Data-driven time propagation of quantum systems with neural networks

James Nelson,[1,*] Luuk Coopmans,[1,2,†] Graham Kells,[2,3,‡] and Stefano Sanvito[1,§]
[1]*School of Physics, AMBER and CRANN Institute, Trinity College, Dublin 2, Ireland*
[2]*Dublin Institute for Advanced Studies, School of Theoretical Physics, 10 Burlington Rd, Dublin, Ireland*
[3]*Dublin City University, School of Physical Sciences, Glasnevin, Dublin 9, Ireland*

(Received 27 January 2022; revised 17 May 2022; accepted 22 June 2022; published 1 July 2022)

We investigate the potential of supervised machine learning to propagate a quantum system in time. While Markovian dynamics can be learned easily, given a sufficient amount of data, non-Markovian systems are nontrivial and their description requires the memory knowledge of past states. Here we analyze the feature of such memory by taking a simple 1D Heisenberg model as many-body Hamiltonian, and construct a non-Markovian description by representing the system over the single-particle reduced density matrix. The number of past states required for this representation to reproduce the time-dependent dynamics is found to grow exponentially with the number of spins and with the density of the system spectrum. Most importantly, we demonstrate that neural networks can work as time propagators at any time in the future and that they can be concatenated in time forming an autoregression. Such neural-network autoregression can be used to generate long-time and arbitrary dense time trajectories. Finally, we investigate the time resolution needed to represent the system memory. We find two regimes: For fine memory samplings the memory needed remains constant, while longer memories are required for coarse samplings, although the total number of time steps remains constant. The boundary between these two regimes is set by the period corresponding to the highest frequency in the system spectrum, demonstrating that neural network can overcome the limitation set by the Shannon-Nyquist sampling theorem.

DOI: 10.1103/PhysRevB.106.045402

I. INTRODUCTION

The notion of a memory kernel arises frequently in the study of dynamical models, where one attempts to understand or recreate dynamical behaviours using only a subset of the total available degrees of freedom. In particular, a kernel becomes necessary in situations where the subset of information available at a given instant in time is not enough to predict the subsequent behavior. In open quantum systems, where a restricted number of degrees of freedom is in contact with a bath, one formally understands this subset restriction as the tracing out of the bath. Here, the need of a memory kernel to obtain accurate long-time trajectories is a defining criteria that distinguishes Markovian from non-Markovian dynamics.

A number of key techniques and approaches have emerged enabling a systematic study of this effect. One such family are the so-called projection techniques (e.g., the Nakajima-Zwanzig equation [1–3] and their time-convolutionless variants [4,5]) that render the history (memory) within an integro-differential equation and can be used to provide a perturbative justification for the Markovian behavior. Another approach for exploring memory effects is to embed the system into an enlarged Hilbert space such that the non-Markovian dynamics of the reduced system can be represented via the Markovian evolution of an enlarged whole [6–11]. In this respect, a useful class of embeddings are the so-called collision models, where ancillas representing the bath interact in a time-dependent fashion. Here the concept of memory depth naturally arises, when one allows these ancillas to interact between themselves.

Both the projection and embedding schemes represent intuitive and inherently human-learned/reasoned approaches for investigating the interplay between a quantum system and its associated bath. Given the now widespread application of machine learning (ML) in the domain of classical and quantum dynamics, it is natural that one might apply machine learning to the same domain. For a selective snapshot of this field we refer the reader to some representative published literature [12–18] and references therein.

When looking at the non-Markovianity of a time evolution, in particular with embeddings and memory kernels, an approach that has recently emerged is to construct a fixed neural network (NN) that learns the behaviours of local observables and then attempts to match the time dependence of the observables beyond the training time. For instance, in Ref. [19] this data-driven method was applied to learn time-local generators of the dynamics of open quantum systems. Similarly, in [20] a related approach was used to find an effective Markovian embedding from which several important properties and the exact dynamics of the reduced system could be extracted.

One of the key advantages of these data-driven methods is their universality. In fact, machine-learning time propagators can be used with any method to generate quantum evolution and show promise for use in investigations into the

*janelson@tcd.ie
†coopmanl@tcd.ie
‡gkells@stp.dias.ie
§sanvitos@tcd.ie

nature of the non-Markovian evolution. In this respect, the methodology blends features of Markovian embeddings and integro-differential memory kernels, but where the functional that encodes the dynamics is fixed so that the embeddings and the kernel/memory encoding are time or state independent.

In this paper we take a broadly similar approach in that we train a NN on a dataset of time-dependent quantum trajectories and then ask it to evolve an arbitrary starting state up to times longer than the final time of the learned trajectories. Distinguishing our paper from previous studies, we look at the complete collection of reduced single-particle density matrices that together make up the full (strongly interacting) system and focus, in particular, on the accuracy of the method with respect to the depth of the sampling history that is used as input for the NN. While in general, as we show below, the Markovian dynamics of the full system can be learned by only one history sample, for the representation in terms of reduced density matrices this is no longer the case. In fact, we find that the sampling history needs to be increased exponentially with system size to keep the error rates on the predicted dynamics below a fixed threshold. Aside from studying such memory effects, an advantage of this approach is that its malleability allows for nonfixed time sampling and naturally protects against oversampling error. Moreover, as the error growth on a given NN is random we can use the idea of neural-network ensembling to monitor and reduce errors.

We structure our paper as follows: In Sec. II we give an overview of various methods to study memory effects in quantum dynamics and introduce our specific setup and method to generate the data of the quantum evolution. In Secs. III and IV we present our results for Markovian and non-Markovian dynamics, respectively, discussing the interplay between the nonlocality in time and space. Then we conclude.

II. METHODOLOGY AND BACKGROUND

A. Brief review of existing methods

We consider a quantum system described by its density matrix $\hat{\rho}$ that consists of a total of N qubits. The reduced density matrices of such system can be obtained by tracing out selected degrees of freedom, here collectively called the *environment* $\hat{\rho}_{\rm red} = {\rm Tr}_{\rm E}\hat{\rho}$, where "E" stands for some environment (see Fig. 1). When the traced-out region is all but one qubit we use the parameterisation $\hat{\rho}_{\rm red} = \frac{1}{2}(\mathbb{1} + \mathbf{r}\cdot\sigma)$, where $\mathbf{r}$ is the Bloch vector. Since the reduced density matrices leave out much information they alone cannot determine the future evolution of the entire system. Thus, for a typical dynamical reconstruction of the reduced quantities over time one may also need some historical knowledge of past states. Several analytical and numerical techniques have been developed to study the memory effects in this type of setup.

Projection methods. A technique often mentioned in literature to render the past knowledge explicit in the time-evolution equations is the Nakajima-Zwanzig projector method [1–3] (see Appendix A for a quick derivation). When the time-evolution starts ($t = 0$) from an initially unentangled system and bath, $\hat{\rho}(t = 0) = \hat{\rho}_{\rm red} \otimes \hat{\rho}_{\rm E}$, this approach allows one to reduce the Liouville-Von Neumann equation for the entire system,

$$\partial_t \hat{\rho} = \frac{i}{\hbar}[\hat{\rho}, \hat{H}] = \hat{L}\hat{\rho}\,, \tag{1}$$

to an equation containing just the relevant reduced density operator $\hat{\rho}_{\rm red}$,

$$\partial_t[\hat{\rho}_{\rm red}] = \hat{\mathcal{P}}\hat{L}[\hat{\rho}_{\rm rel}] + \int_0^t dt'\hat{\mathcal{K}}(t')[\hat{\mathcal{P}}\hat{\rho}_{\rm red}(t - t')]\,, \tag{2}$$

where

$$\hat{\mathcal{K}}(t) = \hat{\mathcal{P}}\hat{L}e^{\hat{\mathcal{Q}}\hat{L}t}\hat{\mathcal{Q}}\hat{L}\hat{\mathcal{P}}\,. \tag{3}$$

Here we have introduced the Liouvillian operator $\hat{L}$, and the projectors $\hat{\mathcal{P}}$ and $\hat{\mathcal{Q}} = \hat{\mathcal{I}} - \hat{\mathcal{P}}$ ($\hat{\mathcal{I}}$ is the identity), which are defined in terms of the reduced density matrix $\hat{\rho}_{\rm red}$ and a fixed reference state $\hat{\rho}_{\rm B}$ of the environment degrees of freedom,

$$\hat{\rho}_{\rm red} = \hat{\mathcal{P}}\hat{\rho} = {\rm Tr}_{\rm B}(\hat{\rho}) \otimes \hat{\rho}_{\rm B}\,. \tag{4}$$

The memory kernel $\hat{\mathcal{K}}(t)$ includes the effects of the history of the reduced state. Solving the Nakajima-Zwanzig equation and finding the memory kernel is in general a computationally hard task. Transfer-tensor methods [21–23] have been recently proposed as efficient techniques to reconstruct the memory kernel. In these the individual quantum trajectories are used to learn dynamical maps that can be converted into the so-called transfer tensors. The transfer tensors are then directly related to the memory kernel of the Nakajima-Zwanzig equation [21].

Embeddings and collision models. An alternative general approach to the study of memory consists in collision models and the so-called Markovian embeddings (see, for instance,

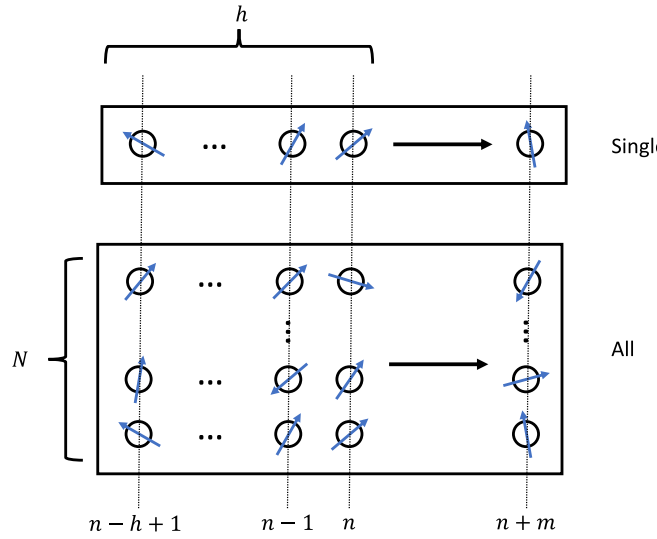

FIG. 1. Here the two types of non-Markovian mappings are shown. The small blue arrows symbolise the Bloch vectors of the individual qubits, while the dashed lines are the time steps. N is the number of qubits, h is the history and m is the future time where we map the evolution to. In the "single" case, the history of a single qubit (evolving in the presence of the others, which then define the *environment*) is used to predict its future dynamics, while in the "all" case, the history of all the qubits, is used to predict the future dynamics of the entire system.

[6–11]). The general idea is that the system interacts in a controlled way with an environment comprising of ancillas that, in turn, can interact amongst themselves in a variety of ways. In this approach, which has been very fruitful for understanding the boundary between Markovian and non-Markovian dynamics, one typically employs some spin Hamiltonian, where the interaction is turned on and off sequentially in some predetermined fashion. For the purposes of our study we similarly employ a one-dimensional spin description of our system, specifically using the Heisenberg model as generator of the dynamics. Thus the Hamiltonian reads,

$$H = -\frac{J}{2}\sum_{j=1}^{N}\sigma^{(j)}\cdot\sigma^{(j+1)}\,, \tag{5}$$

where σ is the vector of Pauli matrices for spin $1/2$, N is the total number of qubits and J defines the time scale of the problem. In particular here we consider finite periodic spin arrangements, namely rings of spins. In our case, since our motivation is to understand how well the full dynamics can be predicted by using a machine-learning (ML) data-driven technique, no piecewise temporal distinctions are made.

B. Data-driven methods

The data driven approach for predictive dynamics used here and, for instance, in references [19,20] has some clear similarities to both the projection and the collision methods. One defines $\mathbf{r}_i^t$ as the Bloch vector for the ith qubit at the time t, which is then discretized into $t_n = n\Delta$ with n an integer and some fundamental time Δ, and sets

$$R^{t_n} = R^n = (\mathbf{r}_1^n, \mathbf{r}_2^n, ..., \mathbf{r}_N^n)\,, \tag{6}$$

namely, R^{t_n} is the collection of all Bloch vectors at a given time. We consider two types of mappings (in the underbraces we show the dimensions): (i) "all", where the entire collection of Bloch vectors is propagated in time

$$\underbrace{(R^n, R^{n-1}, ..., R^{n-h+1})}_{3\times N\times h} \to \underbrace{R^{n+m}}_{3\times N}\,, \tag{7}$$

and (ii) "single", where the time-evolution of only one Bloch vector is computed,

$$\underbrace{(\mathbf{r}^n, \mathbf{r}^{n-1}, ..., \mathbf{r}^{n-h+1})}_{3\times h} \to \underbrace{\mathbf{r}^{n+m}}_{3}\,. \tag{8}$$

Here, the history h is the Markov order, which determines the depth of the time memory, and m is the distance into the future where the prediction will take place. Unlike when propagating the full many-body wavefunction, which grows as 2^N, both mappings scale linearly with the number of qubits N. A schematic of both these mappings is shown in Fig. 1.

In this set-up a Markovian evolution corresponds to the $h = 1$ case, while the accurate prediction of an increasingly non-Markovian dynamics [11] can be obtained by enlarging the memory h. In lay terms, the ability of a NN to predict the dynamics from only a partial knowledge of the full density matrix (the reduced density matrix) is traded off with a nonlocality in time. An important feature of this scheme is that we can perform autoregression, namely we can feed the predictions at some given time back into the model to make predictions at subsequent times. Such ability makes our mappings to be universal time propagators, regardless of whether the dynamics is Markovian or not. Notably, as one can train the model to predict at arbitrary times in the future (one can train different networks for different m's) both short- and long-time trajectories are accessible, with the error being dictated by the accuracy of the specific network.

Although recurrent NNs have been used previously in the literature to model quantum evolution [17,24], here we opt for a fully connected NN to learn the propagator of the dynamics, since we only deal with inputs of a fixed size. In particular, we use a two-layer network with 64 nodes in each hidden layer, and the exponential linear unit (ELU) activation function, which was found to perform best. In order to learn the NN parameters we minimize the trace distance between the predicted and true Bloch vectors. When applied to the density matrix the trace distance reads $D = \frac{1}{2}|\sigma - \rho|$, while for the Bloch vectors $\mathbf{x}$ and $\mathbf{y}$ it reduces to $\frac{1}{2}|\mathbf{x} - \mathbf{y}|$. Note that $0 \leqslant D \leqslant 1$. The neural networks have been implemented using PyTorch [25].

C. Data generation

The time-dependent quantum trajectories used for the training of the NN have been generated by taking two different types of pure initial states, $\hat{\rho}(t = 0)$, namely random and product states. These are, respectively, constructed from initial wavefunctions of the form

$$|\psi(0)\rangle_{\text{rand}} = \sum_{i_1=0,1}\sum_{i_2=0,1}...\sum_{i_N=0,1} a_{i_1 i_2 ... i_N}|i_1 i_2 ... i_N\rangle\,, \tag{9}$$

$$|\psi(0)\rangle_{\text{prod}} = \sum_{i_1=0,1} a_{i_1}|i_1\rangle \sum_{i_2=0,1} a_{i_2}|i_2\rangle ... \sum_{i_N=0,1} a_{i_N}|i_N\rangle\,, \tag{10}$$

where the vector $|i_n\rangle$ spans the Fock space of the nth qubit and the coefficients a_{i_n} and $a_{i_1 i_2 ... i_N}$ are selected randomly in both cases. Note that, on the one hand, the single-particle reduced density matrices associated to product states have all zero von Neumann entanglement entropy, $\text{Tr}(\hat{\rho}_{\text{red}}\log\hat{\rho}_{\text{red}})$, and hence there is no entanglement initially present in the system. On the other hand, random initial states have, on average, a high entanglement entropy, as shown in Fig. 2.

These initial states are then time evolved with the propagator of the total system (the specific time-dynamic generators are discussed below). In particular, at a practical level the equations of motion have been integrated numerically by using the QuTiP Python package [26]. Then, we trace out the environment degrees of freedom (some of the qubits) to generate the quantum trajectories of the single qubits. In all our numerical experiments, unless otherwise specified, we use for each system size N, a dataset consisting of 11 000 sample trajectories. These are generated by randomising the initial states and by propagating those forward in time with the relevant Hamiltonian. The training set contains 8000 samples, while the validation and test subsets are made of 2000 and 1000 samples, respectively. We employ the validation set to determine when to halt the training. Throughout this paper all the results shown are calculated on the test set.

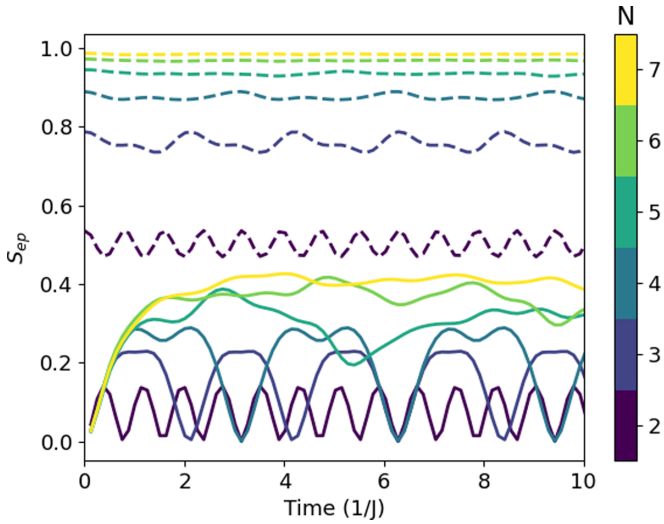

FIG. 2. The mean single-particle entropy $S_{\rm sp} = {\rm Tr}(\hat{\rho}_{\rm red} \log \hat{\rho}_{\rm red})$, as a function of time for differently initialized states. The dashed lines correspond to randomly generated initial states, while the continuous lines are associated to product initial states. The color encodes the total number of qubits in the system. Note that product states have vanishing initial entropy, while the entropy of random states grows with the number of qubits considered.

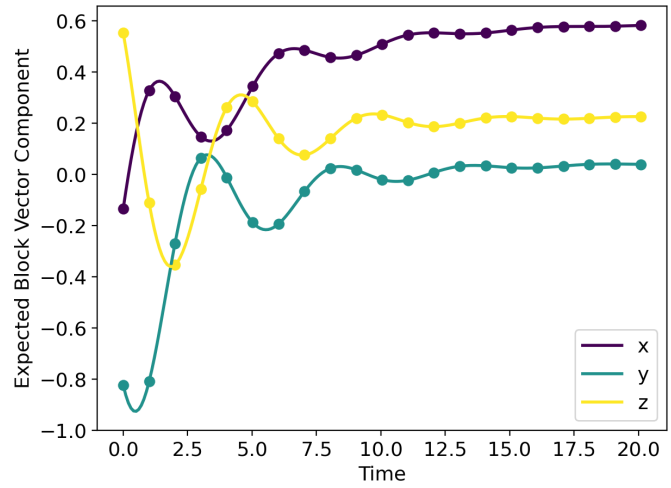

FIG. 3. Comparison between the exact time-dependent trajectories (continuous line) and those predicted by the machine-learned regression (points). Here we plot the time-dependent expectation values of the three Pauli operators over the evolution determined by the Lindblad equation, Eq. (12). The predictions agree with the exact values up to a floating point error.

III. PREDICTING THE EVOLUTION OF MARKOVIAN SYSTEMS

For Markovian dynamics the state of the system at a given time contains sufficient information to predict its future evolution. This is the case when the state is described by the wavefunction or by the full density matrix. Here, since either the Schrödinger or the von Neumann equation imply a linear mapping between states at different times, given a set of trajectories we can use a linear regression to learn the propagator. Let the vector $\mathbf{x}_n$ represent the wavefunction or density matrix of a system at time step n, then the propagator, P, is a matrix with $\mathbf{x}_{n+1} = P\mathbf{x}_n$. Given a dataset with p pairs of single time step evolutions $\{(\mathbf{x}_n^{(1)}, \mathbf{x}_{n+1}^{(1)}), ..., (\mathbf{x}_n^{(p)}, \mathbf{x}_{n+1}^{(p)})\}$ we can learn the propagator matrix by writing

$$P = (X_n^T X_n)^{-1} X_n^T X_{n+1}\,, \tag{11}$$

where X_n is a matrix, whose ith row is $\mathbf{x}_n^{(i)}$. Thus, we can exactly learn the propagator, if the matrix $X_n^T X_n$ is invertible. This is the case when the number of samples p is greater than the dimension of the vector. If Ω is the dimension of the Hilbert space, then we require $2\Omega - 1$ samples to learn the wavefunction propagator and $\Omega^2 - 1$ samples to learn that of the density matrix [27].

As a warm-up example we consider a single qubit evolving via the Lindblad master equation [28]

$$\frac{d\hat{\rho}}{dt} = -i[\hat{H}, \hat{\rho}] + 2\hat{L}\hat{\rho}\hat{L}^\dagger - \{\hat{L}^\dagger\hat{L}, \hat{\rho}\}, \tag{12}$$

where we have chosen $\hat{H} = 0.5\hat{\sigma}_x + 0.3\hat{\sigma}_y + 0.2\hat{\sigma}_z$ and $\hat{L} = \sqrt{0.1}\hat{\sigma}_x$. Note that this is nonunitary but Markovian. Here the propagator is of the form $\mathbf{x}_{n+1} = A\mathbf{x}_n + \mathbf{a}$, where A is a matrix and $\mathbf{a}$ is a vector. However, by defining $\tilde{\mathbf{x}} = (1, x_1, x_2, ...)$ we can write it in the form $\tilde{\mathbf{x}}_{n+1} = P\tilde{\mathbf{x}}_n$ and thus apply Eq. (11).

Figure 3 compares the exacts Bloch vector trajectories with those generated by the learnt propagator. Here four samples were required to learn the propagator and the two trajectories agree up to floating point error.

IV. PREDICTING NON-MARKOVIAN DYNAMICS

Let us now move to the non-Markovian dynamics. In this case our Hamiltonian is the Heisenberg model of Eq. (5), calculated for a number of qubits going from $N = 2$ to $N = 7$. Information about the time evolution can be extracted by plotting the wavefunction fidelity $|\langle\psi(t=0)|\psi(t)\rangle|^2$ as a function of time for a range of random initial states, as presented in Fig. 4. We note that for $N \leqslant 4$ the fidelity appears periodic in time over the time interval considered, with periods $T_2 = \pi/2$, $T_3 = 2\pi/3$, and $T_4 = \pi$, respectively for $N = 2, 3$, and 4 (all times are in units of $1/J$). We also note that after a time of approximately 1 $(1/J)$ the fidelity approaches zero regardless of N. Such time can be defined as the characteristic de-correlation time. Next, we describe our results obtained for the non-Markovian dynamics of single particle Bloch vectors.

A. Memory

Let us now consider the dynamics of the single-particle reduced density matrix and evaluate the memory needed to propagate a system initialized in a random state. In particular, we set the time step at $\Delta = 0.08\pi \approx 0.25$ and predict several times into the future. We consider two different cases. In the first, Fig. 5, the feature vector for the NN comprises the Bloch vectors of all qubits, namely we simultaneously follow the dynamics of the single-particle reduced density matrices of all the qubits. In the second case, Fig. 6, the NN propagates the Bloch vector of a single qubit only. Note that the qubit choice here is irrelevant and the neural network can be used to propagate any one specific qubit. This is because the Hamiltonian generator of the dynamics is translation invariant. In

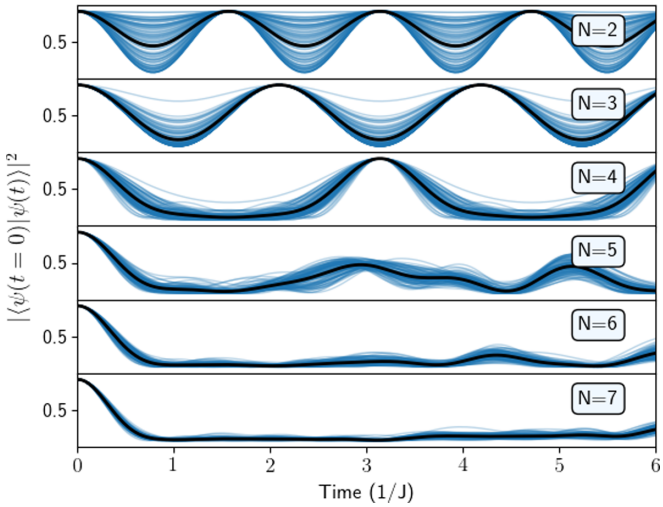

FIG. 4. The fidelity of the wavefunction $|\psi\rangle$ with respect to the initial state $|\psi(t=0)\rangle$ is computed as a function of time for states initialized randomly. Here, the fidelity is defined as $|\langle\psi(t=0)|\psi(t)\rangle|^2$. The blue lines are the individual trajectories, while the black lines their mean. Note that for $N = 2, 3, 4$ the fidelity is periodic in time. The time evolution are for the Heisenberg Hamiltonian of Eq. (5).

the figures we plot the NN error, the mean trace distance (MTD), as a function of the memory (in units of the time step, $\Delta = 0.08\pi$) for NNs that propagate at different times into the future (color code).

In general, in both situations one can always find a sufficiently long memory to converge the NNs to small errors (in the figures an error of 0.01 is indicted as a dashed black line). For the "all" case such memory seems to be rather independent of the final time of propagation, namely it takes approximately the same memory to propagate the entire Bloch-vector manifold to either short or long times. This is more evident when the total number of qubits is small, while some scattering in the data appears for $N \geqslant 5$. In this case propagating at longer times in the future seems to require a

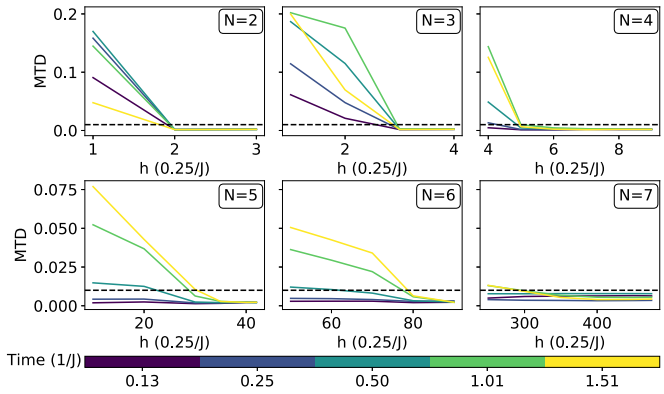

FIG. 5. Error of the NN (the mean trace distance—MTD) as a function of the memory h (in units of $\Delta = 0.08\pi \approx 0.25/J$), for NNs that propagate at different points in the future (color code). Here the results are for NNs that use as feature the reduced density matrix of all qubits ("all" case).

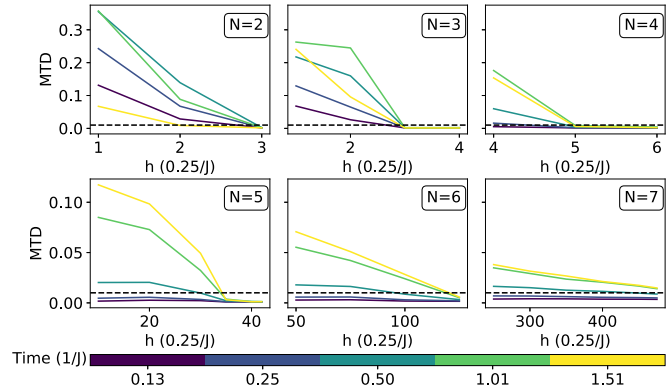

FIG. 6. Error of the NN (the mean trace distance—MTD) as a function of the memory, h, (in units of $\Delta = 0.08\pi \approx 0.25/J$), for NNs that propagate at different points in the future (color code). Here the results are for NNs that use as feature the reduced density matrix of a single qubit ("single" case).

deeper memory. Similar results are found when constructing NNs using a single Bloch vector as feature (see Fig. 6), in particular when the qubit count remains low ($N \leqslant 4$). Most importantly, in both situations the memory needed to converge the NNs increases drastically with the number of qubits.

The scaling of the memory with the number of qubits is presented in the left-hand side panel of Fig. 7 for NNs propagating $0.48\pi \approx 1.5$ in the future ($\Delta = 0.08\pi$). This is a time significant longer than the de-correlation time observed in Fig. 4. We take the operational definition of "necessary" memory $h_{\rm nec}$ as the memory needed to reduce the error below 0.01, and this is plotted on a base-2 logarithmic scale against N. We find that $h_{\rm nec}$ scales exponentially with N, namely $h_{\rm nec} \propto 2^{\alpha N}$, with $\alpha \sim 1$ (this cannot be determined with precision from our limited number of data points). Interestingly, we find little difference in the $h_{\rm nec}$ scaling between the "all" and "single" case, although for large N's $h_{\rm nec}$ is systematically lower when all Bloch vectors are used in the NNs.

In Fig. 7 we also plot $\log_2(h_{\rm nec})$, respectively as a function of $-\log_2(f_{\rm min})$ (middle panel), where $f_{\rm min}$ is the lowest frequency of a given system, and as a function of the $\log_2$

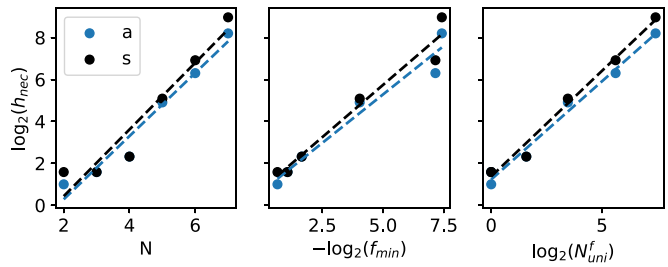

FIG. 7. Scaling of the memory needed to achieve accurate propagation (see definition in the text) $h_{\rm nec}$ as a function of: (left) the number of qubits, (middle) the minimum frequency $f_{\rm min}$ and (right) the number of unique frequencies of the spectrum of the corresponding Heisenberg chain $N^f_{\rm uni}$. Note that all quantities are plot on a $\log_2$ scale. Light (dark) blue symbols are for the "all" ("single") case. Here data are presented for NNs predicting $\Delta = 0.48\pi \sim 1.5/J$ in the future.

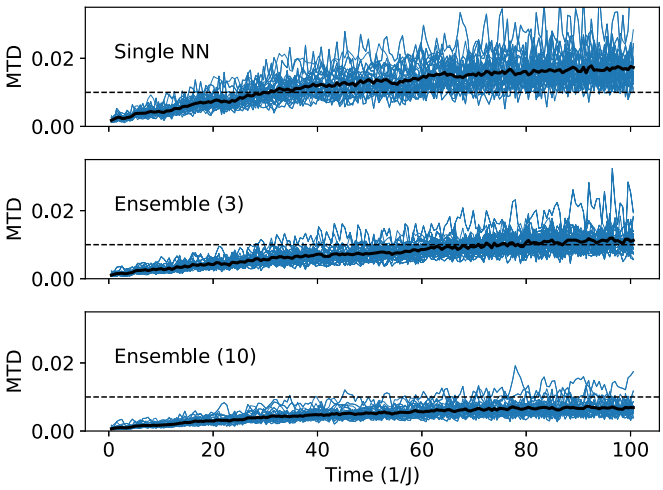

FIG. 8. Demonstration of autoregression, where the output of a NN at time t is used as the input for the propagation at time $t + \Delta t$. Results are presented for NNs with a time time step of $\Delta = 0.16\pi \sim 0.5/J$ and a memory of $h = 45$. The system comprises $N = 6$ qubits, the initial state is a random state and the NNs use the entire manifold of Bloch vectors as feature. The graphs show the mean trace distance error as a function of time for different trajectories (blue lines). The bold line corresponds to the time-averaged mean error. In the top panel we use a single NN, in the middle panel an ensemble of three NNs and in the bottom panel an ensemble of ten. Note that both the time-averaged error and the fluctuations around the average improve as one uses a large number of NNs.

of the number of unique frequencies associated to the spectrum of the corresponding system (right-hand side panel). The frequencies are computed as $\Delta\epsilon_{nm} = \epsilon_n - \epsilon_m$, with ϵ_n being an eigenvalue of the Heisenberg Hamiltonian, obtained by exact diagonalization. We find an extremely good correlation between h_{nec} and the number of unique frequencies, suggesting that the memory needed for an accurate propagation of the time propagation may scale more favourably for systems presenting a limited number of frequencies (e.g., in the case of high degeneracy). This correlation, however, may still be accidental, so that it would be interesting to investigate a many-body Hamiltonian where the multiplicity of the frequency spectrum presents a scaling, as a function of the number of particles, different from that of the Hilbert space.

B. Propagation

In order to show that our NNs can act as true time propagators we need to demonstrate that their time evolution can be concatenated, namely that one can use the predictions made for time t as an input for the subsequent prediction at time $t + \Delta t$. Repeating such a process, an operation called autoregression, allows one to follow the dynamics at, in principle, arbitrary times. Such exercise is performed here for the case of six qubits, $N = 6$, a propagator with a time time step of $\Delta = 0.16\pi \sim 0.5/J$ and a memory of $h = 45$. We consider NNs using the entire set of Bloch vectors as feature (the "all" case) and we initiate the dynamics from a random state.

The results of this test are presented in Fig. 8, where we plot the error (the mean trace distance) as a function of time for propagation up to $100/J$, namely for about 200 time steps. In order to minimize the error we train multiple NNs along the same time trajectories and propagate the Bloch vectors by using the average of the networks' predictions. Thus, in Fig. 8 we show results for a single NN (top panel), an average of three NNs (middle panel) and one of ten (bottom panel). Two main conclusions can be drawn from the figure. On the one hand, it is clear that our NNs are well capable of performing autoregression, with the error growing relatively slowly in time. On the other hand, the figure clearly shows that averaging over several NNs significantly improves the predictions; the average is more accurate and the fluctuations around the average are reduced. This effectively demonstrates that our NNs can be used as universal time-evolution operators.

When considering an ensemble of NNs one can then use the disagreement among the NNs as a measure of the confidence over the prediction. This is essentially the variance of the predicted quantity over the different NNs. Such variance is plotted in the top panel of Fig. 9 for the expectation value of σ_z along the time trajectory of a single qubit in a system of $N = 6$ qubits. Also in this case we simulate the time-evolution of $N = 6$ qubits with a memory $h = 45$. In this particular case the variance does not significantly change over time, indicating that the accuracy is largely preserved over the trajectory. However, one can also note that the variance is larger along particular branches of the trajectory, where the predictions of the NNs agree less well with the exact results. As such, the variance can be used as a measure of the accuracy of the autoregression. Specifically, one can monitor the increase in variance and use it as an indicator of the fidelity of the prediction as a function of time. Finally, we show that NNs trained to propagate the dynamics at different time steps can be concatenated, namely that the output of a network of time step Δt_1 can be used as input for a NN with time step Δt_2. This is shown in the two lower panels of Fig. 9. Firstly, we present the trajectory of the expectation value of σ_z of one qubit, computed with a NN using a time step of $\Delta = 0.16\pi$. Then, we use the output of such NN as input in NNs of steps 0.04π, 0.08π, and 0.12π. This effectively allows us to increase the density of the predicted points along the trajectory. The figure shows clearly that such an operation is possible, without any significant loss of accuracy. This is an important results, as it allows one to construct a range of NNs, predicting at different times in the future, and use appropriate combinations of them to reach any point in time along the trajectory. Crucially, this means that the NNs autoregression can be used to generate long-time and arbitrary dense time trajectories.

C. Dynamics initiated from a product state

So far we have analyzed the dynamics initialized from a random state, namely from a state with a high entanglement entropy. Here we repeat the analysis for an initial state corresponding to a product state. Recalling Fig. 2, the single-particle entropy of a product state grows with the time, meaning that a NN trained from the early-time trajectory will not contain enough information to reproduce the correct

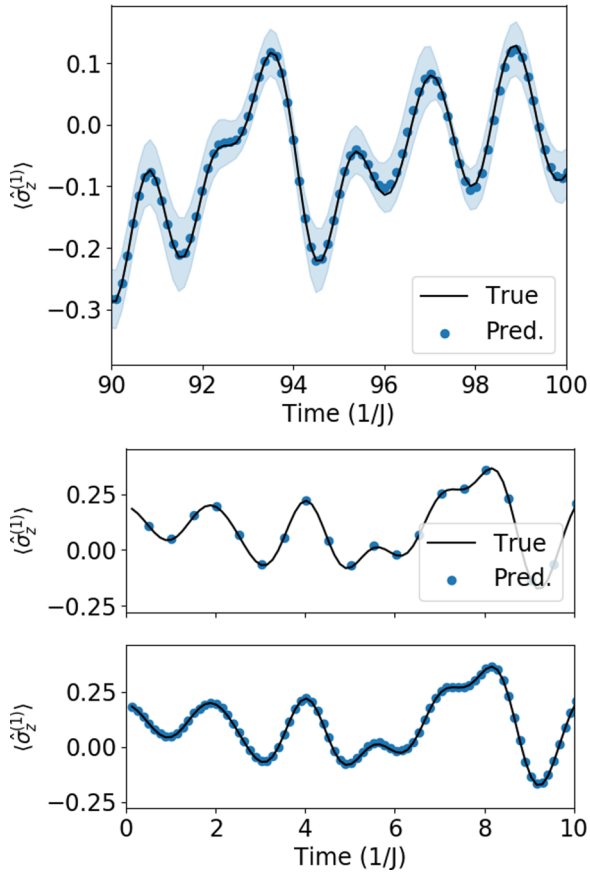

FIG. 9. Time trajectories generated via NN autoregression. Here we consider $N = 6$ qubits and monitor the time dependence of the expectation value of σ_z of one qubit. The memory in this case is $h = 45$. In the top panel we show the predicted trajectory (dots) against the exact one (solid-black line), together with the variance of the prediction over ten NNs (blue shadow). The middle panel show the predictions obtained with a time-propagator NN of step $\Delta = 0.16\pi$, while the bottom one uses the same predictions as starting point for time propagators with time steps 0.04π, 0.08π, and 0.12π. Note NNs trained to predict the dynamics at different time scales can be effectively concatenated.

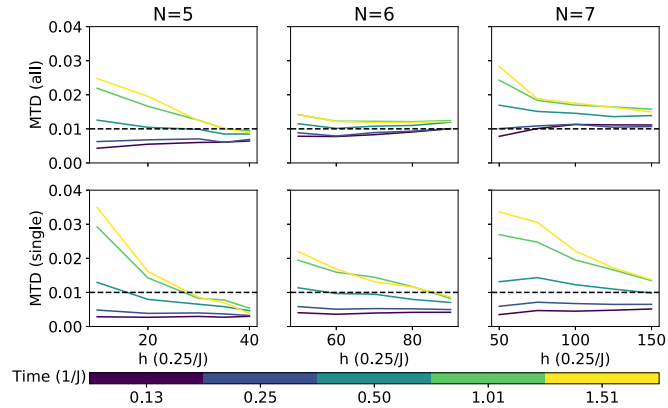

FIG. 10. Error of the NN (the mean trace distance) as a function of the memory h (in units of $\Delta = 0.08\pi \approx 0.25/J$), for NNs that propagate at different points in the future (color code). Here the results are for NNs that use as feature either the reduced density matrix of all qubits ("all" case—top panels) or of a single qubit ("single" case—lower panels).

dynamics and cannot be used in an autoregression. This is because the states encountered during the early dynamics of a product state are qualitatively different from those encountered over long times. As such, we have now trained NNs by using as an initial state the one evolved from a product state at time $0.32\pi \approx 1/J$ (this time is sufficient to reach equilibrium, see Fig. 2), and our results are summarised in Fig. 10 for system with $N \leqslant 5$.

In the figure we show the NN error (the mean trace distance) as a function of the memory for different time propagators computed for both the "all" and "single" case. The results are qualitatively similar to those encountered for networks trained over dynamics initiated with random states (see Figs. 5 and 6), except that the memory required to converge the propagator is now, in general, significantly shorter. This follows from the fact that the dynamics initiated from a product state spans only a subset of the entire Hilbert space, an observation corroborated by the result that product states never evolve to configurations with single-particle entropy close to that of random states (see Fig. 2). Such behavior is not surprising, given the high-symmetry form of the Hamiltonian.

We investigate further this aspect by constructing an autoencoder [29] performing a nonlinear compression of the wavefunctions corresponding to both random and product states. To perform this task the wavefunction is expanded over the full Fock space $\{|i\rangle\}$ and the real and imaginary coefficients of expansion $\psi_i = \langle i|\psi\rangle$ are taken to form a 2Ω-dimensional vector $\mathbf{x}$ with

$$x_j = \begin{cases} \mathrm{Re}(\psi_j) & j \leqslant \Omega \\ \mathrm{Im}(\psi_{j-\Omega}) & j > \Omega \end{cases} \tag{13}$$

where $\sum_{i=1}^{\Omega} |\psi_i|^2 = |\mathbf{x}|^2 = 1$ and Ω is the dimension of the Hilbert space. The encoder and decoder forming the autoencoder are two fully connected NNs, with two 64-nodes hidden layers and the ELU activation function. By minimizing the reconstruction error, as measured by the Euclidean distance between the vector representations, we can quantify the level of compression that a wavefunction can undergo at different times. Figure 11 shows our results obtained with a dataset of 2000 samples, with 1000 in the training set and 500 in the validation and test ones. For this numerical experiment we consider $N = 5$. Clearly, a product state can be compressed much more efficiently than a random one (a 15-dimensional latent space appears to be sufficient at all times), regardless of the time at which the wavefunction is measured. Note here that for $N = 5$ the dimension of the Hilbert space is 32. Therefore, when the latent space is 64-dimensional there is no compression and the autoencoder just learns the identify function. As such, one expects the fidelity to approach unity regardless of the state. The deviation observed for the random state is then attributed to relatively small training set.

D. Frequency of the memory sampling

We now investigate further the nature of the memory required for non-Markovian dynamics. The first question we

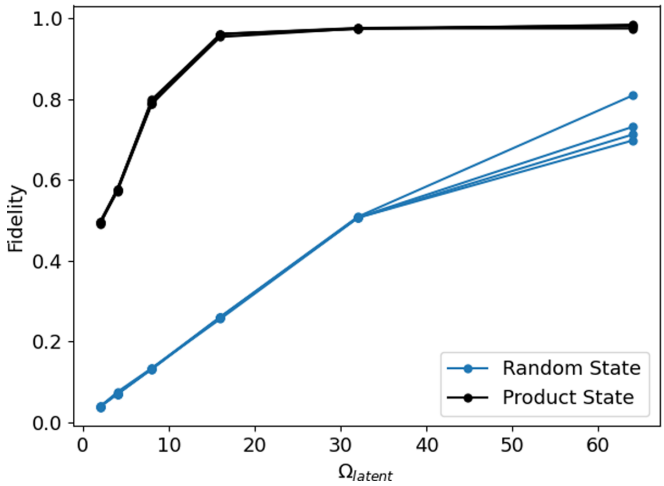

FIG. 11. Wavefunction fidelity as a function of the dimension of the latent space (reduced dimension) for wavefunction evolved to times 0, 0.25, 0.5, and 1 (in units of $1/J$). Here we compare evolutions initiated from either a random or a product state for $N = 5$.

want to address is whether the convergence of the memory is smooth or sharp, namely whether the error in the propagation drops sharply when one reaches the required memory. This is explored in Fig. 12, where we present the error of the NN (the mean trace distance of a single NN for all the samples in the training set) as a function of the memory duration for propagation to a time of $0.48\pi \approx 1.5/J$, where the memory is sampled with the fine time resolution of $0.04\pi \approx 0.13/J$. The exercise is performed for both $N = 5$ (left-hand side panels) and $N = 6$ (right-hand side panels), in the "all" (top panels) and "single" (bottom panels) cases. The graphs clearly show a very sharp transition, with the error abruptly reducing below the 0.01 threshold as soon as the memory reaches a critical duration. This means that, if the memory is accurately sampled, one will just need to construct a NN time propagator with

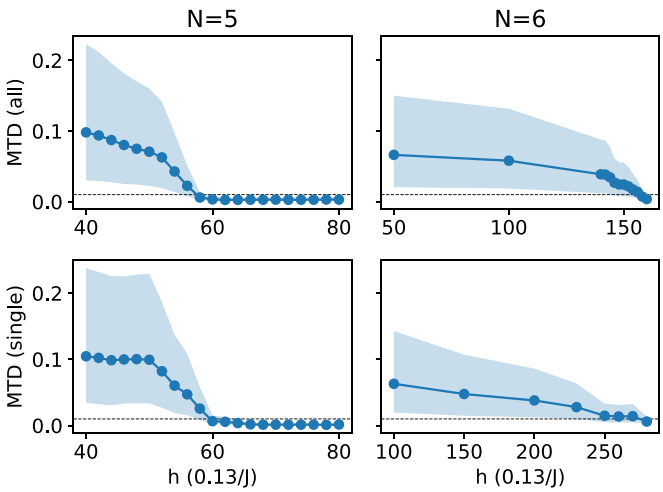

FIG. 12. Error of the NN (the mean trace distance) as a function of the memory h (in units of $\Delta = 0.04\pi \sim 0.13/J$), for $N = 5$ and $N = 6$ and both the "all" (upper panels) and "single" (lower panels) case. In this case the NN propagates $0.48\pi \approx 1.5/J$ in the future. The shaded region corresponds to the 25th and 75th error percentiles.

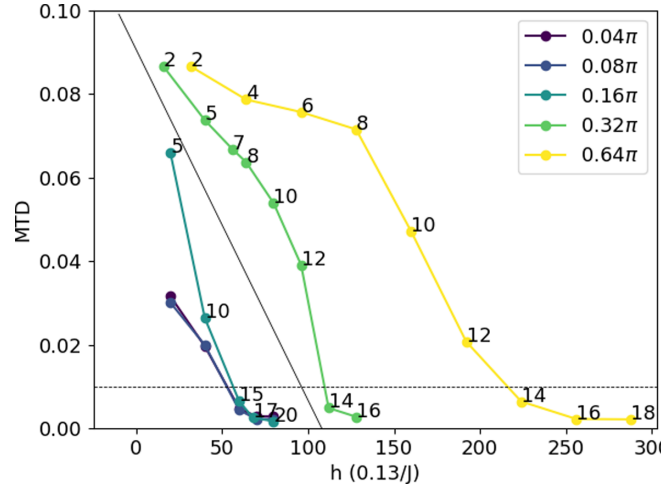

FIG. 13. NN error (the mean trace distance) as a function of the memory for the propagation to the time $0.48\pi \approx 1.5/J$ of a $N = 5$ system. Different curves correspond to memories sampled with different time steps. The numbers on each curve correspond to the number of time steps included in the memory (the dimension of the time component of the feature vector), while the x axis scale is absolute ($\Delta = 0.04\pi$). The black-diagonal line separates the results for which the memory duration is independent from the sampling.

a memory longer than the critical one. In fact, considering longer memories does not improve the convergence. In other words, it appears that the range of the time-propagator kernel is finite and well defined for any given system.

Having established that there is a critical memory for each system, one now needs to find out how finely such memory should be sampled. This question is answered in Fig. 13. Here we have constructed a single NN to propagate a $N = 5$ system $0.48\pi \approx 1.5/J$ in the future ("all" case). The memory is then sampled with different time steps, going from 0.04π to 0.64π. Surprisingly, we find that when the memory is finely sampled ($\Delta = 0.04\pi$, 0.08π, and 0.16π) its duration does not change, namely feature vectors of different length can equally well predict the dynamics as long as enough time-evolution history is learned. In contrast, more coarse samplings ($\Delta > 0.16\pi$) require longer memories. Intriguingly, in this case of coarse sampling, one approximately needs the same number of time steps (about 14), although of different duration, to converge the NN.

Notably, the highest frequency found in the spectrum of the $N = 5$ spin chain is $f_{\max} = 0.99J$. According to the Shannon-Nyquist sampling theorem [30] the largest possible period required to sample a time-dependent dynamics is $\Delta_{\max} = 1/2f_{\max}$, which in our case is $\Delta_{\max} \approx 0.16\pi$. Intriguingly, this corresponds to the critical sampling time above which the memory is no longer time-step independent. We can then conclude that the two regimes found in Fig. 13 are simply separated by the Shannon-Nyquist limit. In general, for a linear model the dynamics cannot be reproduced at all if sampled at a time step larger than $\Delta_{\max}$. Here, however, our time-propagator (the NN) is highly nonlinear and the Shannon-Nyquist limit can be avoided by sampling longer memories at a coarser resolution. Further tests using nonlinear memory samplings have not given conclusive results.

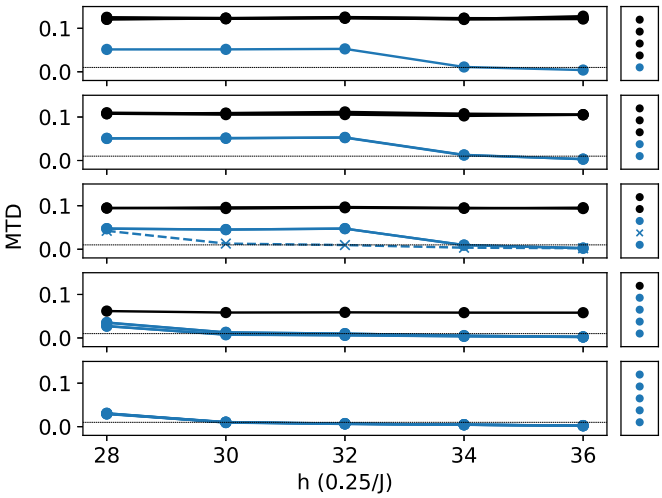

FIG. 14. NN error (the mean trace distance) as a function of the memory for the propagation to the time $0.48\pi \approx 1.5/J$ of a $N = 5$ system. Here the NN is trained by including only the Bloch vector of m qubits, while the remaining $N - m$ are not considered. In the graphs the error over the qubits included in the training is in blue, while that of the qubits outside the NN feature vector are in black.

E. Information locality

Finally we investigate how the information included in the feature vector affects our ability to make predictions. For this experiment we take the $N = 5$ spin chain with dynamics initiated at a random initial state. In this case we compute the error (the mean trace distance) of a single NN as a function of the memory for a generic case where the feature vector contains the Bloch vectors of m qubits. The error is then computed over both the m qubits used in the feature vector and the remaining $N - m$ ones. With this notation $m = 1$ and $m = 5$ corresponds to the "single" and "all" case, respectively. Our results are summarized in Fig. 14.

In general we find that the models cannot satisfactory predict the evolution of qubits, whose Bloch vector was not included in the training, although the error reduces as m gets larger. Interestingly, this is true even for the $m = 4$ case, where only the Bloch vector of one qubit is left outside the NN feature vector. A perhaps more intriguing feature is found for the $m = 3$ case. In this situation the three qubits used for the training are inequivalent, since only one neighbor other qubits included in training the model. In this case we found that the memory required to bring the error below threshold is different depending on the location of the qubit. This suggests that there may be a trade off between time and space locality in the time propagator kernel. Similar results are also found for the $N = 6$ case (not presented here), where several inequivalent configurations can be designed.

V. CONCLUSION

In this paper we have explored the use of machine learning to propagate quantum systems in time, using the Heisenberg Hamiltonian as many-body model. For Markovian dynamics the time propagator can be learnt easily with a linear regression, as long as the training dataset is sufficiently large. In contrast, the propagation of non-Markovian systems requires a history, meaning that the system state at a given time is determined by a number of states in the past. The duration of such memory appears to scale exponentially with the system size, regardless of whether one uses the density matrix of a single qubit or of the entire ensemble as feature. However, shorter memories are required when the system explores only a subset of the available spectrum during the time evolution, as in the case of dynamics initiated from a product state.

Crucially, we have shown that the machine-learning propagators can be concatenated in an autoregression, meaning that the state evolved with one neural network can be used as input for another propagation. This allows us to propagate at arbitrary long times with any desired resolution. Our method can then be applied to quantum dynamical data generated from any computational scheme, whether it be propagation, tensor networks [31], restricted Boltzmann machines [14], or data obtained experimentally. Furthermore, by using an ensemble of machine-learning propagators we can maintain accuracy for a large number of iterations and constantly estimate the error of our predictions.

Finally, we have investigate in detail the time resolution needed to represent the system memory, and found two regimes separated by the Shannon-Nyquist limit. Namely, when the memory is sampled at a time step shorter than the period corresponding to the fastest frequency of the system, the memory remains constant. This means that one has to sample a fixed time interval but different time steps can be used. In contrast, if the time step is longer than such period, the required memory is no longer constant, but the total number of time steps is. This demonstrates that the nonlinearity built in the neural networks can overcome the limitation set by the Shannon-Nyquist sampling theorem.

For future studies it will be useful to compare the method presented here with the other approaches proposed for solving the reduced dynamics of the system, such as the discussed Markovian embeddings, the Nakajima-Zwanzig technique, and also the transfer tensor methods. In particular, it will be interesting to see if a direct connection between the memory kernel appearing in the Nakajima-Zwanzig equation and the history depth h that we observed for the trained NNs can be made.

ACKNOWLEDGMENTS

This work has been supported by the Irish Research Council Postgraduate Scholarship (J.N.) and Advance Laureate Award (S.S. - Award No. IRCLA/2019/127). L.C. and G.K. acknowledge Science Foundation Ireland for financial support through Career Development Award No. 15/CDA/3240. We acknowledge the DJEI/DES/SFI/HEA Irish Centre for High-End Computing (ICHEC) and Trinity Centre for High Performance Computing (TCHPC) for the provision of computational resources. The authors thank Plamen Stamenov and John Goold for interesting discussion.

APPENDIX: NAKAJIMA-ZWANZIG EQUATION

Starting from the Liouville-Von Neumann equation,

$$\partial_t \hat{\rho} = \frac{i}{\hbar}[\hat{\rho}, \hat{H}] = \hat{L}\hat{\rho}\,, \tag{A1}$$

one splits the density operator up, $\hat{\rho} = (\hat{\mathcal{P}} + \hat{\mathcal{Q}})\hat{\rho}$, by means of the projection operators $\hat{\mathcal{P}}$ and $\hat{\mathcal{Q}}$, with $\hat{\mathcal{Q}} = \hat{\mathcal{I}} - \hat{\mathcal{P}}$ ($\hat{\mathcal{I}}$ is the identity). The definition of $\hat{\mathcal{P}}$ and $\hat{\mathcal{Q}}$ is somewhat unusual. Consider a reference state $\hat{\rho}_{\mathrm{B}}$ of the environment and then define $\hat{\mathcal{P}}$ as the projector such that

$$\hat{\rho}_{\mathrm{red}} = \hat{\mathcal{P}}\hat{\rho} = \mathrm{Tr}_{\mathrm{B}}(\hat{\rho}) \otimes \hat{\rho}_{\mathrm{B}} \,, \tag{A2}$$

where ρ_{red} is the reduced density matrix of the degrees of freedom of interest. Thus $\hat{\mathcal{P}}$ is defined as the partial tracing out of the environment and then the product of the resulting state with the predetermined environmental state, which is typically time independent. The definition of $\hat{\mathcal{Q}}$ then follows as

$$\hat{\mathcal{Q}}\hat{\rho} = \hat{\rho} - \mathrm{Tr}_{\mathrm{B}}(\hat{\rho}) \otimes \hat{\rho}_{\mathrm{B}} \,. \tag{A3}$$

Although unusual the projectors are perfectly well defined and it is easy to show that

$$\hat{\mathcal{P}}^2 = \hat{\mathcal{P}} \,, \tag{A4}$$

$$\hat{\mathcal{Q}}\hat{\mathcal{P}} = \hat{\mathcal{P}}\hat{\mathcal{Q}} = 0 \,. \tag{A5}$$

Starting from Eq. (A1) one inserts $\hat{\mathcal{I}} = \hat{\mathcal{P}} + \hat{\mathcal{Q}}$ to obtain

$$\partial_t \hat{\mathcal{P}}\hat{\rho} = \hat{\mathcal{P}}\hat{L}\hat{\mathcal{P}}\hat{\rho} + \hat{\mathcal{P}}\hat{L}\hat{\mathcal{Q}}\hat{\rho} \,, \tag{A6}$$

$$\partial_t \hat{\mathcal{Q}}\hat{\rho} = \hat{\mathcal{Q}}\hat{L}\hat{\mathcal{Q}}\hat{\rho} + \hat{\mathcal{Q}}\hat{L}\hat{\mathcal{P}}\hat{\rho} \,, \tag{A7}$$

and the second identity can be formally solved as

$$\hat{\mathcal{Q}}\hat{\rho} = e^{\hat{\mathcal{Q}}\hat{L}t}\hat{\mathcal{Q}}\hat{\rho}(0) + \int_0^t dt' e^{\hat{\mathcal{Q}}\hat{L}t'}\hat{\mathcal{Q}}\hat{L}\hat{\mathcal{P}}\hat{\rho}(t-t') \,. \tag{A8}$$

This can then be inserted back into the first equation to give

$$\partial_t \hat{\mathcal{P}}\hat{\rho} = \hat{\mathcal{P}}\hat{L}\hat{\mathcal{P}}\hat{\rho} + \hat{\mathcal{P}}\hat{L}e^{\hat{\mathcal{Q}}\hat{L}t}\hat{\mathcal{Q}}\hat{\rho}(0) + \tag{A9}$$

$$\hat{\mathcal{P}}\hat{L}\int_0^t dt' e^{\hat{\mathcal{Q}}\hat{L}t'}\hat{\mathcal{Q}}\hat{L}\hat{\mathcal{P}}\hat{\rho}(t-t') \,. \tag{A10}$$

Finally, if we assume that at the time $t = 0$ the system is a product state $\hat{\rho}(0) = \hat{\rho}_{\mathrm{A}} \times \hat{\rho}_{\mathrm{B}}$, then $\hat{\mathcal{P}}\hat{\rho}(0) = \hat{\rho}(0)$ and $\hat{\mathcal{Q}}\hat{\rho}(0) = 0$, so that we can drop the middle term to obtain

$$\partial_t[\hat{\mathcal{P}}\hat{\rho}] = \hat{\mathcal{P}}\hat{L}[\hat{\mathcal{P}}\hat{\rho}] + \int_0^t dt' \hat{\mathcal{K}}(t')[\hat{\mathcal{P}}\hat{\rho}(t-t')] \,, \tag{A11}$$

where the time kernel writes

$$\hat{\mathcal{K}}(t) = \hat{\mathcal{P}}\hat{L}e^{\hat{\mathcal{Q}}\hat{L}t}\hat{\mathcal{Q}}\hat{L}\hat{\mathcal{P}} \,. \tag{A12}$$

[1] S. Nakajima, On quantum theory of transport phenomena: Steady diffusion, Prog. Theor. Phys. **20**, 948 (1958).

[2] R. Zwanzig, Ensemble method in the theory of irreversibility, J. Chem. Phys. **33**, 1338 (1960).

[3] I. Prigogine, *Non-Equilibrium Statistical Mechanics* (Interscience Publishers, New York, 1962).

[4] F. Shibata, Y. Takahashi and N. Hashitsume, A generalized stochastic Liouville equation. Non-Markovian versus memoryless master equations, J. Stat. Phys. **17**, 171 (1977).

[5] F. Shibata and T. Arimitsu, Expansion formulas in nonequilibrium statistical mechanics, J. Phys. Soc. Jpn. **49**, 891 (1980).

[6] F. Ciccarello and V. Giovannetti, A quantum non-Markovian collision model: Incoherent swap case, Phys. Scr. **T153**, 014010 (2013).

[7] R. McCloskey and M. Paternostro, Non-Markovianity and system-environment correlations in a microscopic collision model, Phys. Rev. A **89**, 052120 (2014).

[8] S. Kretschmer, K. Luoma and W. T. Strunz, Collision model for non-Markovian quantum dynamics, Phys. Rev. A **94**, 012106 (2016).

[9] S. Lorenzo, F. Ciccarello and G. M. Palma, Composite quantum collision models, Phys. Rev. A **96**, 032107 (2017).

[10] B. Çakmak, M. Pezzutto, M. Paternostro and O. E. Müstecaplıoğlu, Non-Markovianity, coherence, and system-environment correlations in a long-range collision model, Phys. Rev. A **96**, 022109 (2017).

[11] S. Campbell, F. Ciccarello, G. M. Palma and B. Vacchini, System-environment correlations and Markovian embedding of quantum non-Markovian dynamics, Phys. Rev. A **98**, 012142 (2018).

[12] A. Mardt, L. Pasquali, H. Wu and F. Noé, VAMPnets for deep learning of molecular kinetics, Nat. Commun. **9**, 5 (2018).

[13] P. G. Breen, C. N. Foley, T. Boekholt and S. P. Zwart, Newton versus the machine: Solving the chaotic three-body problem using deep neural networks, Mon. Not. R. Astron. Soc. **494**, 2465 (2020).

[14] G. Carleo and M. Troyer, Solving the quantum many-body problem with artificial neural networks, Science **355**, 602 (2017).

[15] M. J. Hartmann and G. Carleo, Neural-Network Approach to Dissipative Quantum Many-Body Dynamics, Phys. Rev. Lett. **122**, 250502 (2019).

[16] S.-H. Lin and F. Pollmann, Scaling of neural-network quantum states for time evolution, Physica Status Solidi (b) **259**, 2100172 (2022).

[17] L. Banchi, E. Grant, A. Rocchetto and S. Severini, Modelling non-Markovian quantum processes with recurrent neural networks, New J. Phys. **20**, 123030 (2018).

[18] I. A. Luchnikov, S. V. Vintskevich, D. A. Grigoriev and S. N. Filippov, Machine Learning Non-Markovian Quantum Dynamics, Phys. Rev. Lett. **124**, 140502 (2020).

[19] P. P. Mazza, D. Zietlow, F. Carollo, S. Andergassen, G. Martius, and I. Lesanovsky, Machine learning time-local generators of open quantum dynamics, Phys. Rev. Research **3**, 023084 (2021).

[20] I. A. Luchnikov, E. O. Kiktenko, M. A. Gavreev, H. Ouer-dane, S. N. Filippov and A. K. Fedorov, Probing non-Markovian quantum dynamics with data-driven analysis: Beyond "black-box" machine learning models, arXiv:2103.14490.

[21] J. Cerrillo and J. Cao, Non-Markovian Dynamical Maps: Numerical Processing of Open Quantum Trajectories, Phys. Rev. Lett. **112**, 110401 (2014).

[22] A. Gelzinis, E. Rybakovas and L. Valkunas, Applicability of transfer tensor method for open quantum system dynamics, J. Chem. Phys. **147**, 234108 (2017).

[23] S. Gherardini, A. Smirne, S. Huelga, and F. Caruso, Transfer-tensor description of memory effects in open-system dynamics and multi-time statistics, Quantum Sci. Technol. **7**, 025005 (2022).

[24] Z. Zhang, S. Yang, C. Liu, Y. Han, C. H. Lee, Z. Sun, G. Li and X. Zhang, Predicting quantum many-body dynamics with long short-term memory based neural networks, Chinese Phys. Lett. **37**, 018401 (2020).

[25] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, Pytorch: An imperative style, high-performance deep learning library, in *Advances in Neural Information Processing Systems* (MIT Press, Cambridge, MA, 2019), p. 8026.

[26] J. R. Johansson, P. D. Nation and F. Nori, Qutip 2: A python framework for the dynamics of open quantum systems, Comput. Phys. Commun. **184**, 1234 (2013).

[27] Since linear regressions require the weight matrix to be real we must write the wavefunction as a 2Ω vector and, hence, require 2Ω samples. For the density matrix one needs to evaluate $\Omega(1) + 1/2(\Omega^2 - \Omega)(2) = \Omega^2$ real numbers and hence we need Ω^2 samples. We subtract 1 from both as we have the normalization constraint.

[28] H.-P. Breuer and F. Petruccione, *The Theory of Open Quantum Systems* (Oxford University Press, Oxford, 2002).

[29] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, Cambridge, MA, 2016).

[30] R. L. Allen and D. Mills, *Signal Analysis: Time, Frequency, Scale, and Structure* (John Wiley & Sons, New York, 2004).

[31] S. Paeckel, T. Köhler, A. Swoboda, S. R. Manmana, U. Schollwöck, and C. Hubig, Time-evolution methods for matrix-product states, Ann. Phys. **411**, 167998 (2019).