

## RESEARCH OUTPUTS / RÉSULTATS DE RECHERCHE

### Two methods to approximate the Koopman operator with a reservoir computer

Gulina, Marvyn; MAUROY, ALEXANDRE

*Published in:*

Chaos: an interdisciplinary journal of nonlinear science

*DOI:*

[10.1063/5.0026380](https://doi.org/10.1063/5.0026380)

*Publication date:*

2021

*Document Version*

Early version, also known as pre-print

[Link to publication](#)

*Citation for published version (HARVARD):*

Gulina, M & MAUROY, ALEXANDRE 2021, 'Two methods to approximate the Koopman operator with a reservoir computer', *Chaos: an interdisciplinary journal of nonlinear science*, vol. 31, no. 2, 023116.  
<https://doi.org/10.1063/5.0026380>

#### General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

#### Take down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

## Two methods to approximate the Koopman operator with a reservoir computer

Marvyn Gulina<sup>a)</sup> and Alexandre Mauroy<sup>b)</sup>

*Department of Mathematics and Namur Institute for Complex Systems (naXys),  
University of Namur, Belgium*

(Dated: 2 September 2020)

The Koopman operator provides a powerful framework for data-driven analysis of dynamical systems. In the last few years, a wealth of numerical methods providing finite-dimensional approximations of the operator have been proposed (e.g. extended dynamic mode decomposition (EDMD) and its variants). While convergence results for EDMD require an infinite number of dictionary elements, recent studies have shown that only few dictionary elements can yield an efficient approximation of the Koopman operator, provided that they are well-chosen through a proper training process. However, this training process typically relies on nonlinear optimization techniques.

In this paper, we propose two novel methods based on a reservoir computer to train the dictionary. These methods rely solely on linear convex optimization. We illustrate the efficiency of the method with several numerical examples in the context of data reconstruction, prediction, and computation of the Koopman operator spectrum. These results pave the way to the use of the reservoir computer in the Koopman operator framework.

Keywords: Koopman operator, dictionary learning, reservoir computing, nonlinear dynamics, dynamic mode decomposition.

---

<sup>a)</sup>Electronic mail: marvyn.gulina@unamur.be

<sup>b)</sup>Electronic mail: alexandre.mauroy@unamur.be

The so-called Koopman operator offers the possibility to turn nonlinear dynamical systems into linear ones. In this framework, dynamical systems can be studied with systematic linear techniques and, in particular, they are amenable to spectral analysis. However there is a price to pay. The Koopman operator is infinite-dimensional and must be approximated by a finite-rank operator (i.e. a matrix) as soon as numerical methods come into play. This approximation requires to choose a finite-dimensional subspace, a choice which is not necessarily appropriate since it is made *a priori*. Recent methods have been proposed, using neural networks to “learn” the best finite-dimensional approximation subspace. The main drawback of these methods is that they rely on nonlinear optimization. In this paper, we propose to obtain a finite-dimensional approximation of the Koopman operator by using a reservoir computer. The reservoir computer is a specific recurrent neural network where only the weights of the nodes on the output layer are trained with the data, a training which can be performed with linear, convex optimization. Considering either the internal nodes or the output nodes of the reservoir computer to obtain the finite-dimensional approximation subspace, we derive two novel methods that compute a finite-dimensional approximation of the Koopman operator.

---

## I. INTRODUCTION

Dynamical systems theory plays an important role in the context of data analysis. In fact time-series can often be assumed to be generated by an underlying dynamical system, and to be related to the system orbits through a given observation map. In contrast to this classical description, there exists an alternative description in terms of the observation maps themselves, also called *observables*. The dynamics, and in particular the time evolution of the observables, are then described through the so-called Koopman operator, which is a linear (but infinite-dimensional) operator. In this linear setting, it is natural to study the spectral properties of the operator and relate them to the dynamics of the nonlinear system<sup>26</sup>. The related notion of Koopman modes decomposition is also useful to study the systems in many contexts (e.g. fluids dynamics<sup>30</sup>, power grids<sup>36</sup>, epidemiology<sup>29</sup>, control<sup>25</sup>).

A noticeable fact is that the Koopman operator description is conducive to data analysis. In particular, there exist numerical techniques that can be used to compute a finite-dimensional approximation of the Koopman operator from data. Combined with the spectral analysis relying on Koopman modes, these data-driven techniques lead to the so-called (Extended) Dynamic Mode Decomposition ((E)DMD) method<sup>6,8,31,39</sup>. In practice, EDMD techniques require to choose a specific approximation subspace, or equivalently a finite set of *dictionary functions*. This choice is crucial, but has to be made *a priori* and is therefore not necessarily relevant to provide the best approximation of the operator.

Recently, *Dictionary learning* methods based on neural networks have been proposed to provide a relevant set of dictionary functions that are trained with the data and yield appropriate finite-dimensional representations of the Koopman operator<sup>16,37</sup>. Subsequent developments have also been made in the context of deep learning<sup>5,19,27,40</sup>. All these learning methods showed good performances, thereby demonstrating the effectiveness of dictionary learning in EDMD methods. However these techniques require nonlinear optimization of the neural networks, while the classical EDMD method merely relies on linear optimization (i.e. linear least squares regression).

In this work, we propose a novel dictionary learning method for EDMD, which relies solely on linear optimization. Our key idea is to combine the EDMD method with a reservoir computer<sup>10</sup>. In more general contexts, reservoir computing compete with other algorithms on hard tasks such as channel equalization<sup>32</sup>, phoneme recognition<sup>38</sup>, and prediction<sup>1,28</sup>, amongst others (see the survey<sup>17,18</sup>). To our knowledge, we propose the first use of a reservoir computer for dictionary learning in the Koopman operator framework. Note that, very recently, the work<sup>3</sup> has emphasized a connection between the Koopman operator and the reservoir computer, though in a slightly different setting where the reservoir activation function is linear. This reinforces our claim that reservoir computing is relevant in the context of the Koopman operator. Interestingly the recurrent neural network characterizing the reservoir allows to train the dictionary with a dynamical network rather than with a static one. Hence, generated dictionary functions are nonlinear functions of time-delay coordinates, which are particularly relevant for time-delay systems and also bear some similarity to previous Koopman mode decomposition methods based on delayed coordinates (e.g. prony method<sup>35</sup>, Hankel DMD<sup>2</sup>). We derive two numerical schemes, where the dictionary functions are respectively the internal states and the outputs of the reservoir computer. We

illustrate these two methods with several examples in the context of data reconstruction, prediction, and computation of the Koopman operator spectrum.

The rest of the paper is organized as follows. Section II provides an introduction to the Koopman operator framework, the EDMD method, and the reservoir computer. Section III presents our two methods obtained by combining the EDMD method with the reservoir computer. These two methods are illustrated in Section IV with numerical examples. Finally, concluding remarks are given in Section V.

## II. PRELIMINARIES

### A. The Koopman operator framework

The Koopman operator provides an alternative framework to describe the evolution of nonlinear systems in a purely linear fashion. Consider an autonomous dynamical system

$$\mathbf{x}(t+1) = \mathbf{F}(\mathbf{x}(t)) \quad \mathbf{x} \in \mathcal{X}, \quad (1)$$

where  $\mathcal{X}$  is (an invariant subset of) the state space and  $\mathbf{F} : \mathcal{X} \rightarrow \mathcal{X}$  is a nonlinear map. The Koopman operator is defined as the composition<sup>14</sup>

$$\mathcal{K}f = f \circ \mathbf{F} \quad (2)$$

where  $f : \mathcal{X} \rightarrow \mathbb{C}$  is an *observable* that belongs to some function space  $\mathcal{F}$ . In the following, we will assume that  $\mathcal{F} = L^2(\mathcal{X})$  and that  $\mathcal{X}$  is a compact set. It is clear from (2) that the Koopman operator is linear. Also, while (1) describes the nonlinear dynamics of the state in the space  $\mathcal{X}$ , (2) equivalently describes the linear dynamics of the observables in  $\mathcal{F}$ . Roughly speaking the system described in the space  $\mathcal{X}$  is *lifted* into the space  $\mathcal{F}$  when it is described in the Koopman's framework.

The Koopman operator description can be used for several purposes. For instance, it can be used for prediction. Indeed, the (vector-valued) identity function  $\mathbf{g}(\mathbf{x}) = \mathbf{x}$ , also called projection maps, is characterized by the linear evolution

$$\mathbf{x}(t+1) = \mathcal{K}\mathbf{g}(\mathbf{x}(t)). \quad (3)$$

Provided that the Koopman operator associated with the system is known, (3) allows to predict future trajectories. Moreover the spectral properties of the Koopman operator —

namely the eigenvalues  $\lambda \in \mathbb{C}$  and the associated eigenfunctions  $\phi \in \mathcal{F}$  satisfying  $\mathcal{K}\phi = \lambda\phi$  — provide meaningful information on the underlying dynamical system<sup>24,26</sup>. In particular, the eigenvalues are related to internal frequencies of the dynamics and the eigenfunctions reveal geometric properties in the state space. These spectral properties can also be used for control<sup>9,12,15</sup>, stability analysis<sup>23,34</sup>, time-series classification<sup>33</sup>, analysis and training of neural networks<sup>5,21</sup>, and network identification<sup>22</sup>, to list a few.

## B. Finite-dimensional approximation of the Koopman operator

Since the Koopman operator is infinite-dimensional, it is natural and often necessary to compute a finite-dimensional approximation. This approximation is given by the so-called *Koopman matrix*  $\mathbf{K}$ , which represents the projection  $\mathcal{P}$  of the operator onto a subspace  $\mathcal{F}_D$  spanned by the basis functions  $\psi_k \in \mathcal{F}$ ,  $k = 1, \dots, D$ , also called *dictionary*. More precisely, the  $i^{\text{th}}$  row of  $\mathbf{K}$  is the coordinate vector of  $\mathcal{P}\mathcal{K}\psi_i$  in the dictionary. If one denotes  $\boldsymbol{\psi}(\mathbf{x}) = (\psi_1(\mathbf{x}), \dots, \psi_D(\mathbf{x}))^T$  and  $\mathcal{K}\boldsymbol{\psi}(\mathbf{x}) = (\mathcal{K}\psi_1(\mathbf{x}), \dots, \mathcal{K}\psi_D(\mathbf{x}))^T$ , one has

$$\mathcal{K}\boldsymbol{\psi}(\mathbf{x}) \approx \mathcal{P}\mathcal{K}\boldsymbol{\psi}(\mathbf{x}) = \mathbf{K}\boldsymbol{\psi}(\mathbf{x}),$$

so that one can obtain an approximation of the evolution of the dictionary functions under the action of the Koopman operator. In particular, if the identity belongs to the dictionary, it follows from (3) that an approximation of the system trajectories can be computed.

The finite-dimensional approximation of the Koopman operator can be obtained from data through the so-called Extended Dynamic Mode Decomposition (EDMD) method<sup>39</sup>. Given a set of snapshot pairs  $\{(\mathbf{x}_t, \mathbf{x}'_t = \mathbf{F}(\mathbf{x}_t))\}_{t=1}^T$ , the Koopman matrix is given by

$$\mathbf{K} = \arg \min_{\tilde{\mathbf{K}} \in \mathbb{R}^{D \times D}} \sum_{t=1}^T \left\| \boldsymbol{\psi}(\mathbf{x}'_t) - \tilde{\mathbf{K}}\boldsymbol{\psi}(\mathbf{x}_t) \right\|^2 = \boldsymbol{\Psi}' \boldsymbol{\Psi}^+ \quad (4)$$

where  $^+$  denotes the pseudo-inverse, and where  $\boldsymbol{\Psi}$  and  $\boldsymbol{\Psi}' \in \mathbb{R}^{D \times T}$  denote the matrices whose columns are  $\boldsymbol{\psi}(\mathbf{x}_t)$  and  $\boldsymbol{\psi}(\mathbf{x}'_t)$ , respectively, for  $t \in \{1, \dots, T\}$ . The Koopman matrix is the solution to a least squares problem and therefore represents the approximation of the operator obtained with a discrete orthogonal projection. Note that the specific dictionary  $\boldsymbol{\psi}(\mathbf{x}) = \mathbf{x}$  leads to the classical Dynamic Mode Decomposition (DMD) algorithm<sup>8,30,31</sup>. This statement justifies the term “extended” introduced above.

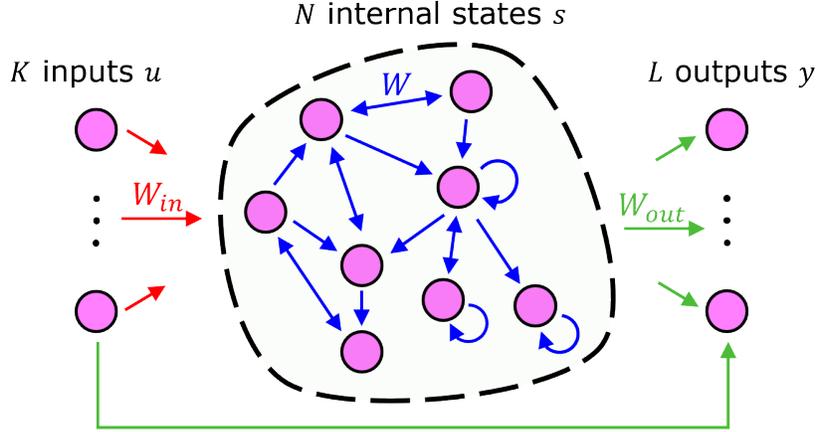


FIG. 1: *Layout of the reservoir computer.*

The finite-dimensional approximation of the operator depends on both the projection and the dictionary of basis functions. In a data-driven context, the discrete orthogonal projection used in the EDMD method is a natural and appropriate projection to use. However, the choice of the dictionary is somehow arbitrary but crucial since it affects the quality of the approximation. The original EDMD method<sup>39</sup> relies on a dictionary that is fixed and chosen *a priori* (e.g. polynomial functions, radial basis functions). Recently, machine learning techniques have been used to guide the choice of the dictionary<sup>16</sup>. Building on this result, we propose to select the dictionary functions through a reservoir computer.

### C. Reservoir computer

The reservoir computer is a discrete-time neural network which consists of three layers: the inputs, the reservoir, and the outputs (FIG. 1). We denote the input signals by  $\mathbf{u} \in \mathbb{R}^K$ , the reservoir states by  $\mathbf{s} \in \mathbb{R}^N$ , and the output signals by  $\mathbf{y} \in \mathbb{R}^L$ . The reservoir states are updated according to the dynamics<sup>10</sup>

$$\mathbf{s}(t+1) = (1 - Ca)\mathbf{s}(t) + C \tanh[\mathbf{W}_{in}\mathbf{u}(t+1) + \mathbf{W}\mathbf{s}(t) + \boldsymbol{\nu}(t)] \quad (5)$$

where  $C$  is a timescale constant,  $a$  is the leaking rate,  $\mathbf{W} \in \mathbb{R}^{N \times N}$  and  $\mathbf{W}_{in} \in \mathbb{R}^{N \times K}$  are the matrices of internal connection weights and input weights, respectively, and  $\boldsymbol{\nu}$  is a noise term. The matrix  $\mathbf{W}$  is typically sparse and its density (*i.e.* the proportion of non-zero elements) is denoted by  $\gamma$ . The nonzero entries of  $\mathbf{W}_{in}$  and  $\mathbf{W}$  are uniformly randomly distributed over  $[-1, 1]$  and  $[-w, w]$ , respectively. The components of  $\boldsymbol{\nu}$  are uniformly distributed over

$[-\varepsilon, \varepsilon]$ . This noise term is added according to the work<sup>10</sup> as an alternative to Tikhonov regularization for the output weights training.

The reservoir can contain loops and is therefore a *recurrent neural network*. Furthermore, the gain  $w$  is chosen such that the spectral radius  $\rho$  of  $\mathbf{W}$  satisfies the echo state property<sup>10,11</sup>

$$|1 - C(a - \rho)| < 1, \quad (6)$$

so that the reservoir “forgets” the initial condition  $\mathbf{s}(0)$ , which is uniformly distributed over  $[0, 1]$ . Hence the computer reservoir is an *echo state network*. In practice, we can discard the first (transient) states corresponding to the initialization of the reservoir.

Finally, the outputs are given by

$$\mathbf{y}(t) = \mathbf{W}_{\text{out}} \bar{\mathbf{s}}(t), \quad (7)$$

where  $\bar{\mathbf{s}}(t) = [\mathbf{s}(t); \mathbf{u}(t)] \in \mathbb{R}^{\bar{N}}$  (with  $\bar{N} = N + K$ ) is the vertical concatenation of internal states and inputs and where  $\mathbf{W}_{\text{out}} \in \mathbb{R}^{L \times \bar{N}}$  is the matrix of output weights. It is noticeable that the outputs are obtained through *linear* combinations of the states and that *only* the output weights are trained. This is a computational advantage of the reservoir computer that we will leverage.

### III. NUMERICAL METHODS

In this Section, we present our two methods, which combine the EDMD technique with the reservoir computer. The key idea is to use (linear combinations of) the internal states of the reservoir as dictionary functions. The first method uses all the states of the reservoir, while the second method selects a subset of these states.

#### A. Method 1: EDMD using a reservoir computer

A straightforward method consists in using all the reservoir internal states as dictionary functions, *i.e.*  $\Psi = \bar{\mathbf{s}}$ . In this case the optimization problem (4) becomes

$$\mathbf{K} = \arg \min_{\tilde{\mathbf{K}} \in \mathbb{R}^{\bar{N} \times \bar{N}}} \sum_{t=1}^{T-1} \left\| \bar{\mathbf{s}}(t+1) - \tilde{\mathbf{K}} \bar{\mathbf{s}}(t) \right\|^2 \quad (8)$$

and its solution is given by

$$\mathbf{K} = \mathbf{S}' \mathbf{S}^+ \quad (9)$$

where  $\mathbf{S}$  and  $\mathbf{S}' \in \mathbb{R}^{\bar{N} \times (T-1)}$  denote the matrices whose columns are  $\bar{\mathbf{s}}(t)$  for  $t \in \{1, \dots, T-1\}$  and  $t \in \{2, \dots, T\}$ , respectively. The internal states evolve according to the dynamics (5), where the input  $\mathbf{u}(t)$  is a trajectory  $\mathbf{x}(t)$  (equivalently,  $\bar{\mathbf{s}} = [\mathbf{s}; \mathbf{x}]$ ). For this reason, the data points are generated from a single trajectory, and not from a set of scattered data pairs.

**Remark 1.** *The proposed dictionary can be interpreted as nonlinear functions of time-delay coordinates. Indeed, since the internal states are solutions to (5), they depend on past values of the input  $\mathbf{u}(t)$ , or equivalently of the state  $\mathbf{x}(t)$ . Moreover, provided that the reservoir satisfies the echo state property and is initialized for a sufficiently long time before generating the data, the internal states do not depend on the (random) initial condition and are time-independent. The use of time-delay coordinates to construct the basis functions is reminiscent of previous works in the context of the Koopman operator<sup>2,13,35</sup>.*

The proposed method is summarized in Algorithm 1.

---

**Algorithm 1** EDMD using a reservoir computer

---

**Input:** Sampled trajectory  $\mathbf{x}(t)$ ,  $t = 1, \dots, T$ ; parameters  $N, L, C, a, \gamma, \rho, w_{in}, \varepsilon$ .

**Output:** Koopman matrix  $\mathbf{K}$ .

- 1: Initialize  $\mathbf{W}_{in}, \mathbf{W}$  (see Section II C) and set random initial reservoir states.
  - 2: Update the reservoir states according to (5) with the input  $\mathbf{u}(t) = \mathbf{x}(t)$ , for  $t = 1, \dots, T-1$ .
  - 3: Compute the Koopman matrix  $\mathbf{K} = \mathbf{S}'\mathbf{S}^+$  (9) (possibly discarding the first values of the states  $\bar{\mathbf{s}}$ ).
- 

**B. Method 2: dictionary learning for EDMD using a reservoir computer**

The method presented above yields a  $\bar{N} \times \bar{N}$  Koopman matrix, which is not so convenient since the number of internal states is typically large. In order to reduce the size of the Koopman matrix, we propose to define the dictionary functions as the output (7) of the reservoir, that is we select  $L \ll N$  linear combinations of the states  $\bar{\mathbf{s}} = [\mathbf{s}; \mathbf{x}]$  obtained through a dictionary learning step. The optimization problem is written as

$$\min_{\mathbf{W}_{out}, \mathbf{K}} \sum_{t=1}^{T-1} \|\mathbf{W}_{out} \bar{\mathbf{s}}(t+1) - \mathbf{K} \mathbf{W}_{out} \bar{\mathbf{s}}(t)\|^2. \quad (10)$$

It is clear that  $\mathbf{W}_{\text{out}} = \mathbf{0}$  is a trivial solution to the optimization problem. We therefore add the projection maps  $\mathbf{x}$  to the outputs, considering the augmented output weights matrix  $\mathbf{W}_{\text{out}} = [\mathbf{W}_1; \mathbf{W}_2] \in \mathbb{R}^{(L+K) \times \bar{N}}$  with the output weights  $\mathbf{W}_1 \in \mathbb{R}^{L \times \bar{N}}$  related to the main dictionary functions and the output weights  $\mathbf{W}_2 \in \mathbb{R}^{K \times \bar{N}}$  related to the projection maps  $\mathbf{x}$ . The dictionary size is given by  $D = L + K$ . Since  $\mathbf{u}(t) = \mathbf{x}(t)$ , we have  $\mathbf{W}_2[\mathbf{s}; \mathbf{u}] = \mathbf{u}$ , so that the constraint  $\mathbf{W}_2 = [\mathbf{0}_{K,N}, \mathbf{I}_K]$  (where  $\mathbf{0}_{K,N}$  is the  $K \times N$  zero matrix and  $\mathbf{I}_K$  is the  $K \times K$  identity matrix) prevents the trivial zero solution. In this case, we only need to optimize over the weights  $\mathbf{W}_1$ .

This method presented can be interpreted as an intermediate method between the first method developed in Section III A and the DMD method. If we set  $L = N$ , the dictionary contains the maximal number of independent functions constructed with the internal states of the reservoir. These functions can be chosen as the internal states themselves and we recover the first method. In contrast, in the case  $L = 0$ , there is no optimization performed on the outputs weights and the optimization on  $\mathbf{K}$  associated with linear basis functions is equivalent to DMD. The proposed method can therefore be seen as a trade-off where an optimal subset of basis functions is obtained through the training process.

Similarly to<sup>16</sup>, (10) can be solved with two alternating steps:

1. (Computation of the Koopman matrix) fix  $\mathbf{W}_1$  and optimize  $\mathbf{K}$  ;
2. (Dictionary learning) fix  $\mathbf{K}$  and optimize  $\mathbf{W}_1$ .

A key advantage is that *both* optimization steps rely on *linear* optimization. It is clear that step 1 is a least squares problem, whose solution is given by

$$\mathbf{K} = \mathbf{W}_{\text{out}} \mathbf{S}' (\mathbf{W}_{\text{out}} \mathbf{S})^+ = \mathbf{W}_{\text{out}} \mathbf{S}' \mathbf{S}^+ \mathbf{W}_{\text{out}}^+. \quad (11)$$

In fact, this is the standard EDMD problem  $\mathbf{K} = \mathbf{\Psi}' \mathbf{\Psi}^+$  in the case of our specific choice of dictionary functions  $\mathbf{\Psi} = \mathbf{W}_{\text{out}} \mathbf{S}$ . We can also note that the matrix  $\mathbf{S}' \mathbf{S}^+$  in (11) is the Koopman matrix computed in the first method. The optimization over the output weight matrix  $\mathbf{W}_{\text{out}}$  somehow acts as a coupling in the optimization problem, where the columns of the Koopman matrix are optimized simultaneously in order to minimize the overall residue in (10).

Step 2 can also be cast into a least squares problem. Denoting

$$\mathbf{K} = \begin{pmatrix} \mathbf{K}_{11} & \mathbf{K}_{12} \\ \mathbf{K}_{21} & \mathbf{K}_{22} \end{pmatrix},$$

we can write problem (10) as the equation

$$\begin{aligned} \mathbf{W}\mathbf{S}' - \mathbf{K}\mathbf{W}\mathbf{S} &= \begin{pmatrix} \mathbf{W}_1\mathbf{S}' \\ \mathbf{W}_2\mathbf{S}' \end{pmatrix} - \begin{pmatrix} \mathbf{K}_{11}\mathbf{W}_1\mathbf{S} + \mathbf{K}_{12}\mathbf{W}_2\mathbf{S} \\ \mathbf{K}_{21}\mathbf{W}_1\mathbf{S} + \mathbf{K}_{22}\mathbf{W}_2\mathbf{S} \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{W}_1\mathbf{S}' \\ \mathbf{0}_{K,T-1} \end{pmatrix} - \begin{pmatrix} \mathbf{K}_{11} \\ \mathbf{K}_{21} \end{pmatrix} \mathbf{W}_1\mathbf{S} - \begin{pmatrix} \mathbf{K}_{12}\mathbf{W}_2\mathbf{S} \\ -\mathbf{W}_2\mathbf{S}' + \mathbf{K}_{22}\mathbf{W}_2\mathbf{S} \end{pmatrix} \quad (12) \\ &= \mathbf{0}_{D,T-1}. \end{aligned}$$

Denoting the independent terms  $\mathbf{C}_1 = \mathbf{K}_{12}\mathbf{W}_2\mathbf{S}$  and  $\mathbf{C}_2 = -\mathbf{W}_2\mathbf{S}' + \mathbf{K}_{22}\mathbf{W}_2\mathbf{S}$ , we compute the optimal output weights  $\mathbf{W}_1$  as the least squares solution given by

$$\mathbf{W}_1(\cdot) = \begin{bmatrix} (\mathbf{S}')^T \otimes \mathbf{I}_L - \mathbf{S}^T \otimes \mathbf{K}_{11} \\ -\mathbf{S}^T \otimes \mathbf{K}_{21} \end{bmatrix}^+ \begin{bmatrix} \mathbf{C}_1(\cdot) \\ \mathbf{C}_2(\cdot) \end{bmatrix}, \quad (13)$$

where  $(\cdot)$  denotes the vectorization of matrices and  $\otimes$  is the Kronecker product.

A variant of the above numerical scheme can also be obtained. Instead of computing the least squares solution (13) in terms of output weights  $\mathbf{W}_1$ , we could replace them by the reservoir outputs  $\Psi_1 = \mathbf{W}_1\mathbf{S}$  in (12). Using  $\mathbf{W}_1 = \Psi_1\mathbf{S}^+$ , we obtain the Sylvester equations

$$\begin{aligned} \Psi_1\mathbf{S}^+\mathbf{S}' - \mathbf{K}_{11}\Psi_1 &= \mathbf{C}_1 \\ -\mathbf{K}_{21}\Psi_1 &= \mathbf{C}_2, \end{aligned}$$

whose least squares solution is

$$\Psi_1(\cdot) = \begin{bmatrix} (\mathbf{S}^+\mathbf{S}')^T \otimes \mathbf{I}_L - \mathbf{I}_{T-1} \otimes \mathbf{K}_{11} \\ -\mathbf{I}_{T-1} \otimes \mathbf{K}_{21} \end{bmatrix}^+ \begin{bmatrix} \mathbf{C}_1(\cdot) \\ \mathbf{C}_2(\cdot) \end{bmatrix}. \quad (14)$$

In the case  $\bar{N} \geq T - 1$ , it is noticeable that  $\mathbf{S}^+\mathbf{S}'$  is a companion matrix, so that only the last column has to be effectively computed. When  $\bar{N} > T - 1$ , optimizing over  $\Psi_1 = \mathbf{W}_1\mathbf{S}$  in the variant of the method makes the optimization problem more constrained, since the basis functions are not described anymore as linear combinations of the state reservoirs but through their values at a smaller number  $T - 1 < \bar{N}$  of sample points. Conversely, the case

$\bar{N} < T - 1$  is a relaxation of the optimization problem which is less computationally efficient since  $\mathbf{S}^+\mathbf{S}'$  is not sparse in this case. The intermediate setting  $\bar{N} = T - 1$ , where  $\mathbf{S}$  is a square matrix and  $\boldsymbol{\Psi}_1 = \mathbf{W}_1\mathbf{S}$ , can be interpreted as a change of variable. It appears as an appropriate choice, but the size of the reservoir becomes forced by the size of the dataset.

Both equations (13) and (14) provide an effective way to solve the second step of the alternating optimization, yielding two variants of our proposed second method. The first variant provides the output weights describing the dictionary functions, while the second variant provides the updated values of the dictionary functions, which can directly be used to compute the Koopman matrix  $\mathbf{K}$ . Both variants require to invert a matrix of  $D(T-1) \times \bar{N}L$  and  $D(T-1) \times (T-1)L$ , respectively. This matrix is not sparse for the first variant, but is sparse for the second variant (provided that  $\bar{N} \geq T - 1$ ). It follows that the second variant is more efficient from a computational point of view.

The method with its two variants is summarized in Algorithm 2.

---

**Algorithm 2** Dictionary learning for EDMD using a reservoir computer

---

**Input:** Sampled trajectory  $\mathbf{x}(t)$ ,  $t = 1, \dots, T$ ; parameters  $N$ ,  $L$ ,  $C$ ,  $a$ ,  $\gamma$ ,  $\rho$ ,  $w_{in}$ ,  $\varepsilon$ .

**Output:** Koopman matrix  $\mathbf{K}$ , output weights  $\mathbf{W}_1$

- 1: Initialize  $\mathbf{W}_{in}$ ,  $\mathbf{W}$  (see Section II C) and set random initial reservoir states and output weights  $\mathbf{W}_1$ .
  - 2: Update the reservoir states according to (5) with the input  $\mathbf{u}(t) = \mathbf{x}(t)$ , for  $t = 1, \dots, T - 1$ .
  - 3: Compute the dictionary values  $\boldsymbol{\Psi} = [\boldsymbol{\Psi}_1; \boldsymbol{\Psi}_2]$  with  $\boldsymbol{\Psi}_1 = \mathbf{W}_1\mathbf{S}$  and  $\boldsymbol{\Psi}_2 = [\mathbf{0}_{K,N}, \mathbf{I}_K]\mathbf{S}$  (possibly discarding the first values of the states  $\bar{\mathbf{s}}$ ).
  - 4: **loop**
  - 5:   Solve step 1: compute the Koopman matrix  $\mathbf{K} = \mathbf{W}_{out}\mathbf{S}'\mathbf{S}^+\mathbf{W}_{out}^+$  (variant 1) or  $\mathbf{K} = \boldsymbol{\Psi}'\boldsymbol{\Psi}^+$  (variant 2).
  - 6:   Solve step 2: update the values  $\mathbf{W}_1$  using (13) (variant 1) or  $\boldsymbol{\Psi}_1$  using (14) (variant 2).
  - 7:   Stop if some criterion is satisfied (e.g. upper bound on the number of iterations, lower bound on the least squares error).
  - 8: If variant 2 is used: compute the output weights  $\mathbf{W}_1 = \boldsymbol{\Psi}_1\mathbf{S}^+$ .
-

## C. Application to prediction and spectral properties

### 1. Reconstruction and prediction

The Koopman matrix  $\mathbf{K}$  provided by Algorithm 1 or 2 is optimized so that

$$\forall t \in \{1, \dots, T-1\} : \Psi(t+1) \approx \mathbf{K} \Psi(t).$$

It follows that we can iterate the matrix to recompute known dictionary values  $\hat{\Psi}(t+1) = \mathbf{K}^t \Psi(1)$  (reconstruction) or predict new values  $\hat{\Psi}(t+T) = \mathbf{K}^t \Psi(T)$  from the last data point (prediction). In particular, predicted states are obtained by considering the values of the dictionary functions related to the projection maps.

It should be noted that our use of the reservoir computer differs from the classic use in the context of prediction and is not aimed at this specific prediction objective. Here the outputs are not optimized so that they provide the best predictions of the state. Instead they provide the basis functions that yield the most accurate approximation of the Koopman operator, which can in turn be used for prediction as a by-product. This observation is particularly relevant to Method 2 and will be discussed with more details in Section IV.

In order to test the quality of the Koopman matrix approximation, we will first compute the optimization residue

$$\sum_{t=1}^T \left\| \psi(\mathbf{x}'_t) - \tilde{\mathbf{K}} \psi(\mathbf{x}_t) \right\|^2. \quad (15)$$

We will also consider the reconstruction error

$$\mathbf{E}(t) = \mathbf{K}^t \Psi(1) - \Psi(t+1) \quad (16)$$

and in particular the Normalized Root Mean Square Error (NRMSE)

$$\mathbf{NRMSE} = \frac{\sqrt{\sum_{t=1}^{T-1} \mathbf{E}(t)^2}}{\sqrt{\sum_{t=1}^{T-1} \left[ \Psi(t+1) - \frac{1}{T-1} \sum_{\tau=1}^{T-1} \Psi(\tau+1) \right]^2}}$$

where the square operations, the square roots and the quotient are considered element-wise. The NRMSE value can be interpreted as follows:  $\mathbf{NRMSE} = 0$  means that the two series perfectly match and  $\mathbf{NRMSE} = 1$  is the error obtained when the reconstructed time serie is a constant value equal to the mean value of the other one. Similarly we will denote by  $\mathbf{nrmse}$  the error restricted to the projection maps.

## 2. *Spectral properties*

The Koopman matrix  $\mathbf{K}$  can be used to compute spectral properties of the Koopman operator  $\mathcal{K}^{39}$ . In particular, the eigenvalues of  $\mathbf{K}$  provide an approximation of the Koopman operator spectrum. Its left eigenvectors  $\mathbf{w}$  provide the expansion of Koopman eigenfunctions  $\phi$  in the basis given by the dictionary functions, *i.e.*  $\phi \approx \mathbf{w}^T \boldsymbol{\psi}$ . Note also that the right eigenvectors are related to the Koopman modes.

## IV. RESULTS

In this Section, we illustrate the performance of our methods with several datasets in the context of trajectory reconstruction, prediction, and computation of spectral properties.

### A. Datasets and parameters

#### 1. *Dynamical systems and data generation*

We consider several systems, including chaotic dynamics.

*a. Van der Pol system.* Using the limit-cycle dynamics

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= \mu(1 - x_1^2)x_2 - x_1\end{aligned}\tag{17}$$

with  $\mu = 1$ , we have generated  $T = 501$  data points over the time interval  $[0, 20]$  (time step  $h = 0.04$ ) for the initial condition  $\mathbf{x}(0) = (-4, 5)$ .

*b. Duffing system.* The dynamics

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\gamma x_2 - (\alpha x_1^2 + \beta)x_1\end{aligned}\tag{18}$$

with  $\alpha = 1$ ,  $\beta = -1$ , and  $\gamma = 0.5$  admit a stable equilibrium at the origin. We have generated  $T = 501$  data points over the time interval  $[0, 20]$  (time step  $h = 0.04$ ) for the initial condition  $\mathbf{x}(0) = (-1.21, 0.81)$ .

c. *Mackey-Glass system.* We consider the following delayed equation<sup>20</sup>:

$$\dot{x} = \frac{\alpha x_\tau}{(1 + x_\tau^n)} - \beta x \quad (19)$$

where  $x_\tau = x(t - \tau)$  with  $\tau = 17$  and  $\alpha = 0.2$ ,  $\beta = 0.1$ , and  $n = 10$ . We have generated  $T = 501$  data points over the time interval  $[0, 500]$  (time step  $h = 1$ ) for the initial condition  $x(t < 0) = 0.1$ . Note that the system is integrated using the Matlab function `dde23`.

d. *Rössler system.* We have used the chaotic dynamics

$$\begin{aligned} \dot{x}_1 &= -x_2 - x_3 \\ \dot{x}_2 &= x_1 + \alpha x_2 \\ \dot{x}_3 &= \beta + (x_1 - \gamma)x_3 \end{aligned} \quad (20)$$

with  $\alpha = 0.1$ ,  $\beta = 0.1$ ,  $\gamma = 14$  to generate  $T = 601$  data points over the time interval  $[0, 300]$  (time step  $h = 0.5$ ) for the initial condition  $\mathbf{x}(0) = (2, 1, 5)$ .

e. *Lorenz-63 system.* Using the chaotic dynamics

$$\begin{aligned} \dot{x}_1 &= s(x_2 - x_1) \\ \dot{x}_2 &= rx_1 - x_2 - x_1x_3 \\ \dot{x}_3 &= x_1x_2 - bx_3 \end{aligned} \quad (21)$$

with  $s = 10$ ,  $r = 28$ ,  $b = 8/3$ , we have generated  $T = 751$  data points over the time interval  $[0, 15]$  (time step  $h = 0.02$ ) for the initial condition  $\mathbf{x}(0) = (3, 3, 19)$ .

**Remark 2.** *The data are scaled through a linear transformation that maps the minimum and maximum values of each state to  $-1$  and  $1$ , respectively. The values of the initial conditions are given before rescaling. The data that support the findings of this study are available from the corresponding author upon reasonable request.*

## 2. Parameter values

The number of basis functions is set to  $D = N + K = 1000$  and  $D = L + K = 15$  for the first and the second method, respectively. Note that the numbers  $N$  and  $L$  depend on

the state dimension  $K$ . We note that  $\tilde{N} = 1000 > T - 1$  for all study cases. Although this choice yields a more constrained optimization problem (for the second variant of the second method, see Section III B), it yields the best results in terms of reconstruction, prediction and spectral properties while remaining computationally efficient.

For the second method, the second variant is used and the number of iterations is limited to 20. In the examples, the EDMD method is also used for comparison purpose with a dictionary of  $N$  Gaussian radial basis functions  $\psi_k(\mathbf{x}) = e^{-\gamma\|\mathbf{x}-\hat{\mathbf{x}}_k\|^2}$  (with  $D = N + K = 1000$ ), where  $\hat{\mathbf{x}}_k$  is the center and with  $\gamma = 0.05$ .

In most cases, the reservoir parameters are kept constant for every systems. The spectral radius of the internal weights is set to  $\rho = 0.79$  for all cases. The leaking rate is set to  $a = 3$  for all systems except for the Mackey-Glass system where  $a = 1$ . The noise level in the reservoir is  $\varepsilon = 10^{-4}$ . The time constant is set to  $C = 0.45$  for the Van der Pol system and the Duffing system, and is set to  $C = 0.11$  for the Mackey-Glass system, the Rössler system, and the Lorenz-63 system.

## B. Reconstruction results

The Koopman matrix computed with the basis functions generated by the reservoir is efficient to reconstruct the trajectories. As shown in FIG. 2, small residues (15) are obtained with all the systems introduced above. Moreover, in each case, the proposed methods outperform the EDMD method using as many radial basis functions as there are internal states in the reservoir. We observe that the first method yields better results than the second method. This can be explained by the fact the second method uses a smaller number  $L < N$  of dictionary functions. We also note that the EDMD method is not able to reconstruct the trajectories of the Mackey-Glass system (residue larger than 1). This suggests that the classical EDMD method with Gaussian radial basis functions cannot capture a time-delayed dynamics, in contrast to the reservoir whose internal states can be seen as functions depending on delayed input values (see Section III A).

The proposed methods provide a Koopman matrix that can be iterated to reconstruct the trajectories from the initial state. This is illustrated by the *nrmse* value shown in TABLE I. We note that a larger error is obtained with the second method for the Rössler system and the Lorenz system, in which case reconstructed trajectories diverge after some

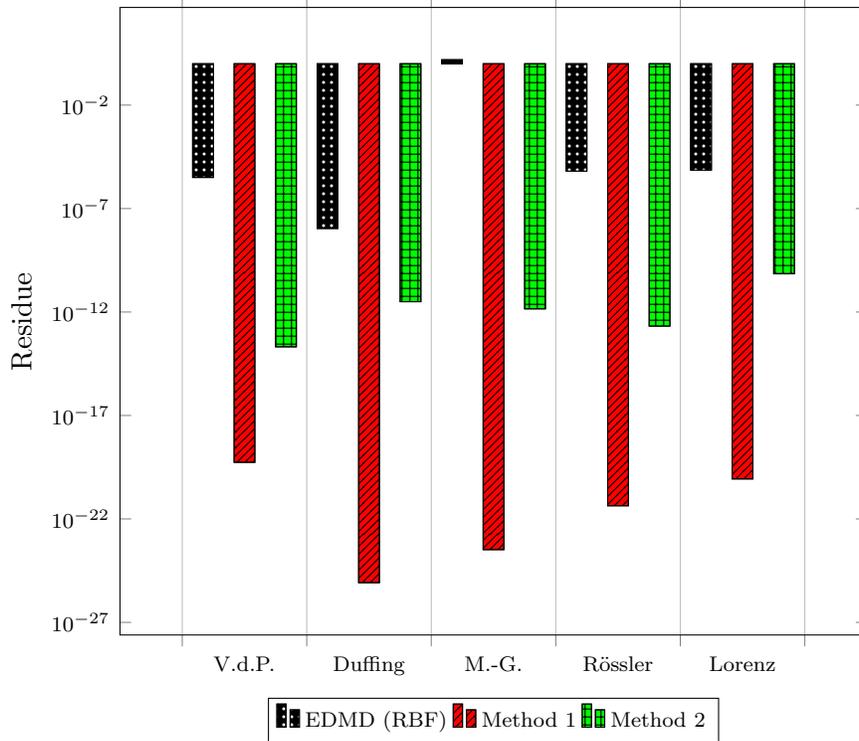


FIG. 2: The optimization residues of (4) are computed for the different systems and show that our methods 1 and 2 outperform EDMD.

System	V.d.P.	Duffing	M.-G.	Rössler	Lorenz
<b>EDMD (RBF)</b>	$7.39 \times 10^1$	$9.88 \times 10^{-7}$	$3.43 \times 10^0$	$9.32 \times 10^{-1}$	$9.02 \times 10^6$
<b>Method 1</b>	$8.21 \times 10^{-11}$	$5.32 \times 10^{-12}$	$5.36 \times 10^{-8}$	$2.80 \times 10^{-8}$	$3.40 \times 10^{-9}$
<b>Method 2</b>	$2.37 \times 10^{-1}$	$2.68 \times 10^{-1}$	$1.84 \times 10^{-1}$	$1.06 \times 10^0$	$1.17 \times 10^0$

TABLE I: The mean value of the *nrmse* vector components is shown for the different systems. The error vectors are computed according to (16) for the first 100 reconstructed points. Our methods 1 and 2 yield better performance.

time. However, in all cases, the proposed methods outperform the EDMD method used with Gaussian radial basis functions. Finally, FIG. 3 illustrates the reconstruction performances of the two methods for the Duffing system and the Mackey-Glass system.

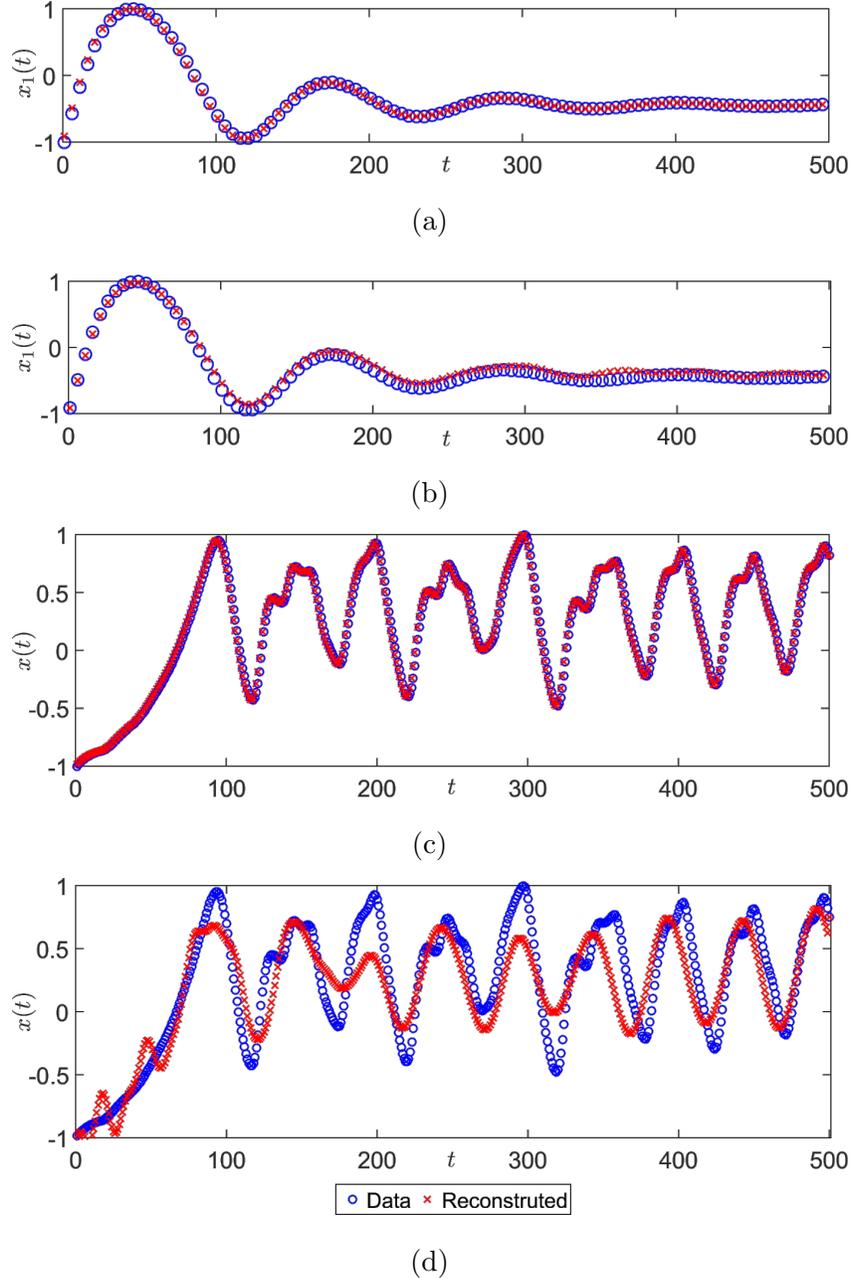


FIG. 3: The trajectories are reconstructed by iterating the Koopman matrix computed with Method 1 or 2 from the initial state. Blue circles and red crosses denote the data and the reconstructed trajectory, respectively (note that the all data points are not shown, so that the sampling period is smaller than it may seem on the figure). The first two panels show the first component of the Duffing system reconstructed by Method 1 (panel (a)) and Method 2 (panel (b)). The second component is not shown but is similar. The last two panels show the trajectory of the Mackey-Glass system reconstructed by Method 1 (panel (c)) and Method 2 (panel (d)).

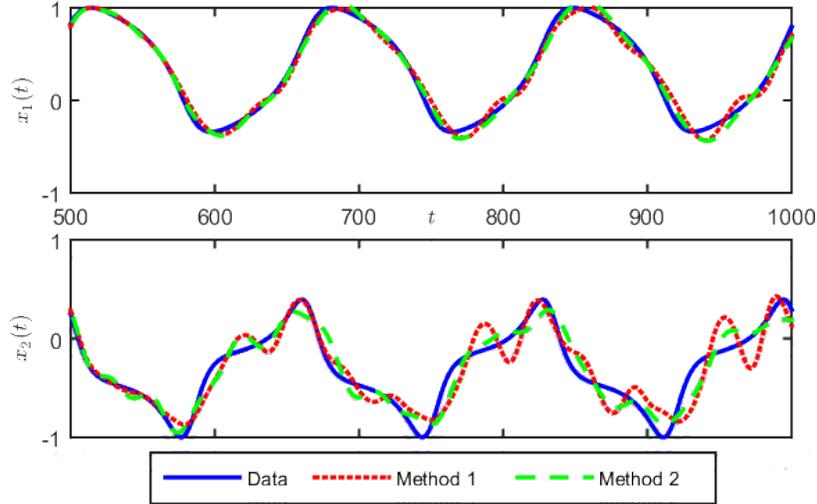


FIG. 4: Both methods correctly predict the trajectory of the Van der Pol oscillator.

### C. Prediction results

In this section, we briefly present prediction results for illustrative purposes. To do so, we consider the last data point of the training stage and iterate the Koopman matrix from this point.

*a. Van der Pol system.* As a first toy example, we consider the Van der Pol oscillator and verify that both methods correctly predict the trajectory (FIG. 4). For the second state variable, the prediction slowly diverges as the number of iterations of the Koopman matrix increases. Although the dynamics are very simple, it is noticeable that the classical EDMD method fails to provide good prediction results. In fact, for all study cases, the predicted trajectory either quickly diverges or converges to a constant. For this reason, we will not show the results obtained with the EDMD method.

*b. Mackey-Glass system.* FIG. 5 shows that the proposed methods are also efficient to predict the trajectory of the chaotic Mackey-Glass dynamics. In the present setting, the first method provides more accurate results for short-term prediction, but the predicted trajectory eventually diverges. In contrast the trajectory predicted with the second method converges to a constant which is close to the mean value of the data (see FIG. 6). This is due to the fact that the Koopman matrix does not have unstable eigenvalues and has only the eigenvalue  $\lambda = 1$  on the unit circle (see Section IVD below). It follows that the predicted trajectory converges to a steady state (associated with that eigenvalue  $\lambda = 1$ )

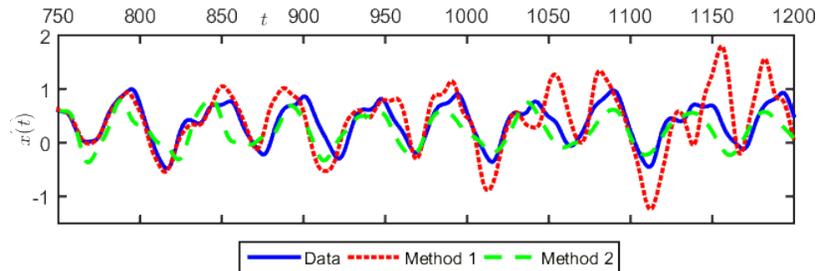


FIG. 5: *The two methods are efficient to predict the trajectory of the Mackey-Glass dynamics over some time horizon.*

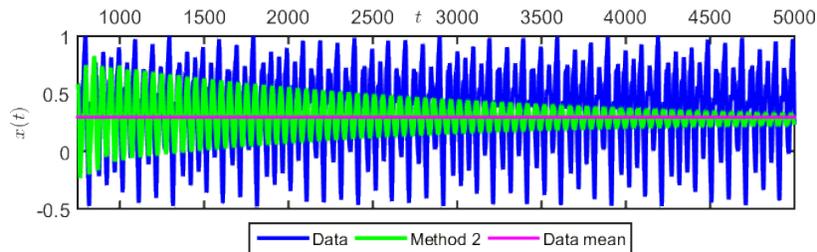


FIG. 6: *The long-term prediction of the second method for the Mackey-Glass system converges to a constant value close to the mean value of the data.*

which corresponds to the average value of the identity observable on the attractor, computed with respect to the stationary density. This value is also equal to the time average of the identity observable along a trajectory, which is close to the mean value of the data.

We finally recall that prediction is not the main goal of our methods. Although the prediction results are decent, successive iterations of the Koopman matrix may lead to divergent prediction errors and could be avoided to improve the prediction results (see Section IV E).

#### D. Spectral properties

In this Section, we compute an approximation of the spectrum of the Koopman operator for the Duffing system and the Rössler system. The results are shown in FIG. 7a and 7b, respectively. In both cases, the EDMD method generates many eigenvalues at the origin due to the rank-deficiency of the Koopman matrix. Our methods are characterized by less redundancy in the dictionary functions and, in particular, the second method provides a full-rank matrix. For the Rössler system, the EDMD method also generates spurious eigenvalues

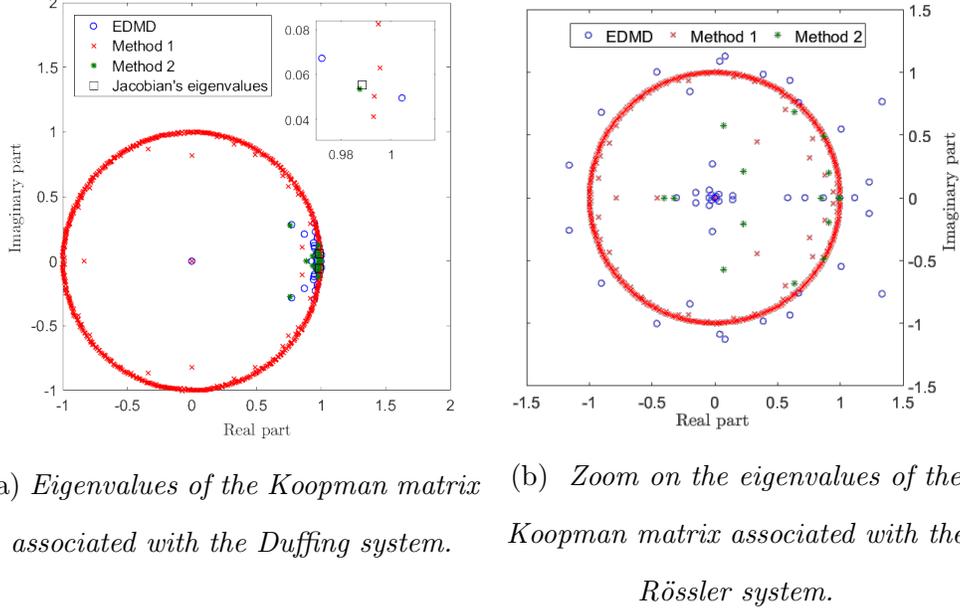


FIG. 7: *Computation of the spectrum of the Koopman operator for the Duffing system and the Rössler system.*

outside the unit circle (eigenvalues with module approximately equal to 10, not shown in the figure). This explains the fast divergence of the trajectories predicted with the EDMD method. In contrast, the first method recovers the whole unit circle, with a few additional eigenvalues inside the circle for both systems. The second method yields eigenvalues around 1 and inside the unit circle for the Duffing system. It yields more scattered eigenvalues inside the unit circle for the Rössler system. The inset in FIG. 7a shows that the eigenvalues of the Jacobian matrix at the stable fixed points are correctly recovered with the second method.

## E. Discussion

### 1. Comparison of the two methods

The first method provides the best reconstruction results since it exploits all the internal states of the reservoir rather than a few linear combinations of them. It should therefore be preferred if one aims at obtaining the most accurate matrix approximation of the Koopman operator. This method is also quite fast since there is no training process.

However the Koopman matrix obtained with the first method may be very large, since

the number of internal states of the reservoir computer is typically large. The second method is motivated by a tradeoff between the quality of the results and the size of the Koopman matrix at the cost of an additional computation time due to the dictionary training. It should be considered if one seeks for a low-dimensional approximation of the Koopman operator. In the context of prediction, it also seems that the second method provides slightly better results. While increasing the number of basis functions, the second method should produce results converging to those yielded by the first method. However the computation time also drastically increases so that the first method appears to be more relevant and efficient in this case. Similarly, if a large dataset is available, the second method might be computationally demanding because of huge matrices needed for the training.

## ***2. Strengths and weaknesses***

A main advantage of the proposed methods is that they rely solely on linear techniques thanks to the reservoir computer framework, in contrast to other Koopman operator-based learning techniques. Moreover, both reconstruction and prediction results obtained with these methods are improved with respect to the results obtained with the classical EDMD method. The Koopman spectrum computed with these methods is consistent and motivates the use of a trained set of basis functions. Finally we note that both methods require very little data to provide accurate results.

A main limitation of the second method is the computational cost of the dictionary training through the reservoir computer framework. We also note that both methods are not designed for prediction and cannot outperform state-of-the-art prediction methods. In particular the method fails to predict trajectories from initial conditions that are not related to the training set.

## ***3. Improving the prediction methods***

Our proposed methods mainly aim at computing the Koopman matrix with appropriate dictionary functions to provide the best global linear approximation of the dynamics. In the context of prediction, however, better results could be obtained with nonlinear approximations of the dynamics. We refer to other works proposing an efficient use of the reservoir

computer for prediction, i.e.<sup>1,28</sup>.

In the context of prediction with nonlinear approximations, we can also note that classical EDMD method could be used to compute a single iteration of the Koopman matrix, extract the updated value of the projection maps  $\boldsymbol{x}$ , and use them to evaluate the iterated values of all dictionary functions. This would allow to project back the predicted trajectory onto the manifold containing the lifted states (see also a similar idea in the work<sup>4</sup>). In fact, this amounts at computing the least squares projection of the system map  $\boldsymbol{F}$  in the span of the dictionary functions. Numerical simulations suggest that this method is very efficient in the context of prediction.

## V. CONCLUSION

We have proposed two novel methods for computing a finite-dimensional approximation of the Koopman operator. These methods combine classical EDMD with the use of a reservoir computer. In the first method, the dictionary functions are chosen to be the internal states of the reservoir. In the second method, the reservoir computer is trained and the dictionary functions are optimized linear combinations of internal states. A key advantage of these two methods is that they rely on linear optimization techniques. The accuracy of the Koopman matrix approximation is assessed in the context of reconstruction, prediction, and computation of the Koopman operator spectrum. The results are encouraging and pave the way to the use of the reservoir computer in the Koopman operator framework.

Several research perspectives can be proposed. First, the method could be improved to achieve better predictive performances, although this is not our main goal in this paper (note also that other Koopman operator-based methods exist for this purpose, see e.g.<sup>7,13,15</sup>). To do so, one could adapt the training of the reservoir according to this specific prediction objective, promote the computation of stable Koopman matrices, and use proper projections between the iterations of the Koopman matrix (see e.g.<sup>4</sup>). Our second method could also be complemented with convergence results for the alternating optimization scheme. Finally, the proposed methods could be used on real datasets, in the context of spectral analysis, network identification, time-series classification, event detection, and predictive control.

## VI. ACKNOWLEDGMENTS

This work is supported by the Namur Institute For Complex Systems (naXys) at University of Namur. M. Gulina is grateful to Adrien Fiorucci and Thomas Evrard for fruitful discussions and comments on the manuscript.

## DATA AVAILABILITY

The data that support the findings of this study are almost all available within the article. Any data that are not available can be found from the corresponding author upon reasonable request.

## REFERENCES

- <sup>1</sup>P. ANTONIK, M. GULINA, J. PAUWELS, AND S. MASSAR, *Using a reservoir computer to learn chaotic attractors, with applications to chaos synchronization and cryptography*, Phys. Rev. E, 98 (2018), p. 012215.
- <sup>2</sup>H. ARBABI AND I. MEZI, *Ergodic Theory, Dynamic Mode Decomposition, and Computation of Spectral Properties of the Koopman Operator*, SIAM Journal on Applied Dynamical Systems, 16 (2017), p. 20962126.
- <sup>3</sup>E. BOLLT, *On Explaining the surprising success of reservoir computing forecaster of chaos? The universal machine learning dynamical system with contrasts to VAR and DMD*, arXiv preprint arXiv:2008.06530, (2020).
- <sup>4</sup>D. BRUDER, B. GILLESPIE, C. D. REMY, AND R. VASUDEVAN, *Modeling and control of soft robots using the koopman operator and model predictive control*, arXiv preprint arXiv:1902.02827, (2019).
- <sup>5</sup>A. S. DOGRA AND W. T. REDMAN, *Optimizing neural networks via Koopman operator theory*, arXiv preprint arXiv:2006.02361, (2020).
- <sup>6</sup>Z. DRMA, I. MEZIC, AND R. MOHR, *Data driven modal decompositions: Analysis and enhancements*, SIAM Journal on Scientific Computing, 40 (2017).
- <sup>7</sup>D. GIANNAKIS, S. DAS, AND J. SLAWINSKA, *Reproducing kernel Hilbert space compactification of unitary evolution groups*, arXiv preprint arXiv:1808.01515, (2018).

- <sup>8</sup>J. H. TU, C. W. ROWLEY, D. M. LUCHTENBURG, S. L. BRUNTON, AND J. NATHAN KUTZ, *On dynamic mode decomposition: Theory and applications*, Journal of Computational Dynamics, 1 (2014), p. 391421.
- <sup>9</sup>B. HUANG, X. MA, AND U. VAIDYA, *Feedback stabilization using koopman operator*, in 2018 IEEE Conference on Decision and Control (CDC), 2018, pp. 6434–6439.
- <sup>10</sup>H. JAEGER, *The echo state approach to analysing and training recurrent neural networks*, GMD-Report 148, German National Research Institute for Computer Science, (2001).
- <sup>11</sup>H. JAEGER AND H. HAAS, *Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication*, Science, 304 (2004), pp. 78–80.
- <sup>12</sup>E. KAISER, J. N. KUTZ, AND S. L. BRUNTON, *Data-driven discovery of Koopman eigenfunctions for control*, arXiv preprint arXiv:1707.01146, (2017).
- <sup>13</sup>M. A. KHODKAR, P. HASSANZADEH, AND A. ANTOULAS, *A Koopman-based framework for forecasting the spatiotemporal evolution of chaotic dynamics with nonlinearities modeled as exogenous forcings*, arXiv preprint arXiv:1909.00076, (2019).
- <sup>14</sup>B. O. KOOPMAN, *Hamiltonian Systems and Transformation in Hilbert Space*, Proceedings of the National Academy of Sciences of the United States of America, 17 (1931), pp. 315–318. 16577368[pmid].
- <sup>15</sup>M. KORDA AND I. MEZI, *Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control*, Automatica, 93 (2018), pp. 149 – 160.
- <sup>16</sup>Q. LI, F. DIETRICH, E. M. BOLLT, AND I. G. KEVREKIDIS, *Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the Koopman operator*, Chaos: An Interdisciplinary Journal of Nonlinear Science, 27 (2017), p. 103111.
- <sup>17</sup>M. LUKOŠEVIČIUS, H. JAEGER, AND B. SCHRAUWEN, *Reservoir computing trends*, KI - Künstliche Intelligenz, 26 (2012), pp. 365–371.
- <sup>18</sup>M. LUKOŠEVIČIUS AND H. JAEGER, *Survey: Reservoir computing approaches to recurrent neural network training*, Comput. Sci. Rev., 3 (2009), p. 127149.
- <sup>19</sup>B. LUSCH, J. N. KUTZ, AND S. L. BRUNTON, *Deep learning for universal linear embeddings of nonlinear dynamics*, Nature Communications, 9 (2018), p. 4950.
- <sup>20</sup>M. MACKEY AND L. GLASS, *Oscillation and chaos in physiological control systems*, Science (New York, N.Y.), 197 (1977), p. 287289.
- <sup>21</sup>I. MANOJLOVI, M. FONOBEROVA, R. MOHR, A. ANDREJUK, Z. DRMA,

- Y. KEVREKIDIS, AND I. MEZI, *Applications of Koopman Mode Analysis to Neural Networks*, arXiv preprint arXiv:2006.11765, (2020).
- <sup>22</sup>A. MAUROY AND J. HENDRICKX, *Spectral identification of networks using sparse measurements*, SIAM Journal on Applied Dynamical Systems, 16 (2017), pp. 479–513. Article.
- <sup>23</sup>A. MAUROY AND I. MEZI, *Global stability analysis Using the eigenfunctions of the Koopman operator*, IEEE Transactions on Automatic Control, 61 (2016), pp. 3356–3369.
- <sup>24</sup>A. MAUROY, I. MEZIĆ, AND J. MOEHLIS, *Isostables, isochrons, and Koopman spectrum for the action–angle representation of stable fixed point dynamics*, Physica D: Nonlinear Phenomena, 261 (2013), pp. 19–30.
- <sup>25</sup>A. MAUROY, Y. SUSUKI, AND I. MEZIĆ, *The Koopman Operator in Systems and Control*, Springer, 2020.
- <sup>26</sup>I. MEZIĆ, *Spectral properties of dynamical systems, model reduction and decompositions*, Nonlinear Dynamics, 41 (2005), pp. 309–325.
- <sup>27</sup>S. PAN AND K. DURAISAMY, *Physics-informed probabilistic learning of linear embeddings of nonlinear dynamics with guaranteed stability*, SIAM Journal on Applied Dynamical Systems, 19 (2020), p. 480509.
- <sup>28</sup>J. PATHAK, A. WIKNER, R. FUSSELL, S. CHANDRA, B. R. HUNT, M. GIRVAN, AND E. OTT, *Hybrid forecasting of chaotic processes: Using machine learning in conjunction with a knowledge-based model*, Chaos: An Interdisciplinary Journal of Nonlinear Science, 28 (2018), p. 041101.
- <sup>29</sup>J. L. PROCTOR AND P. A. ECKHOFF, *Discovering dynamic patterns from infectious disease data using dynamic mode decomposition*, International health, 7 (2015), pp. 139–145. 25733564[pmid].
- <sup>30</sup>C. ROWLEY, I. MEZIC, S. BAGHERI, P. SCHLATTER, AND D. HENNINGSON, *Spectral analysis of nonlinear flows*, Journal of Fluid Mechanics, 641 (2009), pp. 115 – 127.
- <sup>31</sup>P. J. SCHMID, *Dynamic mode decomposition of numerical and experimental data*, Journal of Fluid Mechanics, 656 (2010), pp. 5–28.
- <sup>32</sup>E. SKIBINSKY-GITLIN, M. ALOMAR, C. FRASSER, V. CANALS, E. ISERN, M. ROCA, AND J. L. ROSSELLO, *Cyclic Reservoir Computing with FPGA Devices for Efficient Channel Equalization*, 01 2018, pp. 226–234.
- <sup>33</sup>A. SURANA, *Koopman operator framework for time series modeling and analysis*, Journal of Nonlinear Science, (2018).

- <sup>34</sup>Y. SUSUKI AND I. MEZI, *Nonlinear Koopman modes and power system stability assessment without models*, IEEE Transactions on Power Systems, 29 (2014), pp. 899–907.
- <sup>35</sup>——, *A prony approximation of Koopman Mode Decomposition*, in 2015 54th IEEE Conference on Decision and Control (CDC), 2015, pp. 7022–7027.
- <sup>36</sup>Y. SUSUKI AND I. MEZIĆ, *Nonlinear Koopman modes and coherency identification of coupled swing dynamics*, IEEE Transactions on Power Systems, 26 (2011), pp. 1894–1904.
- <sup>37</sup>N. TAKEISHI, Y. KAWAHARA, AND T. YAIRI, *Learning Koopman invariant subspaces for Dynamic Mode Decomposition*, Advances in Neural Information Processing Systems (Proc. of NIPS), 30 (2017).
- <sup>38</sup>F. TRIEFENBACH, A. JALALVAND, B. SCHRAUWEN, AND J.-P. MARTENS, *Phoneme recognition with large hierarchical reservoirs*, in Advances in Neural Information Processing Systems, J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, eds., vol. 23, Neural Information Processing System Foundation, 2010, p. 9.
- <sup>39</sup>M. O. WILLIAMS, I. G. KEVREKIDIS, AND C. W. ROWLEY, *A data-driven approximation of the Koopman operator: Extending Dynamic Mode Decomposition*, Journal of Nonlinear Science, 25 (2015), pp. 1307–1346.
- <sup>40</sup>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), 2019, pp. 4832–4839.