

Adapting reservoir computing to solve the Schrödinger equation

Cite as: Chaos 32, 063111 (2022); doi: 10.1063/5.0087785

Submitted: 9 February 2022 · Accepted: 17 May 2022 ·

Published Online: 3 June 2022



View Online



Export Citation



CrossMark

L. Domingo,^{1,2,3,a)} J. Borondo,^{4,5,b)} and F. Borondo^{1,2,c)}

AFFILIATIONS

¹Instituto de Ciencias Matemáticas (ICMAT), Campus de Cantoblanco UAM, Nicolás Cabrera, 13-15, 28049 Madrid, Spain

²Departamento de Química, Universidad Autónoma de Madrid, Cantoblanco, 28049 Madrid, Spain

³Grupo de Sistemas Complejos, Universidad Politécnica de Madrid, 28035 Madrid, Spain

⁴Departamento de Gestión Empresarial, Universidad Pontificia de Comillas ICADE, Alberto Aguilera 23, 28015 Madrid, Spain

⁵AgrowingData, Navarro Rodrigo 2 AT, 04001 Almería, Spain

^{a)}Email: laia.domingo@icmat.es

^{b)}Email: jborondo@gmail.com

^{c)}Author to whom correspondence should be addressed: f.borondo@uam.es

ABSTRACT

Reservoir computing is a machine learning algorithm that excels at predicting the evolution of time series, in particular, dynamical systems. Moreover, it has also shown superb performance at solving partial differential equations. In this work, we adapt this methodology to integrate the time-dependent Schrödinger equation, propagating an initial wavefunction in time. Since such wavefunctions are complex-valued high-dimensional arrays, the reservoir computing formalism needs to be extended to cope with complex-valued data. Furthermore, we propose a multi-step learning strategy that avoids overfitting the training data. We illustrate the performance of our adapted reservoir computing method by application to four standard problems in molecular vibrational dynamics.

Published under an exclusive license by AIP Publishing. <https://doi.org/10.1063/5.0087785>

Reservoir computing is a machine learning method that has proven very useful in predicting the evolution of time series, such as weather forecast, stock index, or the dynamics of chaotic systems. In this work, we adapt this machine learning algorithm to predict the evolution of quantum systems whose dynamics is determined by the Schrödinger equation. In this context, the quantum states are described using large complex-valued arrays, which makes the prediction difficult with classical reservoir computing mainly due to overfitting. For this reason, we propose a new reservoir computing-based methodology that allows us to efficiently propagate quantum states in time. We illustrate the method and its performance with several quantum systems.

I. INTRODUCTION

Recurrent neural networks (RNNs)^{1,2} are a deep learning tool used to deal with sequential data, such as text or time series. Within this context, RNNs have proven to give optimal results, due to their ability to keep the knowledge learned from past data. However,

RNNs are known to be computationally expensive during training, and such training may be complicated due to the problem of exploding gradients during backpropagation.³ To solve these problems, a new trend of RNN, called *Reservoir Computing (RC)*, was designed.

Indeed, Jaeger *et al.*^{4,5} proved that as long as the reservoir fulfills certain properties, only training the readout layer is enough to obtain excellent performance in many tasks. The concept of RC was proposed simultaneously as Echo State Networks (ESNs)⁴ in the field of machine learning and as Liquid State Machines (LSMs)⁶ in computational neuroscience. Actually, RC has proven useful for chaotic time series prediction,^{7,8} the underlying idea being that a well-trained reservoir is able to reproduce the attractor of the originating chaotic dynamics.

Recently, relevant advances have been made in developing RC-based algorithms. Multiple stacked reservoirs have been used to increase reservoir performance.^{9,10} Also, RC has been adapted to predict time series with short training data sets.¹¹ For predicting low dimensional time series, a RC algorithm requiring no random matrices and fewer hyperparameters has shown to be equivalent to standard RC.¹² Regarding ordinary differential equations (ODEs), in

Ref. 13, a closed-formula for the training parameters of RC to solve first order non-autonomous linear ODEs without backpropagation and a hybrid approach for non-linear ODEs, were derived. Furthermore, multiple physical implementations of RC have been recently proposed, by replacing the random neural network with a dynamical system. For example, in Ref. 14, the authors implemented RC on a scalable on-chip photonic linear processor, and in Ref. 15, the Schrödinger equation was used as the dynamical system. In the latter case, the trainable weights of the neural networks correspond to the physical quantities in the Schrödinger equation. Finally, in Ref. 16, a discrete soliton chain was used as a reservoir.

In the field of quantum chemistry, machine learning methods are proving to have advantages over the standard computational chemistry approaches,¹⁷ especially in the case of high-dimensional systems. In particular, machine learning has been used to solve the time-independent Schrödinger equation for the electronic^{18,19} and vibrational (ground state^{20–22} and high-energy states²³) parts of the total wavefunction within the Born–Oppenheimer approximation.

In this work, we aim to use RC to propagate wave packets with time, that is, to solve the time-dependent Schrödinger equation in continuous quantum systems. To do so, the RC model needs to be adapted to work with wavefunctions, which are complex-valued high-dimensional matrices. Complex-valued neural networks have been introduced in Ref. 24 for recurrent neural networks. Also, in Ref. 25, the authors proposed a different generalization of RC to complex values especially adapted to analyze synthetic aperture radar measurements. We also propose here a new learning strategy that allows propagating wavefunctions while reducing the overfitting of the training data. Such a learning strategy consists of a two-step training of the readout layer, where the reservoir is shown how predicting unseen data affects the evolution of the internal states. The learning algorithm is adapted to prevent fast error propagation during the test phase. A more detailed explanation of the algorithm is given in Sec. IV B.

The organization of this paper is as follows. In Sec. II, we review the original formulation of RC introduced in Ref. 4. In Sec. III, we present the quantum setting and the challenges that it presents to standard RC. Section IV describes the method we propose to overcome these challenges. Section V describes the four quantum vibrational systems that are used for illustration in this work. The corresponding results are presented in Sec. VI. Finally, Sec. VII ends the paper by summarizing the main conclusions of the present work and presenting an outlook for future work.

II. RESERVOIR COMPUTING

In the RC framework, the learning complexity of the algorithm is reduced to performing a linear regression. The key point is to design a dynamical system (the internal network) that learns the input–output dynamics. For this reason, the internal states of the network are called *echo states* since they can be thought of as an echo of their past.⁴ Such internal states are unambiguously determined by the input and output of the network. The structure of the network is shown in Fig. 1, where

- $W^{\text{in}} \in M(\mathbb{R})_{N \times K}$ gives the weights from the input units to the internal states,

- $W \in M(\mathbb{R})_{N \times N}$ gives the weights between the different internal states,
- $W^{\text{out}} \in M(\mathbb{R})_{L \times N}$ gives the weights from internal states to output units, and
- $W^{\text{back}} \in M(\mathbb{R})_{N \times L}$ gives the weights from the output units to the internal states,

and where

- $u(t) = (u_1(t), u_2(t), \dots, u_K(t))$ is a K -dimensional vector giving the input units at time t ,
- $x(t) = (x_1(t), x_2(t), \dots, x_N(t))$ is an N -dimensional vector giving the internal states at time t , and
- $y(t) = (y_1(t), y_2(t), \dots, y_L(t))$ is an L -dimensional vector giving the output units at time t .

Notice that the matrices W^{in} , W , and W^{back} are fixed so that they do not change during training. The only learnable parameters are the weights W^{out} . The steps to train the ESN are the following:

1. Generate the reservoir matrices (W , W^{in} , W^{back}) randomly.
2. Update the internal states by teacher forcing,

$$\begin{aligned}\tilde{x}(t) &= f(W^{\text{in}}u(t) + Wx(t-1) + W^{\text{back}}y_{\text{teach}}(t-1)), \\ x(t) &= (1 - \alpha)x(t-1) + \alpha\tilde{x}(t),\end{aligned}\quad (1)$$

where y_{teach} is the output that we want our network to predict, $\alpha \in (0, 1]$ is the leaking rate, and f is the activation function. Usually, $f(\cdot) = \tanh(\cdot)$, which is applied component-wise.

3. Discard a transient of t_{min} states to guarantee the convergence of the reservoir dynamics.
4. Find the readout matrix W^{out} by minimizing the mean square error,

$$MSE(y, y_{\text{teach}}) = \frac{1}{T - t_{\text{min}}} \sum_{t=t_{\text{min}}}^T \left[f^{\text{out}-1}(y_{\text{teach}}(t)) - W^{\text{out}}x(t) \right]^2, \quad (2)$$

where f^{out} is the output activation function (usually $f^{\text{out}} = Id$). Notice that this step only requires performing a linear regression.

The steps to make the predictions after training the network are the following:

1. Given an input $u(t)$, update the state of the reservoir,

$$\begin{aligned}\tilde{x}(t) &= f[W^{\text{in}}u(t) + Wx(t-1) + W^{\text{back}}y(t-1)], \\ x(t) &= (1 - \alpha)x(t-1) + \alpha\tilde{x}(t),\end{aligned}\quad (3)$$

where $y(t-1)$ is the prediction of the output at time $t-1$.

2. Compute the prediction $y(t)$,

$$y(t) = f^{\text{out}}[W^{\text{out}}x(t)]. \quad (4)$$

Let us now explain these steps in more detail. To ensure that the RC model works, the internal network must fulfill the *echo state properties*. Roughly speaking, this means that the internal states must have the state and input forgetting property. That is, for t large enough, $x(t)$ does not depend on $x(0)$, $u(0)$, or $y_{\text{teach}}(0)$. For this reason, in step 2 of the training phase, we dismiss the initial values of

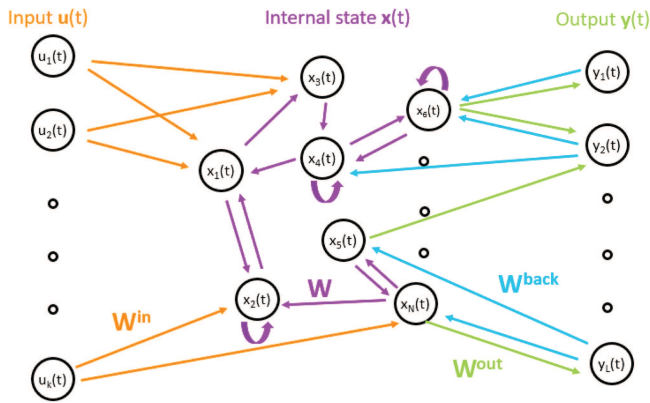


FIG. 1. Architecture of a reservoir computing model. Only the readout layer W^{out} is learnt during training.

the internal states, which could be influenced by the initial parameters of the reservoir. The spectral radius $\rho(W)$ of the internal matrix W also influences the performance of the method. A larger $\rho(W)$ provides longer memory. However, for $\rho(W) \gg 1$, the reservoir will likely have no echo states. In fact, it is a sufficient condition that $\rho(W) < 1$ to have echo states, even though reservoirs with $\rho(W)$ slightly larger than 1 may also give optimal results.⁴ Also, matrix W should be sparse and inhomogeneous so that the internal states contain a diverse set of trajectories. On the other hand, the leaking rate α influences the velocity of the dynamics of the output. A fast-changing output should be trained with greater values of α . However, the RC model is also frequently used without the leaky integration, which is a special case obtained by setting $\alpha = 1$ and, thus, $\tilde{x}(t) = x(t)$.

Once the internal states have been computed, the learning phase only consists of training a linear model to find the mapping W^{out} from the internal states to the output. This linear model is usually a ridge regression, which minimizes the following expression:

$$MSE_r(y, y_{\text{teach}}) = \frac{1}{T - t_{\min}} \sum_{t=t_{\min}}^T \left(j^{\text{out}-1}(y_{\text{teach}}(t)) - W^{\text{out}}x(t) \right)^2 + \gamma \|W^{\text{out}}\|^2. \quad (5)$$

This ridge regression is used to reduce overfitting during training. Other methods, such as adding noise to the input during the teacher forcing phase or adding a random constant input, can also be used to reduce overfitting. Finally, notice that the RC model can also be used without the input layer when we aim to predict a time series without explanatory variables. In this case, the term $W^{\text{in}}u(t)$ is removed from Eqs. (1) and (3).

III. RESERVOIR COMPUTING CHALLENGES IN QUANTUM PROBLEMS

A. The quantum setting

In this work, we aim to solve the time-dependent Schrödinger equation given by

$$i\hbar \frac{\partial \psi(\vec{x}, t)}{\partial t} = \hat{H} \psi(\vec{x}, t), \quad (6)$$

where \hat{H} is the Hamiltonian that describes the quantum system. The solution of Eq. (6), given an initial wavefunction $\psi(\vec{x}, 0)$, provides the time evolution of the associated quantum state. The propagation of a quantum state with certain (mean) energy allows the calculation, by Fourier transformation, of the eigenenergies and eigenfunctions of the quantum system around the same energy. Full details on how to obtain the eigenenergies and eigenfunctions from $\psi(\vec{x}, t)$ are provided in Sec. V. Notice that this method presents great advantages over the usual variational method in the case of excited states since it does not require the calculation of the low lying states.²³

In this work, we aim to develop a RC model that can be trained with the short-term evolution of a wave packet and then use RC to predict its longer-term evolution. The wavefunction $\psi(\vec{x}, t)$ is a complex-valued function of the spatial coordinates \vec{x} , which are usually multidimensional. In the RC framework, $\psi(\vec{x}, t)$ is represented as a set of matrices $\{\psi(\vec{x}, t)\}_t$, where each matrix contains the values of $\psi(\vec{x})$ at time t in a grid of points spanning \vec{x} . The use of $\psi(\vec{x}, t)$ as the target of the reservoir presents two challenges in the RC original formulation, which are described in Secs. III B and III C.

B. Complex numbers

The usual RC framework is done with real numbers. That is, all the inputs $u(t)$, outputs $y(t)$, internal states $x(t)$, and weight matrices W , W^{in} , W^{back} , and W^{out} are real-valued. However, the target time series for this quantum problem is a wavefunction $\psi(\vec{x}, t)$ which, in general, takes complex values. Therefore, we need to extend the update of the internal state and the learning algorithm to complex-valued data.

C. High-dimensional data

The target data are represented as a matrix, where each entry is the value of the wavefunction in a discretized spatial grid. The size of this matrix increases exponentially with the dimension of the physical system, that is, the dimension of \vec{x} . In the usual RC setting, we use a large reservoir W , compared to the dimension of the input. However, in the quantum problems, the target can have a very high dimension, and, thus, it is not feasible to design a reservoir much larger than the target size. Moreover, the bigger the reservoir, the more data are needed to train the linear model. If the complexity of the method is too large, the linear model may overfit the training data. Therefore, we need to adapt the RC framework to avoid this overfitting when dealing with high-dimensional data.

IV. RESERVOIR COMPUTING ADVANCES FOR QUANTUM DATA

This section presents the modifications done in the standard RC framework in order to adapt to the quantum setting.

A. Complex-valued ridge regression

We begin by presenting the extension of the RC framework to complex-valued arrays. The update of the internal state consists of matrix multiplications and an application of a non-linear function, in the following way:

$$\begin{aligned}\tilde{x}(t) &= f[(W^{\text{in}}u(t) + Wx(t-1) + W^{\text{back}}y_{\text{teach}}(t-1)], \\ x(t) &= (1 - \alpha)x(t-1) + \alpha\tilde{x}(t).\end{aligned}\quad (7)$$

As long as the function f is defined for complex numbers, the previous equation holds for complex-valued arrays. In this work, we propose the following activation function:

$$f(x) = \tanh(\Re(x)) + i \tanh(\Im(x)). \quad (8)$$

Also, we set $f^{\text{out}} = Id$, where Id is the identity function. Once the internal states have been calculated, the matrix W^{out} is calculated by minimizing the MSE with L^2 regularization. This corresponds to performing a complex-valued *ridge regression*,

$$\begin{aligned}MSE_r(y, y_{\text{teach}}) &= \frac{1}{T - t_{\min}} \sum_{t=t_{\min}}^T \left| f_{\text{out}}^{-1}(y_{\text{teach}}(t)) \right. \\ &\quad \left. - W^{\text{out}}x(t) \right|^2 + \gamma \|W^{\text{out}}\|^2,\end{aligned}\quad (9)$$

where $|\cdot|$ denotes the L^2 norm in complex values. In matrix form, this equation becomes

$$\begin{aligned}MSE_r(y, y_{\text{teach}}) &= (f_{\text{out}}^{-1}(\vec{y}_{\text{teach}}) - W^{\text{out}}X)^* \cdot (f_{\text{out}}^{-1}(\vec{y}_{\text{teach}}) - W^{\text{out}}X) \\ &\quad + \gamma \|W^{\text{out}}\|^2,\end{aligned}\quad (10)$$

where $*$ denotes the conjugate transpose and X is the matrix containing the internal states. In the case of real values, the conjugate transpose is just the transpose. This linear model has a closed solution

$$W^{\text{out}} = (X^*X + \alpha\mathbb{I})^{-1} \cdot (X^*f_{\text{out}}^{-1}(\vec{y}_{\text{teach}})), \quad (11)$$

and, therefore, an analytical solution for the linear model with complex-valued data can be computed.

B. Multiple-step training

The second and main challenge of dealing with quantum wavefunctions is the high-dimensionality of the target matrices. In order to capture all the underlying information of the dynamical system, the size of the reservoir W should be at least of the order of magnitude of the target data. According to Hoeffding's theorem,²⁶ models with high complexity require a large amount of data to reduce the

variance of the model predictions, following Hoeffding's inequality:

$$E_{\text{out}} \leq E_{\text{in}} + \mathcal{O}\left(\sqrt{\frac{C}{N}}\right), \quad (12)$$

where E_{out} is the error in the test set, E_{in} is the error in the training set, C is a notion of complexity, and N is the number of data samples. In our case, the size of the reservoir is similar to the number of data samples, and, thus, the method is likely to produce overfitting. To reduce this effect, we propose the use of a multi-step learning training. The training steps in this method, depicted schematically in Fig. 2, are the following:

1. Split the training input and output into two parts,

$$\begin{aligned}\vec{y} &= \begin{pmatrix} y(1) \\ \vdots \\ y(t_1) \\ y(t_1+1) \\ \vdots \\ y(T) \end{pmatrix} = \begin{pmatrix} y_1(1) \\ \vdots \\ y_1(t_1) \\ y_2(t_1+1) \\ \vdots \\ y_2(T) \end{pmatrix} = \begin{pmatrix} \vec{y}_1 \\ \vec{y}_2 \end{pmatrix}, \\ \vec{u} &= \begin{pmatrix} u(1) \\ \vdots \\ u(t_1) \\ u(t_1+1) \\ \vdots \\ u(T) \end{pmatrix} = \begin{pmatrix} u_1(1) \\ \vdots \\ u_1(t_1) \\ u_2(t_1+1) \\ \vdots \\ u_2(T) \end{pmatrix} = \begin{pmatrix} \vec{u}_1 \\ \vec{u}_2 \end{pmatrix}.\end{aligned}\quad (13)$$

2. Find the internal states of the reservoir $x(1), \dots, x(t_1)$ with \vec{y}_1 and \vec{u}_1 , and perform a ridge regression to find W^{out} .
3. Evolve the internal states of the reservoir $x(t_1+1), \dots, x(T)$ by letting the reservoir predict the values of \vec{y}_2 .
4. Using all the internal states $x(1), \dots, x(t_1), x(t_1+1), \dots, x(T)$ retrain the linear model to correct W^{out} . That is, perform again a ridge regression to find the *new* W^{out} using the *old* values of W^{out} as the initial guess for the coefficients of the linear model.

When the linear model overfits the training data, small changes in the internal states due to prediction errors produce high errors in the prediction of the wavefunctions. Therefore, the errors tend to propagate fast until the reservoir is not able to correctly propagate the quantum state. In our proposed method, we allow the reservoir to evolve after training by predicting a part of the training data. This simulates the real situation found during testing. During this predicting phase, the internal states are modified using the *predicted* output instead of the *exact* ones [see Eq. (3)]. The predicted outputs have some errors, which are transmitted to the internal states, which are then used to retrain the linear model. This learning strategy shows the reservoir how small predicting errors modify the internal states during the test phase. This reduces overfitting, decreasing the test error significantly.

V. STUDIED QUANTUM SYSTEMS

In this section, we describe the quantum mechanical calculations, and we also present the quantum systems chosen to be studied

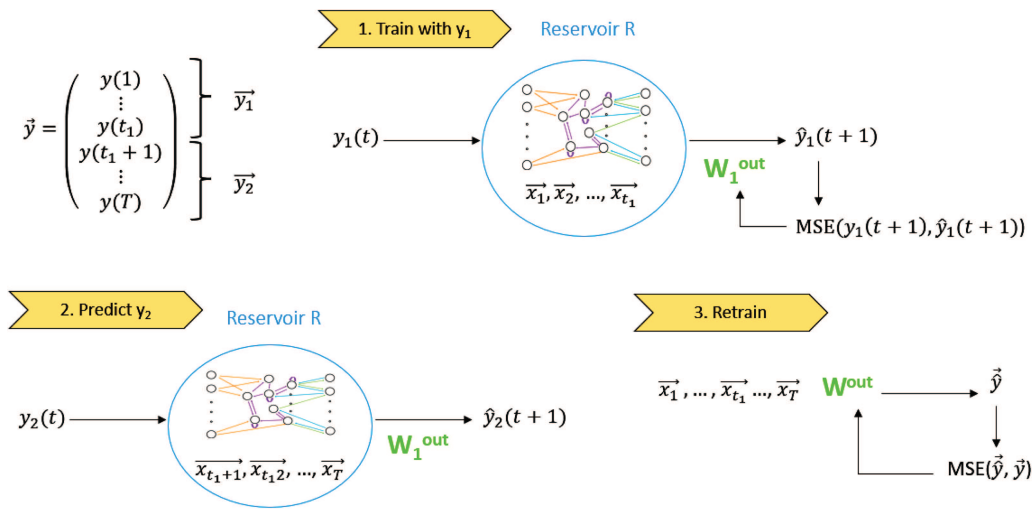


FIG. 2. Architecture of multi-step training of the reservoir computing model.

in this work to illustrate the performance of our multi-step RC method.

A. Quantum mechanical calculations

In all cases, the training procedure is the same, and it consists of the following steps:

1. Generate training and test data by numerically integrating the Schrödinger equation. For this purpose, we used the widespread Fast Fourier Transform (FFT) method proposed by Kosloff and Kosloff.²⁷
2. Train the RC and multi-step RC models. Then, compute the predicted wavefunctions $\psi(\vec{x}, t)$ and evaluate the errors in both cases.

From the time evolution $\psi(\vec{x}, t)$ of a suitable initial quantum state $\psi_0(\vec{x})$, the eigenenergies and eigenfunctions of the quantum system with similar energies to that of $\psi(\vec{x}, 0)$ can be calculated, by subsequent Fourier transform at the quantized energies. The eigenenergies and eigenstates completely describe the stationary

states of the quantum system. Therefore, the error of these eigenenergies gives another measure of the performance of our RC models. To obtain these eigenstates, we have to perform two additional steps.²⁸

3. Calculate the energy spectrum $I(E)$. First, we calculate the time-correlation function

$$A(t) = \langle \psi(t) | \psi(0) \rangle = \int_{\vec{x}} \psi(t)^* \psi(0) d\vec{x}. \quad (14)$$

Then, we obtain the energy spectrum $I(E)$ as the Fourier trans-

TABLE II. Mean squared error (MSE) for the wavefunctions $\psi(\vec{x}, t)$ and eigenenergies for the four quantum systems considered in this work (see Sec. V) using the standard RC model described in Secs. II and IV A and the multi-step RC model described in Secs. IV A and IV B. The table also contains the MSE for the LSTM, out benchmark model.

Quantum system	Model	MSE $\psi(\vec{x}, t)$	MSE energies
Harmonic oscillator 1D	Standard RC	4×10^{-5}	3×10^{-4}
	Multi-step RC	3×10^{-5}	1×10^{-5}
	LSTM	4×10^{-4}	2×10^{-2}
Morse	Standard RC	7×10^{-5}	4×10^{-5}
	Multi-step RC	2×10^{-5}	2×10^{-5}
	LSTM	4×10^{-2}	3×10^{-3}
Polynomial	Standard RC	2×10^{-4}	3×10^{-5}
	Multi-step RC	1×10^{-4}	8×10^{-6}
	LSTM	2×10^{-2}	5×10^{-3}
Harmonic oscillator 2D	Standard RC	1×10^{-4}	4×10^{-3}
	Multi-step RC	2×10^{-5}	8×10^{-5}
	LSTM	2×10^{-3}	2×10^{-2}

TABLE I. Reservoir computing training parameters, as defined in Sec. II, for the four studied quantum systems studied in this work (see Sec. V).

Quantum system	$\rho(W)$	α	t_{\min}	γ	N	W density
Harmonic oscillator 1D	1.10	0.015	300	0.1	2500	0.015
Morse potential	0.75	0.015	500	0.5	1500	0.015
Polynomial potential	0.75	0.017	50	0.05	1500	0.008
Harmonic oscillator 2D	1.10	0.06	300	0.5	2000	0.015

MSE 1D potentials

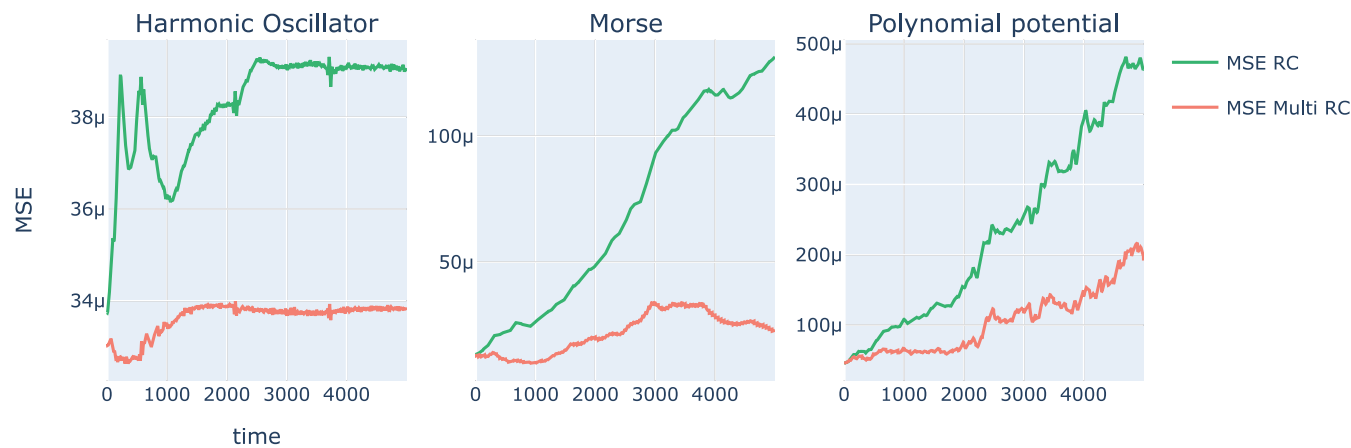


FIG. 3. Time evolution of the mean square error (MSE) for the harmonic, Morse, and polynomial oscillator time-dependent wavefunctions computed with the standard and multi-step reservoir computing methods.

form of the time-correlation function

$$I(E) = \int_t A(t) e^{iEt/\hbar} dt. \tag{15}$$

The eigenenergies E_n correspond to the peaks of the energy spectrum function.

2. Calculate the eigenfunctions $\phi_n(\vec{x})$ by performing the Fourier transform of $\psi(\vec{x}, t)$ at the eigenenergies,

$$\phi_n(\vec{x}) = \int_t \psi(\vec{x}, t) e^{-iE_n t/\hbar} dt. \tag{16}$$

3. Evaluate the errors produced by the RC and multi-step RC models in the eigenstates and eigenenergies.

Harmonic Oscillator 2D

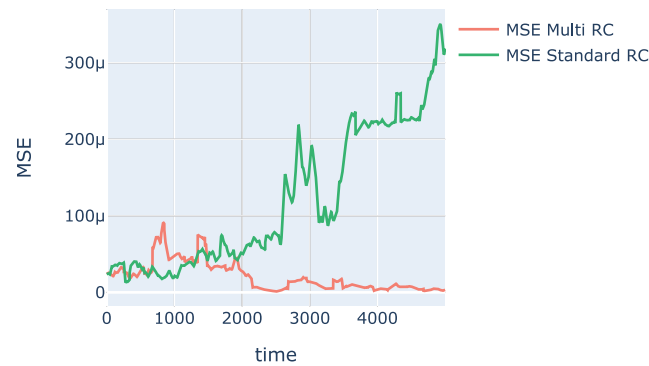


FIG. 4. Time evolution of the mean square error (MSE) of the 2D harmonic oscillator wavefunctions for the standard RC and the multi-step RC.

Now, we present the different quantum systems used for this study, which include three one-dimensional quantum systems and one two-dimensional system.

TABLE III. Training times for the four quantum systems considered in this work (see Sec. V). The standard RC and multi-step RC are the two RC-based model studied in this work. The LSTM and fast Fourier transform²⁷ are the two benchmark models: a standard deep learning model and standard ODE solver, respectively.

Quantum system	Model	Training time (min)
Harmonic oscillator 1D	Standard RC	7
	Multi-step RC	10
	LSTM	51
	FFT ²⁷	44
Morse	Standard RC	6
	Multi-step RC	7
	LSTM	40
	FFT	51
Polynomial	Standard RC	4
	Multi-step RC	6
	LSTM	36
	FFT	48
Harmonic oscillator 2D	Standard RC	10
	Multi-step RC	15
	LSTM	198
	FFT	66

Energy spectrum

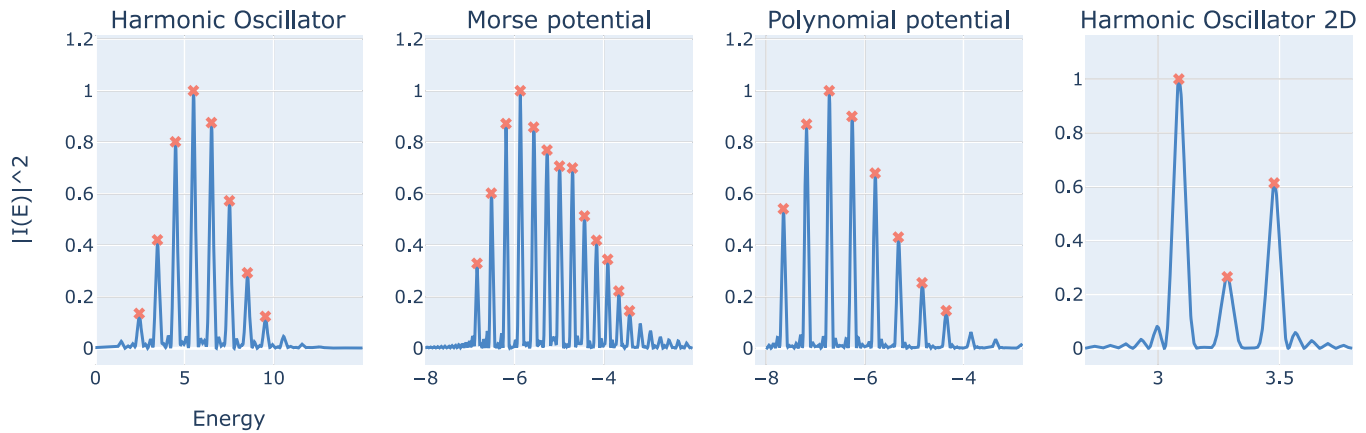


FIG. 5. Energy spectra obtained using the predicted wavefunctions from the multi-step RC for all four studied quantum systems. The orange peaks indicate the eigenenergies of each quantum system. Moreover, the numerical values of the eigenenergies are reported and given in Table IV.

B. 1D systems

We aim to solve the time-dependent Schrödinger equation (6) for 1D systems,

$$i\hbar \frac{\partial \psi(x, t)}{\partial t} = \hat{H} \psi(x, t). \quad (17)$$

In all cases, we set $\hbar = 1$ and $m = 1$ without loss of generality. We use 5000 time steps both for training and testing. The spatial domains are chosen so that they contain the wavefunctions at all times t . The systems under study are the following:

1. **Harmonic oscillator.** In this case, the Hamiltonian \hat{H} is

$$\hat{H}(x) = \frac{1}{2} \frac{\partial^2}{\partial x^2} + \frac{\omega^2}{2} (x - x_0)^2. \quad (18)$$

We choose $x_0 = 0$ and $\omega = 1$. The (computationally effective) spatial domain is $[-10, 10]$ covered with a grid of 200 equidistant points. That is, $\psi(x, t)$ for a fixed t is a vector of size 200. The time interval between time steps is $\Delta t = 0.002$. The initial wavefunction $\psi_0(x)$ is the minimum uncertainty Gaussian wave packet

$$\psi(x, 0) = \left(\frac{1}{\pi}\right)^{1/4} e^{-(x-x_0)^2/2} e^{ip_0 x}, \quad (19)$$

which is centered at $x_0 = 0$, and it has a width of $\Delta x = 1/(2\sqrt{2})$, a phase/momentum $p_0 = 3.35$, and a (mean) energy $E = 6.0$.

2. **Morse potential.** The Morse Hamiltonian adequately represents the potential interaction of a diatomic molecule. We write the corresponding Hamiltonian \hat{H} as

$$\hat{H}(x) = \frac{1}{2} \frac{\partial^2}{\partial x^2} + D_e (e^{-2a(x-x_0)} - 2e^{-a(x-x_0)}), \quad (20)$$

where x is the distance between atoms, x_0 is the corresponding equilibrium bond distance, D_e is the well depth (defined

relative to the dissociated atoms' energy value), and a is a parameter controlling the curvature of the potential at its minimum. We choose $D_e = 7$, $a = 0.09$, and $x_0 = 0$. The spatial domain is $[-10, 25]$ with 140 points. The time interval between time steps is $\Delta t = 0.007$. The initial wavefunction $\psi(x, 0)$ is again the minimum uncertainty packet (19) centered at $x_0 = 0$, with a phase $p_0 = 2$, and an energy $E = -4.8$.

3. **Polynomial potential.** The last quantum system considered corresponds to a quartic polynomial potential

$$\hat{H}(x) = \frac{1}{2} \frac{\partial^2}{\partial x^2} + \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \alpha_3 x^3 + \alpha_4 x^4, \quad (21)$$

where $\alpha_0 = -0.5$, $\alpha_1 = 0.14$, $\alpha_2 = 0.09$, $\alpha_3 = -0.01$, and $\alpha_4 = 0.001$. The spatial domain here is $[-15, 25]$ with 150 points. The time interval between time steps is $\Delta t = 0.007$. The initial wavefunction $\psi(x, 0)$ is again the minimum uncertainty packet

TABLE IV. Eigenenergies obtained from the spectra calculated from the predicted wavefunctions $\psi(\vec{x}, t)$ computed using the multi-step RC for all four studied quantum systems (see Sec. V).

Quantum system	Predicted eigenenergies
Harmonic oscillator 1D	2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5, 9.5
Morse	-6.8326, -6.5040, -6.1834, -5.8710, -5.5666, -5.2704, -4.9822, -4.7022, -4.4302, -4.1664, -3.9106, -3.6630, -3.4234, -3.1920
Polynomial potential	0.1556, 0.6187, 1.0800, 1.5426, 2.0087, 2.4801, 2.9582, 3.4453
Harmonic oscillator 2D	3.0946, 3.2838, 3.4730

Eigenstates Harmonic Oscillator

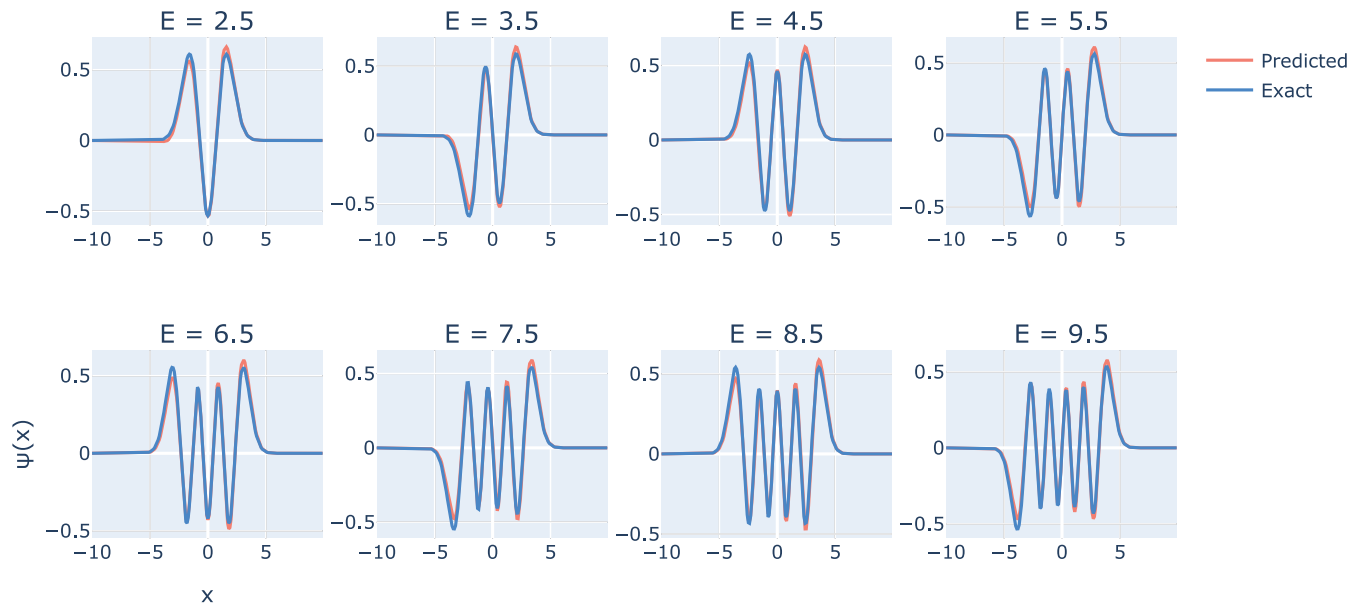


FIG. 6. Exact and predicted eigenfunctions for the harmonic oscillator system (18).

Eigenstates Morse

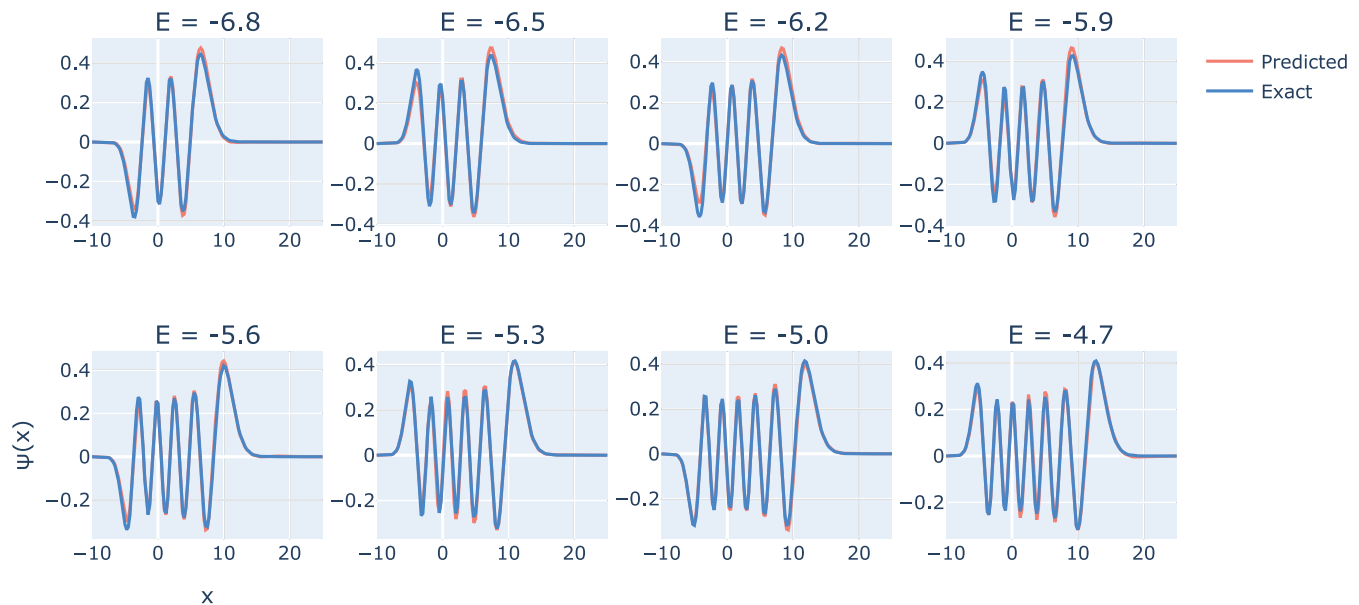


FIG. 7. Same as Fig. 6 for the Morse Hamiltonian system (20.)

Eigenstates Polynomial potential

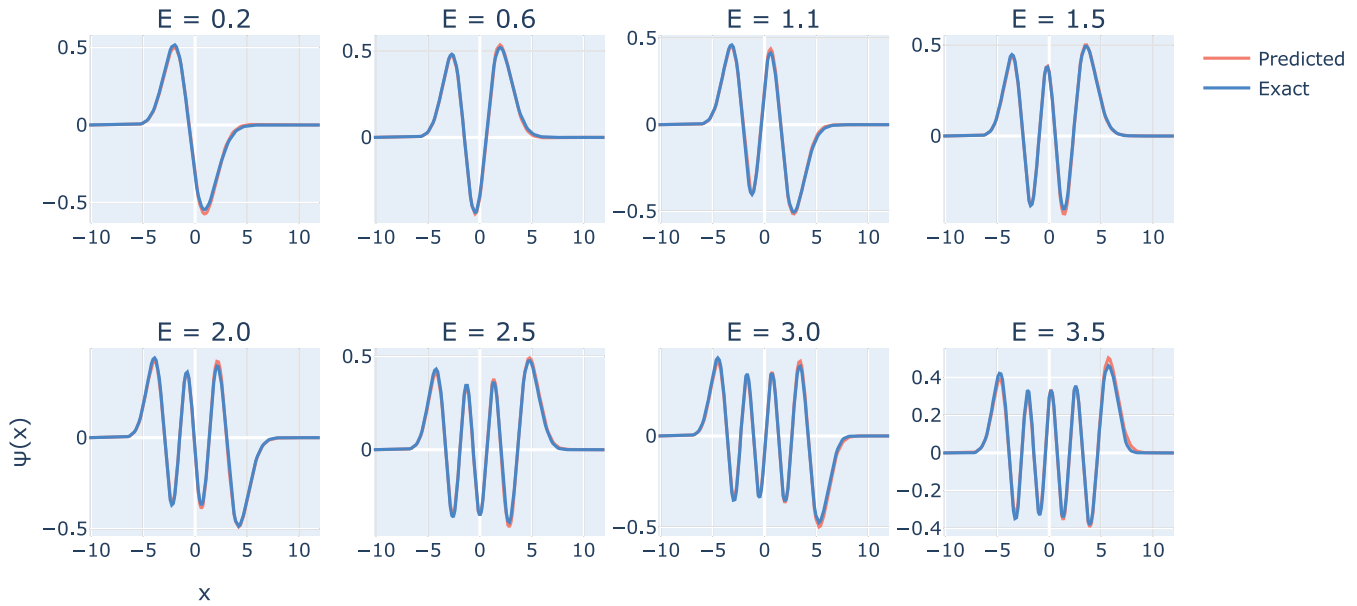


FIG. 8. Same as Fig. 6 for the polynomial Hamiltonian system (21).

(19) centered at $x_0 = 0$, with a phase $p_0 = 2$, and an energy $E = 1.8$.

C. 2D system

Apart from the 1D systems, we also study one 2D system, consisting of a 2D harmonic oscillator potential, whose Hamiltonian is given by

$$\hat{H}(x, y) = \frac{1}{2} \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \right) + \frac{\omega_x^2}{2} (x - x_0)^2 + \frac{\omega_y^2}{2} (y - y_0)^2. \quad (22)$$

We choose $\omega_x^2 = 1$, $\omega_y^2 = \sqrt{2}$, and $x_0 = y_0 = 0$. Notice that this choice of frequencies prevents the appearance of resonances in the dynamics and degeneracies in the eigenenergies. The initial wavefunction $\psi(x, y, 0)$ is the minimum uncertainty Gaussian wave packet with energy $E = 3.2$,

$$\psi_0(x, y, 0) = \left(\frac{1}{\pi} \right)^{1/8} e^{-(x-x_0)^2/(4\Delta x)^2 - (y-y_0)^2/(4\Delta y)^2} e^{ip_0(x-y)}, \quad (23)$$

which is centered at $x_0 = y_0 = 0$, has widths of $\Delta x = \Delta y = 0.9$, and a phase $p_0 = 1.75$. The spatial domain is $[-5, 5] \times [-5, 5]$ with 50 points in each dimension. Therefore, $\psi(x, y, t)$ for each t is a matrix of size 50×50 . Notice that the dimension of the data is of the order of the training data. Therefore, the RC model is very likely to produce overfitting, which motivates the multi-step learning we propose.

Table I shows the training parameters, as defined in Sec. II, used for each of the quantum systems considered in this work. The same parameter is used both for the standard RC and multi-step RC models. For all systems, we used $f(x) = \tanh(\Re(x)) + i \tanh(\Im(x))$ and $f^{\text{out}} = \mathbb{I}$. For the multi-step RC, we used 85% of the training data for the first training step and 15% for the second step. Even though a cumbersome machine learning model such as Bayesian optimization could have been used to select the hyperparameters, we consider that in this case, such methods are not necessary. We follow instead the criteria given in Ref. 29 to choose the appropriate training parameters. For example, the leaking rate α , which determines the velocity of the dynamics of the internal states, is chosen so that the internal states evolve with the same velocity as the input–output dynamics.

VI. RESULTS AND DISCUSSION

In this section, we discuss the results obtained with our multi-step RC method for the four models described in Sec. V, which are compared with those rendered by the standard RC model in order to obtain an estimation of its performance. In addition, a standard recurrent neural network, i.e., a long-short term memory neural network² is also used as a benchmark model. In particular, a LSTM with 1024 neurons has been trained for 1000 epochs, with the same training and test data as the other RC models.

In the first place, we show in Table II the mean square error (MSE) of the wavefunctions $\psi(\vec{x}, t)$ and (mean) energies, for the different systems chosen to study. Let us remark that these results constitute a very strict way of gauging the performance of our

Predicted Eigenstates Harmonic Oscillator

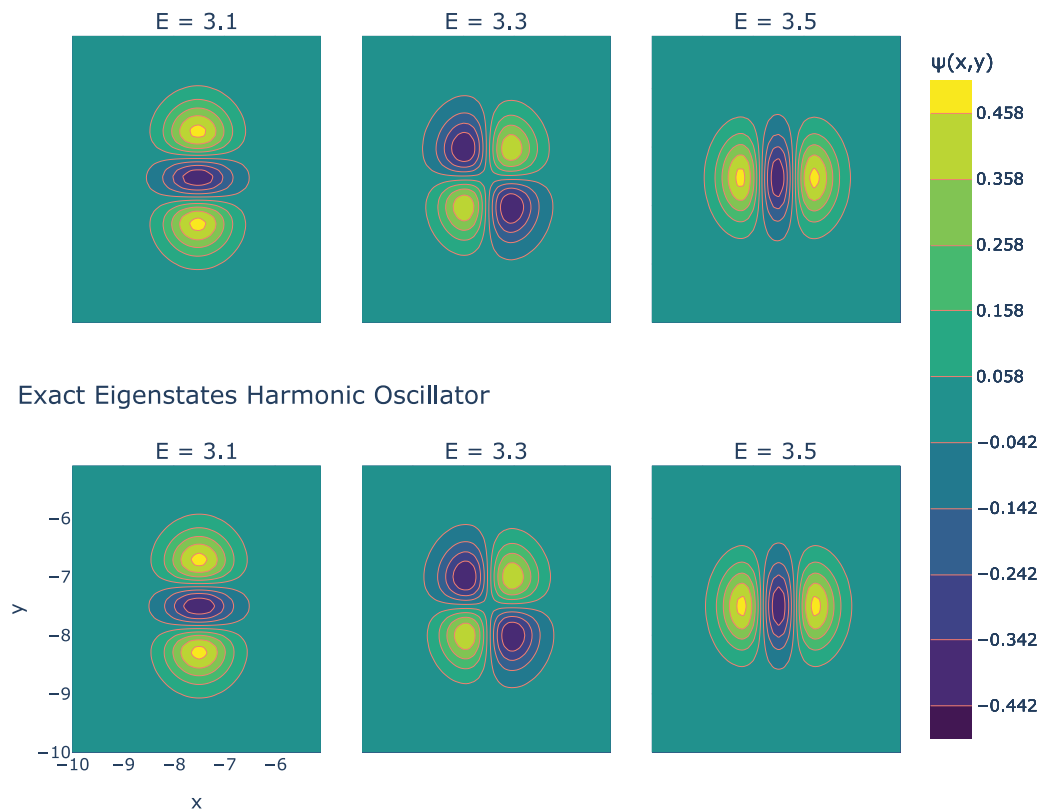


FIG. 9. Predicted (top row) and exact (bottom row) wavefunctions for the 2D harmonic oscillator system of Eq. (23).

method since the quantum properties are evaluated in all space using $\psi^* \psi$ as the probability density, which means that only the regions where the wavefunction is high are meaningful. We see that for all cases the MSE of both the wavefunctions and energies is smaller when using the multi-step RC model. The largest difference in performance appears in the 2D harmonic oscillator, where the MSE is one order of magnitude smaller for the wavefunctions and two orders of magnitude smaller for the energies. Therefore, only the multi-step learning RC algorithm can correctly recover the eigenfunctions and eigenenergies of such a quantum system. Notice that the training data for the 2D harmonic oscillator consist of matrices of size 50×50 (2500 entries), while the 1D data are vectors of size 100–200. Therefore, it is harder to train the RC model on 2D data, since the reservoir size is of the same order of magnitude as the amount of training data. Thus, the standard RC method overfits the training data and does not generalize well to the test data. On the contrary, the multi-step learning strategy refits the linear readout W^{out} by showing the reservoir how to adapt to unseen test data. This prevents having a fast error propagation during the test phase. Table II also shows the MSE of the LSTM for both wavefunctions and energies. We see that the MSE is much larger than the MSE of the RC-based models (one to three orders of magnitude). An

explanation for this fact is that the RC models only need to train the last layer of the network and, thus, have many less trainable parameters than the LSTM. For example, for the 1D harmonic oscillator, the LSTM has more than 12×10^6 trainable parameters, while the RC has 500 000 trainable parameters. For this reason, the LSTM takes longer to train and tends to overfit the training data, thus leading to large generalization errors.

In Figs. 3 and 4, we present the results of Table II for the RC-based models in a graphic way, by plotting the MSE of the predicted wavefunctions $\psi(\vec{x}, t)$ as a function of time. We see that due to error propagation, the MSE tends to increase with time, except for some random fluctuations. For all the studied systems, the MSE increases slowly with time when using the multi-step RC. Moreover, we see that for the 1D and 2D harmonic oscillators and the Morse Hamiltonian, the MSE from the multi-step RC seems to stabilize, while the MSE of the standard RC keeps increasing. Therefore, for a fixed error tolerance, the multi-step RC method allows us to predict a longer time evolution than the standard RC. Table III shows the training times for the three machine learning models studied in this work: the standard RC, the multi-step RC, and the LSTM. It also contains the training time of a standard ODE solver: the fast Fourier transform by Kosloff.²⁷ We see that the multi-step RC is

slightly slower than the standard RC but still much faster than both the LSTM and the FFT. Notice that the RC models do not impose the energy conservation of the solution of the Schrödinger equation, which is a fundamental property of any quantum system. However, during the training phase, the reservoir learns to reproduce the input–output dynamics, which follows this property. Since the predictions of the eigenenergies and eigenfunctions are accurate, the reservoir correctly imposes the energy conservation property during prediction, without being explicitly programmed to do so.

After studying the performance of the RC models when propagating the wavefunctions in time, we can recover the eigenenergies and eigenfunctions around a certain energy. Figure 5 shows the spectra for the different studied systems obtained using the predicted $\psi(\vec{x}, t)$ with the multi-step RC model. The peaks of the spectrum appear at the eigenenergies of each quantum system. The values of these energies are given in Table IV. These eigenenergies are close to the energy of the initial state $\psi(\vec{x}, 0)$ of the system. For example, the initial state of the 1D harmonic oscillator is a minimum uncertainty Gaussian wavepacket [see Eq. (19)] with mean energy $E = 6$. Accordingly, the obtained eigenenergies go from $E = 2.5$ to $E = 9.5$, so they are centered around the initial energy $E = 6$. Therefore, integrating the time-dependent Schrödinger equation allows recovering the eigenstates around certain energy, without needing to compute all the eigenstates with lower energy, as it happens in the usual variational method. Using the eigenenergies, we recover their associated eigenfunctions by computing the Fourier transform of the wavefunction $\psi(\vec{x}, t)$. Figures 6–8 show the predicted and exact eigenfunctions for the 1D harmonic oscillator, the Morse Hamiltonian and the polynomial potential systems, and Fig. 9 shows the predicted and exact eigenfunctions for the 2D harmonic oscillator. The exact eigenstates are calculated analytically for the harmonic oscillator (both 1D and 2D) and the Morse potential, and numerically (using the variational method²³) for the polynomial potential. We see that in all cases, the predicted eigenfunctions are in very good agreement with the exact ones. This fact confirms that the multi-step RC method can correctly propagate a quantum wavepacket with time. All in all, our results show that our adaptation of the RC method allows us to accurately integrate the time-dependent Schrödinger equation, and then obtain, also with high accuracy, the associated eigenenergies and eigenfunctions. This is possible thanks to the fact that our method helps prevent overfitting when working with high-dimensional data, thus, allowing the use of the adapted RC method for time propagation in quantum systems.

VII. CONCLUSIONS AND OUTLOOK

Despite being based on a quite simple training framework, RC has shown remarkable performance in various benchmark tasks, such as time series forecasting³⁰ and image recognition.³¹ Moreover, Pathak *et al.*^{7,8} demonstrated that ESN approaches are useful for chaotic time series prediction.

In this work, we have extended the RC framework to solve quantum problems. For this purpose, we propagate wavepackets with a certain energy and then performed Fourier transform to obtain the eigenenergies around the initial energy and their corresponding eigenfunctions. The use of RC computing to integrate the time-dependent Schrödinger equation presented two main

challenges. The first one is that the data [the quantum states $\psi(\vec{x}, t)$] is complex-valued. To overcome this problem, we proposed to use a complex-valued activation function. Also, we extended the regularized linear model to complex data by extending the ridge regression to the complex domain. The second challenge is derived from the high-dimensionality of the input–output data. These data are usually represented as a matrix containing the values of the wavefunctions in a spatial grid. The size of the matrix increases exponentially with the dimension of the quantum system. Therefore, we need large reservoirs in order to propagate the quantum states, and it can easily happen that the size of the reservoir is similar to or even larger than the number of training samples. In this case, the model is likely to produce overfitting, even when using a regularized model. When the model overfits the training data, it is unable to generalize to unseen data and errors in the predictions propagate fast. To reduce the effect of this problem, we propose the use of a multi-step learning algorithm to train the RC model. This algorithm consists of splitting the training data into two sets. We first train the reservoir with the first t_1 time steps. Then, we make predictions, updating the internal states of the reservoir, with the second part of the training data. In this way, we teach the reservoir how the predictions affect the evolution of its internal states. Last, we retrain the linear readout W^{out} . Since the reservoir has seen how predicting modifies its internal states, small changes to the internal states due to prediction errors will not propagate, leading to large errors in the predicted wavefunctions. This multi-step RC model is designed to work when the dimension of the input/output is larger or comparable to the size of the reservoir. Therefore, it could be useful to train RC models with any high-dimensional data, other than quantum wavefunctions.

As an illustration, we have applied our method to four quantum systems: three 1D systems and one 2D system. It is observed that the MSE of the propagated wavefunctions increases slowly with time when using multi-step learning. This fact is critical in the 2D system, which has higher-dimensional data, thus, leading to more overfitting. In this case, the standard RC model was not able to correctly reproduce the eigenenergies of the system, while the multi-step learning could. Moreover, the multi-step learning RC also allowed us to recover the eigenfunctions of all the quantum systems, proving that the method can correctly predict the time evolution of the wavepackets and the corresponding eigenstates.

Once the efficiency of our multi-step RC has been proved, the present work can be extended by application to other interesting problems. This future work could include, among others, the application to more complex and realistic quantum systems, the computation of eigenstates in a high lying energy window,³² or the calculation of the so-called scarred functions³³ that play a very important role in the field of quantum chaos.

ACKNOWLEDGMENTS

The project that gave rise to these results received support of a fellowship from “la Caixa” Foundation (No. 100010434). The fellowship code is LCF/BQ/DR20/11790028. This work has also been partially supported by the Spanish Ministry of Science, Innovation and Universities, Gobierno de España (Contract No. PGC2018-093854-BI00), ICMAT Severo Ochoa (No. CEX2019-000904-S), and by the People Programme (Marie Curie Actions)

of the European Union's Horizon 2020 Research and Innovation Program (Grant No. 734557).

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

DATA AVAILABILITY

The data that support the findings of this study are openly available in https://github.com/laiadc/RC_quantum, Ref. 34.

REFERENCES

- ¹D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations* (MIT Press, Cambridge, 1986), pp. 318–362.
- ²S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.* **9**, 1735 (1997).
- ³K. Doya, "Bifurcations in the learning of recurrent neural networks," *Proc. IEEE Int. Symp. Circuits Syst.* **6**, 2777 (2000).
- ⁴H. Jaeger, "The "echo state" approach to analysing and training recurrent neural networks-with an erratum note," German National Research Center for Information Technology GMD Technical Report, Vol. 148 (2001).
- ⁵H. Jaeger, "Echo state network," *Scholarpedia* **2**, 2330 (2007).
- ⁶W. Maass, T. Natschlager, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," *Neural Comput.* **14**, 2531 (2002).
- ⁷J. Pathak, Z. Lu, B. Hunt, M. Girvan, and E. Ott, "Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data," *Chaos* **27**, 121102 (2017).
- ⁸J. Pathak, B. Hunt, M. Girvan, Z. Lu, and E. Ott, "Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach," *Phys. Rev. Lett.* **120**, 024102 (2018).
- ⁹T. Akiyama and G. Tanaka, "Analysis on characteristics of multi-step learning echo state networks for nonlinear time series prediction," in *2019 International Joint Conference on Neural Networks (IJCNN)* (2019), pp. 1–8.
- ¹⁰J. Qiao, F. Li, H. Han, and W. Li, "Growing echo-state network with multiple subreservoirs," *IEEE Trans. Neural Netw. Learn. Syst.* **28**, 391 (2017).
- ¹¹P. Chen, R. Liu, K. Aihara, and L. Chen, "Autoreservoir computing for multistep ahead prediction based on the spatiotemporal information transformation," *Nat. Commun.* **11**, 486 (2020).
- ¹²D. Gauthier, E. Bollt, A. Griffith, and W. Barbosa, "Next generation reservoir computing," *Nat. Commun.* **12**, 78 (2021).
- ¹³M. Mattheakis, H. Joy, and P. Protopapas, "Unsupervised reservoir computing for solving ordinary differential equations," *arXiv/2108.11417* (2021).
- ¹⁴M. Nakajima, K. Tanaka, and T. Hashimoto, "Scalable reservoir computing on coherent linear photonic processor," *Commun. Phys.* **4**, 2399–3650 (2021).
- ¹⁵M. Nakajima, K. Tanaka, and T. Hashimoto, "Neural schrödinger equation: Physical law as deep neural network," in *IEEE Transactions on Neural Networks and Learning Systems* (IEEE, 2021), pp. 1–15.
- ¹⁶N. A. Silva, T. D. Ferreira, and A. Guerreiro, "Reservoir computing with solitons," *New J. Phys.* **23**, 023013 (2021).
- ¹⁷A. Ferguson, J. Hachmann, T. Miller, and J. Pfendtner, "Virtual special issue on machine learning in physical chemistry," *J. Phys. Chem. B* **124**, 9767 (2020).
- ¹⁸D. Pfau, J. Spencer, A. Matthews, and W. Foulkes, "Ab initio solution of the many-electron Schrödinger equation with deep neural networks," *Phys. Rev. Res.* **2**, 13890 (2020).
- ¹⁹J. Hermann, Z. Schätzle, and F. Noé, "Deep-neural-network solution of the electronic Schrödinger equation," *Nat. Chem.* **12**, 891 (2020).
- ²⁰S. Manzhos, K. Yamashita, and T. Carrington, "Using a neural network based method to solve the vibrational Schrödinger equation for H₂O," *Chem. Phys. Lett.* **474**, 217–221 (2009).
- ²¹A. Pavlov, J. Serdyuk, and A. Ustinov, "Machine learning and the Schrödinger equation," *J. Phys.: Conf. Series* **1236**, 012050 (2019).
- ²²K. Mills, M. Spanner, and I. Tamblin, "Deep learning and the Schrödinger equation," *Phys. Rev. A* **96**, 2825 (2017).
- ²³L. Domingo and F. Borondo, "Deep learning methods for the computation of vibrational wavefunctions," *Commun. Nonlinear Sci. Numer. Simul.* **103**, 105989 (2021).
- ²⁴C. Trabelsi, O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal, "Deep complex networks," *arXiv/1705.09792* (2017).
- ²⁵B. Konishi, A. Hirose, and R. Natsuaki, "Complex-valued reservoir computing for interferometric SAR applications with low computational cost and high resolution," *IEEE J. Select. Top. Appl. Earth Obs. Remote Sens.* **14**, 7981–7993 (2021).
- ²⁶M. Mohri, A. Rostamzadeh, and A. Talwalkar, *Foundations of Machine Learning*, 2nd ed. (The MIT Press, 2018).
- ²⁷D. Kosloff and R. Kosloff, "A Fourier method solution for the time dependent Schrödinger equation as a tool in molecular dynamics," *J. Comput. Phys.* **52**, 35 (1983).
- ²⁸P. Zdánková and N. Moiseyev, "Complex autocorrelation function and energy spectrum by classical trajectory calculations," *J. Chem. Phys.* **121**, 6175 (2004).
- ²⁹M. Lukoševičius, "A practical guide to applying echo state networks," in *Neural Networks: Tricks of the Trade: Second Edition*, edited by G. Montavon, G. B. Orr, and K.-R. Müller (Springer, Berlin, 2012), pp. 659–686.
- ³⁰R. Gao, L. Du, O. Duru, and K. F. Yuen, "Time series forecasting based on echo state network and empirical wavelet transformation," *Appl. Soft Comput.* **102**, 107111 (2021).
- ³¹A. Jalalvand, W. De Neve, R. Van de Walle, and J.-P. Martens, "Towards using reservoir computing networks for noise-robust image recognition," in *2016 International Joint Conference on Neural Networks (IJCNN)* (2016), pp. 1666–1672.
- ³²F. Revuelta, E. Vergini, R. M. Benito, and F. Borondo, "Short-periodic-orbit method for excited chaotic eigenfunctions," *Phys. Rev. E* **102**, 042210 (2020).
- ³³F. Revuelta, R. Benito, F. Borondo, and E. Vergini, "Using basis sets of scar functions," *Phys. Rev. E* **87**, 042921 (2013).
- ³⁴L. Domingo, J. Borondo, and F. Borondo, *Adapting reservoir computing to solve the Schrödinger equation*, https://github.com/laiadc/RC_quantum (2022).