

Learning Stochastic Dynamics with Statistics-Informed Neural Network

Yuanran Zhu^{*1}, Yu-Hang Tang^{†2}, and Changho Kim^{‡1}

¹Department of Applied Mathematics, University of California, Merced, Merced, CA 95343, USA

²Applied Mathematics and Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA

February 25, 2022

Abstract We introduce a machine-learning framework named statistics-informed neural network (SINN) for learning stochastic dynamics from data. This new architecture was theoretically inspired by a universal approximation theorem for stochastic systems introduced in this paper and the projection-operator formalism for stochastic modeling. We devise mechanisms for training the neural network model to reproduce the correct *statistical* behavior of a target stochastic process. Numerical simulation results demonstrate that a well-trained SINN can reliably approximate both Markovian and non-Markovian stochastic dynamics. We demonstrate the applicability of SINN to model transition dynamics. Furthermore, we show that the obtained reduced-order model can be trained on temporally coarse-grained data and hence is well suited for rare-event simulations.

1 Introduction

The use of machine learning (ML) techniques to model stochastic processes and time series data has seen many contributions in the past years. Two common strategies are to utilize neural networks (NN) to either *solve* or *learn* the associated differential equations.

The ‘solver’ strategy assumes that we know in advance the differential equation governing a dynamical system, and an NN is used to construct a proper numerical solver for the equation. A representative approach using the

^{*}yzhu56@ucmerced.edu

[†]Corresponding author, tang.maxin@gmail.com, tang@lbl.gov

[‡]ckim103@ucmerced.edu

solver strategy is the physics-informed neural network (PINN) [1, 2]. In the PINN approach, an NN serves as a solver that transforms initial and boundary conditions into approximate solutions. The NN solvers are trained using loss functions defined in terms of the underlying differential equations. Minimizing the augmented loss function steers the solver towards producing outputs that conform to the differential equations. More recent members within the PINN family includes sparse physics-informed neural network (SPINN) [3], parareal physics-informed neural network (PPINN) [4], and so on [5].

The ‘learning’ strategy, on the other hand, aims to learn the hidden dynamics from data using NN architectures. This strategy is adopted in the neural ordinary differential equations (NeuralODE) approach [6]. In this framework, we do not solve an equation directly, but rather trains an NN that computes the gradient of the state variable, where an adjoint dynamic system and a reverse-time ODE solver are adopted to facilitate backpropagation. After updating the parameters of the augmented dynamics, solutions to the differential equations can be found to reconstruct and extrapolate the hidden dynamics. Recent developments on top of NeuralODE include NeuralSDE [7, 8], Neural Jump SDE [9], NeuralSPDE [10], Neural Operators [11], infinitely deep Bayesian neural networks [12], *etc.*

Generally speaking, the inverse problem of learning an unknown dynamics is more challenging than solving or simulating a known dynamics, and this is particularly so for stochastic systems. The ever-growing abundance of data necessitates methods that can learn stochastic dynamics in a wide range of scientific disciplines such as, for example, molecular dynamics [13, 14, 15], computational chemistry [16], and ecology [17]. Aside from the aforementioned work based on ML methodologies, recent progress in this direction includes the work of Lu et al. [18, 19, 20] on the learning of interaction kernel of multi-particle systems with non-parametric methods, the kernel-based method [21, 22] for learning discrete non-Markovian time series, and various approximations of the Mori-Zwanzig equation for the learning of non-Markovian stochastic dynamics [13, 14, 23, 24] in molecular dynamics. Typically, these non-ML methods use sophisticated series expansion and regression techniques to learn and construct the desired stochastic model.

Although being theoretically complete and numerically successful within their own applicability, the complexity of such modeling processes makes it hard to extend these frameworks for high-dimensional stochastic dynamical systems or for systems that are highly heterogeneous.

The success of neural network models in dealing with high-dimensional problems for complex systems inspired us to use it to build a simple data-driven, extensible framework for modeling stochastic dynamics. This leads to the development of the statistics-informed neural network (SINN), which we now detail its construction and the main rationale behind it, as well as the major differences from the existing ML frameworks.

To learn stochastic dynamics with NNs, we first consider how to use deterministic NN frameworks such as the convolutional neural network (CNN) or recurrent neural network (RNN) to generate randomness. As an RNN-based model, SINN resolves this issue by feeding discrete Gaussian white noise as an input to the network.

The modeling capacity of this simple construction can be examined using the universal approximation theorem (UAT) for RNNs. Specifically, by mimicking the proof of the UAT for a one-layer RNN for arbitrary deterministic, open dynamical systems [25], we obtain a similar result stating that a one-layer RNN with Gaussian white-noise input can universally approximate arbitrary stochastic systems. We then use the long short-term memory (LSTM) [26] architecture as the building blocks for SINN in order to capture non-Markovian and memory effects that the underlying stochastic system may contain. Contrary to PINNs and the NeuralSDE, SINN is trained on the statistics, such as probability density function (PDF) and time autocorrelation functions (ACFs), of an *ensemble* of trajectories. In a short summary, SINN is mainly different from other ML frameworks in the following three aspects: (I) SINN is entirely equation-free. The training and the modeling do not rely on the underlying stochastic differential equation. (II) An RNN, instead of fully connected or convolutional layers, is used as the primary architecture of the model. (III) Instead of seeking a *pathwise* approximation to the stochastic dynamics, SINN constructs simulated trajectories that converge to the target ones in the sense of probability *measure* and *n*-th *moment*. The computational and modeling merit brought by these three features will be elaborated later with numerical supports.

Numerical experiments will be provided to demonstrate the capability of SINN in approximating Gaussian and non-Gaussian stochastic dynamics as well as its ability to capture the memory effect for non-Markovian systems. We also use SINN to build effective models for transition dynamics that often appears in computational chemistry. Our SINN model can be trained using time coarse-grained trajectories. This feature makes it an efficient simulator for rare events. Besides the application in physics, SINN provides a simple, flexible framework to model arbitrary stochastic processes, hence is generally applicable in the area of uncertainty quantification and time series modeling. Several simple test examples presented in Section 4.2 promisingly show its numerical advantages over established stochastic process modeling tools such as the transformed Karhunen-Loève or polynomial chaos expansion [27, 28, 23, 14].

This paper is organized as follows. In Section 2, we review the established universal approximation theorem for a single-layer RNN model and show that there is a natural extension of this theorem for stochastic systems driven by Gaussian and non-Gaussian white noise. Inspired by this theoretical result, in Section 3, we propose a statistics-informed neural network (SINN) and introduce different types of loss functions. The training method of SINN is provided in Section 4.1. Two simple test examples are presented in Section 4.2 to demonstrate that SINN can well approximate both Gaussian and non-Gaussian stochastic dynamics. In Section 4.3, we apply SINN to the transition dynamics modeling problem and use it as an effective rare-event simulator to evaluate transition rates. The main findings of this paper are summarized in Section 5. We provide a simple comparison between the equation-based modeling diagram and the equation-free modeling diagram that uses neural networks in Appendix A, the proof of main theorems given in the paper in Appendix B, and the further discussion on the predictability of SINN in Appendix C.

2 RNN as a Universal Approximator for Stochastic Dynamics

Recurrent neural network (RNN) is a good candidate architecture for learning the unknown dynamics of a physical system since there is a natural correspondence between the recurrent internal structure of RNN and the time-recursive update rule that quantifies the dynamics. In this section, we discuss the universal approximation properties of RNN for stochastic processes, in particular, for the discrete stochastic process corresponding to the numerical solution of stochastic differential equations (SDEs). We consider a one-layer *deterministic* RNN with *stochastic* input and show that if the model is wide enough, i.e. with a large number of hidden states, it can accurately approximate the finite-difference scheme of a time-homogeneous Markovian SDEs driven by Gaussian white noise.

We first review the universal approximation theorem (UAT) of RNN for deterministic dynamical systems established by Schäfer and Zimmermann in [25]. To this end, let us consider a one-layer RNN model given by the update rule:

$$\begin{aligned} s_{t+1} &= \sigma(As_t + Bx_t - \theta), \\ y_t &= Cs_t, \end{aligned} \tag{1}$$

where $s_t \in \mathbb{R}^H$ is the state vector of the RNN, $x_t \in \mathbb{R}^I$ is the input, σ is the activation function of the network, and $y_t \in \mathbb{R}^N$ is the output. The modeling parameters of this simple, one-layer RNN are the weight matrices $A \in \mathbb{R}^{H \times H}$, $B \in \mathbb{R}^{H \times I}$, $C \in \mathbb{R}^{N \times H}$, and the bias vector $\theta \in \mathbb{R}^H$. An immediate observation is that this update rule is very similar to the structure of a discrete, open dynamical system of the form:

$$\begin{aligned} s_{t+1} &= g(s_t, x_t), \\ y_t &= h(s_t), \end{aligned} \tag{2}$$

where $s_t \in \mathbb{R}^J$, $x_t \in \mathbb{R}^I$, $y_t \in \mathbb{R}^N$, and functions $g(\cdot) : \mathbb{R}^J \times \mathbb{R}^I \rightarrow \mathbb{R}^J$, $h(\cdot) : \mathbb{R}^J \rightarrow \mathbb{R}^N$. In fact, Schäfer and Zimmermann in [25] proved that the one-layer RNN with the update rule (1) can universally approximate the dynamics of the open dynamical system (2) with arbitrary accuracy. Their result can be restated as:

Theorem 1. (Schäfer & Zimmermann [25], UAT for the deterministic RNN). *Let $g(\cdot) : \mathbb{R}^J \times \mathbb{R}^I \rightarrow \mathbb{R}^J$ be measurable and $h(\cdot) : \mathbb{R}^J \rightarrow \mathbb{R}^N$ be continuous, the external input $x_t \in \mathbb{R}^I$, the inner state $s_t \in \mathbb{R}^J$, and the outputs $y_t \in \mathbb{R}^N$ ($t = 1, \dots, T$). Then any discrete, open dynamical system of the form (2) can be approximated by an RNN model of the type (1) arbitrarily accurate.*

Here we note that the exact definition of an RNN model of type (1) and the meaning of the arbitrarily accurate approximation are provided in Appendix B. The proof of this theorem was established based on the UAT for the feedforward neural networks. Here we mention some key points of Theorem 1: (I) The state vector $s_t \in \mathbb{R}^J$ in the dynamical system (2) is *not* the same as the state vector $s_t \in \mathbb{R}^H$ in RNN (1). In fact, to guarantee the accuracy of the

approximation, there normally is an enlargement of the state space in RNN, i.e. $H > J$. This is also to do with the following point. (II) The universal approximation is in the sense of matching the input x_t and the output y_t . This means that, with the exact same input x_t , the output y_t of the RNN should match closely with the output y_t of the dynamical system, while the state vectors s_t of these two systems may be different. (III) The UAT assumes *finite*-step input/output, i.e. $T < +\infty$.

The above UAT clearly indicates that even with a simple architecture such as the one-layer RNN, one can use it to universally approximate any open dynamical system of the general form (2). Although the theorem itself provides little guidance on how to *construct* such an RNN model for a specific dynamical system, and in practice we rarely use a wide-enough RNN to complete the computing task, the theorem indubitably shows the potential of the RNN architecture in modeling/learning dynamical systems. Now we show that by simply replacing the input vector x_t with independent and identically distributed (i.i.d.) Gaussian random variables while leaving all other structures unchanged, a similar UAT holds for the resulting stochastic RNN. This result can be stated as:

Theorem 2. (UAT for RNN with Gaussian inputs). Let $g(\cdot) : \mathbb{R}^J \times \mathbb{R}^I \rightarrow \mathbb{R}^J$ be continuously differentiable and $h(\cdot) : \mathbb{R}^J \rightarrow \mathbb{R}^N$ be continuous, the external input $x_t \in \mathbb{R}^I$ be i.i.d. Gaussian random variables, the inner state $s_t \in \mathbb{R}^J$, and the outputs $y_t \in \mathbb{R}^N$ ($t = 1, \dots, T$). Then any discrete, stochastic dynamical system of the form (2) can be approximated by an RNN model of the type (1) arbitrarily accurate asymptotically almost surely.

Theorem 2 can be proved using a probabilistic variant of the method proposed by Schäfer and Zimmermann [25]. The detailed proof is rather technical, hence will be deferred to Appendix B. An intuitive explanation why the probability of finding a suitable RNN that approximates (2) is only asymptotically 1 is that the key estimate which leads to Schäfer and Zimmermann's deterministic UAT (Theorem 1 in Appendix B) is based on the fact that the finite-step input vector x_t can be bounded in a compact domain of \mathbb{R}^I . Since the Gaussian random input $x_t \in \mathbb{R}^I$ is not compactly supported, but only asymptotically compactly supported, naturally for open dynamical system (2) with Gaussian stochastic input, we can only find its universal approximation asymptotically almost surely. This discussion also implies the following corollary:

Corollary 2.1. (UAT for RNN with compactly supported stochastic input). Let $g(\cdot) : \mathbb{R}^J \times \mathbb{R}^I \rightarrow \mathbb{R}^J$ be continuously differentiable, $h(\cdot) : \mathbb{R}^J \rightarrow \mathbb{R}^N$ be continuous, the external input x_t be i.i.d. random variables with compact support, the inner state $s_t \in \mathbb{R}^J$, and the outputs $y_t \in \mathbb{R}^N$ ($t = 1, \dots, T$). Then any discrete, stochastic dynamical system of the form (2) can be approximated by an RNN model of the type (1) arbitrarily accurate almost surely.

Proof. The proof is easy to obtain following the above arguments and Appendix B. □

As an example, consider i.i.d. stochastic input x_t being uniformly distributed in $[a, b]^I$. Then, with probability 1 one can find an RNN model of the type (1) that accurately approximates the open stochastic dynamics (2). For the

proposed RNN with stochastic inputs, what the RNN learns is the deterministic update rule that matches the input x_t and the output y_t . This is the fundamental reason why the proof of the UAT for RNN with deterministic inputs can be modified to obtain the UAT for RNN with stochastic inputs.

UAT for SDEs. The above UATs for RNN with stochastic input can be immediately applied to address the learning and approximation problem of SDEs. Consider Itô's diffusion on \mathbb{R}^d :

$$dX(t) = b(X(t)) + \sigma(X(t))d\mathcal{W}(t),$$

where $b(X(t)) \in \mathbb{R}^d$ is the vector field, $\sigma(X(t)) \in \mathbb{R}^{d \times m}$ is the diffusion matrix, and $\mathcal{W}(t) \in \mathbb{R}^m$ is the standard Wiener process. Any (explicit) finite difference scheme corresponding to Itô's diffusion can be written in the form of (2) with i.i.d. Gaussian input. For instance, the Euler-Maruyama scheme is given by

$$\begin{aligned} X(t + \Delta t) &= X(t) + \Delta t b(X(t)) + \sigma(X(t)) \sqrt{\Delta t} \xi(t) \\ &= g(X(t), \xi(t), \Delta t), \end{aligned} \tag{3}$$

where $\xi(t)$ are i.i.d. standard normal random variables. For fixed Δt , (3) corresponds to the update rule for the state vector in (2) where $x_t = \xi(t)$. Any phase space observables of the stochastic system $y_t = h(X_t)$ can be the output. Suppose $g(X(t), \xi(t), \Delta t)$ is continuously differentiable with respect to $X(t)$ and $\xi(t)$. Then, immediately we obtain the following UAT for the (discrete) Itô's diffusion:

Corollary 2.2. (UAT for the RNN of Itô's diffusion) The finite-step updates of the Euler-Maruyama scheme (3) with fixed step size Δt and continuously differentiable $g(X(t), \xi(t), \Delta t)$ can be approximated by an RNN model of the type (1) arbitrarily accurate asymptotically almost surely.

Since any time-inhomogeneous SDE admits a time-homogeneous extended dynamics by choosing $t = Y(t)$ as another state variable. Therefore, the universal approximation result naturally extends to time-inhomogeneous SDEs. Similarly, if a non-Markovian SDE admits a suitable embedded Markovian dynamics representation, one can approximate it with the RNN model (1) by using the latter representation. As an example, consider the generalized Langevin equation (GLE) [13] that is frequently used in the coarse-grained modeling of large-scale molecular systems:

$$\begin{cases} \dot{q} &= p, \\ \dot{p} &= F(q) - \int_0^t K(t-s)p(s)ds + f(t), \end{cases} \tag{4}$$

where $q(t)$, $p(t)$ are the *effective* position and momentum of a coarse-grained particle (assuming unit mass $m = 1$), $F(q)$ is the effective potential energy force, $f(t)$ is the fluctuation force which is often assumed to be a colored Gaussian stochastic process, and the time autocorrelation function of $f(t)$ yields the memory kernel $K(t) = \langle f(t)f(0) \rangle$. GLE (4) is a non-Markovian stochastic system because of the time-convolution term $\int_0^t K(t-s)p(s)ds$. It is shown in [13] and many recent works that for many molecular dynamical systems, the GLE (4) for a coarse-grained particle often admits a Markovian embedded approximation:

$$\begin{cases} dq &= p dt, \\ dp &= [F(q) + Z^T s] dt, \\ ds &= [Bs - QZp] dt + dW(t), \end{cases} \quad (5)$$

where vector s consists of auxiliary variables whose length depends on the order of approximation, and Z, B, Q are the corresponding auxiliary matrices. For such Markovian embedded dynamics, the proposed RNN has the capacity to approximate its output $q(t), p(t)$ according to Corollary 2.2.

3 Statistics-Informed Neural Network

The universal approximation theorem shows the potential of RNNs in simulating stochastic dynamics at the large-width limit. In this section, we put this theoretical insight into practice as a framework called the statistics-informed neural network (SINN) to learn stochastic dynamics from data. In particular, we use the long short-term memory (LSTM) architecture to capture non-Markovian memory effects. A set of training algorithms and statistics-based loss functions are devised to train SINN to reproduce the statistical characteristics of a target stochastic system.

3.1 Model Architecture

A graphical illustration of the SINN architecture is shown in Figure 1. The network consists of a multi-layer LSTM component to learn the temporal dynamics of stochastic processes, and a dense layer attached to the output gate of the LSTM as a ‘read-out’ device. Dropout layers can be optionally placed between the layers to control overfitting.

As suggested by the UAT, we use a stream of i.i.d. random numbers as the input to the model, which only carries out deterministic operations, in order to generate different realizations of a stochastic process. The preferred distributions for the input noise are the ones with maximum entropy. This means the normal distribution for outputs with infinite support, the uniform distribution for outputs with compact support, and the exponential distribution for outputs with support on \mathbb{R}^+ . From another perspective, the i.i.d. noise sequences can be viewed as the entropy source for SINN, which in turn can be viewed as a transformer between the input and output stochastic

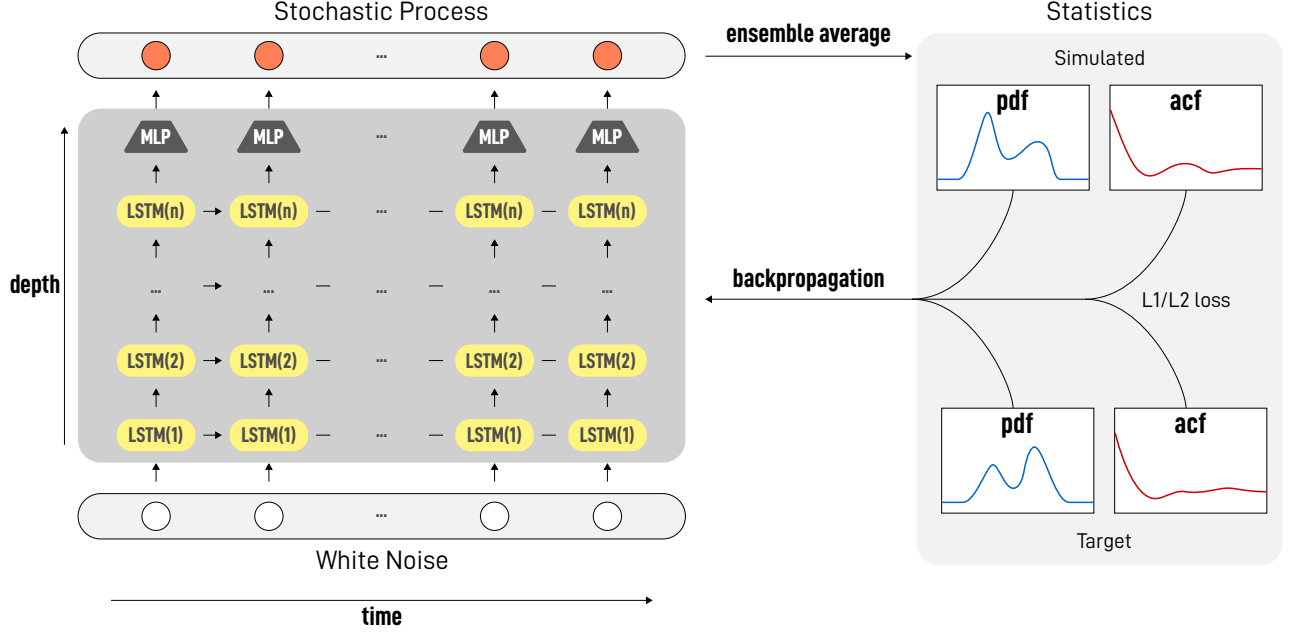


Figure 1: SINN architecture

processes. Since information can be lost during the calculation, the maximum entropy distribution ensures that the transformation process will not starve in terms of entropy. On the other hand, the maximum entropy principle also implies that this is the best choice when we assume minimum prior knowledge about the stochastic process.

If the white noise sequence is denoted as ξ_t , in the first LSTM layer, the output is computed as:

$$f_t^{(1)} = \sigma_g(W_f \xi_t + U_f h_{t-1}^{(1)} + b_f), \quad (6)$$

$$i_t^{(1)} = \sigma_g(W_i \xi_t + U_i h_{t-1}^{(1)} + b_i), \quad (7)$$

$$o_t^{(1)} = \sigma_g(W_o \xi_t + U_o h_{t-1}^{(1)} + b_o), \quad (8)$$

$$\tilde{c}_t^{(1)} = \sigma_c(W_c \xi_t + U_c h_{t-1}^{(1)} + b_c), \quad (9)$$

$$c_t^{(1)} = f_t^{(1)} \circ c_{t-1}^{(1)} + i_t^{(1)} \circ \tilde{c}_t^{(1)}, \quad (10)$$

$$h_t^{(1)} = o_t^{(1)} \circ \sigma_h(c_t^{(1)}), \quad (11)$$

where $f_t^{(1)}$, $i_t^{(1)}$, $o_t^{(1)}$ are known as the forget gate, input gate and the output gate of the first layer LSTM, respectively, $\tilde{c}_t^{(1)}$ is the cell input activation, and $c_t^{(1)}$ is the cell state [26]. Then the first-layer output $h_t^{(1)}$ replaces ξ_t as the input for the second layer and so on. The final output χ_t is then calculated as $\chi_t = W_m h_t^{(n)}$, where $h_t^{(n)}$ is the output of n -th LSTM layer, $W_m \in \mathbb{R}^x$, and x is the size of the output vector χ_t . Here we emphasize that only the input sequence is random, while the entire SINN model is deterministic. This makes training of the network very efficient and straightforward.

3.2 Loss Function.

Rather than seeking a pathwise approximation to the stochastic dynamics, we compare various density functions and the statistics of the trajectories obtained from SINN with those for the target processes. By doing so, we avoid tracking and storing the input Gaussian white noise used in generating the target processes. Moreover, the input noise of SINN can also be time coarse-grained if the output matches the coarse-grained target processes. Specifically, we use a linear combination of L_p norms between the following statistical quantities of the training and simulated processes as detailed below.

Autocorrelation Function (ACF). The ACF of a sequence X is a function of lag τ defined as

$$\text{ACF}_X(\tau) = \mathbb{E}[X_t, X_{t+\tau}]. \quad (12)$$

Two common approaches exist for computing the ACF of discrete time series data. The first approach, which we call the *brute force* method, simply uses the definition in (12) to compute $\text{ACF}(\tau)$ for every τ . For a sequence of length n , computing its ACF up to $\tau = n$ using brute force requires $O(n^2)$ operations. The second approach, which we call the *fast Fourier transform (FFT)* method, uses the Wiener-Khinchin theorem to efficiently compute the ACF using the Fourier transform of the sequence as $\text{ACF}_X = \text{FFT}^{-1}(\text{FFT}(X) \cdot \text{FFT}(X)^*)$, where the asterisk (*) denotes the complex conjugate. The FFT approach requires only $O(n \log n)$ operations for computing the ACF up to $\tau = n$. However, due to the periodicity assumption as implied by the Fourier transformation, the computed ACF can deviate considerably from the ground truth for large τ . The problem is particularly serious if the ACF does not decay close enough to zero at the length of the sequence data. In the following numerical examples, both the brute force and FFT approaches are used as appropriate. They also permit efficient backpropagation of ACF-based losses to the NN model using popular tensor algebra libraries such as PyTorch [29] and JAX [30].

Probability Density. Binning-based probability density function (PDF) estimators are not differentiable due to the discrete nature of the histogram operation. Therefore, we compute and compare the empirical PDFs of both the target and simulated trajectories using kernel density estimation (KDE):

$$\widehat{f}_h(x) = \frac{1}{n} \sum_i^n K_h(x - x_i), \quad (13)$$

where K is a non-negative *kernel* while h is a smoothing parameter. $K_h(d) \doteq \frac{1}{h} K(\frac{d}{h})$ is the scaled kernel. We use the Gaussian kernel $K^{\text{Gauss}}(d) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{d^2}{2})$ with a bandwidth parameter $h = |X|^{-\frac{1}{5}}$ [31], where $|X|$ is the length of the sequence X .

4 Numerical Experiments

4.1 Training Method

The SINN model is trained with stochastic gradient descent (SGD) using the Adam optimizer. The learning rate is set to be 10^{-3} with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ which can be adjusted with respect to the magnitude of the error. Training and validation losses are tracked throughout the training process of 100 steps. *Different* sample trajectories of the Gaussian white noise are introduced throughout the training process to ensure that the learned SINN model is generalizable and not overfitting to a particular realization of the stochastic processes (see the choice of Input Sequence). The training ‘batch’ and the target data consist of 400 sequences, while the validation set consists of 800 sequences.

Evaluation of Loss. Instead of comparing the ACFs over the entire lag range $1, \dots, n$, we randomly select a number of lag values τ_1, \dots, τ_m with $m \ll n$ during each SGD step and compare the ACFs at the selected lags. Typical values of m is around 20. This procedure is particularly needed when the brute force ACF estimator is employed due to its high computational cost.

Input Sequence. The white noise sequence, which serves as the input to the SINN model as described earlier in Section 3.1, is always created *afresh* at each SGD iteration.

Validation Sequence. The validation data is a fixed number of target sequences that are used to monitor training and diagnose overfitting. The losses computed on the validation sequence do not participate in backpropagation.

Computational Cost. All computations are performed using a workstation with 16 AMD Zen3 cores at 3.0 GHz and one NVIDIA A100 accelerator. A SINN model with 2 LSTM layers each with a size of 25 hidden states can be trained 1200 SGD iterations within approximately 1 minute.

Before presenting numerical results, we comment in advance on the modeling advantages of SINN, which echos the three architectural differences we mentioned in Introduction. First of all, SINN is essentially equation-free since the modeling and training of SINN use no equations (see more in Appendix A). This feature allows the generated dynamics to have *tunable* coarse-grained time scales, which makes it particularly suitable for capturing the long-time behavior of stochastic systems. Further discussion in this regard is provided in Section 4.3. Secondly, the architecture of SINN is based on RNN, hence what we learned is the *deterministic* update rule for SDEs which is similar to the Euler-Maruyama scheme (3). It is very natural to do time-domain extrapolation and expect a certain predictability of SINN (see more in Appendix C). Lastly, the convergence we seek is defined in terms of statistical moments and

probability measure. In many cases, such as the transition dynamics simulation in Section 4.3, it can be shown that such convergence is already enough to capture the physics we are interested in.

4.2 Validation Cases

We present two test cases here to show that SINN can well approximate Gaussian and non-Gaussian stochastic dynamics.

Ornstein-Uhlenbeck process. Consider the Ornstein-Uhlenbeck (OU) process given by Itô-type SDE:

$$\frac{dx}{dt} = -\theta x + \sigma \xi(t), \quad (14)$$

where σ and θ are positive parameters and $\xi(t)$ is the standard Gaussian white noise with correlation function $\langle \xi(t)\xi(s) \rangle = \delta(t-s)$. As is well-known, the OU process is ergodic and admits a stationary (equilibrium) Gaussian distribution $\mathcal{N}(0, \sigma^2/2\theta)$. In addition, the ACF of $x(t)$ in the equilibrium is an exponentially decaying function, $C(\tau) = \langle x(t+\tau)x(t) \rangle = \frac{\sigma^2}{2\theta} e^{-\theta\tau}$. With the parameter values $\sigma = 0.5$ and $\theta = 1$, we generate approximated dynamics for $x(t)$ by using the proposed SINN architecture with two LSTM layers and one hidden state. The analytical stationary ACF and the equilibrium PDF are used as the target and the loss function to train the NN parameters. Figure 2 clearly shows that the statistics of the OU process is faithfully reproduced by the trajectories simulated by SINN. Here we note that the time step of SINN ($dt = 0.2$) is much larger than the MD time step used ($\Delta t = 10^{-3}$). This time coarse-grained feature of SINN is one of its main characteristics which makes it particularly useful in rare-event simulations (See Section 4.3).

The error plot also shows that the generalization error gets smaller while at the same time it keeps the same magnitude with respect to the training error during the training process. Hence over-fitting does not happen. Since the correlation function and the equilibrium probability density *uniquely* determined a Gaussian process $x(t)$, we conclude that the stochastic process generated by SINN faithfully represents the dynamics of an OU process.

Langevin Dynamics. Consider the Langevin dynamics for an anharmonic oscillator:

$$\begin{cases} \dot{q} = p, \\ \dot{p} = -V'(q) - \gamma p + \sigma \xi(t), \end{cases} \quad (15)$$

where $V(q) = \frac{\alpha}{2}q^2 + \frac{\theta}{4}q^4$ is the Fermi-Pasta-Ulam (FPU) potential and $\xi(t)$ is Gaussian white noise. Parameters γ and σ are linked by the fluctuation-dissipation relation $\sigma = (2\gamma/\beta)^{1/2}$, where β is proportional to the inverse of the thermodynamic temperature. The Langevin dynamics (15) for the FPU oscillator admits the Gibbs-form equilibrium

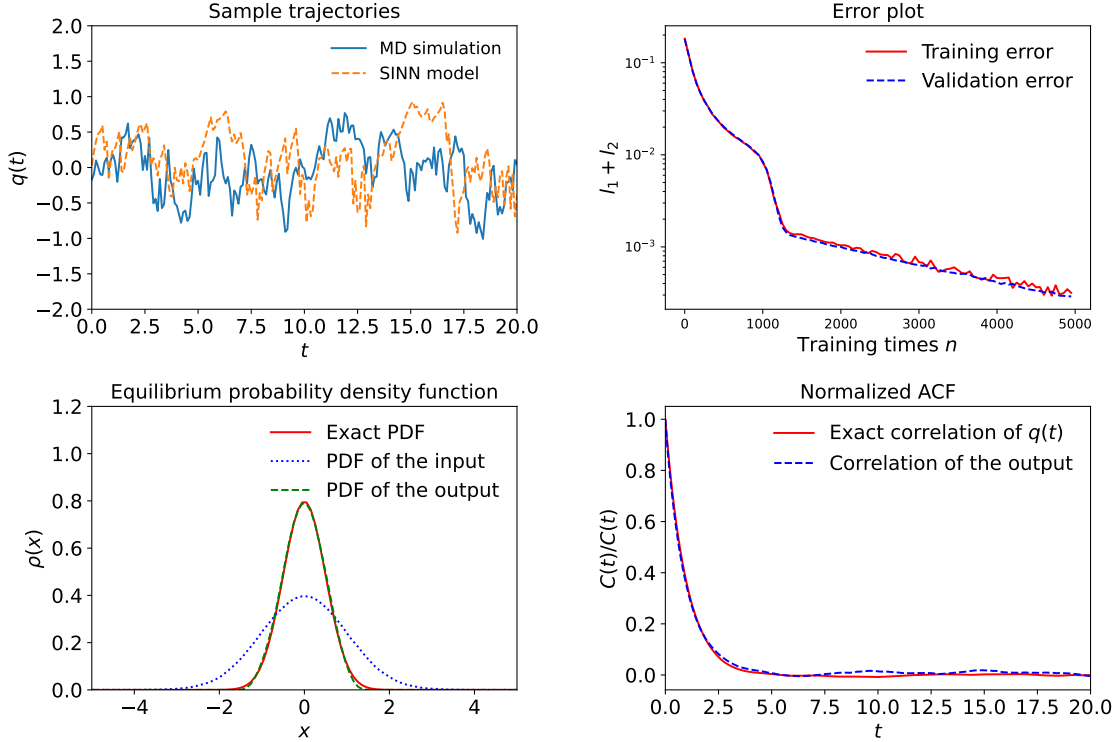


Figure 2: Comparison of the dynamics of $q(t)$ generated by MD simulation and the SINN model. The MD simulation results of the sample trajectories (Top Left) are obtained using the Euler-Maruyama scheme for (14) with step size $\Delta t = 10^{-3}$. The target processes are filtered sample trajectories of $q(t)$ with step size $dt = 0.2$. Note that sample trajectories simulated by SINN thus have natural coarse-grained time scale $dt \gg \Delta t$. The output statistics (PDF and ACF) are evaluated by taking the ensemble average over 5000 trajectories which are generated using a new set of Gaussian white noise as the SINN input.

distribution $\rho_{eq} \propto e^{-\beta H}$, where $H = \frac{p^2}{2} + V(q)$. We use the same SINN model with two LSTM layers and one hidden state to generate approximated dynamics for $q(t)$. Contrary to the OU process case, here we do not have an analytical expression for the ACF of $q(t)$. Hence, by using the sample trajectories of $q(t)$ obtained by numerically solving (15), we determine the ACF and then calculate the loss. Since $q(t)$ is no longer a Gaussian process, its PDF and ACF *cannot* uniquely determine its dynamics. To ensure the validity of the model, we include the stationary ACF for $q^2(t)$ as an extra loss function to train the neural network. The simulation results are presented in Figure 3. It shows that the SINN architecture can well approximate the dynamics of the non-Gaussian process (15).

Remark 1. According to the Kramers–Moyal expansion [32] for an arbitrary Markov process, by including more higher-order moments, one can construct a better approximation to the master equation corresponding to the stochastic process. This is the reason why we added the ACF for $q^2(t)$ as an additional criterion to train the model for non-Gaussian dynamics. We note that higher-order moments such as $\langle q^4(t)q^4(0) \rangle$ can also be easily added into the total loss function. Due to this extensibility of SINN, it is fairly simple to ensure that the higher-order information of the generated stochastic process faithfully approximates that of the original stochastic process. We note that this is not generally guaranteed by established methods in stochastic modeling such as the transformed Karhunen–Loève

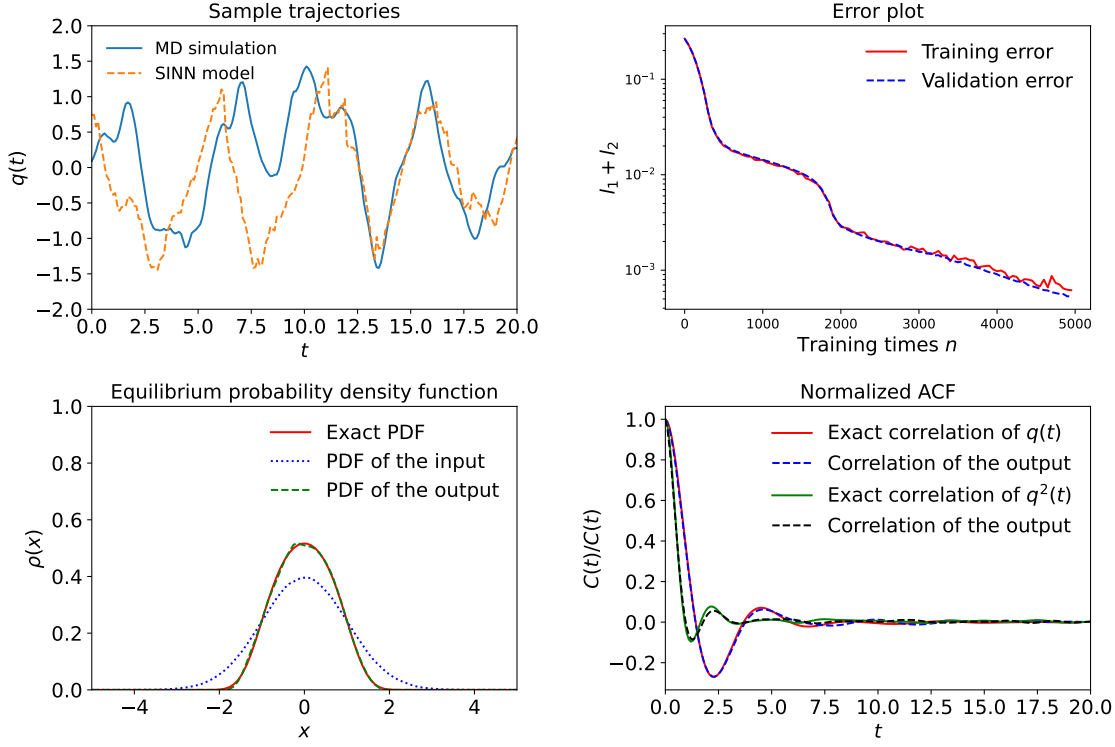


Figure 3: Comparison of the dynamics of $q(t)$ generated by MD simulation and the SINN model. The setting is exactly the same as the one used in Figure 2 except that we added the ACF $\langle q^2(t)q^2(0) \rangle$ into the total loss function.

or polynomial chaos expansion [27, 28].

Remark 2. As a data-driven framework, the *equation-free* feature of SINN makes it desirable for applications in reduced-order modeling problems, where the effective dynamics for the low-dimensional resolved observables is generally hidden and has to be extracted from the underlying high-dimensional dynamical systems through coarse-graining procedures. Generally speaking, the dimension-reduction leads to memory effects in the reduced-order dynamics. We emphasize that these effects can be captured by the LSTM modules of SINN. The previously studied Langevin dynamics (15) provides a good example for this. The system as a whole is Markovian for the state variables $\{p(t), q(t)\}$. However, the reduced-order effective dynamics for the observable $q(t)$ alone is non-Markovian. Using the Mori-Zwanzig framework [33, 14], one can derive the following evolution equation for $q(t)$:

$$\frac{d}{dt}q(t) = \Omega q(t) + \int_0^t K(t-s)q(s)ds + f(t), \quad (16)$$

where Ω is a modeling constant, $K(t)$ is the memory kernel, and $f(t)$ is the stochastic fluctuation force. In (16), the memory effect is encoded by the convolution integral $\int_0^t K(t-s)q(s)ds$, where $K(t)$ is generally unknown. The SINN model provides a novel mechanism to quantify this complicated memory effect by “storing” it within the LSTM cell state vectors c_t , which can be learned through the simulation data.

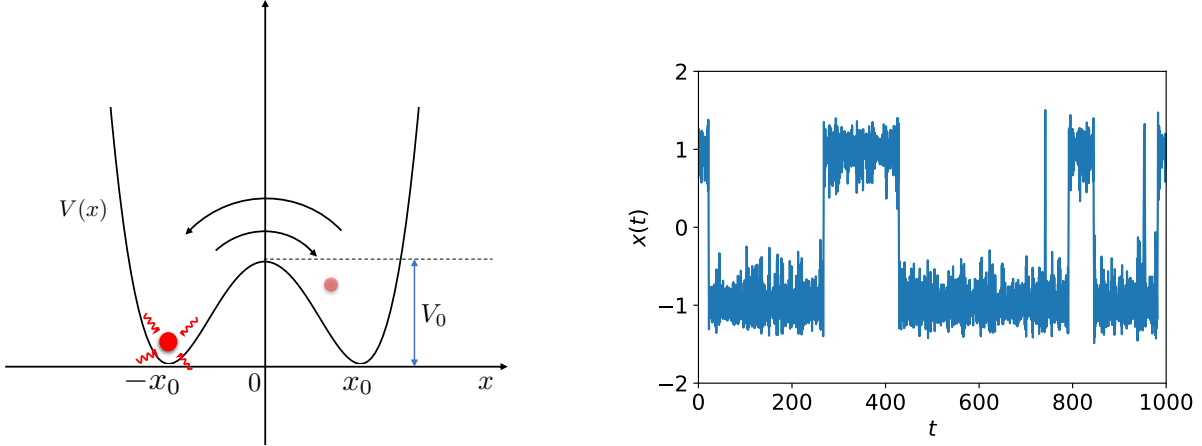


Figure 4: (Left) Schematic illustration of the hopping events between two states for the reaction coordinate $x(t)$. Through thermodynamic interactions with the environment, an imaginary particle may gain enough energy to cross the energy barrier and make a transition from one well to the other. (Right) Sample trajectory of $x(t)$ simulated using (17). The modeling parameters are chosen to be $V_0 = 5$, $x_0 = 1$, and $\beta = 1$. One can see that hopping between these two states is a rare event for the given height of energy barrier.

4.3 Transition Dynamics Modeling and Rare-event Simulations

In this section, we use a practical example to further test the capabilities of SINN in modeling non-Gaussian, non-Markovian stochastic dynamics and verify whether it has long-time predictability. The application we consider involves the study of the transition dynamics and the associated rare-event simulations. The physical motivation for the transition dynamics stems from the calculation of the reaction rate of a chemical reaction. While this is an important problem, determining the reaction rate numerically becomes extremely difficult when the reaction is rare, i.e. happens in a low probability, since it requires a long simulation time to fully capture the reaction dynamics, which often exceeds the typical time scale of MD simulations. In statistical physics and physical chemistry, chemical reactions are often explained using a transition dynamics for the reaction coordinate $x(t)$ [16, 34]. To this end, consider a toy example (see its schematic illustration in Figure 4) for the transition dynamics given by the Langevin equation for a double-well system:

$$\begin{cases} \dot{x} = p, \\ \dot{p} = -V'(x) - \gamma p + \sigma \xi(t), \end{cases} \quad (17)$$

where $V(x) = V_0 [1 - (x/x_0)^2]^2$ is a symmetric double-well potential with depth V_0 and two basins around x_0 and $-x_0$. The two potential wells correspond to the two states of the reaction coordinate $x(t)$, which can be, say the different dihedral angles of n -butane in the isomerization process [35]. We aim to use SINN to construct an effective, reduced-order model for the reaction coordinate $x(t)$ based on the *short-time* simulation data. Once the effective model is built, one may use it as a surrogate model to perform Monte-Carlo simulations or to extrapolate the *long-time* trajectories of $x(t)$. From this, the transition rate of the rare event can be calculated in an economical manner. In

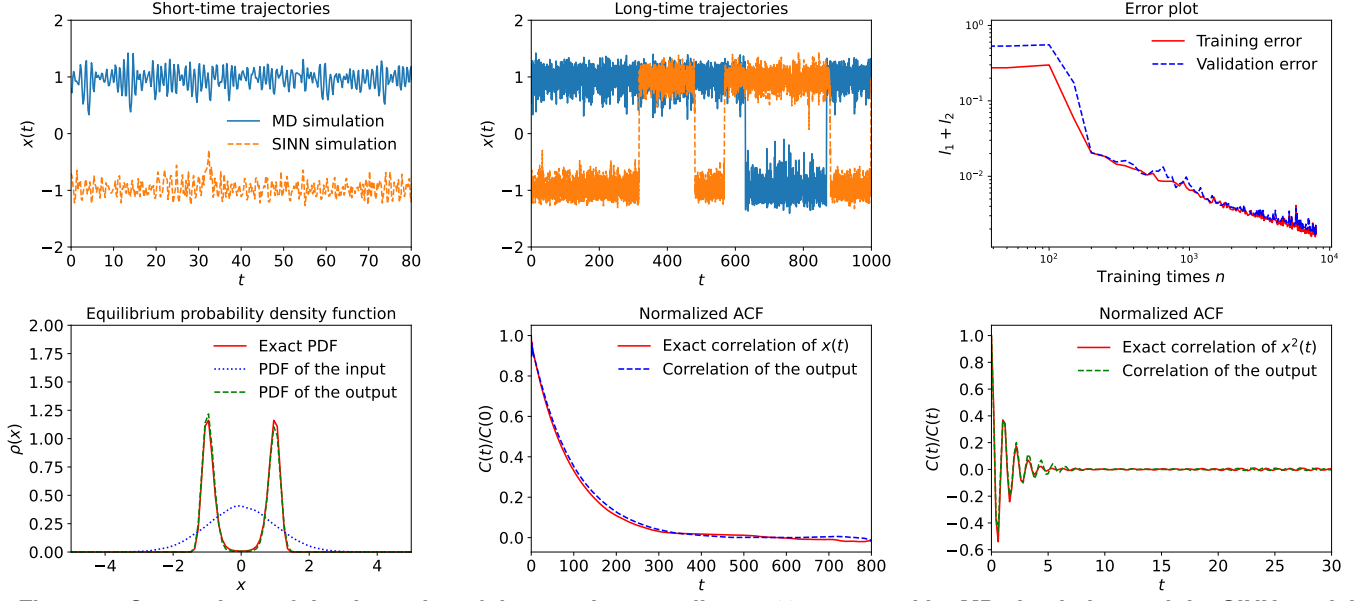


Figure 5: Comparison of the dynamics of the reaction coordinate $x(t)$ generated by MD simulation and the SINN model. The exact statistics, including the PDF and the ACFs for $x(t)$ and $x^2(t)$, are obtained through the MD simulation of (17) by averaging over 5×10^4 trajectories. The statistics for the SINN outputs are similarly calculated.

practical applications, one normally obtain the sample trajectories for the reaction coordinate $x(t)$ from large-scale MD simulations that model the whole physical system. Here, we use a toy model (17) to quickly generate sample trajectories of $x(t)$ for the purpose of demonstrating the learning ability of SINN and its validity in simulating rare events.

Since the transition dynamics is more complicated than the examples considered in Section 4.2, we employ an SINN model with 2 LSTM layers and 25 hidden states to build the stochastic model. This structure may not be optimal in terms of complexity or efficiency, but is found to be sufficient for our study. To train the neural network, we solve (17) using the Euler-Maruyama scheme with step size $\Delta t = 10^{-3}$ and obtain 400 sample trajectories. To use the time coarse-grained trajectories as the target, we use time points with a fixed step size $dt = 0.2$ in each trajectory. The length of each trajectory is 400 steps, i.e. the simulated dynamics is in time domain $[0, 80]$. The equilibrium PDF and the ACFs for $x(t)$ and $x^2(t)$ are used to construct the loss function. The training results are presented in Figure 5. One can see that SINN yields an overall good approximation for the transition dynamics. Remarkably, it actually simulates hopping events between two states. A more qualitative evaluation of SINN on describing the transition dynamics relies on the calculation of the transition rate from the generated samples trajectories, which can be found in the last paragraph of this section.

Remark. From the comparison of the long-time trajectories and the normalized ACF shown in Figure 5, we can see that the trained SINN model yields a good prediction/extrapolation of the long-time dynamics of $x(t)$. This is remarkable because training was carried out only using data from the time domain $[0, 80]$. In Appendix C, we

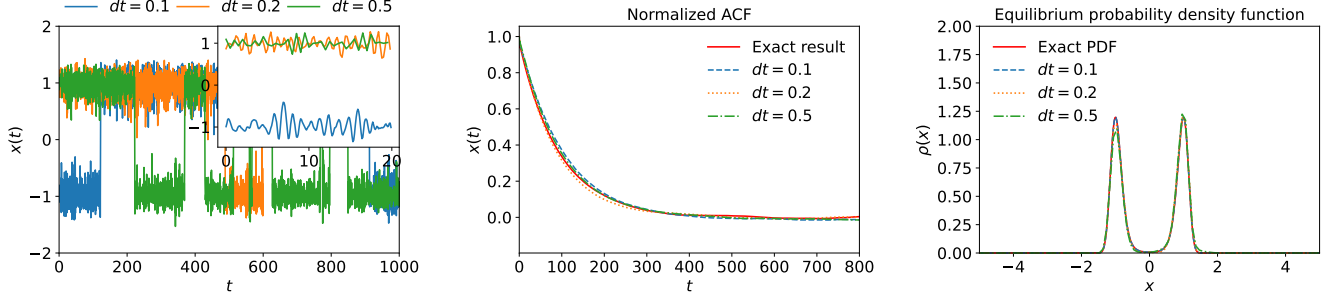


Figure 6: Comparison of the SINNs with different coarse-grained time scales dt . As dt increases, the local information gets gradually filtered as shown in the short-time trajectories (see the inset of the left panel). However, the ACF and PDF of the simulated trajectories remain essentially the same.

provide numerical results to further discuss the long-time predictability of SINN and the numerical convergence of the training result.

SINN as a Coarse-grained Time Integrator. The SINN models we have used so far are trained using the coarse-grained sample trajectories of $x(t)$ with the time step size $dt = 0.2$, which is much larger than the MD integration time step size $\Delta t = 10^{-3}$. As a result, the output of SINN, i.e. the approximated trajectories of $x(t)$, has the same step size, which makes the SINN a natural coarse-grained time integrator for the reduced-order dynamics of $x(t)$. This coarse-grained nature of our SINN models provides an efficient algorithm to generate the approximated long-time trajectories of $x(t)$ because the sampling gets 200 times faster. For the calculation of physical quantities such as the reaction rate where the local fast-time dynamics becomes irrelevant, this gives huge computational advantages. In Figure 6, we compare sample trajectories of $x(t)$ generated by well-trained SINNs with different time step sizes $dt = 0.1, 0.2, 0.5$. For these three time-scales, the well-trained SINNs all reproduced the correct hopping dynamics. It is also clearly observed that while fast-time dynamics is filtered as dt increases, the statistics (i.e. PDF and ACFs) of these trajectories remained essentially the same. This demonstrates the advantages of SINN in modeling the transition dynamics. On one hand, SINN can capture the memory effect of non-Markovian dynamics through the LSTM structure (see discussion in Appendix C). On the other hand, it is also a coarse-grained time integrator while other stochastic models such as the Mori-Zwanzig equation/ GLE and NeuralSDE are not.

Calculation of the Transition Rate. We use our SINN model as the simulator for the transition dynamics and assess how well it predicts the transition rate for rare events. To calculate the transition rate between the two states, we first divide the phase space for $x(t)$ into two regions: $A = (-\infty, 0]$ and $B = (0, +\infty)$. Obviously, $-x_0 \in A$ and $x_0 \in B$. Consider the equilibrium time correlation function $C_{A,B}(t)$ defined by:

$$\frac{C_{A,B}(t)}{C_A} = \frac{\langle h_A(x(0))h_B(x(t)) \rangle}{\langle h_A(x(0)) \rangle}, \quad (18)$$

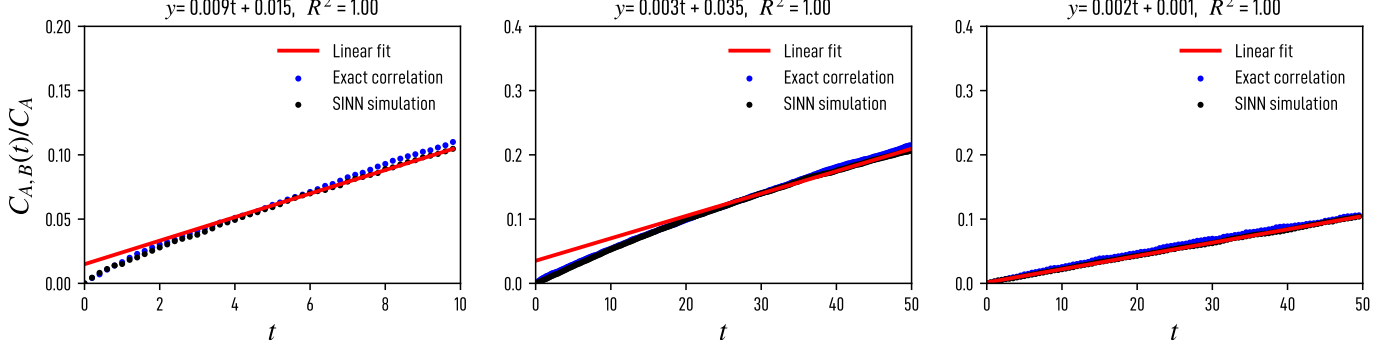


Figure 7: Prediction of the transition rate using SINN as the simulator for the rare events. The time profiles of the equilibrium correlation function $C_{A,B}(t)/C_A$ for double-well dynamics (17) are plotted for different values of the barrier depth V_0 and the coarse-grained time scale dt : (left) $V_0 = 4$, $dt = 0.2$; (middle) $V_0 = 5$, $dt = 0.2$; (right) $V_0 = 6$, $dt = 0.5$. The results obtained from SINN are compared with the numerical simulation results obtained from long-time MD trajectories. The linear regression is used in fitting $C_{A,B}(t)/C_A$ in between the transient time scale τ_{mol} and the exponential relaxation time scale τ_{rxn} in order to evaluate k_{AB} . The specific time domains for the linear regression are chosen to be (from left to right) $[5, 10]$, $[25, 50]$ and $[25, 50]$, respectively. R^2 is the coefficient of determination.

where $h_A(x(t))$ is the characteristic function of the configuration space satisfying $h_A(x(t)) = 1$ if $x(t) \in A$ and $h_A(x(t)) = 0$ if $x(t) \notin A$; $h_B(x(t))$ can be similarly defined. Thus, the ratio (18) is the probability of finding the system in state B after time t when the system is initially at state A . As a result, the transition rate from A to B can be calculated as [36, 37]:

$$k_{AB} = \frac{d}{dt} \frac{C_{A,B}(t)}{C_A}, \quad \tau_{mol} < t \ll \tau_{rxn}, \quad (19)$$

which is the slope of the curve after a short transient time scale τ_{mol} and before its exponential relaxation time $\tau_{rxn} = 1/(k_{AB} + k_{BA})$.

The numerical results are summarized in Figure 7. The time profiles of the equilibrium time correlation function match well with the correlation results obtained by MD simulations. The resulting values of k_{AB} are approximately estimated to be 0.009, 0.003, and 0.002 for transition dynamics with energy barrier height values $V_0 = 4, 5$, and 6, respectively, which agree with the values obtained by MD simulations. By calculating the transition rate using SINNs trained with coarse-grained trajectories with different values of dt , we also observe that the time coarse-graining of the trajectory does not significantly influence the final calculation result of the reaction rate, which is consistent with our previous analysis. The successful prediction of the transition rate k_{AB} indicates that, with the equilibrium PDF and ACFs for $x(t)$ and $x^2(t)$, it is *practically* sufficient to create a reliable numerical approximate for the reduced-order dynamics $x(t)$ using SINN, although, in principle, this information is not enough to *theoretically* guarantee the uniqueness of the non-Gaussian process.

5 Conclusion

In this paper, we introduced a statistics-informed neural network (SINN) for learning stochastic dynamics. The establishment of SINN is theoretically inspired by the universal approximation theorem for the one-layer RNN with stochastic inputs. This new model uses Gaussian white-noise sequences as the input and layers of long short-term memory (LSTM) cells as the functional units to generate output sequences. The statistics of the target stochastic process, such as equilibrium probability density and time autocorrelation functions of different order, are used in the loss function to train the parameters. SINN has a relatively simple architecture, which is easy to implement and train, that applies deterministic models to random input. Numerical simulation results have shown that SINN can effectively approximate Gaussian and non-Gaussian dynamics for both Markovian and non-Markovian stochastic systems. The successful application of SINN in modeling the transition dynamics clearly indicates that it can serve as a useful surrogate model to simulate rare events. Moreover, the coarse-grained nature and the long-time predictability of SINN makes it an efficient and reliable framework for reduced-order modeling. Further applications and extensions of this framework in the general area of stochastic modeling, uncertainty quantification, and time series analysis can be expected.

6 Acknowledgment

This work was supported in part by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Scientific Discovery through Advanced Computing (SciDAC) Program through the FASTMath Institute under Contract No. DE-AC02-05CH11231 at Lawrence Berkeley National Laboratory. Y. Zhu would like to thank Prof. Huan Lei for the valuable discussion of the transition dynamics.

References

- [1] Xiaoli Chen, Jinqiao Duan, and George Em Karniadakis. Learning and meta-learning of stochastic advection–diffusion–reaction systems from sparse measurements. *European Journal of Applied Mathematics*, 32(3):397–420, 2021.
- [2] Dongkun Zhang, Lu Lu, Ling Guo, and George Em Karniadakis. Quantifying total uncertainty in physics-informed neural networks for solving forward and inverse stochastic problems. *Journal of Computational Physics*, 397:108850, 2019.
- [3] Amuthan A. Ramabathiran and Prabhu Ramachandran. SPINN: Sparse, physics-based, and partially interpretable neural networks for PDEs. *Journal of Computational Physics*, 445:110600, 2021.

- [4] Xuhui Meng, Zhen Li, Dongkun Zhang, and George Em Karniadakis. PPINN: Parareal physics-informed neural network for time-dependent PDEs. *Computer Methods in Applied Mechanics and Engineering*, 370:113250, 2020.
- [5] Dongkun Zhang, Ling Guo, and George Em Karniadakis. Learning in modal space: Solving time-dependent stochastic PDEs using physics-informed neural networks. *SIAM Journal on Scientific Computing*, 42(2):A639–A665, 2020.
- [6] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural Ordinary Differential Equations. *arXiv:1806.07366 [cs, stat]*, December 2019.
- [7] Xuanqing Liu, Tesi Xiao, Si Si, Qin Cao, Sanjiv Kumar, and Cho-Jui Hsieh. Neural SDE: Stabilizing Neural ODE Networks with Stochastic Noise. *arXiv:1906.02355 [cs, stat]*, June 2019.
- [8] Xuechen Li, Ting-Kam Leonard Wong, Ricky TQ Chen, and David K Duvenaud. Scalable gradients and variational inference for stochastic differential equations. In *Symposium on Advances in Approximate Bayesian Inference*, pages 1–28. PMLR, 2020.
- [9] Junteng Jia and Austin R. Benson. Neural Jump Stochastic Differential Equations. *arXiv:1905.10403 [cs, stat]*, January 2020.
- [10] Cristopher Salvi, Maud Lemerrier, and Andris Gerasimovics. Neural stochastic partial differential equations, 2021.
- [11] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces, 2021.
- [12] Winnie Xu, Ricky T. Q. Chen, Xuechen Li, and David Duvenaud. Infinitely Deep Bayesian Neural Networks with Stochastic Differential Equations. *arXiv:2102.06559 [cs, stat]*, August 2021.
- [13] H. Lei, N.A. Baker, and X. Li. Data-driven parameterization of the generalized Langevin equation. *Proc. Natl. Acad. Sci.*, 113(50):14183–14188, 2016.
- [14] Yuanran Zhu and Huan Lei. Effective Mori-Zwanzig equation for the reduced-order modeling of stochastic systems. *Discrete & Continuous Dynamical Systems - S*, 2021.
- [15] Weiqi Chu and Xiantao Li. The Mori-Zwanzig formalism for the derivation of a fluctuating heat conduction model from molecular dynamics. *Communications in Mathematical Sciences*, 17:539–563, 2019.
- [16] Mark Tuckerman. *Statistical mechanics: theory and molecular simulation*. Oxford university press, 2010.

- [17] Yael Katz, Kolbjørn Tunstrøm, Christos C Ioannou, Cristián Huepe, and Iain D Couzin. Inferring the structure and dynamics of interactions in schooling fish. *Proceedings of the National Academy of Sciences*, 108(46):18720–18725, 2011.
- [18] Fei Lu, Mauro Maggioni, and Sui Tang. Learning interaction kernels in heterogeneous systems of agents from multiple trajectories. *J. Mach. Learn. Res.*, 22:32–1, 2021.
- [19] Quanjun Lang and Fei Lu. Learning interaction kernels in mean-field equations of 1st-order systems of interacting particles. *arXiv preprint arXiv:2010.15694*, 2020.
- [20] Fei Lu, Mauro Maggioni, and Sui Tang. Learning interaction kernels in stochastic systems of interacting particles from multiple trajectories. *Foundations of Computational Mathematics*, pages 1–55, 2021.
- [21] Faheem Gilani, Dimitrios Giannakis, and John Harlim. Kernel-based prediction of non-Markovian time series. *Physica D: Nonlinear Phenomena*, 418:132829, 2021.
- [22] John Harlim, Shixiao W Jiang, Senwei Liang, and Haizhao Yang. Machine learning for prediction with missing dynamics. *Journal of Computational Physics*, 428:109922, 2021.
- [23] Yuanran Zhu and Daniele Venturi. Generalized langevin equations for systems with local interactions. *Journal of Statistical Physics*, pages 1–31, 2020.
- [24] Z. Li, , X. Bian, X. Li, and G. E. Karniadakis. Incorporation of memory effects in coarse-grained modeling via the Mori-Zwanzig formalism. *J. Chem. Phys*, 143:243128, 2015.
- [25] Anton Maximilian Schäfer and Hans Georg Zimmermann. Recurrent neural networks are universal approximators. In *International Conference on Artificial Neural Networks*, pages 632–640. Springer, 2006.
- [26] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [27] KK Phoon, HW Huang, and ST Quek. Simulation of strongly non-Gaussian processes using Karhunen–Loeve expansion. *Probabilistic engineering mechanics*, 20(2):188–198, 2005.
- [28] Shigehiro Sakamoto and Roger Ghanem. Polynomial chaos decomposition for the simulation of non-Gaussian nonstationary stochastic processes. *Journal of engineering mechanics*, 128(2):190–201, 2002.
- [29] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle,

- A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems* 32, pages 8024–8035. Curran Associates, Inc., 2019.
- [30] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.
- [31] Bernard W Silverman. *Density estimation for statistics and data analysis*. Routledge, 2018.
- [32] Hannes Risken. Fokker-planck equation. In *The Fokker-Planck Equation*, pages 63–95. Springer, 1996.
- [33] Yuanran Zhu and Daniele Venturi. Hypoellipticity and the Mori–Zwanzig formulation of stochastic differential equations. *Journal of Mathematical Physics*, 62(10):103505, 2021.
- [34] Peter Hänggi, Peter Talkner, and Michal Borkovec. Reaction-rate theory: fifty years after kramers. *Reviews of modern physics*, 62(2):251, 1990.
- [35] Phillip L Geissler, Christoph Dellago, and David Chandler. Kinetic pathways of ion pair dissociation in water. *The Journal of Physical Chemistry B*, 103(18):3706–3710, 1999.
- [36] Peter G Bolhuis, David Chandler, Christoph Dellago, and Phillip L Geissler. Transition path sampling: Throwing ropes over rough mountain passes, in the dark. *Annual review of physical chemistry*, 53(1):291–318, 2002.
- [37] Christoph Dellago, Peter G Bolhuis, Félix S Csajka, and David Chandler. Transition path sampling and the calculation of rate constants. *The Journal of chemical physics*, 108(5):1964–1977, 1998.
- [38] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*, pages 1–6. Ieee, 2017.
- [39] Yuxin Tian, Xueqing Deng, Yi Zhu, and Shawn Newsam. Cross-time and orientation-invariant overhead image geolocalization using deep local features. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2512–2520, 2020.
- [40] Tyler W Hughes, Ian AD Williamson, Momchil Minkov, and Shanhui Fan. Wave physics as an analog recurrent neural network. *Science advances*, 5(12):eaay6946, 2019.
- [41] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.
- [42] Yuanran Zhu, Huan Lei, and Changho Kim. Generalized second fluctuation-dissipation theorem in the nonequilibrium steady state: Theory and applications. *arXiv preprint arXiv:2104.05222*, 2021.

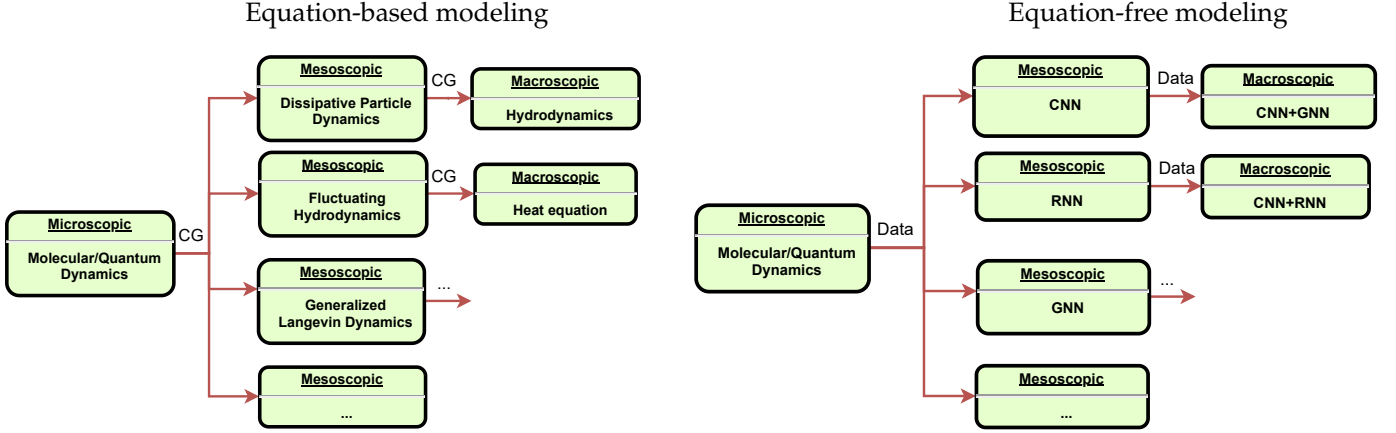


Figure 8: Equation-based and equation-free modeling diagrams in different spatial-temporal scales.

A A paradigm shift in mathematical modeling

From a modeling perspective, the machine learning architectures provide a data-driven, equation-free diagram to discover the hidden dynamics of a physical system at different spatial-temporal scales. Specifically, the classical, equation-based modeling diagram (see Figure 8) normally start with a microscopic model for the molecular/quantum dynamics, then a certain coarse-grained (CG) procedure such as the Bogoliubov–Born–Green–Kirkwood–Yvon (BBGKY) hierarchy, projection operator method, or mean-field approximation, is applied to reduce the dimensionality of the system and obtain mesoscopic models. Similar CG procedures can be further applied to get macroscopic models. At different scales, the established physical models have a general form:

$$\partial_t u = F(u, k, t) \quad (\text{A.1})$$

where u is the unknown (vector) variable or a (vector) field $u(x, t)$, $F(u, k, t)$ is the combination of functions, stochastic processes, and operators with modeling parameters k . On the contrary, the machine-learning framework provides equation-free models such as the Convolutional Neural Network (CNN) [38, 39], Recurrent Neural Network (RNN) [40], or Graph Neural Network (GNN) [41] for a physical process. The training of the neural network requires data such as the sample trajectories of an observable in the phase space. The general form of the modeling ansatz for the unknown function $u(x, t)$ relies on the multi-fold function compositions:

$$u(x, t) = f_1(f_2(f_3(\cdots), k_3), k_2), k_1) \quad (\text{A.2})$$

where f_i is the activation function of i -th neuron and k_i is the corresponding modeling parameter. Some ML models such as PINNs still use equations to define the loss function and train parameters while the modeling anstaz is of the form (A.2). Other examples such as the NeuralODE/SDE/SPDE use (A.2) to model the derivatives function, i.e. the right hand side of equation (A.1). In comparison, SINN is completely equation-free during the training and the

modeling process.

B Universal approximation theorem for RNN with Gaussian stochastic inputs

Following [25], we introduce some useful definitions and established universal approximation results for the deterministic RNN.

Definition 1. For any (Borel-)measurable function $f(\cdot) : \mathbb{R}^I \rightarrow \mathbb{R}^J$ and $I, N \in \mathbb{N}$. $\Sigma^{I,N}(f)$ is called a function class for NN (three-layer feedforward neural network) if any $g \in \Sigma^{I,N}(f)$ is of the form:

$$g(x) = Vf(Wx - \theta), \quad \text{where} \quad x \in \mathbb{R}^I, \quad V \in \mathbb{R}^{N \times J}, \quad W \in \mathbb{R}^{J \times I}, \quad \theta \in \mathbb{R}^J, \quad J \in \mathbb{N}.$$

The above three-layer feedforward neural network has I input neurons, J hidden neurons and N output neurons. To be noticed that the function $f : \mathbb{R}^I \rightarrow \mathbb{R}^J$ is defined to be component-wise with

$$f(Wx - \theta) := \begin{bmatrix} f(W_1x - \theta_1) \\ f(W_2x - \theta_2) \\ \vdots \\ f(W_Jx - \theta_J) \end{bmatrix} \quad (\text{B.1})$$

Definition 2. A subset S of a metric space (X, ρ) is called ρ -dense in a subset U , if there exists, for any $\epsilon > 0$ and any $u \in U$, a $s \in S$ such that $\rho(s, u) < \epsilon$.

Definition 3. Let $C^{I,N} : \mathbb{R}^I \rightarrow \mathbb{R}^N$ be the set of all continuous functions. A subset $S \subset C^{I,N}$ is uniformly dense on a compact domain in $C^{I,N}$ if for any compact subset $K \subset \mathbb{R}^I$, S is ρ_K -dense in $C^{I,N}$, where $\rho_K(f, g) := \sup_{x \in K} \|f(x) - g(x)\|_\infty$.

Definition 4. A function σ is called a sigmoid function if σ is monotonically increasing and bounded.

Common choice of sigmoid function of neural networks are ReLU and $\tanh(x)$. The following result is the well-known universal approximation theorem for feedforward neural networks.

Theorem 3. (UAT for feedforward neural networks) For any sigmoid activation function σ , and any dimension I, N , $\Sigma^{I,N}(\sigma)$ is uniformly dense on a compact domain in $C^{I,N}$.

The above theorem simply implies that for any sigmoid function σ , as long as $J \in \mathbb{N}$ is large enough, i.e. the number of hidden state (neuron) is large enough, then a three-layer feedforward neural networks can approximate any continuous function in any compact domain arbitrarily accurate. This theorem was used [25] to prove the universal approximation theorem for RNN of type (2) when the input x_t is deterministic. To this end, we introduce the following definition of the RNN class:

Definition 5. Let $\sigma(\cdot) : \mathbb{R}^J \rightarrow \mathbb{R}^J$ be an arbitrary sigmoid function and $I, N, T \in \mathbb{N}$. The class $RNN^{I,N}(\sigma)$ refers to discrete RNN system of the form (1), i.e.

$$\begin{aligned} s_{t+1} &= \sigma(As_t + Bx_t - \theta) \\ y_t &= Cs_t, \end{aligned}$$

where $x_t \in \mathbb{R}^I$, $s_t \in \mathbb{R}^J$, and $y_t \in \mathbb{R}^N$ for all $t = 1, \dots, T$. To be noticed that here $\sigma(As_t + Bx_t - \theta)$ is calculated component-wise as of (B.1). We also define $o(RNN^{I,N}(\sigma))$ to be the set of all possible output y_t for the RNN of the class $RNN^{I,N}(\sigma)$.

It is proved in [25] that $RNN^{I,N}(\sigma)$ is “dense” in the “space of discrete open dynamical systems”, in the sense that for any sigmoid f , there exists a $\tilde{y}_t \in o(RNN^{I,N}(\sigma))$ such that $\|\tilde{y}_t - y_t\|_\infty < \delta, \forall \delta > 0$, where y_t is the output of a M -dimensional open system:

$$\begin{aligned} s_{t+1} &= g(s_t, x_t) \\ y_t &= h(s_t) \end{aligned} \tag{B.2}$$

where $g(\cdot) : \mathbb{R}^M \rightarrow \mathbb{R}^M, h(\cdot) : \mathbb{R}^M \rightarrow \mathbb{R}^N$. For RNN with stochastic input, the proof is similar. But the corresponding universal approximation theorem, i.e. Theorem 2, only holds in the sense of probability essentially because whether one can find a RNN model such that $\|\hat{y}_t - y_t\|_\infty < \delta, \forall \delta > 0$ becomes a random event. Using the above definitions and Theorem 3, we can get the proof.

Proof of Theorem2. We first show that the dynamics of a M -dimensional open dynamical system with $s_{t+1} = g(s_t, x_t)$ can be represented by an RNN with update function the form $\bar{s}_{t+1} = \sigma(A\bar{s}_t + Bx_t - \theta)$ for all $t = 1, \dots, T$ asymptotically almost surely. For any realization of x_t , Theorem 3 implies that for any compact set $K \subset \mathbb{R}^M \times \mathbb{R}^I$ which contains (s_t, x_t) , one can find suitable $\bar{g}(s_t, x_t) \in \Sigma^{I+M,M}(\sigma)$ with weight matrices $V \in \mathbb{R}^{M \times J}, W \in \mathbb{R}^{J \times M}, B \in \mathbb{R}^{J \times I}$, and a bias $\theta \in \mathbb{R}^J$ such that for all $t = 1, \dots, T$, we have

$$\sup_{x_t, s_t \in K} \|g(s_t, x_t) - \bar{g}(s_t, x_t)\|_\infty \leq \delta, \quad \text{where} \quad \bar{g}(s_t, x_t) = V\sigma(Ws_t + Bx_t - \theta). \tag{B.3}$$

Here $\delta > 0$ is an arbitrary constant and σ is an arbitrary component-wise applied sigmoid activation function. Now we denote approximated dynamics generated by the feedback neural network by

$$\bar{s}_{t+1} = \bar{g}(\bar{s}_t, x_t) = V\sigma(W\bar{s}_t + Bx_t - \theta).$$

Further assuming that $\bar{s}_t \in K$, then for any $\delta > 0$, we can find suitable W, B, V, θ such that

$$\begin{aligned} \|s_t - \bar{s}_t\|_\infty &= \|g(s_{t-1}, x_{t-1}) - g(\bar{s}_{t-1}, x_{t-1}) + g(\bar{s}_{t-1}, x_{t-1}) - \bar{s}_t\|_\infty \\ &\leq \|g(s_{t-1}, x_{t-1}) - g(\bar{s}_{t-1}, x_{t-1})\|_\infty + \|g(\bar{s}_{t-1}, x_{t-1}) - \bar{s}_t\|_\infty \\ &\leq \|g(s_{t-1}, x_{t-1}) - g(\bar{s}_{t-1}, x_{t-1})\|_\infty + \delta. \end{aligned}$$

Since g is continuous differentiable, in the compact set K , it is also Lipschitz continuous. This implies for any $\epsilon > 0$, there is a $\delta > 0$, therefore suitable W, B, V, θ , such that

$$\|s_t - \bar{s}_t\|_\infty \leq C\|s_{t-1} - \bar{s}_{t-1}\|_\infty + \delta \leq \delta(1 + C + \dots C^{T-1}) = \delta \frac{1 - C^T}{1 - C} \leq \epsilon, \quad (\text{B.4})$$

where we have used $s_0 = \bar{s}_0$. Estimate (B.4) indicates for deterministic inputs x_t , the open dynamical system update function $g(s_t, x_t)$ can be universally approximated by the feedward neural network update function $\hat{g}(\hat{s}_t, x_t)$ since the output of each step, i.e. s_t , can be approximated arbitrarily accurate. For RNN with Gaussian random input x_t , since x_t is not compactly supported, whether $x_t, s_t \in K$ becomes a random event. For any *fixed* $K \subset \mathbb{R}^M \times \mathbb{R}^I$ and sigmoid function σ , one can associate it with a NN function class $\Sigma^{I+M,M}(\sigma)$. The probability of whether one can find suitable approximation to function $g(s_t, x_t)$ within $\Sigma^{I+M,M}(\sigma)$ for any initial $s_0 \in \mathbb{R}^M$ and $x_t \in \mathbb{R}^I$ can be written as

$$Pr \left[\inf_{\hat{g} \in \Sigma^{I+M,M}(\sigma)} \sup_{x_t \in \mathbb{R}^I, s_0 \in \mathbb{R}^M} \|g(s_t, x_t) - \hat{g}(\bar{s}_t, x_t)\|_\infty \leq \epsilon, \forall \epsilon > 0 \right], \quad t = 1, \dots, T. \quad (\text{B.5})$$

To be noticed that this probability depends on K , in particular, the size¹ of it, which is denoted as $|K|$. Taking the limit $|K| \rightarrow +\infty$, we obtain

$$\begin{aligned} &\lim_{|K| \rightarrow +\infty} Pr \left[\inf_{\hat{g} \in \Sigma^{I+M,M}(\sigma)} \sup_{x_t \in \mathbb{R}^I, s_0 \in \mathbb{R}^M} \|g(s_t, x_t) - \hat{g}(\bar{s}_t, x_t)\|_\infty \leq \epsilon, \forall \epsilon > 0 \right], \quad t = 1, \dots, T \\ &= \lim_{|K| \rightarrow +\infty} Pr \left[\inf_{\hat{g} \in \Sigma^{I+M,M}(\sigma)} \sup_{x_t, s_t, \bar{s}_t \in K} \|g(s_t, x_t) - \hat{g}(\bar{s}_t, x_t)\|_\infty \leq \epsilon, \forall \epsilon > 0 \right] * Pr[x_t, s_t, \bar{s}_t \in K] \\ &= \lim_{|K| \rightarrow +\infty} Pr[x_t, s_t, \bar{s}_t \in K] \end{aligned} \quad (\text{B.6})$$

¹ $|K|$ can be defined as, e.g. $|K| := \sup_{x \in K} \|x\|_2$, where $\|x\|_2$ is the l_2 -norm of the vector.

Here we used (B.3) and (B.4) to show that under the condition $x_t, s_t \in K$, $\inf_{\bar{g} \in \Sigma^{l,N}} \sup_{x_t, s_t \in K} \|g(s_t, x_t) - \bar{g}(\bar{s}_t, x_t)\|_\infty < \delta$, $\forall \delta > 0$ is a certain event hence happens with probability 1. Now the problem of calculating the probability (B.5) in the limit $|K| \rightarrow +\infty$ boils down to the calculation of $\lim_{|K| \rightarrow +\infty} Pr[x_t, s_t, \bar{s}_t \in K]$. To this end, we note that if we can choose a compact set $K_0 \subset \mathbb{R}^M \times \mathbb{R}^J$ such that $x_t, s_0 \in K_0$ for all $t = 1, \dots, T$, then $\|s_1\| = \|g(s_0, x_0)\| \leq C_0$ for some $C_0 > 0$ because g is a continuous function therefore bounded in K_0 . Choose another compact set $K_0 \cup B(0, C_0) \subset K_1 \subset \mathbb{R}^J \times \mathbb{R}^J$, where $B(0, C_0)$ is a ball centered at 0 with radius C_0 . Then we must have $\|s_2\| = \|g(s_1, x_1)\| \leq C_1$ since $x_1, s_1 \in K_1$. Continuing this procedure for T times, we can find a compact set $K_T \subset \mathbb{R}^M \times \mathbb{R}^J$ such that $s_t, x_t \in K_T$. The same logic applies to \bar{s}_t , hence we can find a compact $K \subset \mathbb{R}^M \times \mathbb{R}^J$ such that $s_t, x_t, \bar{s}_t \in K_T$. On the other hand, Chernoff inequality for a standard normal random variable X implies

$$Pr[X \geq b] \leq \exp\{-b^2/2\} \quad \Rightarrow \quad Pr[X < b] \geq 1 - \exp\{-b^2/2\}. \quad (\text{B.7})$$

Then for i.i.d random variable X_1, \dots, X_T , we have

$$Pr[X_1 < b, \dots, X_T < b] = \prod_{i=1}^T Pr[X_i < b] \geq [1 - \exp\{-b^2/2\}]^T.$$

Therefore for fixed T , we have

$$\lim_{b \rightarrow +\infty} Pr[X_1 < b, \dots, X_T < b] \geq \lim_{b \rightarrow +\infty} [1 - \exp\{-b^2/2\}]^T = 1 \quad (\text{B.8})$$

This means asymptotically almost surely, one can find a compact subset B such that $x_t \in B$ for all $t = 1, \dots, T$. As a direct result of this, we have

$$\lim_{|K| \rightarrow +\infty} Pr[x_t, s_t, \bar{s}_t \in K] = 1 \quad (\text{B.9})$$

Combining (B.9) and (B.6), we get that for RNN with i.i.d Gaussian input x_t , asymptotically almost surely, one can find suitable W, B, V, θ such that $\|s_t - \bar{s}_t\|_\infty < \epsilon$ for any sigmoid f and any $\epsilon > 0$. Further let

$$s'_{t+1} = \sigma(W\bar{s}_t + Bx_t - \theta)$$

which yields $\bar{s}_t = Vs'_t$. Define $A := WV(\in \mathbb{R}^{J \times J})$, we can get that

$$s'_{t+1} = \sigma(As'_t + Bx_t - \theta) \quad (\text{B.10})$$

Since for every t , we have $\bar{s}_t = Vs'_t$. The dynamics of the RNN update function (B.10) *encodes* (not equals) the dynamics of the open dynamical systems. Hence we claim that the dynamics of a M -dimensional open dynamical system with $s_{t+1} = g(s_t, x_t)$ can be represented by an RNN with update function the form $\bar{s}_{t+1} = \sigma(A\bar{s}_t + Bx_t - \theta)$ for all $t = 1, \dots, T$ asymptotically almost surely. To be noticed that the transformation $\bar{s}_t = Vs'_t$ often involve an enlargement of the hidden state dimensionality since $A \in \mathbb{R}^{J \times J}$, where J is set to be large enough to guarantee the validity of the universal approximation.

The second part of the proof is to show that the output of the dynamical system, i.e. $y_t = h(s_t)$ can be approximated by the output of a RNN $\tilde{y}_t = \tilde{C}\tilde{s}_t$ asymptotically almost surely where \tilde{s}_t is an *extended* vector satisfying the RNN update rule: $\tilde{s}_{t+1} = \sigma(\tilde{A}\tilde{s}_t + \tilde{B}x_t - \tilde{\theta})$. For RNN with deterministic input, the proof is done in [25], hence we simply state the result obtained therein.

Claim: Suppose $x_t, s_t, \bar{s}_t \in K \subset \mathbb{R}^M \times \mathbb{R}^I$, then there exists enlarged matrix $\tilde{A}, \tilde{B}, \tilde{C}$ and $\tilde{\theta}$ for a RNN model:

$$\begin{aligned}\tilde{s}_{t+1} &= \sigma(\tilde{A}\tilde{s}_t + \tilde{B}x_t - \tilde{\theta}) \\ \tilde{y}_t &= \tilde{C}\tilde{s}_t,\end{aligned}\tag{B.11}$$

such that the output vector $\|\tilde{y}_t - y_t\| \leq \epsilon$ for all $\epsilon > 0$. In (B.11), we have

$$\begin{aligned}\tilde{J} &= J + \bar{J}, \quad r_t = \sigma(Es'_t + Fx_t - \bar{\theta}) \in \mathbb{R}^{\bar{J}}, \quad E \in \mathbb{R}^{\bar{J} \times J}, F \in \mathbb{R}^{\bar{J} \times I}, \bar{\theta} \in \mathbb{R}^{\bar{J}}, \quad \tilde{s}_t = \begin{bmatrix} s'_t \\ r_t \end{bmatrix} \in \mathbb{R}^{\bar{J}} \\ \tilde{A} &= \begin{bmatrix} A & 0 \\ E & 0 \end{bmatrix} \in \mathbb{R}^{\bar{J} \times \bar{J}}, \quad \tilde{B} = \begin{bmatrix} B \\ F \end{bmatrix} \in \mathbb{R}^{\bar{J} \times I}, \quad \tilde{C} = [0 \quad D] \in \mathbb{R}^{N \times \bar{J}}, \quad \text{and} \quad \tilde{\theta} = \begin{bmatrix} \theta \\ \bar{\theta} \end{bmatrix} \in \mathbb{R}^{\bar{J}}\end{aligned}$$

Proof.

$$\begin{aligned}\|y_t - \tilde{y}_t\| &= \|y_t - D\sigma(Es'_{t-1} + Fx_{t-1} - \bar{\theta})\| \\ &\leq \|y_t - Q\sigma(GV\sigma(As'_{t-1} + Bx_{t-1} - \theta) - \hat{\theta})\| + \|Q\sigma(GV\sigma(As'_{t-1} + Bx_{t-1} - \theta) - \hat{\theta}) - D\sigma(Es'_{t-1} + Fx_{t-1} - \bar{\theta})\|.\end{aligned}$$

Here $Q\sigma(GV\sigma(As'_{t-1} + Bx_{t-1} - \theta) - \hat{\theta})$ is a bounded function defined in compact domain K , hence the universal approximation theorem implies for any $\epsilon_1 > 0$, we can find suitable $D, E, F, \bar{\theta}$ such that $\|Q\sigma(GV\sigma(As'_{t-1} + Bx_{t-1} - \theta) - \hat{\theta}) - D\sigma(Es'_{t-1} + Fx_{t-1} - \bar{\theta})\| \leq \epsilon_1$.

$\hat{\theta}) - D\sigma(Es'_{t-1} + Fx_{t-1} - \bar{\theta})\| \leq \epsilon_1$. Then we obtain

$$\begin{aligned}
\|y_t - \tilde{y}_t\| &\leq \|y_t - Q\sigma(GV\sigma(As'_{t-1} + Bx_{t-1} - \bar{\theta}) - \hat{\theta})\| + \epsilon_1 \\
&= \|y_t - Q\sigma(GVs'_t - \hat{\theta})\| + \epsilon_1 \\
&= \|y_t - Q\sigma(G\bar{s}_t - \hat{\theta})\| + \epsilon_1 \\
&\leq \|y_t - h(\bar{s}_t)\| + \|h(\bar{s}_t) - Q\sigma(G\bar{s}_t - \hat{\theta})\| + \epsilon_1
\end{aligned}$$

Again applying the universal approximation theorem, we have $\|h(\bar{s}_t) - Q\sigma(G\bar{s}_t - \hat{\theta})\| \leq \epsilon_2, \forall \epsilon_2 > 0$. On the other hand, $h(x)$ is continuously differentiable hence Lipschitz in a compact set K , we have $\|y_t - h(\bar{s}_t)\| = \|h(s_t) - h(\bar{s}_t)\| \leq C\|s_t - \bar{s}_t\| \leq C\epsilon_0$. This leads to

$$\|y_t - \tilde{y}_t\| \leq C\epsilon + \epsilon_1 + \epsilon_2 \leq \delta, \forall \delta > 0.$$

□

This above claim indicates that under the condition $x_t, s_t, \bar{s}_t \in K \subset \mathbb{R}^M \times \mathbb{R}^I$, the probability of finding suitable $\tilde{A}, \tilde{B}, \tilde{\theta}, \tilde{C}$ such that $\|\tilde{y}_t - y_t\| \leq \delta, \forall \delta > 0$ is 1. Hence using again relations (B.5), (B.6), and (B.9), we have

$$\begin{aligned}
&\lim_{|K| \rightarrow +\infty} Pr \left[\inf_{\tilde{y}_t \in \mathcal{O}(\text{RNN}^{I,N}(\sigma))} \sup_{x_t \in \mathbb{R}^I, s_0 \in \mathbb{R}^M} \|y_t - \tilde{y}_t\|_\infty \leq \delta, \forall \delta > 0 \right], \quad t = 1, \dots, T \\
&= \lim_{|K| \rightarrow +\infty} Pr \left[\inf_{\tilde{y}_t \in \mathcal{O}(\text{RNN}^{I,N}(\sigma))} \sup_{x_t, s_t, \bar{s}_t \in K} \|y_t - \tilde{y}_t\|_\infty \leq \delta, \forall \delta > 0 \mid x_t, s_t, \bar{s}_t \in K \right] * Pr[x_t, s_t, \bar{s}_t \in K] \\
&= \lim_{|K| \rightarrow +\infty} Pr[x_t, s_t, \bar{s}_t \in K] = 1.
\end{aligned} \tag{B.12}$$

This implies any open dynamical system (2) with Gaussian inputs x_t can be approximated arbitrarily accurate asymptotically almost surely by a *deterministic* RNN model of the form (1) with the *same stochastic* input x_t and an arbitrary sigmoid function σ .

C Long-time predictability of SINN

In this section, we discuss the long-time predictability of SINN. As a neural network based on the LSTM architecture, SINN makes prediction of the long-time dynamics of the reduced-order observable $x(t)$ by quantifying the memory effect of the non-Markovian system, which is similar to the reduced-order modeling using the Mori-Zwanzig equation/Generalized Langevin equation (MZ/GLE). Since the MZ/GLE is proven to have predictability of the long-time stochastic dynamics [13, 42], it is reasonable to expect SINN has similar function. This is indeed the case as we

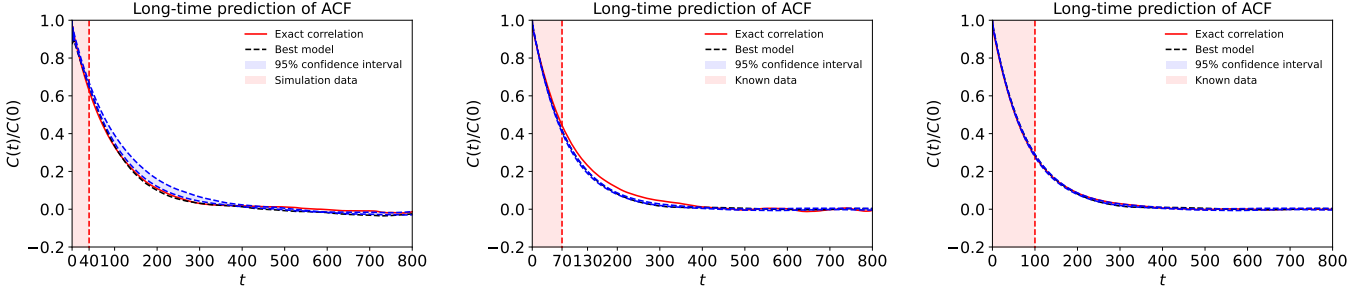


Figure 9: Long-time prediction of the ACF of $x(t)$ using SINN. In each figure, the ACF of the top 5 *qualified* SINN model are used to calculate the 95% confidence interval of the predicted dynamics. For the *qualified* SINN models, the one with the smallest validation error ϵ_V is selected to be the best model.

already shown in Figure 5 that using the short-time simulation data for $t \in [0, 80]$, the SINN model can well predict the long-time dynamics of $x(t)$. However, SINN has a major difference from the MZ/GLE model. Due to the usage of stochastic optimizer, the learned SINN model hence its “memory” effect would be *different* for each training. This arises a reasonable doubt that whether the showed long-time predictability of SINN is merely a coincidence. Due to the well-known difficulties on the theoretical convergence analysis for deep neural networks, here we only provide numerical studies. The predictability of the obtained SINN model is evaluated by the long-time tail of the ACF for $x(t)$.

We performed an ensemble of independent, repeated training of SINN based on the same data set. Specifically, we obtained three ensembles of SINN models. The first ensemble was trained using 400 simulated trajectories of $x(t)$ for $t \in [0, 40]$ with coarse-grained step size $\Delta t = 0.2$, while the time domain for the second and third ensemble trajectories are $[0, 70]$ and $[0, 100]$ respectively. In each ensemble, we repeat the training process and get 20 *candidate* SINN models. Each candidate model is obtained by independent training of SINN until the training and validation error satisfy $\epsilon_T, \epsilon_V = l_1 + l_2 \leq 10^{-3}$. From these candidate SINN models, we perform time extrapolation to generate long-time dynamics of $x(t)$ and re-evaluate the validation error ϵ_V to select the top 5 *qualified* SINN models with the smallest ϵ_V . The evaluation time domain for ϵ_V has to be $[0, 40]$, $[0, 70]$, or $[0, 100]$ respectively since that is the only time frame we know the exact evolution of $x(t)$. This procedure ensures the qualified SINN models are stationary time sequences which imitate the equilibrium dynamics of $x(t)$. The simulation results and the analysis are presented in Figure 9. We can see that with simulation data as short as $[0, 40]$, the *qualified* SINN models, i.e. the ones with the small enough validation errors, yield overall good predictions of the correct long-time dynamics of $x(t)$. This validates the long-time predictability of SINN. As we gradually increase the length of the target trajectories from $t \in [0, 40]$ to $t \in [0, 100]$, the 95% confidence interval of the predicted dynamics gets smaller which indicates the output of the ensemble of SINNs converges to the correct dynamics of $x(t)$. Hence a numerical validation of the convergence of SINN is established in terms of the statistics of the input-output.