A quantitative analysis of Koopman operator methods for system identification and predictions

Christophe Zhang Enrique Zuazua

July 1, 2021

Abstract

We give convergence and cost estimates for a data-driven system identification method: given an unknown dynamical system, the aim is to recover its vector field and its flow from trajectory data. It is based on the so-called Koopman operator, which uses the well-known link between differential equations and linear transport equations. Data-driven methods recover specific finite-dimensional approximations of the Koopman operator, which can be understood as a transport operator. We focus on such approximations given by classical finite-elements spaces, which allow us to give estimates on the approximation of the Koopman operator as well as the solutions of the associated linear transport equation. These approximations are thus relevant objects to solve the system identification problem.

We then analyze the convergence of a variant of the generator Extended Dynamic Mode Decomposition (gEDMD) algorithm, one of the main algorithms developed to compute approximations of the Koopman operator from data. We find however that, when combining this algorithm with classical finite elements spaces, the results are not satisfactory numerically, as the convergence of the data-driven approximation is too slow for the method to benefit from the accuracy of finite elements spaces. In particular, for problems in dimension 1 it is less efficient than direct interpolation methods to recover the vector field. We provide some numerical examples to illustrate this last point.

Contents

1 Introduction

1.1 Unknown dynamics and system identification

Many problems in applied mathematics rely on the existence of a mathematical model for the system under consideration. Given the potential complexity of systems that arise in all scientific fields, from econometry to molecular dynamics, finding relevant and numerically tractable models for applications such as forecasting and control is a central challenge.

The traditional approach to mathematical modelling consists in deriving governing equations from fundamental laws of physics such as Newton's second law or the principle of least action, after a series of assumptions and ideal simplifications. Models derived in this way have been thoroughly studied, giving rise to rich theories and a profound understanding of some of the phenomena they describe. But these remain ideal models, which sometimes fall short of real-life observations and applications. Indeed, as real-life systems become more complex, the relevance of such models for applications comes into question, as there is a risk that usual, convenient simplifications could lead to essential features of the dynamics being overlooked.

On the other hand, the development of data-driven techniques has brought a new perspective to this challenge (see for example [39, 5, 46]): how can one complement traditional modelling approaches with data-driven considerations? Can relevant features of the dynamics of the system be recovered from data? More generally, how can one learn a model from data, and propose a system of governing equations that allow for reliable predictions, and effective control?

Mathematically speaking, we can cast this as follows: suppose that, for a given system, its behaviour can be relevantly modelled with an autonomous, nonlinear differential equation

$$\dot{x} = f(x), \quad x \in \mathbb{R}^d. \tag{1}$$

How can one learn an approximation of the vector field f from observational data, that is, samples of trajectories? Given initial conditions, can one produce predictions on the evolution of the system in a data driven way?

This is a form of inverse problem, which differs from more classical inverse problems (see for example [13, 23, 63]) in that the latter rely on the a priori choice of a class of models (heat equation, polynomial differential equations, balance laws...) so that one has to recover parameters of the given model (diffusivity, coefficients...).

1.2 Regression, interpolation and operator theoretic approach

The task of recovering a function $f : \mathbb{R}^d \to \mathbb{R}^d$ from a data sample of values $\{x_i, y_i = f(x_i)\}_{i=1}^N$ in order to make predictions lies at the heart of statistical regression analysis (linear regression being perhaps the simplest example), with examples dating back as early as Legendre's works on planetary orbits using a least squares method ([38]). More recently, a variety of supervised learning and deep learning techniques have been developed to tackle this problem, such as kernel-based methods, neural networks, or deep neural networks. We refer to [8, 7, 19] for a comprehensive overview and further references.

The core idea of these regression techniques is to find a function $\hat{f}$ belonging to a certain class of functions $\mathcal{S}$ (linear functions, polynomials, wavelets...), that minimizes the loss

$$\sum_{i=1}^{N} \|\hat{f}(x_i) - y_i\|^2, \tag{2}$$

in such a way that $\hat{f}$ will also be a good approximation of f on new data. The first approach to solving (2) would be to find, if it exists, a function in $\mathcal{S}$ that interpolates the data, that is,

$$\hat{f} \in \mathcal{S}, \quad \sum_{i=1}^{N} \|\hat{f}(x_i) - y_i\|^2 = 0 \quad \text{i.e.} \quad \hat{f}(x_i) = y_i = f(x_i), \quad i = 1, \cdots, N. \tag{3}$$

A typical example of functions used for interpolation are polynomials, as there exists a unique polynomial of degree at most $N-1$ that satisfies (3). We refer to [53, Chapter 3] for more details on these interpolation results. However, although $\hat{f}$ fits the given data perfectly, it has no reason a priori to be a good approximation of f on new data. Nonetheless under some assumptions on the x_i and on f, there are some well-known estimates for interpolation by polynomials and continuous piecewise polynomial functions (we refer again to [53, Chapter 3]).

In general, regression techniques are not designed to interpolate the data exactly, as in the above, but rather to reach a compromise between a small loss function in (2) and a good overall approximation of the target function f. The goal is to ensure a good generalization of $\hat{f}$ to new data, without resorting to the aforementioned general interpolation estimates. There are then two important aspects to consider: first, the choice of the family of functions $\mathcal{S}$ and its representation, which can benefit from prior knowledge on f. Second, the method used to minimize the loss, which can involve for example regularization to avoid overfitting $\hat{f}$ to the data to the detriment of generalization.

In particular, if one has the prior knowledge that f is linear, then the above problem amounts to a multivariate linear regression, and the least squares problem (2) has a well-known solution. This, among other things, has motivated an alternative approach system identification, the so-called operator theoretic approach. Instead of treating system identification directly as a general nonlinear problem, one can recast this problem as a linear one, using the well-known connection between differential equations and linear transport equations. This is appealing both on a theoretical level, as it gives an elegant framework to understand some intricate properties of nonlinear dynamical systems, and on a practical level, due to the many tools available both for forecasting and control of linear systems. As we will see, with this approach system identification then becomes a linear regression, but in a potentially much larger space.

2 Koopman operators

2.1 Definitions

Consider a vector field $f : \mathbb{R}^d \to \mathbb{R}^d$. One can view it as the vector field of a nonlinear ordinary differential equation:

$$\dot{x} = f(x), \quad x \in \mathbb{R}^d, \tag{4}$$

but also as the velocity field of the linear transport equation:

$$\frac{\partial \varphi}{\partial t}(t, x) = f(x) \cdot \nabla \varphi. \tag{5}$$

The equivalence between these two points of view has been long known. Existence and uniqueness results for nonlinear differential equations with Sobolev vector fields were obtained by studying the corresponding transport equation ([18, 22]). Conversely, the method of characteristics solves (5) using (4): if f is regular enough, it generates a flow Φ_f^t which solves (4), and the solutions of (5) are constant along the characteristics $(t, \Phi_f^t(x)),\ t \geq 0,\ x \in \mathbb{R}^d$. The solution of (5) with initial condition $\varphi_0 \in L^2(\mathbb{R}^d)$ is then given by

$$\varphi(t,x) := \varphi_0\left(\Phi_f^t(x)\right). \tag{6}$$

This corresponds to the following semigroup:

$$\varphi_0 \in L^2(\mathbb{R}^d) \mapsto \varphi_0 \circ \Phi_f^t, \tag{7}$$

which is a simple composition operator with the following infinitesimal generator

$$\mathcal{K} := f(x) \cdot \nabla, \tag{8}$$

which is simply the transport operator with velocity field f, on the domain

$$\mathcal{D}(\mathcal{K}) = \{\varphi \in L^2(\mathbb{R}^d), \quad f \cdot \nabla\varphi \in L^2(\mathbb{R}^d)\}. \tag{9}$$

In what follows, we will denote by $e^{t\mathcal{K}}$ the semigroup (7), generated by the Koopman operator $\mathcal{K}$.

The semigroup and its generator were studied and used by B.O.Koopman in [28] to study properties of fluid dynamics. They have since then been named after him: what is known as the *Koopman operator of f* is the operator $\mathcal{K}$ given by (8), and the *Koopman semigroup of f* is the semigroup $e^{t\mathcal{K}}$ given by (7). Practically speaking, for a function $\varphi \in L^2(\mathbb{R}^d)$ and a trajectory $x(t)$ of the differential equation (4), the Koopman semigroup describes the evolution of $t \mapsto \varphi(x(t))$.

Remark 2.1. *In applications involving discrete-time dynamical systems*

$$x^+ = T(x), \quad x \in \mathbb{R}^d, \tag{10}$$

the corresponding discrete-time Koopman operator is simply the composition operator

$$U: \quad g \mapsto g \circ T. \tag{11}$$

Equivalently, when dealing with continuous-time systems, the Koopman operator is sometimes defined as the bounded operator $e^{\tau\mathcal{K}}$ with the notations of (7)*, for some time increment $\tau > 0$.*

Historically, B.O.Koopman used the linear composition operator (7) to study measure-preserving systems. It was notably instrumental in J. Von Neumann's proof of the mean ergodic theorem [47], and a first connection of chaotic behavior to the existence of continuous parts in the spectrum of the Koopman semigroup was noted in [29]. More recently, spectral properties of the Koopman semigroup have been shown to be related to a variety of aspects of the underlying dynamical system such as ergodicity, stability, existence of different time scales, mixing and non-mixing properties (see for example [43, 44, 37, 42, 45, 36]). An important feature in these works that the spectral analysis of the Koopman operator is done in a rather geometrical spirit in combination with linearization results and conjugation techniques, focusing on local behaviour near equilibria or attractors.

2.2 Koopman operator and data-driven modelling

In addition to this renewed theoretical exploration, data-driven applications of Koopman operator theory have gained quite some interest in the last decade, in particular for system identification, modelling, predictions and control problems [9, 30, 26]. In these works, among others, the idea is that one can identify the underlying nonlinear dynamical system, (or at least some features of this system), by identifying its Koopman operator, or its Koopman semigroup, or the discrete-time Koopman operator, depending on the nature of the system.

There are two potential benefits to this approach:

1. The linearity of the Koopman operators, which offers the possibility to describe nonlinear dynamics in a linear way.
2. As a consequence, the possibility to make predictions by computing solutions to the linear equations given by these operators, instead of integrating a nonlinear differential equation.

Numerically speaking, the focus of these methods lies on finite-dimensional reductions of the Koopman operator, as it is not possible to recover an infinite-dimensional operator from a finite amount of data.

The question then becomes: which finite-dimensional reductions are accurate and relevant for system identification and predictions? What is their dimension? How does one recover them from data? At what cost?

- **Exact reductions.** In some cases, there exist finite-dimensional subspaces V of functions which are invariant under the action of the Koopman operator. Then, the linear transport equation (5) reduces exactly to a linear differential equation on these subspaces:

  $$\dot{\varphi} = \mathcal{K}\varphi, \quad \varphi \in V.$$

  The main example of such a situation can be found in [9], where a system of 2 polynomial differential equations are lifted to a system of 3 linear differential equations by considering a basis of multinomials (note that this is a particular case where the Carleman linearization [14] of a polynomial system is exact and does not need to be truncated).

  Although the existence of such subspaces holds great interest for applications, and there have been efforts to recover them from data [56], generally speaking, this seems to be a very rare situation. Let us point out also that even though in this example, the dimension of the invariant subspace is small, in general it can be much larger the dimension d of the underlying problem.

  Another way to find invariant subspaces is to find eigenfunctions of the Koopman operator. Then, any subspace spanned by a finite number of eigenfunctions $\varphi_1, \cdots, \varphi_N$ with eigenvalues $\lambda_1, \cdots, \lambda_N$ will be invariant under the action of the Koopman operator, and the linear transport equation (5) reduces exactly to the following diagonal system of linear differential equations:

  $$\dot{\varphi}_i = \lambda_i \varphi_i, \quad i \in [\![1, N]\!].$$

  However, even if there exist eigenfunctions to the Koopman operator, there is no reason a priori for them to span an approximation of the Koopman operator that is relevant for system identification.

- **Spectral properties of the Koopman operator.** On the other hand, the complete spectral analysis of the Koopman operator is a delicate topic, and depends heavily on the choice of a function space. The transport operator

  $$\mathcal{K} = f(x) \cdot \nabla, \mathcal{D}(\mathcal{K}) = H^1(\mathbb{R}^d)$$

  does not have a natural basis of eigenfunctions, sometimes no eigenfunctions at all. This is related to the fact that in general transport operators do not have a compact resolvent, and are not generally self-adjoint unless the velocity field f has zero divergence.

  Let us nevertheless point out an interesting recent development for the discrete time Koopman operator introduced in Remark 2.1. In the case where Ω is the unit circle in $\mathbb{R}^2$, [54] shows that an adequate choice of a function space (namely, Hilbert-Hardy spaces) leads to interesting spectral results. Indeed, by restricting the function space to a class of analytical functions, the adjoint of the Koopman operator (the Perron-Frobenius operator) becomes compact. By duality, this means that in this case, by extending the discrete time Koopman operator to a larger space containing L^2, it becomes compact. When it is self-adjoint, which is equivalent to the existence of an invariant measure, it then has a basis of eigenfunctions. This result however concerns a certain class of discrete-time dynamical systems, and analogous results in the continuous time setting, regarding the Koopman generator (8), have yet to be explored.

In terms of data-driven methods, spectral analysis of the Koopman operator from data has received a lot of attention (see [24, 32, 20, 33]), in particular the data-driven recovery of eigenfunctions when they exist. As is often the case in numerical analysis, only the smallest eigenvalues can be accurately approximated. This is of interest to isolate components of the observed system that evolve slowly in time (or are invariant, if the eigenvalue is 0). However, these give only a very partial picture of how the system actually evolves.

As we have pointed out earlier, in general, transport operators (Koopman operators) do not have nice spectral properties. For system identification and predictions, there seems to be better hope in finding accurate and appropriate finite-dimensional approximations of the Koopman operator independently of such properties.

- **Galerkin projections and Extended Dynamic Mode Decomposition.** Classical Galerkin methods used in numerical analysis, which approximate solutions of partial differential equations using approximation spaces, seem a natural solution to provide reliable approximations of the Koopman operator. They have indeed proven to be an efficient way to obtain reduced-order models of partial differential equations without having to use spectral decompositions.

  To recover a data-driven approximation of such a Galerkin projection, one of the main algorithms is the so-called Extended Dynamic Mode Decomposition (EDMD) (introduced in [61]). Its predecessing algorithm, Dynamic Mode Decomposition (DMD), computes the "best fit" system of linear differential equations for the unknown system (4) in the state space $\mathbb{R}^d$. EDMD does this in the subspace of functions given by the Galerkin projection. As this corresponds to recovering the best linear approximation of a linear infinite-dimensional operator, this has a better chance of yielding convincing results than DMD.

  EDMD was first introduced in the discrete-time framework, to recover approximations of the discrete-time Koopman operator, that is, the classical composition operator. It has been actively explored as a way to perform data-driven system identification and predictions [31, 40, 25]. It can also provide approximations of so-called coarse-grain models (see [27]) of highly complex systems such as molecular dynamics ([26]). It has been used in combination with model predictive control to achieve data-driven control of systems with unknown dynamics [30], and combined with deep learning techniques [2, 51, 34, 49, 64].

  In continuous time, there are two main approaches using EDMD to recover an approximation of the Koopman operator.

  1. The "Koopman lifting method", proposed in [41], where the authors apply EDMD to a continuous time system to obtain an approximation of the Koopman semigroup $U(\tau)$ for a sampling time $\tau > 0$. Then, they compute its matrix logarithm to produce an approximation of the infinitesimal generator $\mathcal{K}$.
  2. Generator Extended Dynamic Mode Decomposition (gEDMD), proposed in [26], where EDMD is adapted to directly produce an approximation of $\mathcal{K}$ by processing data in different way.

In this article, we implement a variant of gEDMD, using classical finite elements spaces to define finite-dimensional approximations of the Koopman operator. Our central aim is to quantify the quality and cost of system identification and predictions by this method. More precisely, we will focus on the case of an unknown vector field f of a certain regularity, on a *bounded* domain $\Omega \in \mathbb{R}^d$, as it would be unreasonable to expect in general a good approximation of f on all of $\mathbb{R}^d$ from a finite number of observations. Moreover, to simplify matters and alleviate notations, we will focus on the case where the flow Φ_f^t satisfies

$$\Phi_f^t(\Omega) \subset \Omega. \tag{12}$$

In Section 3, we give estimates on the finite-dimensional approximations of the Koopman operator, thanks to properties of finite elements spaces. We show how these approximations can be used to solve the system identification problem. The estimates depend on the size of the mesh and on the degree of the finite elements functions we consider, which are related to the dimension of the finite elements space. Accurate Galerkin projections require finite elements spaces of considerable dimension, which increases exponentially with the dimension d of the system under consideration.

In Section 4 we devise a variant of gEDMD to compute a data-driven approximation of these Galerkin projections. The convergence of this approximation depends on the number of data samples, and is essentially based on Monte-Carlo approximation of integrals.

In Section 5 we then analyze the total approximation error of gEDMD with finite elements, and estimate its computational cost, in order to quantify whether this method can be efficient for system identification and predictions in practice.

This allows us to give some clear insights on the challenges and limitations of this method. The overall conclusion is that, even when using accurate approximation spaces such as finite elements, the approximation error decreases relatively slowly, because on one hand the convergence of the data-driven approximation is quite slow (in $m^{-\frac{1}{2}}$, m being the number of samples), but also because this convergence is further slowed if one chooses a finite elements space of greater dimension (i.e. refines the mesh or considers higher-order finite elements).

Finally in Section 6 we give some numerical illustrations in dimension $d = 1$ that illustrate the clear advantage of direct interpolation methods, given the same data sample, due to the slowness of the convergence of gEDMD.

2.3 Notations

We denote the usual Lebesgue measure by μ.

For the rest of this article, when there is no ambiguity we will note $L^2(\mathbb{R}^d, \mu) = L^2$ to alleviate notations. Notice that if $f \in L^\infty(\mathbb{R}^d, \mathbb{R}^d)$, by the definition (9) we have

$$H^1(\mathbb{R}^d) \subset \mathcal{D}(\mathcal{K}). \tag{13}$$

We denote by $\|\cdot\|$ the usual L^2 norm, $\|\cdot\|_{H^s(\mathbb{R}^d)}$ the usual Sobolev norms for $s \geq 0$, and $|\cdot|_{H^k(\mathbb{R}^d)}$ the Sobolev seminorms for $k \in \mathbb{N}$:

$$|\varphi|^2_{H^k(\Omega)} = \sum_{\substack{\alpha \in \mathbb{N}^d \\ |\alpha| = k}} \|D^\alpha \varphi\|^2, \quad \varphi \in H^k(\Omega). \tag{14}$$

In Euclidean spaces, $\|\cdot\|$ denotes the usual Euclidean norm.

For matrices $|||\cdot|||$ denotes the operator norm induced by the Euclidean norm, $\|\cdot\|_F$ denotes the Frobenius norm, $\|\cdot\|_\infty$ denotes the max norm

$$\|A\|_\infty = \max_{ij} |A_{ij}|\,.$$

The sign † denotes the Moore-Penrose pseudo-inverse, and $\top$ will denote the usual transposition.

3 Finite-dimensional approximations of the Koopman operator

3.1 Restricting the Koopman operator to a subdomain Ω

As we have mentioned in the introduction, we will focus on studying the Koopman operator of a vector field f on a bounded domain $\Omega \subset \mathbb{R}^d$. Indeed, from a finite data sample one cannot expect to obtain a good approximation of f on all of $\mathbb{R}^d$.

When dealing with a vector field that is indeed defined on all of $\mathbb{R}^d$, this means we are studying the Koopman operator $\mathcal{K}_\Omega$ of the restriction $f_{|\Omega}$:

$$\mathcal{K}_\Omega = f_{|\Omega} \cdot \nabla. \tag{15}$$

Again, $\mathcal{K}_\Omega$ is an unbounded operator, with domain

$$\mathcal{D}(\mathcal{K}_\Omega) = \{\varphi \in L^2(\Omega), \quad f \cdot \nabla\varphi \in L^2(\Omega)\}. \tag{16}$$

However, with this domain, in general $\mathcal{K}_\Omega$ does not generate a semigroup as in (5). Indeed in general the flow Φ_f^t does not preserve Ω. Noting

$$\partial\Omega^{out} = \{x \in \partial\Omega, \quad f(x) \cdot \nu(x) > 0\}, \tag{17}$$

where $\nu(\cdot)$ denotes the outward normal unit vector, it is necessary to specify boundary conditions on $\partial\Omega^{out}$ in order for the corresponding linear transport equation to be well posed. Typically, the partial differential equation

$$\begin{cases} \dfrac{\partial\varphi}{\partial t} = f(x) \cdot \nabla\varphi, \quad x \in \Omega, \\ \varphi(t,x) = 0 \quad x \in \partial\Omega^{out}, \end{cases} \tag{18}$$

is well posed. Accordingly, the unbounded operator

$$\tilde{\mathcal{K}}_\Omega := f \cdot \nabla, \quad \mathcal{D}(\tilde{\mathcal{K}}_\Omega) = \{\varphi \in H^1(\Omega), \quad \varphi = 0 \text{ on } \partial\Omega^{out}\}, \tag{19}$$

generates the C^0-semigroup

$$e^{t\tilde{\mathcal{K}}_\Omega}\varphi_0(x) = \begin{cases} \varphi_0(\Phi_f^t(x)) & \text{if } \Phi_f^t(x) \in \Omega, \\ 0 & \text{otherwise,} \end{cases} \qquad \forall\varphi_0 \in L^2(\Omega), \quad \forall x \in \Omega. \tag{20}$$

3.2 System identification and predictions with the Koopman operator

We now show how the Koopman operator can be used to recover the vector field f, and in some cases, its flow Φ_f^t.

Let us note the coordinate functions

$$g_i(x) = x_i, \quad x \in \Omega, \ i = 1, \cdots, d. \tag{21}$$

Then, as they are polynomial, and Ω is a bounded domain,

$$g_i \in \mathcal{D}(\mathcal{K}_\Omega), \quad i = 1, \cdots, d. \tag{22}$$

Moreover,

$$\mathcal{K}_\Omega g_i = f \cdot \nabla g_i = f_i, \quad i = 1, \cdots, d, \tag{23}$$

where the f_i denote the coordinates of the vector field f in (4). It is thus possible, with the restricted Koopman operator $\mathcal{K}_\Omega$, to recover the vector field f. On the other hand, in general

$$g_i \notin \mathcal{D}(\tilde{\mathcal{K}}_\Omega)$$

due to the boundary conditions in (19). However, we can still apply the semigroup defined in (20):

$$e^{t\tilde{\mathcal{K}}_\Omega}g_i(x) = \begin{cases} (\Phi_f^t(x))_i & \text{if } \Phi_f^t(x) \in \Omega, \\ 0 & \text{otherwise,} \qquad \forall x \in \Omega, \end{cases} \tag{24}$$

where the $(\Phi_f^t(\cdot))_i$ denote the coordinates of the flow (4). Thus, it is possible, with the restricted Koopman operator $\tilde{\mathcal{K}}_\Omega$, to recover the flow Φ_f^t, and thus make predictions, as long as the flow does not exit Ω.

To make matters simpler, in order to bring forth our conclusions with more clarity, we will for the rest of this article focus on the case where

$$\mu_{\partial\Omega}\left(\partial\Omega^{out}\right) = 0,$$

i.e., the case where Ω is forward invariant under Φ_f^t. Then, we will set, to alleviate notations,

$$\mathcal{K} := \mathcal{K}_\Omega = f \cdot \nabla, \quad \mathcal{D}(\mathcal{K}) := \mathcal{D}(\mathcal{K}_\Omega) = \mathcal{D}(\tilde{\mathcal{K}}_\Omega).$$

With this domain, $\mathcal{K}$ generates the semigroup:

$$e^{t\mathcal{K}}\varphi_0(x) = \varphi_0\left(\Phi_f^t(x)\right), \quad \forall \varphi_0 \in L^2(\Omega), \quad \forall x \in \Omega. \tag{25}$$

i.e. the linear transport equation

$$\begin{cases} \dfrac{\partial \varphi}{\partial t}(t,x) = f(x) \cdot \nabla\varphi(t,x), \quad t \geq 0, \quad x \in \Omega, \\ \quad \varphi(0,\cdot) = \varphi_0(\cdot) \in L^2(\Omega), \end{cases} \tag{26}$$

is well-posed.

In particular, (23) remains unchanged, and (24) becomes

$$e^{t\mathcal{K}} g_i(x) = \left(\Phi_f^t(x)\right)_i, \quad \forall x \in \Omega. \tag{27}$$

3.3 Galerkin projection of the Koopman operator

We now consider a bounded domain $\Omega \subset \mathbb{R}^d$, and we focus on the case where f is such that

$$\Phi_f^t(\Omega) \subset \Omega.$$

On this domain, the Koopman operator of f

$$\mathcal{K} = f \cdot \nabla$$

with domain

$$\mathcal{D}(\mathcal{K}) = H^1(\Omega)$$

generates the semigroup

$$e^{t\mathcal{K}}\varphi_0 = \varphi_0 \circ \Phi_f^t, \quad \forall \varphi_0 \in L^2.$$

The idea of Koopman operator methods is to observe trajectories of the dynamical system (4), but instead of trying to identify the vector field f directly, to apply linear regression methods to identify the linear operator $\mathcal{K}$.

In practice, identifying $\mathcal{K}$ numerically is impossible as it is infinite dimensional. Instead, the goal is to identify a Galerkin projection of $\mathcal{K}$, which corresponds to a reduced-order model of the linear transport equation (5). The projection is determined by a family of linearly independent functions $\Psi := \{\psi_1, \cdots, \psi_N\} \subset L^2$, which span a finite-dimensional subspace $V_N \subset L^2$ which should satisfy

$$V_N \subset \mathcal{D}(\mathcal{K}). \tag{28}$$

Then, the Galerkin projection of $\mathcal{K}$ is given by

$$\mathcal{K}_N := \Pi_N \mathcal{K} \Pi_N, \tag{29}$$

where Π_N denotes the L^2-orthogonal projection on V_N. In particular,

$$\mathcal{K}_N\varphi = \Pi_N \mathcal{K}\varphi, \quad \varphi \in V_N. \tag{30}$$

In this article we will focus on a specific choice of V_N, given by continuous finite elements. Classical approximation results for these spaces then ensure the Galerkin projection is a relevant approximation of the actual Koopman operator.

3.4 Galerkin projection on finite elements spaces

In order to alleviate notations, and simplify matters pertaining to the choice and regularity of meshes, we fix for the remainder of this article

$$\Omega = \mathcal{Q} := [0, L]^d \tag{31}$$

We then define the family of rectangulations of Q:

$$\mathcal{T}_M = \prod_{i=1}^{d} \left[p_i \frac{L}{M}, (p_i+1)\frac{L}{M} \right], \quad 0 \le p_i \le M-1, \ M \in \mathbb{N}^*. \tag{32}$$

In what follows, we will note $\mathcal{T}_h := \mathcal{T}_M$ where

$$h = \frac{L}{M}.$$

These rectangulations form a regular family of rectangulations of $\mathcal{Q}$.

For $k \ge 1$, we define $\mathbb{Q}_k$ the space of polynomials of degree at most k in each variable, and the following continuous finite elements spaces:

$$V_h^k = \left\{ v \in C^0(Q), \quad v_{|K} \in \mathbb{Q}_k, \ \forall K \in \mathcal{T}_h \right\}. \tag{33}$$

We consider the nodes

$$\left\{ \hat{x}_\alpha = \frac{h}{k}\alpha, \quad \alpha \in \left[\!\left[0, k\frac{L}{h} \right]\!\right]^d \right\},$$

and note

$$N_h^k = (1+kM)^d = \left(1 + \frac{kL}{h}\right)^d \tag{34}$$

in order to re-index the nodes by

$$\hat{x}_j, \quad j \in [\![1, N_h^k]\!].$$

Then the classical shape functions ψ_j which satisfy

$$\psi_j(\hat{x}_k) = \delta_{jk} \quad \forall j, k \in [\![1, N_h^k]\!],$$

form a basis of V_h^k, so that

$$\dim V_h^k = N_h^k, \tag{35}$$

i.e.,

$$h = \frac{kL}{\left(N_h^k\right)^{\frac{1}{d}} - 1}. \tag{36}$$

One can see from (35) that both refining the mesh ($h \to 0$) and taking finite elements of higher degree ($k \to \infty$) will increase the dimension of the approximation subspace at the same rate.

We now recall classical projection inequalities for finite elements spaces. Note r_h^k the classical finite elements interpolation operator

$$r_h^k(\varphi) = \sum_{j=1}^{N_h^k} \varphi(x_j)\psi_j, \quad \varphi \in C^0(\overline{\Omega}). \tag{37}$$

The classical interpolation inequalities for triangular finite elements (see for example Quarteroni and Valli [53, Section 3.4]) can be adapted to the rectangulation $\mathcal{T}_h$:

Proposition 3.1. *Let $\mathcal{T}_h$ be the rectangulation defined by* (32)*. Let $s > d/2$, $\nu \in \{0, 1\}$, and define $l = \min(k, s-1)$. Then, for $\varphi \in H^s(\Omega)$, $\varphi \in C^0(\overline{\Omega})$ so that $r_h^k(\varphi)$ is well defined, and there exists $C > 0$ independent of h such that*

$$\|r_h^k(\varphi) - \varphi\|_{H^\nu(Q)} \leq Ch^{l+1-\nu}|\varphi|_{H^{l+1}(Q)}, \quad \forall \varphi \in H^s(Q). \tag{38}$$

Denoting Π_h^k the L^2-orthogonal projection on V_h^k, one gets, by property of orthogonal projections in Hilbert spaces:

Corollary 3.1. *With the notations above,*

$$\|\Pi_h^k(\varphi) - \varphi\|_{H^\nu(Q)} \leq Ch^{l+1-\nu}|\varphi|_{H^{l+1}(Q)}, \quad \forall \varphi \in H^s(Q). \tag{39}$$

Now, we have

$$V_h^k \subset \mathcal{D}(\mathcal{K}). \tag{40}$$

Following (29), the Galerkin projection of $\mathcal{K}$ is then given by

$$\mathcal{K}_h^k := \Pi_h^k \mathcal{K} \Pi_h^k.$$

3.5 System identification and predictions

We know show how the approximation power of finite elements can be combined with (23) and (27) to obtain approximations of the vector field f and its flow Φ_f^t.

First, as the g_i are linear, we clearly have

$$g_i \in V_h^k, \quad \forall i \in [\![1, d]\!]. \tag{41}$$

We then have, from (23),

$$\mathcal{K}_h^k g_i = \Pi_h^k \mathcal{K} g_i = \Pi_h^k f_i, \tag{42}$$

which can be understood as a finite elements approximation of (23).

In the same spirit, we obtain a finite elements approximation of (26) by the Galerkin method:

$$\begin{cases} \dfrac{d}{dt}\varphi_i^{k,h} = \mathcal{K}_h^k \varphi_i^{k,h}, \quad \varphi_i^{k,h} \in V_h^k, \\ \varphi_i^{k,h}(0, \cdot) = g_i(\cdot), \quad i \in [\![1, d]\!]. \end{cases} \tag{43}$$

Remark 3.1. *In accordance with the assumptions we have made on the domain Ω and the behavior of the flow, there is no need here to specify boundary conditions for the Galerkin method, as the actual linear transport equation* (26) *is well-posed without having to specify any boundary conditions.*

Noting

$$\varphi_i^{k,h}(t, \cdot) = \sum_{j=1}^{N_h^k} (\upvarphi_i^{k,h}(t))_j \psi_j, \; g_i = \sum_{j=1}^{N_h^k} (\mathbf{g}_i)_j \psi_j,$$

We have the following system of linear differential equations

$$\begin{cases} M_h^k \dfrac{d}{dt}\upvarphi_i^{k,h} = R_h^k \upvarphi_i^{k,h}, \\ \quad \upvarphi_i^{k,h}(0) = \mathbf{g}_i, \end{cases} \tag{44}$$

where $M_h^k = (\langle \psi_i, \psi_j \rangle)_{i,j}$, $R_h^k = \left(\langle \psi_i, \mathcal{K}_h^k \psi_j \rangle\right)_{i,j}$ are the **mass and stiffness matrices** of the basis (ψ_j), for the operator $\mathcal{K}_h^k$: noting K_h^k its matrix in the basis (ψ_j), we have

$$K_h^k = \left(M_h^k\right)^{-1} R_h^k. \tag{45}$$

We see from (42) and (43) that solving the system identification problem with a Galerkin projection corresponds to projecting the vector field on the finite elements space V_h^k, and approximating the solutions of the linear transport equation (26) as in classical numerical analysis. With this in mind, the choice of classical approximation spaces used in numerical analysis for the approximation of partial differential equations seems adequate.

Accordingly, under some conditions on the order k of the finite elements, and the regularity of the vector field f, we have the following estimates both on the approximation of f and its flow Φ_f^t:

Proposition 3.2. *Let $k+1 > d/2$. There exist constants $C_1, C_2(f) > 0$ such that, for $f \in C^{k+1}(\overline{\Omega}, \mathbb{R}^d)$, the following error estimates hold:*

$$\|\mathcal{K}_h^k g_i - f_i\| \leq C_1 h^{k+1} |f_i|_{k+1}. \tag{46}$$

$$\left\| \left(\Phi_f^t\right)_i - \varphi_i^{k,h}(t,\cdot) \right\| \leq C_2(f) h^k. \tag{47}$$

Proof. From (42), we have

$$\|\mathcal{K}_h^k g_i - f_i\| = \|\Pi_h^k f_i - f_i\|. \tag{48}$$

To estimate the right hand-term, we apply Corollary 3.1 with $s = k+1$ to the $f_i \in C^{k+1}(\Omega) \subset H^{k+1}(\Omega)$ (as Ω is bounded), which directly yields (46).

Now consider the solutions of the following linear transport equations:

$$\begin{cases} \dfrac{\partial \varphi_i}{\partial t}(t,x) = f(x) \cdot \nabla \varphi_i(t,x), \quad t \geq 0, \quad x \in \Omega, \\ \varphi_i(0,\cdot) = g_i(\cdot), \quad i \in [\![1,d]\!], \end{cases} \tag{49}$$

Recalling (27), these solutions are given by

$$\varphi_i(t,x) = g_i\left(\Phi_f^t(x)\right) = \left(\Phi_f^t(x)\right)_i. \tag{50}$$

As $f \in C^{k+1}(\Omega, \mathbb{R}^d)$, by the Cauchy-Lipschitz theorem, for all $i \in [\![1,d]\!]$,

$$(t,x) \mapsto \left(\Phi_f^t(x)\right)_i \tag{51}$$

is in $C^{k+1}([0,T] \times \overline{\Omega})$. Hence, $\varphi_i \in L^2\left(0,T; H^{k+1}(\Omega)\right) \cap H^1\left(0,T; H^k(\Omega)\right)$. Adapting [53, Chapter 14, Section 3] to our setting, which is simplified by the absence of boundary conditions and the fact that $\varphi_i^{k,h}(0,\cdot) \in V_h^k$, we can then estimate the convergence of solutions of (43) to the φ_i: there exists a constant $C_2(f) > 0$ such that

$$\left\| \varphi_i(t,\cdot) - \varphi_i^{k,h}(t,\cdot) \right\| \leq C_2(f) h^k. \tag{52}$$

Together with (50), this directly yields (47). □

Thus, Galerkin projections of the Koopman operator on finite elements spaces are relevant objects to approximately solve the system identification problem, both to recover the vector field f and to approximate its flow Φ_f^t. However the accuracy obtained in the estimates (46) and (47) comes at a cost, as we will see in the next section.

3.6 Curse of dimensionality

In Proposition 3.2 we have given the error estimate (46) for system identification using the Galerkin projection $\mathcal{K}_h^k$ of $\mathcal{K}$ on finite elements spaces. In terms of the dimension of the finite elements space, we have the following:

Proposition 3.3. *The dimension N_h^k of the finite elements spaces needed to ensure the quality of approximation $\varepsilon > 0$ for system identification grows exponentially with the dimension d of the underlying space:*

$$N_h^k \geq C''(L, f) k^d \varepsilon^{-\frac{d}{k+1}}, \tag{53}$$

for some constant $C'''(L, f) > 0$.

Proof. If one requires the quality of approximation in (46):

$$C_1 h^{k+1} |f|_{k+1} \leq \varepsilon, \tag{54}$$

then

$$h \leq C'(f) \varepsilon^{\frac{1}{k+1}}. \tag{55}$$

From (34) we then get

$$N = \left(1 + \frac{kL}{h}\right)^d \leq C''(L, f) k^d \varepsilon^{\frac{d}{k+1}}, \tag{56}$$

which proves (53). □

This is a form of curse of dimensionality, a phenomenon first pointed out in [6] in the context of dynamic programming. It is present in a wide range of situations such as function approximation ([17, 35]), integral computation ([48]), approximation of parametric PDEs ([11, 16]), where the complexity of a problem increases exponentially with its underlying dimension. On the other hand, for a fixed dimension d, the required dimension N_h^k is polynomial in the error ε.

This curse of dimensionality is a common feature in approximation theory. It stems from the use of a mesh to define the approximation spaces (here, continuous functions that are polynomial on each cell of the mesh). This allows a very accurate approximation, but the complexity of the mesh increases exponentially with the dimension, hence the curse of dimensionality.

Recalling that predictions are made by integrating the system of N_h^k linear differential equations (44), this means that in high dimension d, making predictions by using Galerkin projections of the Koopman operator becomes prohibitively expensive.

In the next sections we will also see that this curse of dimensionality has an impact on the data-driven approximation of the Galerkin projections.

4 A variant of generator Extended Dynamic Mode Decomposition

Having given estimates on how the Galerkin projection $\mathcal{K}_h^k$ converges to the Koopman operator $\mathcal{K}$ under suitable regularity assumptions on the vector field f, it remains to compute a data-driven approximation of $\mathcal{K}_h^k$. To achieve this, we present a modified version of the generator Extended Dynamic Mode Decomposition (gEDMD) algorithm.

4.1 The algorithm

Recalling the identity (45), we see that there two distinct components in the Galerkin projection $\mathcal{K}_h^k$: the mass matrix M_h^k, which depends on the space V_h^k and the basis (ψ_j) (thus, essentially, on the domain Ω), and the stiffness matrix R_h^k which depends on the vector field f, the space V_h^k, and the basis (ψ_j). In general, the mass matrix can be approximated by numerical integration methods such as quadrature formulae, or Monte-Carlo integration. In the case of finite elements, in particular linear finite elements, it can even be given explicitly. In any case, this does not require trajectory data, as it does not depend on f, and can thus be done separately.

On the other hand, the stiffness matrix R_h^k depends on f. The algorithm we now lay out mainly consists in approximating this matrix from observations of trajectories of the system (4). To our knowledge, this approach, consisting in separating the approximation of the mass and stiffness matrices, is new. As we will see, it has the advantage of allowing the gEDMD method to be extended to localized bases such as finite elements, or wavelets.

Practically speaking, we consider a collection of points $\{x^l \in \Omega, l \in [\![1, m]\!]\}$, and for each $l \in [\![1, m]\!]$, the solution $x^l(t)$ of (4) with initial condition x^l.

To account for the potentially random nature of measurements for real-life systems, we make the common assumption that the x^l are drawn independently and uniformly with respect to the Lebesgue measure on Ω. From an adequate sampling of the corresponding trajectories $x^l(t), l \in [\![1, m]\!]$, the initial velocities $\dot{x}^k(0) = f(x^k)$ can be approximated by total variation regularized differentiation, which is also used in other methods such as SINDy (see [10, 15]). We then proceed as follows:

Step 1 From the data set $\{x^l, y^l := f(x^l),\ l \in [\![1, m]\!]\}$, define the following matrices:

$$G^m_{k,h} = \left(\psi_i(x^j)\right)_{\substack{i\in[\![1,N]\!]\\ j\in[\![1,m]\!]}}, \quad A^m_{k,h} = \left(y^j \cdot \nabla\psi_i(x^j)\right)_{\substack{i\in[\![1,N]\!]\\ j\in[\![1,m]\!]}}, \tag{57}$$

and compute the following:

$$R^m_{k,h} := \frac{1}{m} G^m_{k,h} \left(A^m_{k,h}\right)^\top. \tag{58}$$

As we will establish later on, $R^m_{k,h}$ is an approximation of the stiffness matrix R^k_h of (44).

Step 2 Solve the following least-squares problem:

$$\min_{K\in\mathbb{R}^{N\times N}} \|M^k_h K - R^m_{k,h}\|^2_F. \tag{59}$$

The solution is then given by

$$K^m_{k,h} := \left(M^k_h\right)^{-1} R^m_{k,h}. \tag{60}$$

4.2 Convergence

We will now prove that $K^m_{k,h}$ converges to the matrix K^k_h given by (45).

Now, as we have considered random points $(x^l)^m_{l=1}$, this means that the coefficients of

$$K^k_h - K^m_{k,h}$$

are random variables. The notion of convergence of random variables we consider here is then the notion of convergence in distribution:

Definition 4.1 (Convergence in distribution of a sequence of random variables)**.** *A sequence $(X_n)_{n\in\mathbb{N}}$ of real random variables is said to converge in distribution to a real random variable X:*

$$X_n \xrightarrow[n\to\infty]{\mathcal{D}} X$$

if their cumulative distribution functions converge pointwise to the cumulative distribution function of X on its continuity set.

We can now state the following convergence result (see also [60, 31, 55]), which focuses on the convergence of the coefficients of the approximate stiffness matrix $R^m_{k,h}$:

Proposition 4.1 (Convergence and complexity of gEDMD)**.** *The computational complexity of gEDMD has the upper bound $\mathcal{O}(mN^2 + N^3)$.*

For $f \in L^\infty(\Omega, \mathbb{R}^d)$, the following convergence in distribution holds:

$$\sqrt{m}\left(\left(R^m_{k,h}\right)_{ij} - \left(R^k_h\right)_{ij}\right) \xrightarrow[m\to\infty]{\mathcal{D}} \mathcal{N}\left(0, \sigma^{k,h}_{ij}\right), \tag{61}$$

where

$$\left(\sigma^{k,h}_{ij}\right)^2 = \int_\Omega \psi_i^2(x)\left(f(x)\cdot\nabla\psi_j(x)\right)^2 dx - \left(\int_\Omega \psi_i(x) f(x)\cdot\nabla\psi_j(x) dx\right)^2, \tag{62}$$

and $\mathcal{N}(0, \sigma^{k,h}_{ij})$ is the normal distribution with mean 0 and standard deviation $\sigma^{k,h}_{ij}$.

Proof. The upper bound for the complexity is given by the complexity of the linear regression that is performed to compute $K^m_{k,h}$. It is well-known (see [57, Appendix D]) that this complexity is bounded by $\mathcal{O}(mN^2 + N^3)$ due to the matrix product operation, and the pseudo-inversion that follows.

As for the convergence in distribution, notice that the coefficients of $R^m_{k,h}$ are given by

$$\left(R^m_{k,h}\right)_{ij} = \frac{1}{m}\sum_{l=1}^{m} \psi_i(x^l) f(x^l) \cdot \nabla\psi_j(x^l),$$

which is none other than the classical Monte-Carlo approximation of the integral

$$\langle \psi_i, f \cdot \nabla\psi_j \rangle = \int_\Omega \psi_i(x) f(x) \cdot \nabla\psi_j(x) dx = \left(R^k_h\right)_{ij},$$

as the sampling points x^l are drawn independently and uniformly with respect to the Lebesgue measure.

Convergence (61) then follows directly from a classical result of Monte-Carlo integration (see [12, Theorem 2.1]:

$$\sqrt{m}\left(\left(\frac{1}{m}\sum_{l=1}^{m} \psi_i(x^l) f(x^l) \cdot \nabla\psi_j(x^l)\right) - \int_\Omega \psi_i(x) f(x) \cdot \nabla\psi_j(x) dx\right) \xrightarrow[m\to\infty]{\mathcal{D}} \mathcal{N}(0, \sigma^{k,h}_{ij}). \tag{63}$$

□

Remark 4.1. *In the usual implementation of gEDMD, the mass matrix is not computed offline, but also approximated with the Monte-Carlo approximation*

$$M^m_{k,h} := \frac{1}{m} G^m_{k,h} \left(G^m_{k,h}\right).$$

The matrix $K^m_{k,h}$ is then given by

$$K^m_{k,h} = \left(M^m_{k,h}\right)^\dagger R^m_{k,h},$$

which converges to K^k_h provided the pseudo-inverses converge. By a classical result due to R.Penrose [50], this is the case if, for m large enough, the $M^m_{k,h}$ are invertible with probability 1. This corresponds to the following condition:

$$\varphi(x^1) = \cdots = \varphi(x^m) = 0 \implies \varphi = 0, \quad \forall \varphi \in V^k_h. \tag{64}$$

When the $(x^j)_j$ are drawn uniformly, this is equivalent to

$$\forall \varphi \in V_N \setminus \{0\}, \quad \mu\{x \in \Omega,\ \varphi(x) = 0\} = 0, \tag{65}$$

which is not satisfied by the shape functions $(\psi_j)_j$, nor by other classical localized approximation functions such as wavelets.

In our case, to ensure (64), it would be necessary to draw sample points in every element of the rectangulation $\mathcal{T}_h$, which would be extremely impractical.

4.3 Convergence in matrix norms

Proposition 4.1 essentially means that, for a given confidence rate $\alpha \in [0, 1)$, there exists a constant $C^{k,h}_{ij}(\alpha)$ given by the normal distribution $\mathcal{N}(0, \sigma^{k,h}_{ij})$, such that for large enough m the following estimate holds with probability at least α:

$$\left|\left(R^m_{k,h}\right)_{ij} - \left(R^k_h\right)_{ij}\right| \leq C^{k,h}_{ij}(\alpha) m^{-\frac{1}{2}}. \tag{66}$$

This probabilistic estimate tells us that the coefficients converge in $m^{-\frac{1}{2}}$, with a constant which depends on the standard deviation $\sigma_{ij}^{k,h}$, that is, recalling its expression (62), on the shape functions (ψ_j) and f. We then note

$$C_h^k(f,\alpha) := \max_{ij}(C_{ij}^{k,h}(\alpha)). \tag{67}$$

Now as we work with L^2 functions, we need to relate this matrix norm to the operator norm on $L^2(\Omega)$. We define the following norm on $\mathbb{R}^{N_h^k}$:

$$\|c\|_{L^2}^2 = \left\|\sum_{j=1}^{N_h^k} c_j\psi_j\right\|^2 = c^\top M_h^k c = \left\|(M_h^k)^{\frac{1}{2}}c\right\|^2. \tag{68}$$

The matrix M_h^k is symmetric definite positive, we note $\rho_{k,h}^+$ and $\rho_{k,h}^-$ its largest and smallest eigenvalue respectively. We then have

$$\sqrt{\rho_{k,h}^-}\|c\| \le \|c\|_{L^2} \le \sqrt{\rho_{k,h}^+}\|c\|, \quad \forall c \in \mathbb{R}^{N_h^k}. \tag{69}$$

With this norm we then define the induced operator norm on $\mathbb{R}^{N_h^k\times N_h^k}$:

$$|||A|||_{L^2} = \sup_{\|c\|_{L^2}=1} \|Ac\|_{L^2}. \tag{70}$$

Applying (69), we then have

$$\sqrt{\frac{\rho_{k,h}^-}{\rho_{k,h}^+}}|||A|||_{L^2} \le |||A||| \le \sqrt{\frac{\rho_{k,h}^+}{\rho_{k,h}^-}}|||A|||_{L^2}, \quad \forall A \in \mathbb{R}^{N_h^k\times N_h^k}. \tag{71}$$

We can now state the following proposition:

Proposition 4.2 (Convergence of gEDMD in matrix norm)**.** *Let $\alpha \in [0,1)$. For $f \in L^\infty(\Omega,\mathbb{R}^d)$, there exists a constant $C_h^k(f,\alpha)$ such that, for m large enough and with probability at least α,*

$$|||K_h^k - K_{k,h}^m|||_{L^2} \le \frac{N_h^k}{\rho_{k,h}^-}\sqrt{\frac{\rho_{k,h}^+}{\rho_{k,h}^-}}C_h^k(f,\alpha)m^{-\frac{1}{2}}. \tag{72}$$

Proof. From the probabilistic estimate (66) we can then deduce the following estimate on the max matrix norm: for m large enough, with probability at least α,

$$\|R_{k,h}^m - R_h^k\|_\infty \le C_h^k(f,\alpha)m^{-\frac{1}{2}} \tag{73}$$

Now, we recall the result of [58] on sharp equivalence constants for matrix norms:

$$\frac{1}{N_h^k}|||A||| \le \|A\|_\infty \le |||A|||, \quad \forall A \in \mathbb{R}^{N_h^k\times N_h^k}. \tag{74}$$

Putting (74), (71) and (73) together, we get, for m large enough and with probability at least α:

$$|||R_h^k - R_{k,h}^m|||_{L^2} \le N_h^k\sqrt{\frac{\rho_{k,h}^+}{\rho_{k,h}^-}}C_h^k(f,\alpha)m^{-\frac{1}{2}} \tag{75}$$

Finally, by submultiplicativity of operator norms, and the expression of $K_{k,h}^m$, we have

$$|||K_h^k - K_{k,h}^m|||_{L^2} = |||(M_h^k)^{-1}(R_h^k - R_{k,h}^m)|||_{L^2} \le |||(M_h^k)^{-1}|||_{L^2}|||R_h^k - R_{k,h}^m|||_{L^2}. \tag{76}$$

Now, as

$$|||(M_h^k)^{-1}|||_{L^2} = |||(M_h^k)^{-1}||| = \frac{1}{\rho_{k,h}^{-}}, \tag{77}$$

we finally deduce (72) from (75). □

This shows that the convergence of K_h^k to $K_{k,h}^m$, i.e. the convergence of gEDMD, is asymptotically in $m^{-\frac{1}{2}}$, which is quite slow but does not depend on the underlying dimension d of the problem (4). It is however further slowed down by numerous factors: the standard deviation $\sigma_{ij}^{k,h}$, which represents the oscillatory behavior of the shape functions, the conditioning number and smallest eigenvalue of the mass matrix M_h^k, and the dimension of the finite elements space V_h^k.

5 Quantitative analysis

5.1 Using gEDMD for system identification and predictions: convergence analysis

We now use the matrix $K_{k,h}^m$ computed by gEDMD to give an approximation of f and its flow Φ_f^t.

Recalling (23) and (42), an approximation of the f_i is given by

$$\mathcal{K}_{k,h}^m g_i, \quad i \in \llbracket 1, d \rrbracket. \tag{78}$$

Recalling (26), (27), (43), (45) and (60), an approximation of the $\left(\Phi_f^t\right)_i$ is given by the solutions to

$$\begin{cases} \dfrac{d}{dt}\varphi_{i,m}^{k,h} = \mathcal{K}_{k,h}^m \varphi_{i,m}^{k,h}, \quad \varphi_{i,m}^{k,h} \in V_h^k, \\ \varphi_{i,m}^{k,h}(0,\cdot) = g_i(\cdot), \quad i \in \llbracket 1, d \rrbracket. \end{cases} \tag{79}$$

In the basis (ψ_j), noting $g_i = \sum_{j=1}^{N_h^k} \gamma_j^i \psi_j$, the components of the $\mathcal{K}_{k,h}^m g_i$ are given by

$$K_{k,h}^m \gamma^i \in \mathbb{R}^{N_h^k}, \quad i \in \llbracket 1, d \rrbracket \tag{80}$$

and the components of the $\varphi_{i,m}^{k,h}(t,\cdot)$ are given by the solutions to

$$\begin{cases} \dfrac{d}{dt}\upvarphi_{i,m}^{k,h} = K_{k,h}^m \upvarphi_{i,m}^{k,h}, \quad \upvarphi_{i,m}^{k,h} \in \mathbb{R}^{N_h^k}, \\ \upvarphi_{i,m}^{k,h}(0,\cdot) = \gamma^i, \quad i \in \llbracket 1, d \rrbracket. \end{cases} \tag{81}$$

We now give quantitative estimates for system identification and predictions, using gEDMD with finite elements spaces:

Proposition 5.1. *Let $f \in C^{k+1}(\overline{\Omega})$. There exists constants $C_1, C_2(f), C_3 > 0$ such that, for a given probability rate $\alpha \in [0,1]$, one has for m large enough and with probability at least α:*

$$\begin{aligned} \|\mathcal{K}_{k,h}^m g_i - f_i\| &\leq \frac{N_h^k}{\rho_{k,h}^{-}} \sqrt{\frac{\rho_{k,h}^{+}}{\rho_{k,h}^{-}}} C_h^k(f,\alpha) m^{-\frac{1}{2}} \|g_i\| + C_1 h^{k+1} |f_i|_{k+1}, \\ \left\| \varphi_{i,m}^{k,h}(t,\cdot) - \left(\Phi_f^t\right)_i \right\| &\leq C_3 t e^{t |||K_h^k|||_{L^2}} \frac{N_h^k}{\rho_{k,h}^{-}} \sqrt{\frac{\rho_{k,h}^{+}}{\rho_{k,h}^{-}}} C_h^k(f,\alpha) m^{-\frac{1}{2}} \|g_i\| + C_2(f) h^k, \qquad \forall i \in \llbracket 1, d \rrbracket. \end{aligned} \tag{82}$$

Proof. Recalling (23) and (42), we have the following: for $i \in [\![1, d]\!]$,

$$\begin{aligned} \|\mathcal{K}^m_{k,h} g_i - f_i\| &\le \|\mathcal{K}^m_{k,h} g_i - \mathcal{K}^k_h g_i\| + \|\mathcal{K}^k_h g_i - f_i\| \\ &\le |\!|\!| K^m_{k,h} - K^k_h |\!|\!|_{L^2} \|g_i\| + \|\mathcal{K}^k_h g_i - f_i\|. \end{aligned} \tag{83}$$

Applying (72) and (46) yields the first inequality of (82).

Recalling (27) and (43), we have the following: for $i \in [\![1, d]\!]$,

$$\begin{aligned} \left\| \varphi^{k,h}_{i,m}(t,\cdot) - \left(\Phi^t_f\right)_i \right\| &\le \left\| \varphi^{k,h}_{i,m}(t,\cdot) - \varphi^{k,h}_i(t,\cdot) \right\| + \left\| \varphi^{k,h}_i(t,\cdot) - \varphi_i(t,\cdot) \right\| \\ &\le |\!|\!| e^{tK^m_{k,h}} - e^{tK^k_h} |\!|\!|_{L^2} \|g_i\| + \left\| \left(\Phi^t_f\right)_i - \varphi^{k,h}_i(t,\cdot) \right\|. \end{aligned} \tag{84}$$

By property of the matrix exponential,

$$|\!|\!| e^{tK^m_{k,h}} - e^{tK^k_h} |\!|\!|_{L^2} \le t |\!|\!| K^m_{k,h} - K^k_h |\!|\!|_{L^2} e^{t|\!|\!| K^k_h |\!|\!|_{L^2}} e^{t|\!|\!| K^m_{k,h} - K^k_h |\!|\!|_{L^2}} \tag{85}$$

Combining (85) with (84) and (72) then yields the second inequality of (82). □

With these inequalities it is now clear that the "slowest" term is the data-driven approximation error term in $m^{-\frac{1}{2}}$. It depends both on the number of samples m and the dimension of the Galerkin projection space V^k_h.

More precisely, for given k and h, we see from the first inequality of (82) that for the data-driven approximation error to reach the same accuracy as the finite elements approximation error, m has to satisfy

$$m \ge \left(\frac{N^k_h}{\rho^-_{k,h}} \right)^2 \frac{\rho^+_{k,h}}{\rho^-_{k,h}} C^k_h(f, \alpha)^2 h^{-2k-2}. \tag{86}$$

For a fixed k, choosing more accurate finite elements spaces by decreasing the mesh size h has a double impact on the required number of samples m: on the h^{-2k-2} term, on one hand, and on the quantity $(N^k_h/\rho^-_{k,h})^2(\rho^+_{k,h}/\rho^-_{k,h})C^k_h(f,\alpha)^2$ on the other hand. Moreover, due to the presence of N^k_h we see that this number of samples must also increase exponentially with d. Thus, indirectly, the convergence of gEDMD is also subject to the curse of dimensionality.

Overall, the slowness of convergence of classical Monte-Carlo approximations usually employed in gEDMD conceals the high accuracy one can obtain from classical approximation spaces, which in a way defeats the purpose of choosing good approximation spaces.

In the next section we will focus on the case $d = 1$ and fix $k = 1$, in order to give precise estimates of the terms in (86). This will in turn allow us to give a precise estimate of the computational cost of gEDMD and give a clear comparison to direct interpolation methods.

5.2 Comparison with direct interpolation in 1-D for linear finite elements

In this subsection, $d = 1$ and $k = 1$. That is, we focus on gEDMD for systems on the interval $[0, L]$, using Galerkin projections on linear finite elements.

System identification using the Koopman operator appears as a quite indirect method, as it consists in observing trajectories of the differential equation, working with the associated transport equation, then coming back to the differential equation.

In the gEDMD algorithm, the values $f(x^l)$ at the sample points are approximated. Instead of using them to compute an approximation of the Koopman operator, one could also use them to interpolate f directly, without resorting to Koopman operator methods. In that case, predictions are then made by integrating the nonlinear ODE

$$\dot{x} = \tilde{f}(x), \quad x \in \mathbb{R}^d,$$

where $\tilde{f}$ is the interpolant of f, instead of a Galerkin method of the linear transport equation such as (81). We will focus here on linear and cubic spline interpolation.

In theory, the linear interpolant $\ell(f, x_i)$ and the natural cubic spline interpolant $\sigma(f, x_i)$ (with derivative 0 at the endpoints) satisfy the following inequalities

$$\begin{aligned} \|\ell(f, x_i) - f\| &\le C_1\delta^2|f|_2, \quad f \in H^2(0, L), \\ \|\sigma(f, x_i) - f\| &\le C_2\delta^4|f|_4, \quad f \in H^4(0, L), \end{aligned} \tag{87}$$

where

$$\delta = \max\{x_{n+1} - x_n, x_1 + L, L - x_m, \; n \in [\![1, m]\!]\} \tag{88}$$

The inequality for the linear interpolant follows from classical inequalities on C^2 functions, for the cubic spline interpolation error bounds, we refer to [21].

The m samples are drawn in the whole interval, and for large m we have (see [52])

$$\delta \propto \frac{\log(m)}{m}$$

so that (3) becomes

$$\begin{aligned} \|\ell(f, x_i) - f\| &\le C_2\left(\frac{\log(m)}{m}\right)^2|f|_2, \quad f \in H^2(0, L), \\ \|\sigma(f, x_i) - f\| &\le C_2\left(\frac{\log(m)}{m}\right)^4|f|_4, \quad f \in H^4(0, L). \end{aligned} \tag{89}$$

Comparing with the $m^{-\frac{1}{2}}$ rate of convergence of gEDMD, we get

$$\begin{aligned} m^{\frac{1}{2}}\left(\frac{\log(m)}{m}\right)^2 &= \left(\frac{\log(m)}{m^{\frac{3}{4}}}\right)^2 \xrightarrow[m\to\infty]{} 0 \\ m^{\frac{1}{2}}\left(\frac{\log(m)}{m}\right)^4 &= \left(\frac{\log(m)}{m^{\frac{7}{8}}}\right)^4 \xrightarrow[m\to\infty]{} 0. \end{aligned} \tag{90}$$

Hence, for a given number of samples m, gEDMD is less accurate than direct interpolation methods.

To give a more thorough comparison, let $\varepsilon > 0$. We now compare the number m of samples required in each method in order to to achieve an accuracy of at most ε.

For linear interpolation, the required number of samples satisfies:

$$\frac{\log(m)}{m} \sim \varepsilon^{\frac{1}{2}} \implies \varepsilon^{-\frac{1}{2}} \le m \le C\varepsilon^{-\frac{1}{2}+\epsilon}$$

For cubic spline interpolation, the required number of samples satisfies:

$$\frac{\log(m)}{m} \sim \varepsilon^{\frac{1}{4}} \implies \varepsilon^{-\frac{1}{4}} \le m \le C\varepsilon^{-\frac{1}{4}+\epsilon}$$

For modified gEDMD, first consider the Galerkin projection. From (53), h must satisfy:

$$h \le C\varepsilon^{\frac{1}{2}}. \tag{91}$$

Recalling (86):

$$m \ge \left(C_h^1(f, \alpha)\frac{1 + \frac{1}{h}}{\rho_{1,h}^-}\right)^2 c_h^1\, h^{-4}$$

and using the estimates (110), (111), and (120) from Appendix A, one has:

$$\begin{aligned} m &\ge C'\left(C(\alpha)\|f\|_{L^\infty}h^{-2}\frac{1 + \frac{1}{h}}{h}\right)^2 \varepsilon^{-2} \\ &\ge C''C(\alpha)^2\|f\|_{L^\infty}^2 h^{-8}\varepsilon^{-2} \\ &\ge C'''C(\alpha)^2\|f\|_{L^\infty}^2\varepsilon^{-6} \end{aligned} \tag{92}$$

Now, it is well-known that interpolation methods have linear complexity (see [59]). On the other hand, we have seen in Proposition 4.1 that the computational complexity of gEDMD is in $\mathcal{O}(mN^2+N^3)$. Regarding N, (91) implies

$$N \geq C'\varepsilon^{-\frac{1}{2}}.$$

Together with (92), this yields the following complexity in terms of ε:

$$\mathcal{O}\left(\varepsilon^{-7}+\varepsilon^{-\frac{3}{2}}\right)=\mathcal{O}(\varepsilon^{-7})$$

.

We summarize the above in the following table:

	Parameters	**Cost**
Koopman operator method	h k m	ε^{-7}
Linear interpolation	m	$\varepsilon^{-\frac{1}{2}+\epsilon}$
Splines interpolation	m	$\varepsilon^{-\frac{1}{4}+\epsilon}$

This clearly shows that interpolation methods are more cost-effective than gEDMD with linear finite elements.

6 Numerical illustration

We show the results of some implementations of gEDMD, focusing on the approximation of the vector field f. Given our remarks on the curse of dimensionality in Section 3.6, we focus on the case $d=1$. We have given precise estimates in the previous section that show how the convergence of the data-driven approximation is affected by the choice of the Galerkin projection. Given (53) and (86), this problem worsens in higher dimensions.

We take f given by combinations of trigonometric functions, polynomials, and exponentials on the interval $[0,1]$. We implement gEDMD with linear finite elements to recover an approximation of f. We compare the results with linear interpolation and spline interpolation on the same data sample.

6.1 Linear finite elements

Data sample we form the data set:

$$\{x^n, f(x^n), \quad n \in \llbracket 1, m \rrbracket\} \tag{93}$$

To simplify matters we directly take the values of the nonlinearity f we have chosen, at the sample points x^n. In practice, f is unknown, so one obtains approximate values of f at the sample points by observing the trajectories of the system with initial conditions x^n, and approximating the initial velocities by finite differences (see Section 4 and [15]).

gEDMD. Compute the following matrices from the data:

$$\begin{aligned} G_{1,h}^m &:= (\psi_i(x_j))_{\substack{i\in\llbracket 1,N\rrbracket_h \\ j\in\llbracket 1,m\rrbracket}} \\ A_{1,h}^m &:= (\dot{x_j}\cdot\nabla\psi_i(x_j))_{\substack{i\in\llbracket 1,N\rrbracket_h \\ j\in\llbracket 1,m\rrbracket}}\,, \end{aligned} \tag{94}$$

and define

$$R_{1,h}^{m} = \frac{1}{m} G_{1,h}^{m} \left(A_{1,h}^{m}\right)^{\top}.$$

Perform the linear least squares regression:

$$K_{1,h}^{m} := \operatorname{argmin}_{A \in \mathbb{R}^{N \times N}} \left\| M_{h}^{1} A - R_{1,h}^{m} \right\|^{2} = \left(M_{1,h}^{m}\right)^{-1} R_{1,h}^{m}. \tag{95}$$

Identifying the nonlinearity in the differential equation. Now consider the identity function $g(x) = x$: $g \in \mathbb{P}_1$ as it is linear, and

$$g(x) = \sum_{j=0}^{M} jh\psi_j(x) = c^{\top}\psi(x), \quad \forall x \in [0, L]. \tag{96}$$

To recover f, we compute

$$\tilde{c}^{\top} := c^{\top} K_{1,h}^{m} \tag{97}$$

to get the following approximation of f on the interval $[0, L]$:

$$\tilde{f}_{1,h}^{m}(x) = \tilde{c}^{\top}\psi(x) = \sum_{j=0}^{M} \tilde{c}_j \psi_j(x), \quad \forall x \in [0, L]. \tag{98}$$

6.2 Numerical simulations

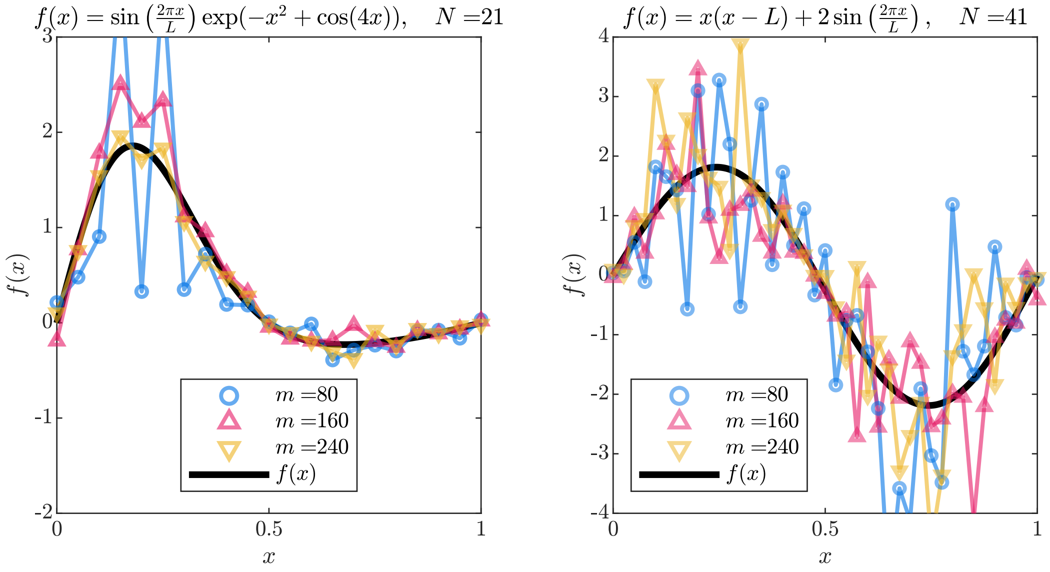

Figure 1: Identification of different functions using gEDMD with linear finite elements.

Figure 1 shows approximations of f (given by the thicker black curve) computed with gEDMD for a fixed dimension of the finite elements space, and different numbers of samples m. One can see that for a small number of samples, there are sizeable discrepancies. For a larger number of samples the curves obtained by gEDMD seem to match the curve rather well.

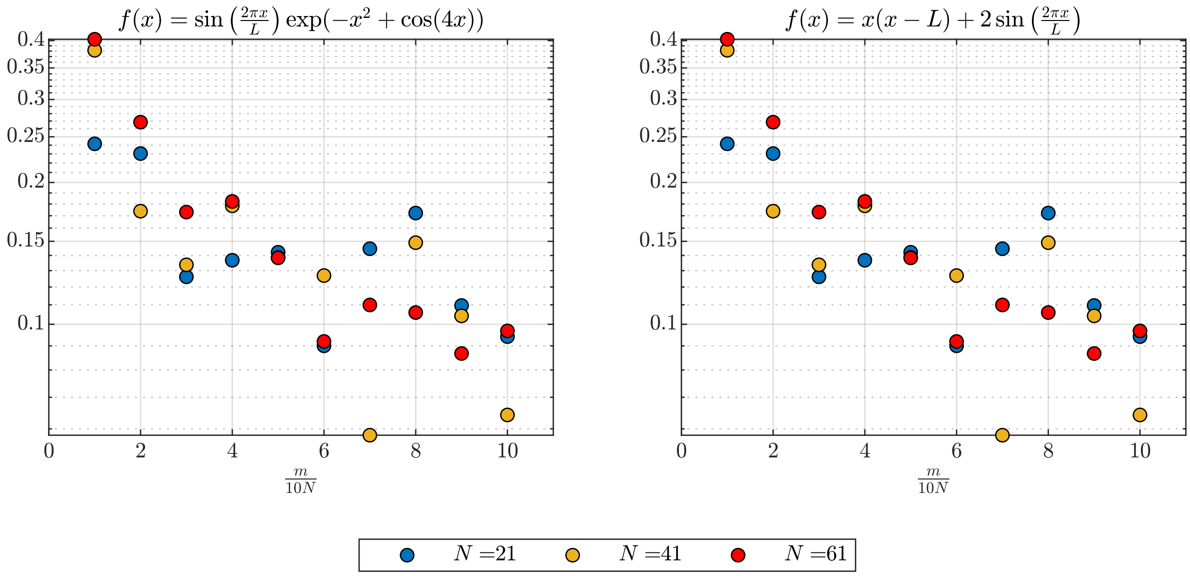

Figure 2: Relative error of gEDMD with linear elements for different values of N. Dependence on the number m of samples.

In Figure 2, we plot the relative approximation error

$$\frac{\|\tilde{f}_N^m - f\|}{\|f\|}$$

as a function of the number of samples m, for several values of N: as expected, increasing the number of samples decreases the error. This decrease appears to be slower for greater values of N (see the red dots in the figure below), which is consistent with (86).

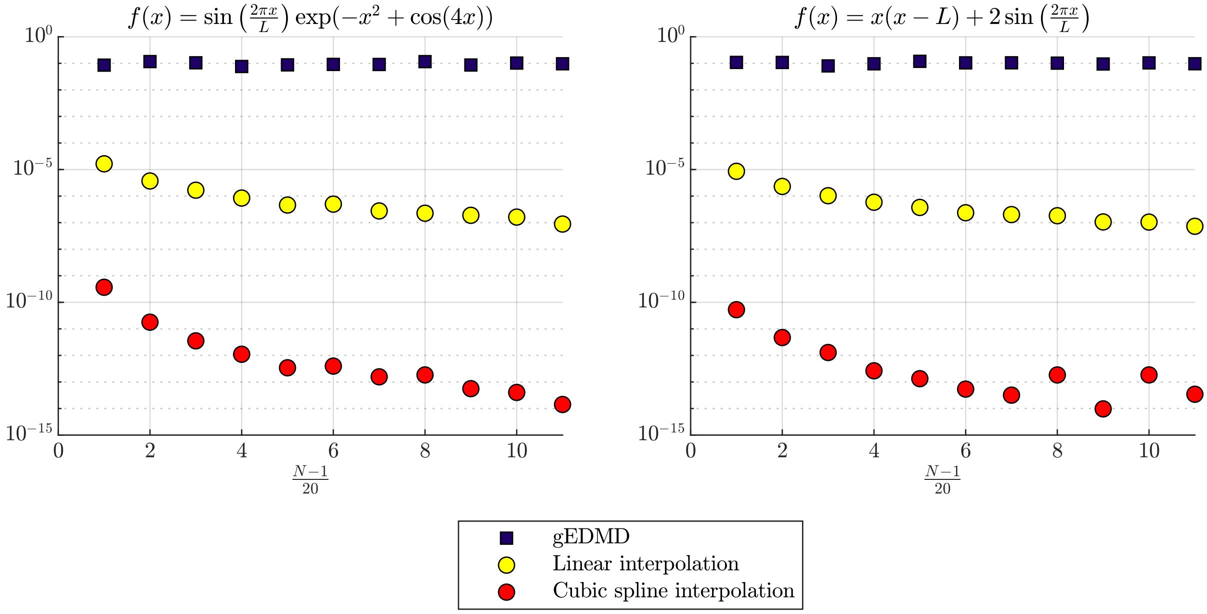

Figure 3: Comparison with direct interpolation methods. $m = 100(N-1)$.

Finally, in Figure 3 we compare the relative approximation error of gEDMD with that of direct interpolation methods. We implement gEDMD for increasing values of N, with $m = 100N$ samples. We compare the results with linear interpolation and spline interpolation with m random samples. There is a clear hierarchy between the three methods, gEDMD being the least accurate by far.

7 Conclusion

In this article, we have presented the implementation of a **system identification** method, based on finite-dimensional, data-driven approximations $\mathcal{K}_N^m$ of the so-called **Koopman operator**, which is the transport operator associated to the unknown vector field f:

$$f(x) \cdot \nabla.$$

From trajectory data on the system

$$\dot{x} = f(x),$$

measured at randomly drawn sample points $(x^n)_{n \in [\![1,m]\!]}$ one can compute the sample values

$$f(x^n), \quad x^n \in \Omega, \quad n \in [\![1,m]\!].$$

From this data set, the method provides an approximation of the vector field f and its flow Φ_f^t, using the coordinate functions $g_i(x) = x_i$:

$$f_i \approx \mathcal{K}_N^m g_i, \quad \left(\Phi_f^t\right)_i \approx e^{t\mathcal{K}_N^m} g_i.$$

The potential advantage of this method lies in its **linearity** of the Koopman operator: as can be seen above, the nonlinear flow Φ_f^t is approximated thanks to the linear flow $e^{t\mathcal{K}_N^m}$, which is however defined on a subspace of functions rather than the state space $\Omega \subset \mathbb{R}^d$.

We have focused on the case of Galerkin projections of the Koopman operator on finite elements spaces. They are generic approximation spaces, widely used in numerical analysis to approximate PDEs. In this case, the approximating properties of these spaces guarantee that the corresponding Galerkin projection of the Koopman operator is a reliable approximation, without having to rely on hypothetical spectral properties. Indeed the Koopman operator (which is essentially a transport operator) does not have nice spectral properties in general.

For 1-dimensional problems, we have compared this method with **direct interpolation** of the vector field f from the sample values

$$f(x^n), \quad x^n \in \Omega, \quad n \in [\![1, m]\!].$$

Interpolation yields an approximation $\tilde{f}$ of f. Predictions can then be made by integrating the nonlinear differential equation

$$\dot{x} = \tilde{f}(x).$$

We have found that this Koopman operator based system identification method is generally not advantageous in practice, due to two drawbacks.

1. **Dimension of the Galerkin projections** With Galerkin projections on classical approximation spaces, high accuracy requires high-dimensional spaces of functions. This drawback, due to the well-known *curse of dimensionality*, is a recurring feature of approximation spaces. On these spaces, the approximation of the linear transport equation associated to the Koopman operator then comes at a high computational cost.

2. **Data-driven approximation of the Galerkin projections** Galerkin projection on finite elements spaces provide accurate, albeit high-dimensional, approximations of the Koopman operator for system identification and predictions. However, for the system identification problem we have formulated, these projections must be recovered from a finite amount of data. To that effect, we have brought a novel modification to the usual gEDMD algorithm. As we are working with classical finite elements spaces, the mass matrix can be computed offline. Then, recovering the Galerkin projection of the Koopman operator reduces to approximating the stiffness matrix from data. To approximate the integrals in the stiffness matrix, given the nature of the data set, the natural path is to use Monte-Carlo integration (as is the case for regular EDMD in the literature).

   In addition to the inherent slowness of Monte-Carlo integration, we furthermore observe that the rate of convergence of the data-driven approximation has multiple dependencies in the parameters h (mesh size) and k (degree) of the finite elements space. As a consequence, the more accurate Galerkin projection we take, the harder it becomes to approximate from data by the gEDMD method.

   We have quantified this precisely in 1-D, which allows for a clear comparison between recovery of the vector field f by the gEDMD method and direct interpolation (linear or cubic splines) of f.

We have based our analysis on a specific choice of approximation space, but our estimates illustrate what is probably a general phenomenon: the Galerkin projection of the Koopman operator on generic approximation spaces is prohibitively slow to approximate from data.

Let us conclude with some prospects that could be explored to overcome these drawbacks:

Galerkin projections of low dimension Considering that classical interpolation methods (or SINDy with sparsity constraints) provide a very good approximation of the vector field, the goal for the gEDMD method is to provide a numerically interesting approximation of the flow Φ_f^t in order to make predictions. This will be the case if adequate lower-dimensional Galerkin projections can be found, which still allow for reliable estimates as in Proposition 3.2 with finite elements spaces. Finding such spaces would also reduce the impact of the choice of the Galerkin projection on the Monte-Carlo convergence, which in addition to being inherently slow, is further slowed down when taking higher-dimensional finite elements spaces.

It seems very likely that there are many situations where this is not possible (chaotic systems are a frequently cited example [3, 4]). Between the universal solution of classical approximation spaces, and the ideal situation of subspaces of functions that are invariant under the Koopman operator (see [9]), the complete picture is still missing: what characterizes the existence of these relevant low-dimensional Galerkin projections? How "rare" is it? Is there a data-driven way to determine if they exist and find them?

Use of deep neural networks There have been many recent attempts to find such low-dimensional Galerkin projections, using deep neural networks [51, 64, 62, 34, 56]. The output of such networks is a family of functions $\psi_1, \cdots, \psi_N$, where N is fixed beforehand. These functions are taken from a large class of functions (essentially given by the architecture of the network) and the network is trained to minimize the prediction error made when performing EDMD with the subspace $V_N = Span\{\psi_1, \cdots, \psi_N\}$, while additionally penalizing situations where V_N does not contain the g_i, for the purpose of system identification (see Section 5.1). Although numerical implementations yield promising results, there is still much work to be done to interpret the outputs of these deep neural networks. Moreover, as is mentioned in [56], there is no guarantee that the output is not a local minimizer.

Acknowledgements We would like to thank Pr.Lars Grüne for fruitful discussions on this topic.

We are greatly indebted to Jesus Oroya Villalta, for his help on the numerical experiments and his astute comments.

A Linear 1-D finite elements

We fix a mesh size

$$h := \frac{1}{M}, \quad M \in \mathbb{N}^*, \tag{99}$$

we define the mesh

$$\mathcal{T}_h = \{jh, \ j \in [\![0, M]\!]\}, \tag{100}$$

which is a subdivision of $[0, 1]$. We define the finite elements of degree 1:

$$V_h^1 := \left\{v \in C^0\left([0,1]\right), \quad v_{|[jh,(j+1)h]} \text{ is affine}, \ j \in [\![0, M-1]\!]\right\}. \tag{101}$$

We note

$$N := \dim V_h^1 = M + 1 = \frac{1}{h} + 1. \tag{102}$$

We denote by $\psi_j, j \in [\![0, M]\!]$ the node functions, which satisfy

$$\psi_j(kh) = \delta_{jk}, \quad j, k[\![0, M]\!], \tag{103}$$

they form a basis of V_h^1.

Explicitly, for a given node jh, ψ_j is defined by

$$j \in [\![1, M-1]\!], \ \psi_j(x) = \begin{cases} 0 & \text{on} \quad [-L, (j-1)h] \cup [(j+1)h, L] \\ \dfrac{x-(j-1)h}{h} & \text{on} \quad [(j-1)h, jh] \\ \dfrac{(j+1)h - x}{h} & \text{on} \quad [jh, (j+1)h] \end{cases} \tag{104}$$

$$\Psi_0(x) = \begin{cases} \dfrac{h-x}{h} & \text{on} \quad [0, h] \\ 0 & \text{on} \quad [h, L] \end{cases} \tag{105}$$

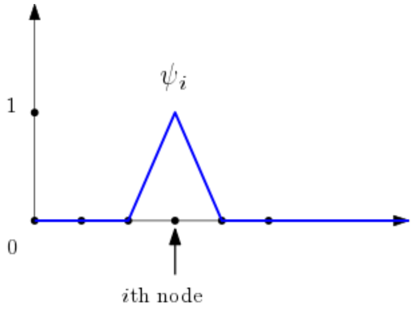

Figure 4: The shape function ψ_j

$$\Psi_M(x) = \begin{cases} 0 & \text{on} \quad [-L, L-h] \\ \dfrac{x-L+h}{h} & \text{on} \quad [L-h, L] \end{cases} \tag{106}$$

Classically, the mass matrix is given by:

$$M_h^1 = h \begin{pmatrix} 1/3 & 1/6 & 0 & \cdots & \cdots & \cdots & 0 \\ 1/6 & 2/3 & 1/6 & 0 & \ddots & \cdots & \vdots \\ 0 & 1/6 & 2/3 & 1/6 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 1/6 & 2/3 & 1/6 \\ 0 & \cdots & \cdots & \cdots & 0 & 1/6 & 1/3 \end{pmatrix} \tag{107}$$

and its eigenvalues are given by (after a slight modification of [1, Exercise 7.4.1]):

$$\lambda_j = \frac{h}{3}\left(2 + \cos\left(\frac{jh\pi}{1+2h}\right)\right), \quad j \in [\![1, N]\!]. \tag{108}$$

Consequently,

$$\rho_{1,h}^{+} = \frac{h}{3}\left(2 + \cos\left(\frac{\pi h}{1+2h}\right)\right), \tag{109}$$

$$\rho_{1,h}^{-} = \frac{h}{3}\left(2 + \cos\left(\frac{N\pi h}{1+2h}\right)\right) \xrightarrow[h \to 0]{} 0, \tag{110}$$

and

$$c_h^1 = \frac{\rho_{1,h}^{+}}{\rho_{1,h}^{-}} = \frac{2 + \cos\left(\frac{\pi h}{1+2h}\right)}{2 + \cos\left(\frac{\pi(h+1)}{1+2h}\right)} \xrightarrow[h \to 0]{} 3. \tag{111}$$

Let us now give estimates on the variances of the coefficients of the rigidity matrix, which is involved in the convergence of the Monte-Carlo method in Section 4.2. First, an obvious remark is that

$$\sigma_{ij}^{k,h} = 0, \quad |i-j| > 1. \tag{112}$$

On the other hand, assuming that $f \in C^2(\overline{\Omega})$: for $i \in [\![0, M-1]\!]$,

$$\int_\Omega \left(\psi_i(x) f(x) \psi'_{i+1}(x)\right)^2 dx - \frac{f(x_i)^2}{h^2} \int_{x_i}^{x_{i+1}} \psi_i(x)^2 dx = \frac{1}{h^2} \int_{x_i}^{x_{i+1}} \psi_i(x)^2 (f(x)^2 - f(x_i)^2) dx \leq \|f'\|_\infty \frac{C_1}{h}, \tag{113}$$

for some constant $C_1 > 0$. Similarly,

$$\int_\Omega \left(\psi_{i+1}(x) f(x) \psi'_i(x)\right)^2 dx - \frac{f(x_i)^2}{h^2} \int_{x_i}^{x_{i+1}} \psi_{i+1}(x)^2 dx = \frac{1}{h^2} \int_{x_i}^{x_{i+1}} \psi_{i+1}(x)^2 (f(x)^2 - f(x_i)^2) dx \leq \|f'\|_\infty \frac{C_2}{h}, \tag{114}$$

and

$$\left(\int_\Omega \psi_i(x) f(x) \psi'_{i+1}(x) dx\right)^2 - \frac{f(x_i)^2}{h^2} \left(\int_{x_i}^{x_{i+1}} \psi_i(x) dx\right)^2 \leq \|f'\|_\infty \frac{C_3}{h}, \tag{115}$$

$$\left(\int_\Omega \psi_{i+1}(x) f(x) \psi'_i(x) dx\right)^2 - \frac{f(x_i)^2}{h^2} \left(\int_{x_i}^{x_{i+1}} \psi_{i+1}(x) dx\right)^2 \leq \|f'\|_\infty \frac{C_4}{h}, \tag{116}$$

for some constants $C_2, C_3, C_4 > 0$.

Putting (113) and (115) together, we get, for $i \in [\![0, M-1]\!]$,

$$\left(\sigma_{i,i+1}^{k,h}\right)^2 = \frac{1}{3h^2} f(x_i)^2 + \mathcal{O}\left(\frac{1}{h}\right), \tag{117}$$

and similarly with (114) and (116),

$$\left(\sigma_{i+1,i}^{k,h}\right)^2 = \frac{1}{3h^2} f(x_i)^2 + \mathcal{O}\left(\frac{1}{h}\right), \tag{118}$$

and, for $i \in [\![0, M]\!]$,

$$\left(\sigma_{ii}^{k,h}\right)^2 = \frac{2}{3h^2} f(x_i)^2 + \mathcal{O}\left(\frac{1}{h}\right). \tag{119}$$

Recalling (67), this yields

$$C_h^1(f, \alpha) = \frac{2}{3h^2} \|f\|_\infty^2 + \mathcal{O}\left(\frac{1}{h}\right). \tag{120}$$

References

[1] Grégoire Allaire. *Numerical analysis and optimization: an introduction to mathematical modelling and numerical simulation.* Oxford university press, 2007.

[2] H. Arbabi, M. Korda, and I. Mezić. A Data-Driven Koopman Model Predictive Control Framework for Nonlinear Partial Differential Equations. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 6409–6414, 2018.

[3] Hassan Arbabi and Igor Mezić. Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the koopman operator. *SIAM Journal on Applied Dynamical Systems*, 16(4):2096–2126, 2017.

[4] Hassan Arbabi and Themistoklis Sapsis. Generative stochastic modeling of strongly nonlinear flows with non-gaussian statistics, 2020.

[5] Bassam Bamieh and Laura Giarré. Identification of linear parameter varying models. *International Journal of Robust and Nonlinear Control*, 12(9):841–853, 2002. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rnc.706.

[6] Richard Bellman. *Dynamic Programming.* Princeton University Press, Princeton, NJ, USA, 1 edition, 1957.

[7] Peter Binev, Albert Cohen, Wolfgang Dahmen, and Ronald DeVore. Universal algorithms for learning theory. part ii: Piecewise polynomial functions. *Constructive approximation*, 26(2):127–152, 2007.

[8] Peter Binev, Albert Cohen, Wolfgang Dahmen, Ronald DeVore, and Vladimir Temlyakov. Universal algorithms for learning theory part i: piecewise constant functions. *Journal of Machine Learning Research*, 6(Sep):1297–1321, 2005.

[9] Steven L. Brunton, Bingni W. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Koopman Invariant Subspaces and Finite Linear Representations of Nonlinear Dynamical Systems for Control. *PLOS ONE*, 11(2):1–19, 2016.

[10] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

[11] Hans-Joachim Bungartz and Michael Griebel. Sparse grids. *Acta Numerica*, 13:147–269, 2004.

[12] Russel E. Caflisch. Monte carlo and quasi-monte carlo methods. *Acta Numerica*, 7:1–49, 1998.

[13] Alberto P Calderón. On an inverse boundary value problem. *Computational & Applied Mathematics*, 25(2-3):133–138, 2006.

[14] Torsten Carleman. Application de la théorie des équations intégrales linéaires aux systèmes d'équations différentielles non linéaires. *Acta Mathematica*, 59:63–87, 1932. Publisher: Institut Mittag-Leffler.

[15] Rick Chartrand. Numerical differentiation of noisy, nonsmooth, multidimensional data. 12 2017.

[16] Abdellah Chkifa, Albert Cohen, and Christoph Schwab. Breaking the curse of dimensionality in sparse polynomial approximation of parametric PDEs. *Journal de Mathématiques Pures et Appliquées*, 103(2):400 – 428, 2015.

[17] Ronald A DeVore. Nonlinear approximation. *Acta numerica*, 7:51–150, 1998.

[18] R. J. DiPerna and P. L. Lions. Ordinary differential equations, transport theory and Sobolev spaces. *Inventiones mathematicae*, 98(3):511–547, October 1989.

[19] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* MIT Press, 2016. `http://www.deeplearningbook.org`.

[20] Nithin Govindarajan, Ryan Mohr, Shivkumar Chandrasekaran, and Igor Mezic. On the Approximation of Koopman Spectra for Measure Preserving Transformations. *SIAM Journal on Applied Dynamical Systems*, 18(3):1454–1497, 2019. _eprint: https://doi.org/10.1137/18M1175094.

[21] Charles A. Hall and W. Weston Meyer. Optimal error bounds for cubic spline interpolation. *Journal of Approximation Theory*, 16(2):105–122, 1976.

[22] Maxime Hauray, Claude Le Bris, and Pierre-Louis Lions. Deux remarques sur les flots généralisés d'équations différentielles ordinaires. *Comptes Rendus Mathematique*, 344(12):759 – 764, 2007.

[23] Victor Isakov. *Inverse problems for partial differential equations*, volume 127. Springer, 2006.

[24] Eurika Kaiser, J. Nathan Kutz, and Steven L. Brunton. Data-driven discovery of koopman eigenfunctions for control, 2017.

[25] Stefan Klus, Feliks Nüske, and Boumediene Hamzi. Kernel-Based Approximation of the Koopman Generator and Schrödinger Operator. *Entropy*, 22(7):722, June 2020. Publisher: MDPI AG.

[26] Stefan Klus, Feliks Nüske, Sebastian Peitz, Jan-Hendrik Niemann, Cecilia Clementi, and Christof Schütte. Data-driven approximation of the koopman generator: Model reduction, system identification, and control. *Physica D: Nonlinear Phenomena*, 406:132416, 2020.

[27] Sebastian Kmiecik, Dominik Gront, Michal Kolinski, Lukasz Wieteska, Aleksandra Elzbieta Dawid, and Andrzej Kolinski. Coarse-Grained Protein Models and Their Applications. *Chemical Reviews*, 116(14):7898–7936, July 2016. Publisher: American Chemical Society.

[28] B. O. Koopman. Hamiltonian Systems and Transformation in Hilbert Space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931. Publisher: National Academy of Sciences _eprint: https://www.pnas.org/content/17/5/315.full.pdf.

[29] BO Koopman and JV Neumann. Dynamical systems of continuous spectra. *Proceedings of the National Academy of Sciences of the United States of America*, 18(3):255—263, March 1932.

[30] Milan Korda and Igor Mezić. Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *Automatica*, 93:149 – 160, 2018.

[31] Milan Korda and Igor Mezić. On Convergence of Extended Dynamic Mode Decomposition to the Koopman Operator. *Journal of Nonlinear Science*, 28(2):687–710, April 2018.

[32] Milan Korda and Igor Mezić. *Optimal construction of Koopman eigenfunctions for prediction and control*. 2018. _eprint: 1810.08733.

[33] Milan Korda, Mihai Putinar, and Igor Mezić. Data-driven spectral analysis of the Koopman operator. *Applied and Computational Harmonic Analysis*, 48(2):599 – 629, 2020.

[34] Milan Korda, Yoshihiko Susuki, and Igor Mezić. Power grid transient stabilization using koopman model predictive control. *IFAC-PapersOnLine*, 51(28):297 – 302, 2018. 10th IFAC Symposium on Control of Power and Energy Systems CPES 2018.

[35] Robert J. Kunsch. Breaking the curse for uniform approximation in Hilbert spaces via Monte Carlo methods. *Journal of Complexity*, 48:15 – 35, 2018.

[36] Matthew D. Kvalheim and Shai Revzen. Existence and uniqueness of global koopman eigenfunctions for stable fixed points and periodic orbits. *arXiv: Dynamical Systems*, 2019.

[37] Yueheng Lan and Igor Mezić. Linearization in the large of nonlinear systems and koopman operator spectrum. *Physica D: Nonlinear Phenomena*, 242(1):42 – 53, 2013.

[38] Adrien Marie Legendre. *Nouvelles méthodes pour la détermination des orbites des comètes*. F. Didot, 1805.

[39] Ljung Lennart. System identification: theory for the user. *PTR Prentice Hall, Upper Saddle River, NJ*, pages 1–14, 1999.

[40] Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):1–10, 2018. Publisher: Nature Publishing Group.

[41] A. Mauroy and J. Goncalves. Koopman-based lifting techniques for nonlinear systems identification. *IEEE Transactions on Automatic Control*, 65(6):2550–2565, 2020.

[42] Alexandre Mauroy and Igor Mezic. Global Stability Analysis Using the Eigenfunctions of the Koopman Operator. *IEEE Transactions on Automatic Control*, 61(11):3356–3369, November 2016. Publisher: Institute of Electrical and Electronics Engineers (IEEE).

[43] Igor Mezić. Spectral Properties of Dynamical Systems, Model Reduction and Decompositions. *Nonlinear Dynamics*, 41(1):309–325, August 2005.

[44] Igor Mezić. Analysis of fluid flows via spectral properties of the koopman operator. *Annual Review of Fluid Mechanics*, 45(1):357–378, 2013.

[45] Mezić Igor. Spectrum of the Koopman Operator, Spectral Expansions in Functional Spaces, and State-Space Geometry. *Journal of Nonlinear Science*, 2019.

[46] Oliver Nelles. *Nonlinear system identification: from classical approaches to neural networks and fuzzy models*. Springer Science & Business Media, 2013.

[47] J. v. Neumann. Proof of the quasi-ergodic hypothesis. *Proceedings of the National Academy of Sciences*, 18(1):70–82, 1932.

[48] Erich Novak and Klaus Ritter. The curse of dimension and a universal method for numerical integration. In *Multivariate approximation and splines*, pages 177–187. Springer, 1997.

[49] Samuel Otto and Clarence Rowley. Linearly-recurrent autoencoder networks for learning dynamics. *SIAM Journal on Applied Dynamical Systems*, 18, 12 2017.

[50] R. Penrose. A generalized inverse for matrices. *Mathematical Proceedings of the Cambridge Philosophical Society*, 51(3):406–413, 1955.

[51] Zuowei Ping, Zhun Yin, Xiuting Li, Yefeng Liu, and Tao Yang. Deep Koopman model predictive control for enhancing transient stability in power grids. *International Journal of Robust and Nonlinear Control*, n/a(n/a). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rnc.5043.

[52] R. Pyke. Spacings. (With discussion.). *J. Roy. Statist. Soc. Ser. B*, 27:395–449, 1965.

[53] Alfio Quarteroni and Alberto Valli. *Numerical approximation of partial differential equations*, volume 23 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 1994.

[54] Julia Slipantschuk, Oscar F. Bandtlow, and Wolfram Just. Dynamic mode decomposition for analytic maps. *Communications in Nonlinear Science and Numerical Simulation*, 84:105179, 2020.

[55] Péter Koltai Stefan Klus and Christof Schütte. On the numerical approximation of the perron-frobenius and koopman operator, 2016.

[56] Naoya Takeishi, Yoshinobu Kawahara, and Takehisa Yairi. Learning koopman invariant subspaces for dynamic mode decomposition. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 1130–1140. Curran Associates, Inc., 2017.

[57] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., USA, 2005.

[58] Andrew Tonge. Equivalence constants for matrix norms: a problem of goldberg. *Linear Algebra and its Applications*, 306(1):1–13, 2000.

[59] Kazuo Toraichi, Kazuki Katagishi, Iwao Sekita, and Ryoichi Mori. Computational complexity of spline interpolation. *International Journal of Systems Science*, 18(5):945–954, 1987. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00207728708964021.

[60] Matthew O. Williams, Maziar S. Hemati, Scott T. M. Dawson, Ioannis G. Kevrekidis, and Clarence W. Rowley. Extending Data-Driven Koopman Analysis to Actuated Systems. *IFAC-PapersOnLine*, 49(18):704 – 709, 2016.

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

[62] Yongqian Xiao, Xinglong Zhang, Xin Xu, Xueqing Liu, and Jiahang Liu. A deep learning framework based on koopman operator for data-driven modeling of vehicle dynamics, 2020.

[63] Masahiro Yamamoto. Carleman estimates and an inverse heat source problem for the thermoelasticity system. *Inverse Problems*, 27(1):015006, December 2010. Publisher: IOP Publishing.

[64] E. Yeung, S. Kundu, and N. Hodas. Learning deep neural network representations for koopman operators of nonlinear dynamical systems. In *2019 American Control Conference (ACC)*, pages 4832–4839, 2019.