 (Aug. 2010), pp. 5–28. ISSN: 1469-7645, 0022-1120. DOI: `10.1017/S0022112010001217`.

[23] Gregory Snyder and Zhuoyuan Song. *Koopman Operator Theory for Nonlinear Dynamic Modeling using Dynamic Mode Decomposition.* Tech. rep. arXiv:2110.08442. arXiv, Oct. 2021. DOI: `10.48550/arXiv.2110.08442`.

[24] Lloyd N. Trefethen and Mark Embree. *Spectra and Pseudospectra: The Behavior of Nonnormal Matrices and Operators.* en. Princeton University Press, Aug. 2005. ISBN: 978-0-691-11946-5.

[25] Jonathan H. Tu et al. "On Dynamic Mode Decomposition: Theory and Applications". In: *Journal of Computational Dynamics* 1.2 (2014), pp. 391–421. ISSN: 2158-2505. DOI: `10.3934/jcd.2014.1.391`.

[26] Matthew O. Williams, Ioannis G. Kevrekidis, and Clarence W. Rowley. "A Data-Driven Approximation of the Koopman Operator: Extending Dynamic Mode Decomposition". In: *Journal of Nonlinear Science* 25.6 (Dec. 2015), pp. 1307–1346. ISSN: 0938-8974, 1432-1467. DOI: `10.1007/s00332-015-9258-5`.

[27] Matthew O. Williams, Clarence W. Rowley, and Ioannis G. Kevrekidis. *A Kernel-Based Approach to Data-Driven Koopman Spectral Analysis.* Tech. rep. arXiv:1411.2260. arXiv, July 2015. DOI: `10.48550/arXiv.1411.2260`.

[28] Thomas G. Wright and Lloyd N. Trefethen. "Pseudospectra of rectangular matrices". In: *IMA Journal of Numerical Analysis* 22.4 (Oct. 2002), pp. 501–519. ISSN: 0272-4979. DOI: `10.1093/imanum/22.4.501`.