

DATA-DRIVEN EFFECTIVE MODELING OF STOCHASTIC CHEMICAL REACTION NETWORKS

YUAN CHEN*, WEIZE MAO*, AND DONGBIN XIU*

Abstract. The Stochastic Simulation Algorithm (SSA), widely considered an exact algorithm for stochastic chemical reaction networks, suffers from high computational cost. In this work, we propose a data-driven effective model that operates on a user-defined coarse time step independent of the underlying microscopic reaction-event scale. This is accomplished by directly approximating the finite-time transition kernel of the continuous-time Markov chain induced by SSA, using a generative machine learning model trained on short bursts of SSA simulation data. The trained model constructs a stochastic propagator that recursively generates statistically consistent trajectories at the constant coarse time step, with significantly reduced computational cost. In this paper, we employ conditional normalizing flow as the stochastic propagator. A comprehensive set of numerical examples is presented to demonstrate the accuracy and efficiency of the proposed method.

Key words. Effective Modeling, Generative Model, Normalizing Flow, Stochastic Reaction Network, Stochastic Simulation Algorithm

MSC codes. 60J27, 60J28, 65C05, 65C40, 92C42

1. Introduction. Stochastic chemical kinetics of well-mixed reacting systems are commonly modeled by the Chemical Master Equation (CME), which describes the evolution of the probability distribution of the underlying reaction network. The corresponding stochastic process is a continuous-time Markov jump process whose exact sample trajectories can be generated by the Stochastic Simulation Algorithm (SSA), also known as the Gillespie algorithm. Since its introduction in [22], SSA has become one of the most widely used computational tools for stochastic chemical reaction systems.

Despite its exactness, SSA suffers from high computational cost. The algorithm advances the system by simulating one reaction event at a time, and the corresponding firing times can become exceedingly small for systems with large populations or stiff reaction rates. Consequently, long-time simulations often require an extremely large number of reaction events. Numerous acceleration and approximation techniques have been developed to reduce this cost, including the next-reaction method [21], modified direct method [10], sorting direct method [35], moment closure approximations [25], and tau-leaping methods. Among these approaches, tau-leaping advances the system over a finite time interval by approximating the collective statistics of many reaction events [23]. This idea has led to extensive developments, including stability and error analysis [1, 29, 43], nonnegativity-preserving variants [7, 11], and adaptive or implicit schemes for stiff systems [9]. Related coarse-graining and reduction strategies for stochastic reaction systems have also been studied from both computational and mathematical perspectives [27, 28]. We refer to [2, 26, 44] for broader reviews and background. These methods are effective in many regimes, but they typically rely on assumptions regarding slowly varying propensity functions and can suffer from loss of accuracy or stability when these assumptions fail.

In this work, we pursue a different direction. Instead of approximating the individual reaction events or modifying the internal mechanics of SSA, we directly approximate the finite-time transition law of the Markov jump process induced by

*E-mail addresses: chen.11050@osu.edu, weizemao339@gmail.com, xiu.16@osu.edu. Department of Mathematics, The Ohio State University, Columbus, OH 43210, USA. Funding: This work was partially supported by AFOSR FA9550-24-1-0237.

SSA. More precisely, let $X(t)$ denote the stochastic process generated by the reaction network. For any finite time lag $\Delta > 0$, the process admits a transition kernel

$$(1.1) \quad P_{\Delta}(x, x') = \mathbb{P}(X(t + \Delta) = x' \mid X(t) = x),$$

which defines the finite-time Markov semigroup associated with the CME. The objective of this work is to construct a data-driven approximation of P_{Δ} directly from short bursts of SSA simulation data.

This viewpoint leads naturally to a coarse stochastic propagator operating on a user-defined time step Δ , which can be substantially larger than the average SSA firing time. Unlike tau-leaping methods, which accelerate SSA by approximating the numbers of reaction firings over a finite interval under assumptions on the variation of propensity functions, the proposed approach does not approximate the reaction-event dynamics. Instead, it directly targets the finite-time transition kernel of the SSA-induced continuous-time Markov chain and learns this kernel from SSA data. Consequently, the learned model should be viewed as a data-driven numerical approximation of the exact Markov semigroup associated with the underlying continuous-time Markov chain, rather than as an approximation of the reaction-event dynamics.

The proposed method follows the framework of stochastic Flow Map Learning (sFML) [16], which extends deterministic flow map learning [40] to stochastic dynamical systems. In deterministic systems, flow map learning seeks to approximate the map between consecutive states over a finite time interval. In stochastic systems, however, the finite-time evolution is no longer deterministic and is instead characterized by a conditional probability distribution. Consequently, the stochastic flow map is interpreted as a stochastic realization operator whose generated samples follow the finite-time transition kernel.

To realize the stochastic flow map, we employ a conditional normalizing flow model. Normalizing flows (see, e.g., [38]) provide expressive generative models capable of representing highly non-Gaussian conditional distributions while maintaining tractable likelihood evaluation and efficient sampling. This property is particularly important for stochastic chemical reaction systems, whose finite-time transition distributions are frequently strongly non-Gaussian and highly state-dependent.

The resulting framework constructs a stochastic propagator directly from short bursts of SSA trajectory data. Once trained, the model can efficiently produce long-time trajectories at a prescribed coarse time step while preserving important statistical properties of the underlying stochastic dynamics. The main contributions of this work are summarized as follows:

- We reinterpret stochastic flow map learning for chemical reaction systems as a data-driven approximation of the finite-time transition kernel associated with the SSA Markov process.
- We construct a coarse stochastic propagator operating on a user-defined time step independent of the microscopic reaction-event scale.
- We employ conditional normalizing flows to approximate non-Gaussian transition distributions arising from stochastic chemical kinetics.
- We demonstrate through a comprehensive set of numerical examples that the learned propagator produces statistically consistent trajectories while significantly reducing the number of simulation steps compared with SSA.

2. Preliminary. We first present the problem setup and our main objective.

2.1. Problem Setup. Consider a stochastic chemical reaction network with N species S_i , $i = 1, \dots, N$, and M reaction channels

$$(2.1) \quad R_\mu : \quad \sum_{i=1}^N r_{i\mu} S_i \xrightarrow{c_\mu} \sum_{i=1}^N p_{i\mu} S_i, \quad \mu = 1, \dots, M,$$

where c_μ denotes the reaction rate constant, and $r_{i\mu}, p_{i\mu}$ are nonnegative integers representing the stoichiometric coefficients.

Let N -dimensional vector

$$\mathbf{X}(t) = [X_1(t), \dots, X_N(t)]^T$$

denote the vector of molecule numbers at time t , in the state space $\mathbb{Z}_{\geq 0}^N$, the set of N -dimensional vectors of non-negative integers. Under the well-mixed assumption, the evolution of $\mathbf{X}(t)$ is modeled as a continuous-time Markov chain (CTMC). Each reaction channel R_μ is characterized by:

- the stoichiometric vector $\mathbf{s}_\mu = [p_{1\mu} - r_{1\mu}, \dots, p_{N\mu} - r_{N\mu}]^T$, which specifies the state change caused by the reaction;
- the propensity function $a_\mu(\mathbf{x}) = c_\mu h_\mu(\mathbf{x})$, where $h_\mu(\mathbf{x})$ represents the number of available reactant combinations at state $\mathbf{X}(t) = \mathbf{x}$.

The stochastic process admits an infinitesimal generator

$$(2.2) \quad (\mathcal{L}f)(\mathbf{x}) = \sum_{\mu=1}^M a_\mu(\mathbf{x}) (f(\mathbf{x} + \mathbf{s}_\mu) - f(\mathbf{x})),$$

which defines the corresponding chemical master equation.

The SSA algorithm generates exact sample trajectories of this CTMC. Starting from state $\mathbf{X}(t) = \mathbf{x}$, SSA samples:

- (1) a firing time

$$\tau \sim \text{Exp} \left(\sum_{\mu=1}^M a_\mu(\mathbf{x}) \right),$$

- (2) a reaction index λ with probabilities

$$\mathbb{P}(\lambda = \mu) = \frac{a_\mu(\mathbf{x})}{\sum_{\nu=1}^M a_\nu(\mathbf{x})},$$

and updates the state according to

$$\mathbf{X}(t + \tau) = \mathbf{X}(t) + \mathbf{s}_\lambda.$$

Although SSA evolves through random microscopic firing events, the resulting process is globally characterized by a finite-time transition kernel. For any $\Delta > 0$, define transition kernel $P_\Delta : \mathbb{Z}_{\geq 0}^N \times \mathbb{Z}_{\geq 0}^N \rightarrow [0, 1]$

$$(2.3) \quad P_\Delta(\mathbf{x}, \mathbf{x}') = \mathbb{P}(\mathbf{X}(t + \Delta) = \mathbf{x}' | \mathbf{X}(t) = \mathbf{x}).$$

Equivalently, for any observable ϕ ,

$$(2.4) \quad (\mathcal{P}_\Delta \phi)(\mathbf{x}) = \mathbb{E}[\phi(\mathbf{X}(t + \Delta)) | \mathbf{X}(t) = \mathbf{x}]$$

defines the Markov semigroup associated with the process.

For each fixed $\mathbf{x}$, $P_\Delta(\mathbf{x}, \cdot)$ defines a (conditional) probability measure on $\mathbb{Z}_{\geq 0}^N$ where $P_\Delta(\mathbf{x}, A) = \mathbb{P}(\mathbf{X}(t + \Delta) \in A | \mathbf{X}(t) = \mathbf{x})$ for any event set A . Modeling this conditional distribution is the central goal of this paper.

2.2. Objective. The objective of this work is to construct a data-driven approximation of the finite-time transition kernel P_Δ (2.3) for a prescribed coarse time step Δ . More importantly, Δ is chosen according to the time scale of the macroscopic behavior of the system and is independent of the microscopic SSA firing times τ . Therefore, Δ may be substantially larger than the average reaction-event scale, i.e., $\Delta \gg \mathbb{E}[\tau]$.

2.3. Related Work. This work belongs to the broader area of learning dynamical systems from data, which has attracted growing attention due to the rapid development of machine learning. Most existing methods focus on deterministic systems, see, for example, [5, 12, 18, 30, 32, 37, 40, 41, 42].

For stochastic systems, most existing approaches are ‘model-based’, i.e., observational data is assumed to be sampled from a known model with unknown components, such as Itô stochastic differential equations which approximately admit Gaussian transition distributions with unknown drift and diffusion functions. These unknown functions are approximated by analyzing and matching statistical properties of data, such as moments, likelihoods, and densities. Such methods have been developed in conjunction with Gaussian processes [3, 19, 36, 51] and deep neural networks (DNNs) [13, 14, 20, 33, 48, 50, 52].

For processes generated by SSA, and CTMCs more broadly, such methods are less capable since the transition distributions are *non-Gaussian* and the underlying mathematical structure does not conform to a standard parametric form. Some recent explorations using machine learning methods can be found in [6, 46, 47]. In this work, we adopt the sFML framework [16], which incorporates generative models to approximate the stochastic map between consecutive states in distribution. Many generative models are compatible with this framework, including GANs (generative adversarial networks) [16], autoencoders [49], normalizing flows [15, 17], and diffusion models [31, 45]. Recently, it was also extended to stochastic systems subjected to external excitation and multiscale systems [15, 17].

3. Main Method: Effective Stochastic Propagator.

3.1. Finite-Time Stochastic Flow Map. Let $t_n = n\Delta$, $n = 0, 1, \dots$ be a sequence of discrete time points with constant time stepsize Δ and we denote $\mathbf{X}(t_n)$ by $\mathbf{X}_n$ for convenience. Then the transition from $\mathbf{X}_n$ to $\mathbf{X}_{n+1}$ is governed by the transition kernel

$$\mathbf{X}_{n+1} \sim P_\Delta(\mathbf{X}_n, \cdot).$$

We therefore define the stochastic flow map as a stochastic realization operator

$$(3.1) \quad \mathbf{X}_{n+1} = \mathbf{G}_\Delta(\mathbf{X}_n, \boldsymbol{\xi}_n),$$

where $\boldsymbol{\xi}_n$ is an auxiliary random variable independent of $\mathbf{X}_n$ and follows a prescribed distribution $\mu_{\boldsymbol{\xi}}$. The stochastic flow map is required to match the conditional distribution

$$(3.2) \quad \mathbf{G}_\Delta(\mathbf{x}, \boldsymbol{\xi}) \sim P_\Delta(\mathbf{x}, \cdot), \quad \text{as } \boldsymbol{\xi} \sim \mu_{\boldsymbol{\xi}}.$$

In other words, $\mathbf{G}_\Delta(\mathbf{x}, \boldsymbol{\xi})$ provides a sampling realization of the transition kernel $P_\Delta(\mathbf{x}, \cdot)$.

In the following sections, we construct a numerical approximation $\mathbf{G}_\Theta$ of the stochastic flow map, parameterized by Θ , such that the conditional distribution of $\mathbf{G}_\Theta(\mathbf{x}, \boldsymbol{\xi})$ approximates the finite-time transition kernel $P_\Delta(\mathbf{x}, \cdot)$. Consequently, the

learned model acts as a sampling realization of the finite-time transition kernel of the SSA Markov process. Once trained, repeated application of the stochastic flow map produces a discrete-time stochastic process

$$(3.3) \quad \tilde{\mathbf{X}}_{n+1} = \mathbf{G}_\Theta(\tilde{\mathbf{X}}_n, \boldsymbol{\xi}_n),$$

whose finite-time statistics approximate those of the original SSA process sampled at time interval Δ .

3.2. Short-Burst Data Generation. The training data are generated directly from SSA simulations. Let $\Delta > 0$ denote the prescribed coarse time step and let

$$\left\{ \mathbf{X}_0^{(i)} \right\}_{i=1}^K$$

be randomly sampled initial conditions from a prescribed distribution with size of dataset $K > 0$.

For each initial condition $\mathbf{X}_0^{(i)}$, we run SSA up to time Δ and obtain

$$\mathbf{X}_1^{(i)} = \mathbf{X}^{(i)}(\Delta).$$

This produces a collection of independent sample pairs

$$(3.4) \quad \left(\mathbf{X}_0^{(i)}, \mathbf{X}_1^{(i)} \right), \quad i = 1, \dots, K.$$

Each sample pair represents one realization of the finite-time transition kernel

$$\mathbf{X}_1^{(i)} \sim P_\Delta \left(\mathbf{X}_0^{(i)}, \cdot \right).$$

An important feature of this construction is that only short independent trajectory bursts are required. The training process therefore avoids the need for extremely long SSA simulations and is naturally parallelizable.

3.3. Conditional Normalizing Flow Approximation. To approximate the stochastic flow map, we employ a conditional normalizing flow model. Let

$$\mathbf{z} \sim \mathcal{N}(0, \mathbf{I}_N)$$

be an N -dimensional standard Gaussian random variable.

We seek a parameterized map

$$(3.5) \quad \mathbf{X}_1 = \mathbf{G}_\Theta(\mathbf{X}_0, \mathbf{z}),$$

such that the conditional distribution of $\mathbf{G}_\Theta(\mathbf{X}_0, \mathbf{z})$ approximates the finite-time transition kernel

$$P_\Delta(\mathbf{X}_0, \cdot).$$

In this work, the map $\mathbf{G}_\Theta$ is realized through a conditional normalizing flow. More specifically, we introduce an invertible transformation

$$\mathbf{T}_\theta : \mathbb{R}^N \rightarrow \mathbb{R}^N,$$

whose parameters depend on the conditioning variable $\mathbf{X}_0$ through a neural network with hyper-parameter Θ :

$$(3.6) \quad \theta = \mathbf{N}_\Theta(\mathbf{X}_0).$$

The stochastic flow map is then defined as

$$(3.7) \quad \mathbf{X}_1 = \mathbf{T}_{\mathbf{N}_\Theta(\mathbf{X}_0)}(\mathbf{z}).$$

The invertibility of the map $\mathbf{T}$ enables tractable likelihood evaluation through the change-of-variable formula. Let

$$\mathbf{S}_\theta = \mathbf{T}_\theta^{-1}.$$

Then the conditional density is

$$(3.8) \quad p(\mathbf{X}_1|\mathbf{X}_0; \Theta) = p_{\mathbf{z}}(\mathbf{S}_{\mathbf{N}_\Theta(\mathbf{X}_0)}(\mathbf{X}_1)) \left| \det \mathbf{D}\mathbf{T}_{\mathbf{N}_\Theta(\mathbf{X}_0)} \right|^{-1},$$

where $p_{\mathbf{z}}$ denotes the standard Gaussian density function and $\mathbf{D}\mathbf{T}(\mathbf{y}) = \partial\mathbf{T}/\partial\mathbf{y}$ denotes the Jacobian of the mapping $\mathbf{T}$.

The model parameters Θ are obtained by maximizing the conditional likelihood over the training dataset, equivalently minimizing the empirical negative log-likelihood loss

$$(3.9) \quad \mathcal{L}(\Theta) = -\frac{1}{K} \sum_{i=1}^K \log p(\mathbf{X}_1^{(i)}|\mathbf{X}_0^{(i)}; \Theta).$$

An illustration of the proposed sFML model structure can be found in Figure 1. This is in direct correspondence of (3.7).

We emphasize that the conditional normalizing flow serves only as one realization strategy for the stochastic flow map. The proposed framework itself is independent of the particular generative architecture and may incorporate alternative conditional generative models.

3.4. Recursive System Prediction. Once the model is trained, long-time trajectories are generated recursively by

$$(3.10) \quad \begin{cases} \tilde{\mathbf{X}}_0 = \mathbf{X}_0, \\ \tilde{\mathbf{X}}_{n+1} = \mathbf{R}(\mathbf{G}_\Theta(\tilde{\mathbf{X}}_n, \mathbf{z}_n)), \end{cases} \quad n = 0, 1, 2, \dots,$$

where $\mathbf{z}_n$ are standard Gaussian random samples independent of $\tilde{\mathbf{X}}_n$ and $\mathbf{R}$ denotes an enforcing operator to satisfy the physical constraints such as nonnegativity and integer-valued molecule counts – see the Section 3.5 below.

The resulting process is designed to provide a weak approximation to the true stochastic dynamics at the discrete time points, i.e.,

$$(\tilde{\mathbf{X}}_0, \tilde{\mathbf{X}}_1, \dots, \tilde{\mathbf{X}}_n) \stackrel{d}{\approx} (\mathbf{X}_0, \mathbf{X}_1, \dots, \mathbf{X}_n),$$

but operates only on a constant coarse time step Δ independent of the microscopic SSA firing times. Consequently, the learned stochastic propagator can significantly reduce the number of simulation steps required for long-time stochastic simulations.

3.5. Computational Details for Enforcing Operator.

Conservation of Molecule Numbers. In some reaction networks, the total molecule count is a conserved quantity, i.e.,

$$(3.11) \quad \mathbf{X}_n \cdot \mathbf{1}_N = \mathbf{X}_0 \cdot \mathbf{1}_N, \quad \text{for all } n = 0, 1, \dots,$$

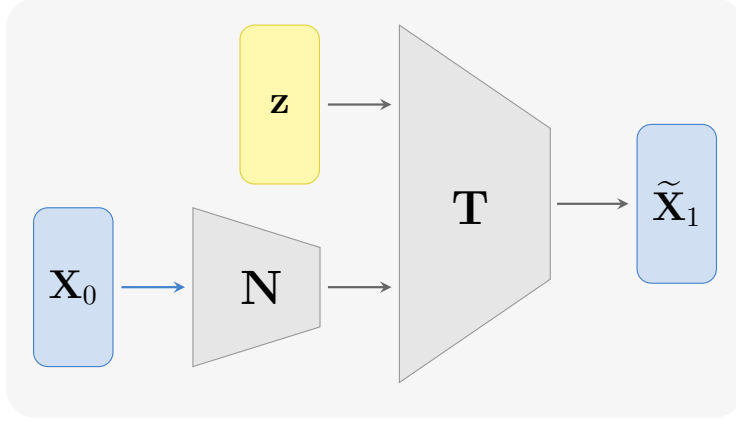


FIG. 1. The DNN model structure for the proposed normalizing flow sFML method (3.7).

where $\mathbf{1}_N$ denotes the N -dimensional vector of all ones. In this case, the transition kernel $P_\Delta(\mathbf{x}, \cdot)$ is supported on the $(N - 1)$ -dimensional hyperplane $\{\mathbf{x}' : \mathbf{x}' \cdot \mathbf{1}_N = \mathbf{x} \cdot \mathbf{1}_N\}$, a constraint that will not be preserved automatically by generic generative models. We enforce this constraint strongly into the model construction.

Let $\mathbf{A} \in \mathbb{R}^{(N-1) \times N}$ be a slicing matrix consisting of the first $N - 1$ rows of the $N \times N$ identity matrix. We replace the training dataset with the projected pairs

$$(3.12) \quad (\mathbf{Y}_0^{(i)}, \mathbf{Y}_1^{(i)}) := (\mathbf{X}_0^{(i)}, \mathbf{A}\mathbf{X}_1^{(i)}), \quad i = 1, \dots, K,$$

and train an $(N - 1)$ -dimensional conditional normalizing flow model based on the new dataset

$$(3.13) \quad \mathbf{Y}_1 = \hat{\mathbf{T}}_{\mathbf{N}_\Theta(\mathbf{Y}_0)}(\hat{\mathbf{z}})$$

following the procedure stated in previous sections. Here $\hat{\mathbf{z}}$ follows an $(N - 1)$ -dimensional unit Gaussian distribution.

After training, the full N -dimensional model prediction is then recovered using the dimension-extension operator $\mathbf{E} : \mathbb{R}^N \times \mathbb{R}^{N-1} \rightarrow \mathbb{R}^N$, defined by

$$(3.14) \quad \mathbf{E}(\mathbf{x}, \mathbf{y}) := [\mathbf{y}; \mathbf{x} \cdot \mathbf{1}_N - \mathbf{y} \cdot \mathbf{1}_{N-1}]^T,$$

which reconstructs the N -th component so that the total count is exactly preserved. The resulting model takes the form

$$(3.15) \quad \mathbf{G}_\Theta(\mathbf{x}, \hat{\mathbf{z}}) = \mathbf{E}\left(\mathbf{x}, \hat{\mathbf{T}}_{\mathbf{N}_\Theta(\mathbf{x})}(\hat{\mathbf{z}})\right),$$

which satisfies $\mathbf{G}_\Theta(\mathbf{x}, \hat{\mathbf{z}}) \cdot \mathbf{1}_N = \mathbf{x} \cdot \mathbf{1}_N$ for any Θ by construction. Other invariants of this type, including those expressible as signal temporal logic (STL) requirements [6], are left for future work.

Post-processing. A notable feature of chemical reaction systems is that the state $\mathbf{X}(t)$ takes values of non-negative integers, as each component represents a molecule count. Since the output of the normalizing flow lies in $\mathbb{R}^N$, we apply a deterministic post-processing operator $\mathbf{R} : \mathbb{R}^N \rightarrow \mathbb{Z}_{\geq 0}^N$ after each recursive step, defined

by

$$(3.16) \quad \mathbf{R}(\mathbf{x}) = \text{Round}(\max(\mathbf{x}, \mathbf{0})),$$

where $\max(\cdot, \mathbf{0})$ enforces nonnegativity componentwise and $\text{Round}(\cdot)$ maps each component to the nearest integer. The operator $\mathbf{R}$ ensures that all generated states remain physically admissible. This projection onto the discrete state space may be interpreted as an abstraction of the learned real-valued model, we refer to [4] for interested readers.

4. Numerical Examples. In this section, we present several numerical experiments to demonstrate the performance of the proposed sFML model across the following reaction networks:

- Transfer process;
- Lotka-Volterra model;
- The Brusselator;
- The Autocatalysis;
- The Oregonator.

In all examples, the reaction networks are known, and the training data are generated via SSA following the procedure described in Section 3.2. Initial conditions are sampled uniformly from a bounded domain specified for each example, along with the chosen coarse time step Δ . Once trained, the model is used to generate ensemble predictions with new initial conditions, which are compared against the SSA ground truth and the classical acceleration method tau-leaping using the following quantities:

- Number of simulation steps required to reach the same termination time, used as a measure of computational efficiency;
- Simulated trajectories and their mean and standard deviation, for visual comparisons as well as basic statistics;
- Fourier modes for examples exhibiting periodic behavior;
- Distributional comparison of the learned stochastic flow map $\mathbf{G}_\Theta(\mathbf{x}, \cdot)$ and the SSA transition kernel $P_\Delta(\mathbf{x}, \cdot)$ at fixed conditioning states $\mathbf{x}$.

We use the Masked Autoregressive Flow (MAF, c.f. [39]) model as a technical realization of the normalizing flow. Upon use of the model, there are 4 coupling flow mappings constructed with independent training parameters. The training data size, batch size, and number of training epochs will be specified in each example. We use fully connected neural networks with 3 layers, each of which has 20 nodes, with a tanh activation function. We employ the cyclic learning rate with a base rate 3×10^{-4} and a maximum rate 5×10^{-4} , $\gamma = 0.99999$, and step size 10,000. Simultaneously, the whole cycle is set to decay with a scale 0.5 for every 40,000 training epoch. To reduce the overfitting phenomenon, we also add a small weight decay with a scale 0.01 on the updated gradients.

For SSA, one simulation step corresponds to one reaction firing event. For sFML, one simulation step corresponds to one recursive evaluation of the learned stochastic flow map $\mathbf{G}_\Theta$ over the prescribed coarse time step Δ , and therefore the number of sFML steps is exactly T/Δ . We also include tau-leaping as a classical accelerated SSA method. When the propensity functions remain approximately unchanged over a time interval τ , tau-leaping approximates the reaction firing counts by independent Poisson random variables,

$$(4.1) \quad \mathbf{X}(t + \tau) = \mathbf{X}(t) + \sum_{\mu=1}^M \mathbf{s}_\mu \mathcal{P}_\mu(a_\mu(\mathbf{X}(t))\tau),$$

where $\mathcal{P}_\mu(\lambda)$ is the Poisson random variable with mean λ ; see [8, 23]. We use tau-leaping in two different ways. The first is adaptive tau-leaping, where the internal step size is chosen automatically to satisfy the leaping condition. This is used in Table 1 to compare the number of simulation steps. The second is fixed-step tau-leaping with $\tau = \Delta$, which is used as an equal-step baseline against sFML to compare approximation accuracy in Table 3. In this paper, the tau-leaping method is implemented with the Python package GillesPy2 **TauLeapingSolver** [34].

Table 1 compares the simulation steps of SSA, adaptive tau-leaping, and sFML. The SSA and adaptive tau-leaping steps are averaged over simulated trajectories, while the sFML steps are fixed by T/Δ . Across the examples, sFML takes substantially fewer steps than SSA and adaptive tau-leaping.

TABLE 1

Comparison of simulation steps. SSA and tau-leaping steps are estimated as averages over 10,000 simulated trajectories and rounded to two significant figures. sFML steps equal T/Δ exactly. The ratios for SSA and adaptive tau-leaping are defined as their respective steps divided by the sFML steps.

Example	T	sFML Steps	SSA		Adaptive Tau-leaping	
			Steps	Ratio	Steps	Ratio
Transfer	10	1.0×10^2	1.9×10^2	$1.93\times$	1.8×10^2	$1.75\times$
LV (slow)	20	2.0×10^2	7.2×10^3	$36\times$	2.5×10^3	$12.3\times$
LV (fast)	20	2.0×10^3	6.0×10^5	$300\times$	3.2×10^4	$15.8\times$
Brusselator	15	1.5×10^3	1.6×10^6	$1070\times$	2.1×10^5	$143\times$
Autocatalysis	10	1.0×10^3	3.2×10^5	$317\times$	7.6×10^3	$7.65\times$
Oregonator	6	6.0×10^2	7.1×10^5	$1187\times$	9.0×10^4	$149\times$

We next evaluate the local accuracy of the learned one-step transition kernel. For a fixed conditioning state $\mathbf{x}$, let $\{\mathbf{X}_1^{(i)}\}_{i=1}^m$ be samples generated by SSA from $P_\Delta(\mathbf{x}, \cdot)$, and let $\{\tilde{\mathbf{X}}_1^{(j)}\}_{j=1}^n$ be samples generated by $\mathbf{G}_\Theta(\mathbf{x}, \boldsymbol{\xi})$. We compare these two empirical distributions using the empirical estimator of the (square of) Maximum Mean Discrepancy (MMD), defined by

$$(4.2) \quad \widehat{\text{MMD}}_h^2 = \frac{1}{m^2} \sum_{i,i'=1}^m k_h(\mathbf{X}_1^{(i)}, \mathbf{X}_1^{(i')}) + \frac{1}{n^2} \sum_{j,j'=1}^n k_h(\tilde{\mathbf{X}}_1^{(j)}, \tilde{\mathbf{X}}_1^{(j')}) - \frac{2}{mn} \sum_{i=1}^m \sum_{j=1}^n k_h(\mathbf{X}_1^{(i)}, \tilde{\mathbf{X}}_1^{(j)}),$$

with the Gaussian kernel

$$(4.3) \quad k_h(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|_2^2}{2h^2}\right).$$

The number reported in Table 2 is $\widehat{\text{MMD}}_h$, where h is set to the median pairwise distance of the samples [24]. For each example, we take one SSA test trajectory, randomly select 30 conditioning states along that trajectory, and compute the MMD at each selected state with 10,000 conditional samples for both SSA and sFML. Table 2 reports the minimum, median, and maximum MMD values over these 30 conditioning states. These values reflect the one-step distributional accuracy along the test manifolds sampled by the SSA trajectories.

TABLE 2

One-step transition-kernel comparison. For each example, MMD is computed at 30 conditioning states sampled from one SSA test trajectory. The table reports the minimum, median, and maximum MMD over the selected states.

Example	Min MMD	Median MMD	Max MMD
Transfer	1.89×10^{-2}	5.59×10^{-2}	1.05×10^{-1}
LV (slow)	1.72×10^{-2}	4.46×10^{-2}	9.60×10^{-2}
LV (fast)	1.55×10^{-2}	5.07×10^{-2}	8.72×10^{-2}
Brusselator	8.09×10^{-2}	2.29×10^{-1}	2.83×10^{-1}
Autocatalysis	1.81×10^{-2}	4.18×10^{-2}	9.35×10^{-2}
Oregonator	5.28×10^{-2}	1.53×10^{-1}	3.09×10^{-1}

Finally, we compare long-time ensemble statistics over the full recursive rollout. Let $t_k = k\Delta$, $k = 0, \dots, K$, and define

$$(4.4) \quad \boldsymbol{\mu}_k = \mathbb{E}(\mathbf{X}_k), \quad \tilde{\boldsymbol{\mu}}_k = \mathbb{E}(\tilde{\mathbf{X}}_k), \quad \boldsymbol{\sigma}_k = \text{STD}(\mathbf{X}_k), \quad \tilde{\boldsymbol{\sigma}}_k = \text{STD}(\tilde{\mathbf{X}}_k).$$

The trajectory-level relative errors are

$$(4.5) \quad E_\mu = \left(\frac{\sum_{k=0}^K \|\boldsymbol{\mu}_k - \tilde{\boldsymbol{\mu}}_k\|_2^2}{\sum_{k=0}^K \|\boldsymbol{\mu}_k\|_2^2} \right)^{1/2}, \quad E_\sigma = \left(\frac{\sum_{k=0}^K \|\boldsymbol{\sigma}_k - \tilde{\boldsymbol{\sigma}}_k\|_2^2}{\sum_{k=0}^K \|\boldsymbol{\sigma}_k\|_2^2} \right)^{1/2}.$$

These errors measure the discrepancy of the mean and standard-deviation curves over the whole discrete trajectory. Table 3 reports these errors for sFML and fixed-step tau-leaping with the same coarse step size $\tau = \Delta$. This provides a quantitative counterpart to the mean and standard-deviation plots shown below for each example.

TABLE 3

Global trajectory comparison with fixed-step tau-leaping. Tau-leaping uses the same coarse step size $\tau = \Delta$ as sFML. Errors are estimated from 10,000 trajectories.

Example	T	Δ	Method	E_μ	E_σ
Transfer	10	0.1	sFML	1.78×10^{-3}	7.92×10^{-2}
			Tau-leaping	7.64×10^{-3}	4.81×10^{-2}
LV (slow)	20	0.1	sFML	2.16×10^{-2}	3.07×10^{-2}
			Tau-leaping	1.70×10^{-1}	3.04×10^0
LV (fast)	20	0.01	sFML	8.31×10^{-3}	5.43×10^{-2}
			Tau-leaping	unstable	unstable
Brusselator	15	0.01	sFML	6.87×10^{-2}	9.31×10^{-2}
			Tau-leaping	2.44×10^{-1}	2.93×10^{-1}
Autocatalysis	10	0.01	sFML	2.18×10^{-2}	6.91×10^{-2}
			Tau-leaping	2.68×10^{-1}	3.67×10^0
Oregonator	6	0.01	sFML	4.41×10^{-1}	2.78×10^{-1}
			Tau-leaping	3.72×10^{-1}	3.92×10^{-1}

4.1. Transfer Process. We first consider the following transfer process



where there are 3 species and 2 reactions, with reaction rates $c_1 = c_2 = 1.0$. This system has a conserved total molecule count $X_1 + X_2 + X_3$, so the dimension-reduction procedure of Section 3.5 is applied. A total of 40,000 training samples are generated with initial conditions drawn from $\mathcal{U}([0, 100] \times [0, 60] \times [50, 180])$. The time step is set to $\Delta = 0.1$, and the model is trained for 200,000 epochs.

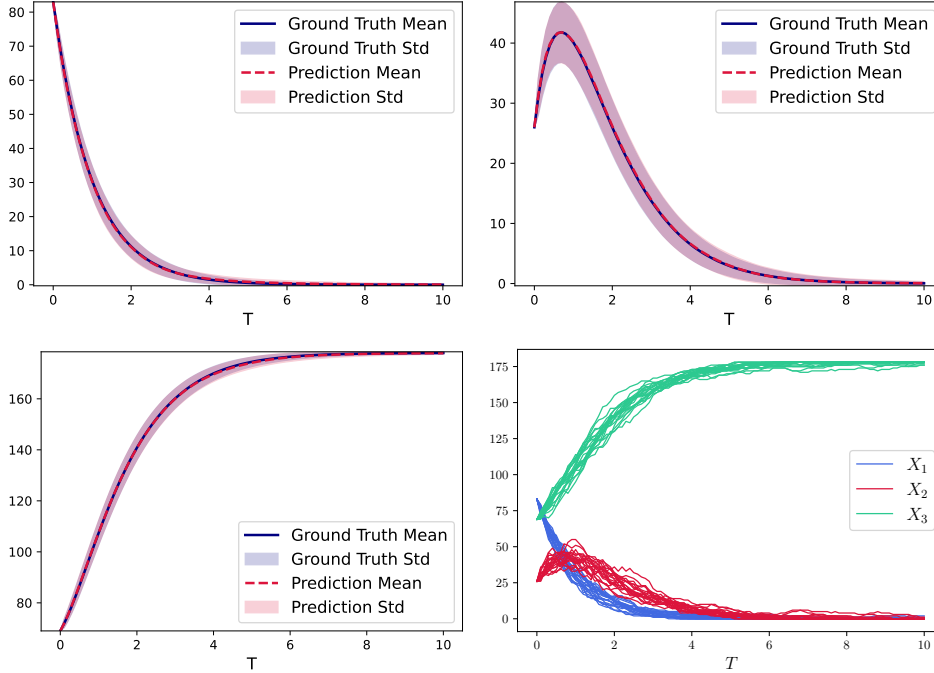


FIG. 2. Mean and standard deviation (STD) of Example 4.1 with initial condition (83, 26, 69): upper left: X_1 , upper right: X_2 , lower left: X_3 . Lower right: sample trajectories of Example 4.1 with initial condition (83, 26, 69).

The trained model is tested with the initial condition (83, 26, 69). The lower right panel of Figure 2 shows several independent predicted trajectory samples up to $T = 10.0$, along with the mean and standard deviation averaged over 10,000 trajectories. The predictions are in close agreement with the reference statistics over a long time period, which demonstrates the accuracy of our proposed method.

4.2. Lotka-Volterra Model. We next consider the Lotka-Volterra model



where there are 2 species and 3 reactions with rates c_1 , c_2 , and c_3 . This is the classical biological model for the interaction between a predator population (X_2) and a prey

population (X_1). We examine two parameter regimes that pose qualitatively different challenges to the learned model.

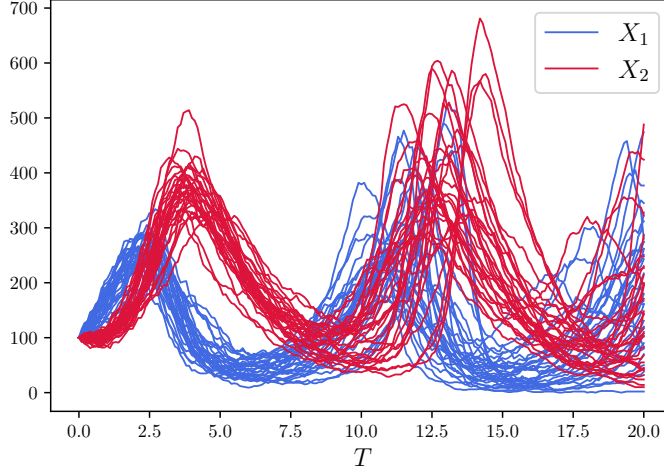


FIG. 3. Test sample trajectories of Example 4.2 with initial condition $(100, 100)$ and parameter $c_1 = 1.0$, $c_2 = 0.005$ and $c_3 = 0.6$ up to $T = 20$.

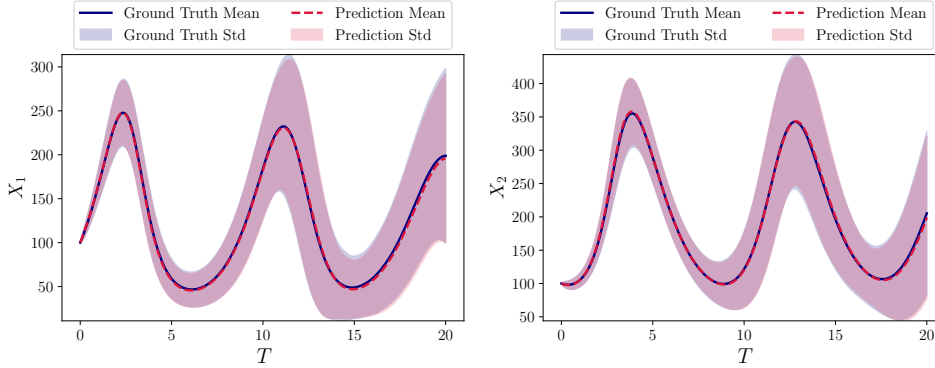


FIG. 4. Mean and standard deviation (STD) of Example 4.2 with initial condition $(100, 100)$ and parameters $c_1 = 1.0$, $c_2 = 0.005$ and $c_3 = 0.6$ up to $T = 20$: left: number of X_1 , right: number of X_2 .

Slow Reaction. With $c_1 = 1.0$, $c_2 = 0.005$, and $c_3 = 0.6$, the system evolves on a long time scale with mild, irregular oscillations. A total of 120,000 training samples are generated with initial conditions drawn from $\mathcal{U}([0, 650] \times [0, 700])$. The time step is $\Delta = 0.1$, and the model is trained for 200,000 epochs.

The trained model is tested with the initial condition $\mathbf{X}_0 = (100, 100)$. Figure 3 shows independent sample trajectories produced by the learned model, and Figure 4 presents the mean and standard deviation comparison using those samples. The results are in close agreement with the reference.

Fast Reaction. We consider the parameter set from [22], with $c_1 = 10$, $c_2 = 0.01$, and $c_3 = 10$. Under such a setting, the system exhibits sharp transients and rapid

oscillations. A total of 120,000 training samples are generated with initial conditions drawn from $\mathcal{U}([0, 5000]^2)$. The time step is $\Delta = 0.01$, chosen to resolve the sharp transients in the dynamics, and the model is trained for 400,000 epochs.

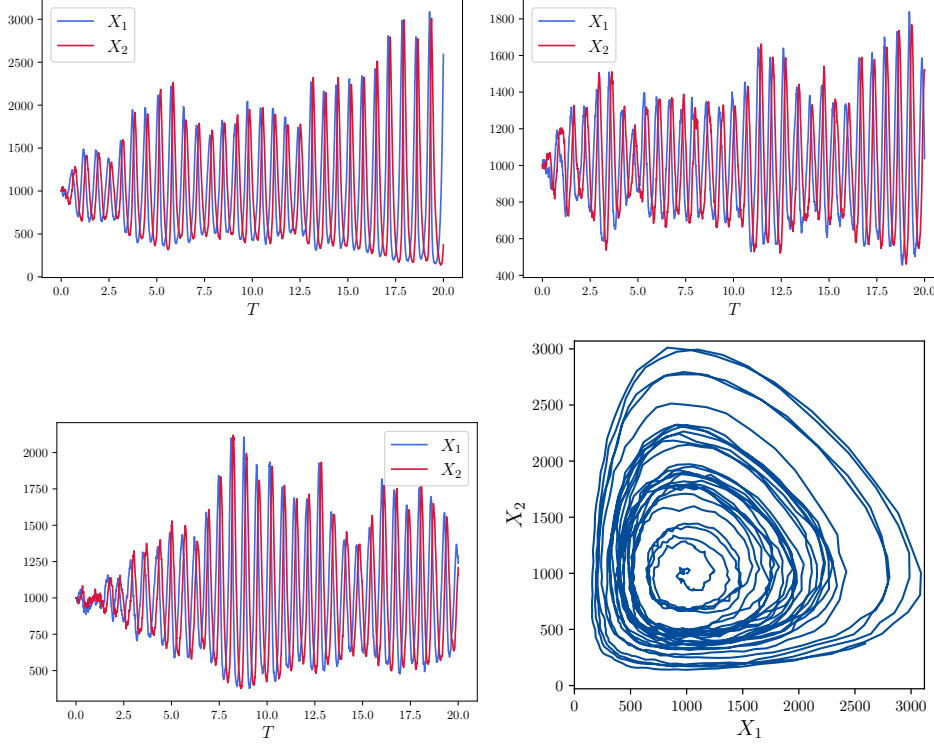


FIG. 5. Sample trajectories produced by trained model of Example 4.2 with initial condition $(1000, 1000)$ and parameter $c_1 = 10$, $c_2 = 0.01$ and $c_3 = 10$ up to $T = 20$. Upper left, upper right, lower left: 3 independent simulations, lower right: phase plot of upper left.

Upon successful training, the model is tested with an initial condition $\mathbf{X}_0 = (1000, 1000)$. Figure 5 presents 3 independent trajectory samples in state space and the phase space plot of the upper left case produced by the learned model. Figure 6 shows the corresponding SSA reference trajectories. The two sets exhibit consistent qualitative behavior. To further assess accuracy, we compare the one-step conditional distributions at condition $\mathbf{x} = (540, 734)$: 10,000 samples are drawn from the trained model $\mathbf{G}_\Theta(\mathbf{x}, \cdot)$ and compared against the SSA referenced transition kernel $P_\Delta(\mathbf{x}, \cdot)$ in Figure 7. The marginal distributions of X_1 and X_2 are also shown and confirm the accuracy of the learned model.

4.3. The Brusselator.

We consider the Brusselator



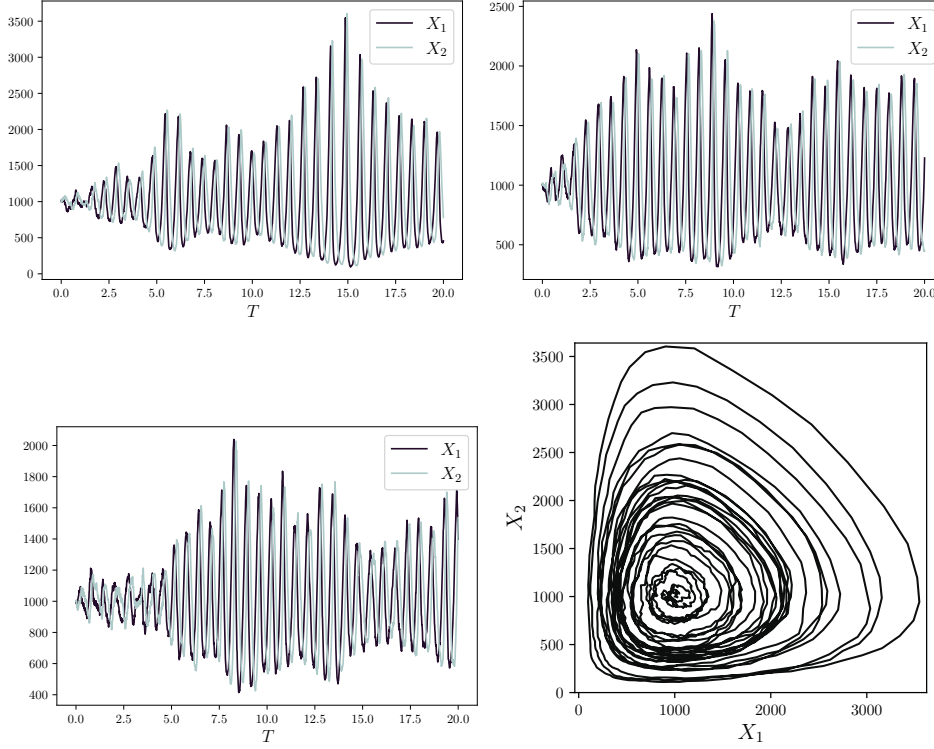


FIG. 6. Sample trajectories produced by SSA of Example 4.2 with initial condition (1000, 1000) and parameter $c_1 = 10$, $c_2 = 0.01$ and $c_3 = 10$ up to $T = 20$. Upper left, upper right, lower left: 3 independent simulations, lower right: phase plot of upper left.

where there are 2 species and 4 reactions, with $\bar{Y}_1$ and $\bar{Y}_2$ treated as parameters and c_1 , c_2 , c_3 , and c_4 the reaction rates. A total of 120,000 training samples are generated with initial conditions drawn from $\mathcal{U}([0, 5000]^2)$, parameter values $c_1\bar{Y}_1 = 5000$, $c_2\bar{Y}_2 = 50$, $c_3 = 5 \times 10^{-5}$, $c_4 = 5$. The time step is set to $\Delta = 0.01$, and the model is trained for 400,000 epochs.

Upon successful training, the model is tested with an initial condition $\mathbf{X}_0 = (1000, 2000)$. Figure 8 presents a comparison of trajectories (in both state and phase spaces) generated by SSA and by the trained model. A consistent, nearly periodic behavior is observed across both sets of results. To further validate the dynamical properties, we apply the Fast Fourier Transform (FFT) to compute the Fourier modes of the two trajectories, presented in Figure 10. The first, second, and third dominant modes of the two trajectories agree closely, demonstrating that the learned model has captured the correct oscillation frequency. Furthermore, a comparison of histograms of the conditional distributions is presented in Figure 9. With condition $\mathbf{x} = (1959, 971)$, 10,000 samples are generated from the trained model $\mathbf{G}_\Theta(\mathbf{x}, \cdot)$ and displayed in the upper right of Figure 9, while the corresponding ground-truth histogram obtained from SSA transition kernel $P_\Delta(\mathbf{x}, \cdot)$ is shown in the upper left. The marginal distributions of X_1 and X_2 are also compared, confirming the accuracy of the learned model.

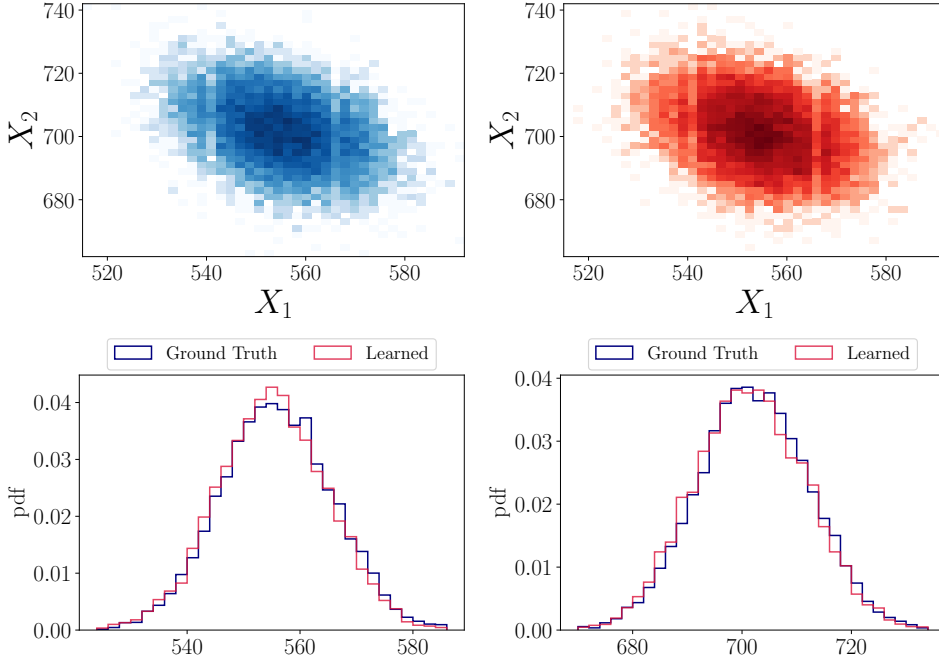


FIG. 7. Comparison of one-step conditional distribution of Example 4.2 with condition $\mathbf{x} = (540, 734)$ and parameter $c_1 = 10$, $c_2 = 0.01$ and $c_3 = 10$. Upper left: ground truth of joint distribution of (X_1, X_2) , upper right: learned joint distribution of (X_1, X_2) , lower left: comparison of the marginal distribution for X_1 , lower right: comparison of the marginal distribution for X_2 .

4.4. The Autocatalysis. We consider the autocatalysis process



where there are 3 species and 3 reactions with rates $c_1 = c_2 = c_3 = 0.002$. A total of 120,000 training samples are generated with initial conditions drawn from $\mathcal{U}([1000, 4000]^3)$. The time step is taken as $\Delta = 0.01$. The training epochs are taken to be 400,000.

This system conserves the total molecule count, so the dimension-reduction procedure of Section 3.5 applies. The trained model is tested with an initial condition $\mathbf{X}_0 = (3000, 2000, 2000)$. Figure 11 compares trajectories in the state space produced by SSA and the learned model. The system exhibits periodic oscillations, and the dominant Fourier modes of the two trajectories, presented in Figure 12, are in close agreement. This confirms that the cyclic oscillation frequency is correctly learned.

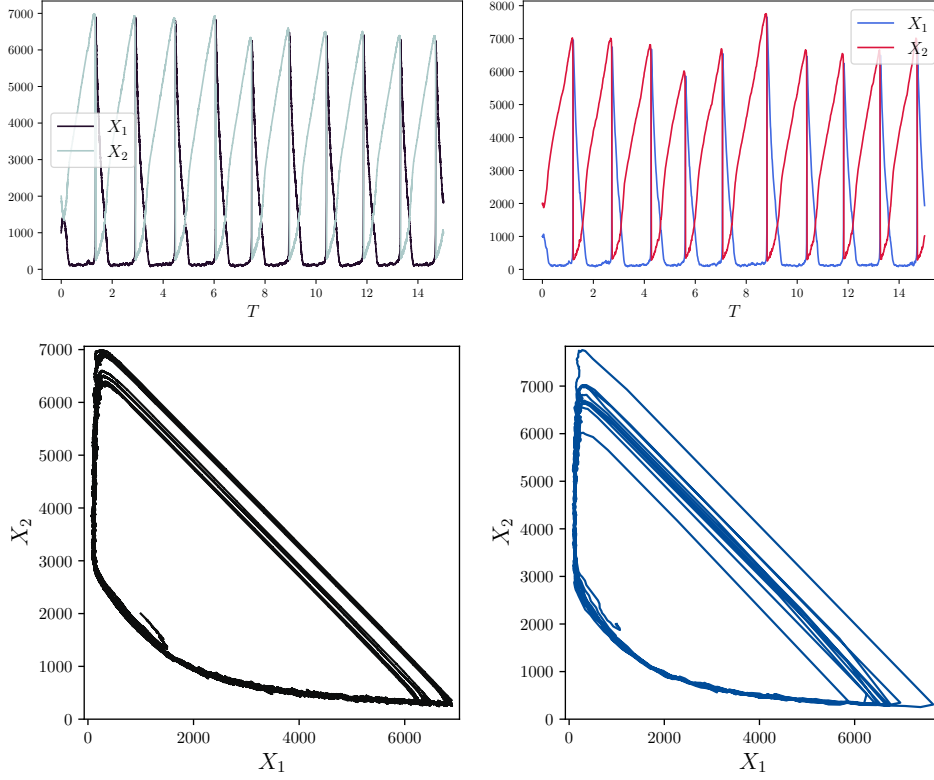


FIG. 8. Comparison of sample trajectories of Example 4.3 with initial condition $(1000, 2000)$ and parameter $c_1\bar{Y}_1 = 5000$, $c_2\bar{Y}_2 = 50$, $c_3 = 5 \times 10^{-5}$ and $c_4 = 5$ up to $T = 15$. Upper left: a sample trajectory in state space produced by SSA, upper right: a sample trajectory in state space produced by learned model, lower left: phase plot of upper left, lower right: phase plot of upper right.

4.5. The Oregonator. We consider the Oregonator model

$$(4.10a) \quad \bar{Y}_1 + X_2 \xrightarrow{c_1} X_1$$

$$(4.10b) \quad X_1 + X_2 \xrightarrow{c_2} Z_1$$

$$(4.10c) \quad \bar{Y}_2 + X_1 \xrightarrow{c_3} 2X_1 + X_3$$

$$(4.10d) \quad 2X_1 \xrightarrow{c_4} Z_2$$

$$(4.10e) \quad \bar{Y}_3 + X_3 \xrightarrow{c_5} X_2,$$

where there are 3 species and 5 reactions, $\bar{Y}_1$, $\bar{Y}_2$ and $\bar{Y}_3$ are treated as parameters, c_1 , c_2 , c_3 , c_4 and c_5 are reaction rates, with $c_1\bar{Y}_1 = 2$, $c_2 = 0.1$, $c_3\bar{Y}_2 = 104$, $c_4 = 0.016$ and $c_5\bar{Y}_3 = 26$. A total of 120,000 training samples are generated with initial conditions drawn from $\mathcal{U}([0, 10000]^3)$. The time step is taken as $\Delta = 0.01$. The training epochs are taken to be 400,000.

The trained model is tested with initial condition $\mathbf{X}_0 = (500, 1000, 2000)$. In Figures 13 and 14, we show a comparison of trajectories (in state and phase spaces) with both SSA and our trained model. The dominant Fourier modes of the two trajectories, presented in Figure 15, agree closely, confirming the accuracy of the learned model.

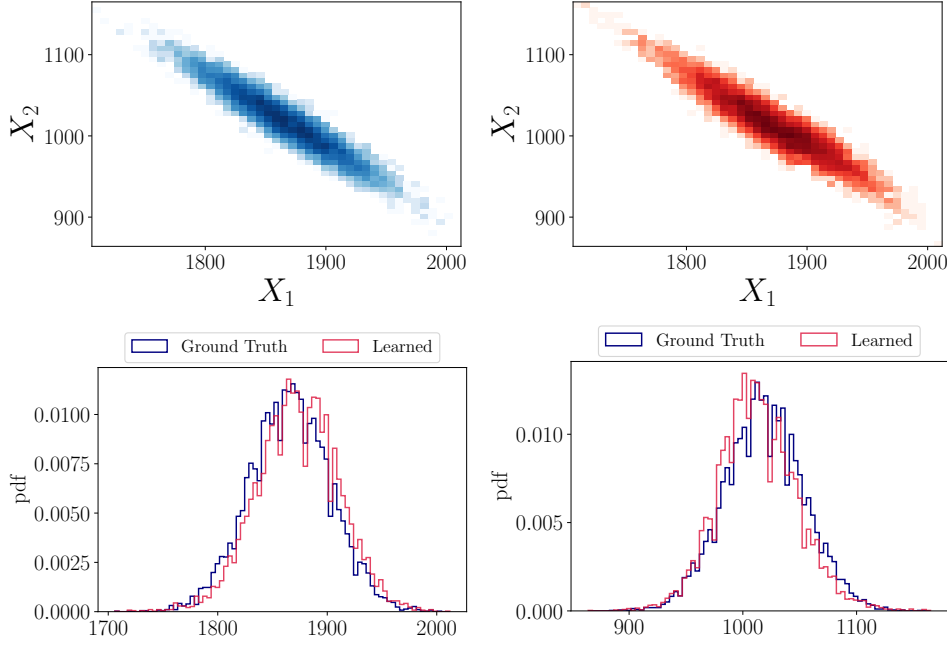


FIG. 9. Comparison of one-step conditional distribution of Example 4.3 with initial condition $(1959, 971)$ and parameter $c_1\bar{Y}_1 = 5000$, $c_2\bar{Y}_2 = 50$, $c_3 = 5 \times 10^{-5}$ and $c_4 = 5$. Upper left: ground truth of joint distribution of (X_1, X_2) , upper right: learned joint distribution of (X_1, X_2) , lower left: comparison of the marginal distribution for X_1 , lower right: comparison of the marginal distribution for X_2 .

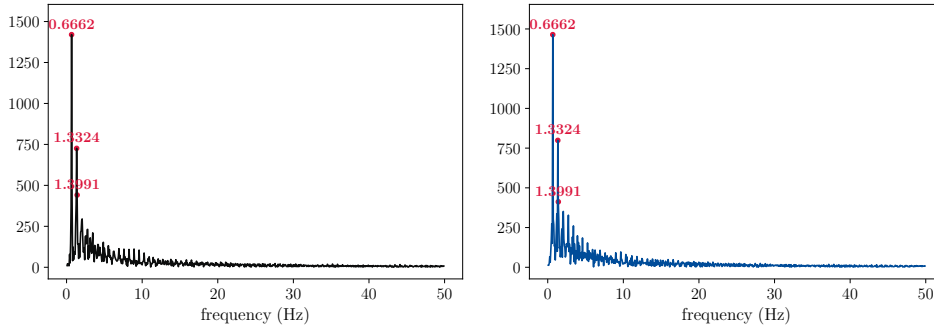


FIG. 10. Comparison of Fourier modes for sFML generated trajectory and ground truth of Example 4.3 with initial condition $(1000, 2000)$ and parameter $c_1\bar{Y}_1 = 5000$, $c_2\bar{Y}_2 = 50$, $c_3 = 5 \times 10^{-5}$ and $c_4 = 5$ up to $T = 15$. Left: Fourier modes of SSA trajectory (upper left of Figure 8), right: Fourier modes of model trajectory (upper right of Figure 8).

5. Conclusion. In this paper, we propose a data-driven effective model for the Stochastic Simulation Algorithm of stochastic reaction networks. The method directly approximates the transition kernel of the underlying continuous-time Markov chain from short bursts of SSA simulation data using a generative model. This approach naturally accommodates highly non-Gaussian transition distributions and provides a flexible framework for constructing coarse stochastic models from exact SSA data. The resulting model serves as a stochastic propagator that generates long-time trajectories

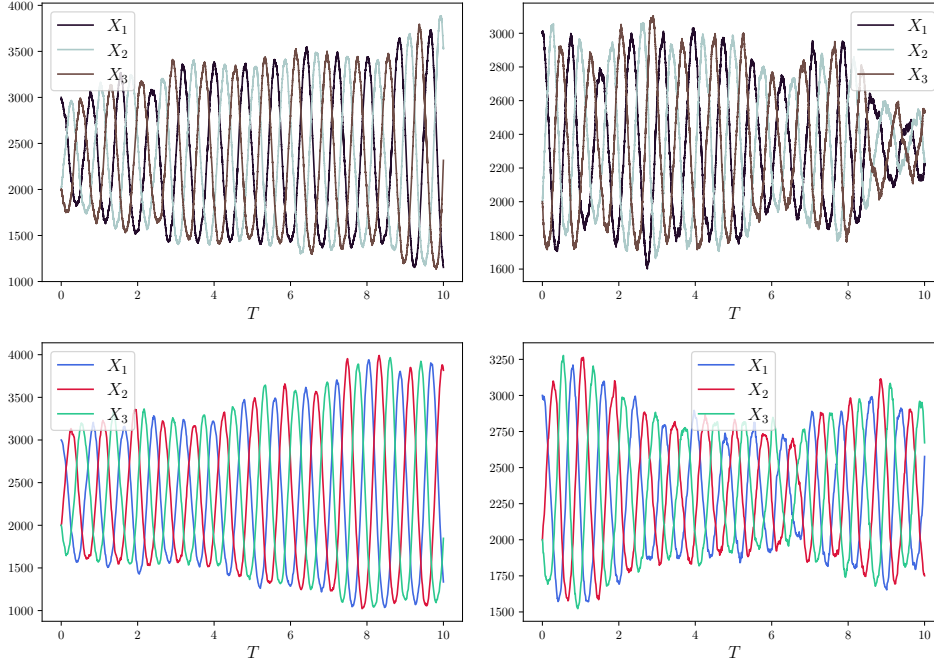


FIG. 11. Comparison of sample trajectories of Example 4.4 with initial condition (3000, 2000, 2000) and parameter $c_1 = 0.002$, $c_2 = 0.002$ and $c_3 = 0.002$ up to $T = 10$. Upper left, upper right: 2 independent sample trajectories produced by SSA, lower left, lower right: 2 independent sample trajectories produced by the trained model.

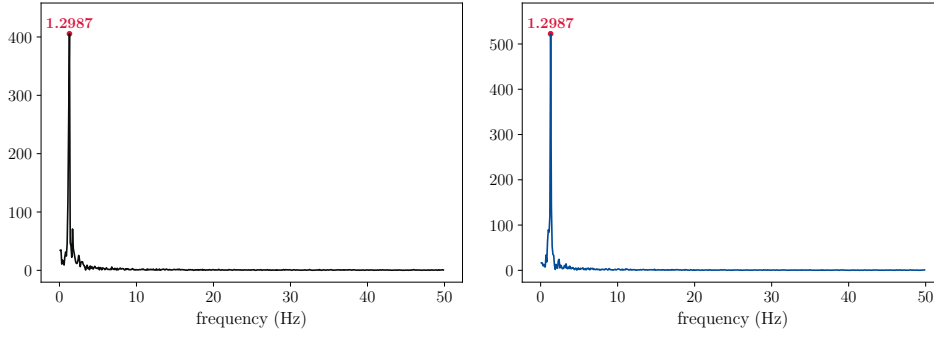


FIG. 12. Comparison of Fourier modes for sFML generated trajectory and ground truth of Example 4.4 with initial condition (3000, 2000, 2000) and parameter $c_1 = 0.002$, $c_2 = 0.002$ and $c_3 = 0.002$ up to $T = 10$. Left: Fourier modes of SSA trajectory (upper right of Figure 11), right: Fourier modes of model trajectory (lower right of Figure 11).

at a user-defined, constant coarse time step, independent of the microscopic firing times. As a consequence, the number of simulation steps is reduced significantly while the statistical properties of the true dynamics are accurately reproduced. In this paper, we employ a conditional normalizing flow as the realization of such a stochastic propagator. By using several numerical examples, we demonstrate the effectiveness and efficiency of the proposed method.

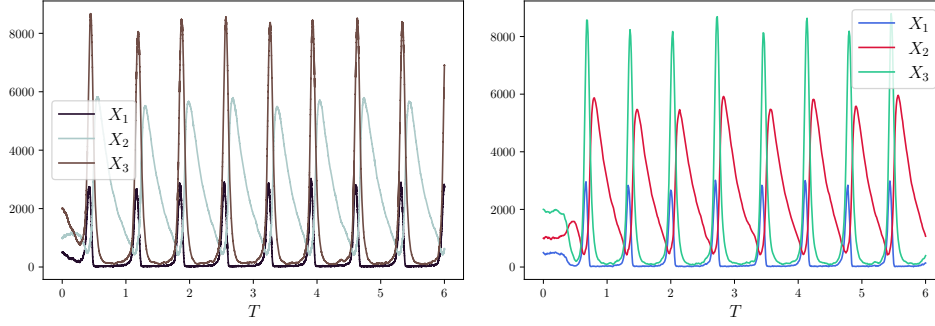


FIG. 13. Comparison of sample trajectories of Example 4.5 in state space with initial condition $(500, 1000, 2000)$ and parameter $c_1\bar{Y}_1 = 2$, $c_2 = 0.1$, $c_3\bar{Y}_2 = 104$, $c_4 = 0.016$ and $c_5\bar{Y}_3 = 26$ up to $T = 6$. Left: a sample trajectory produced by SSA, right: a sample trajectory produced by the trained model.

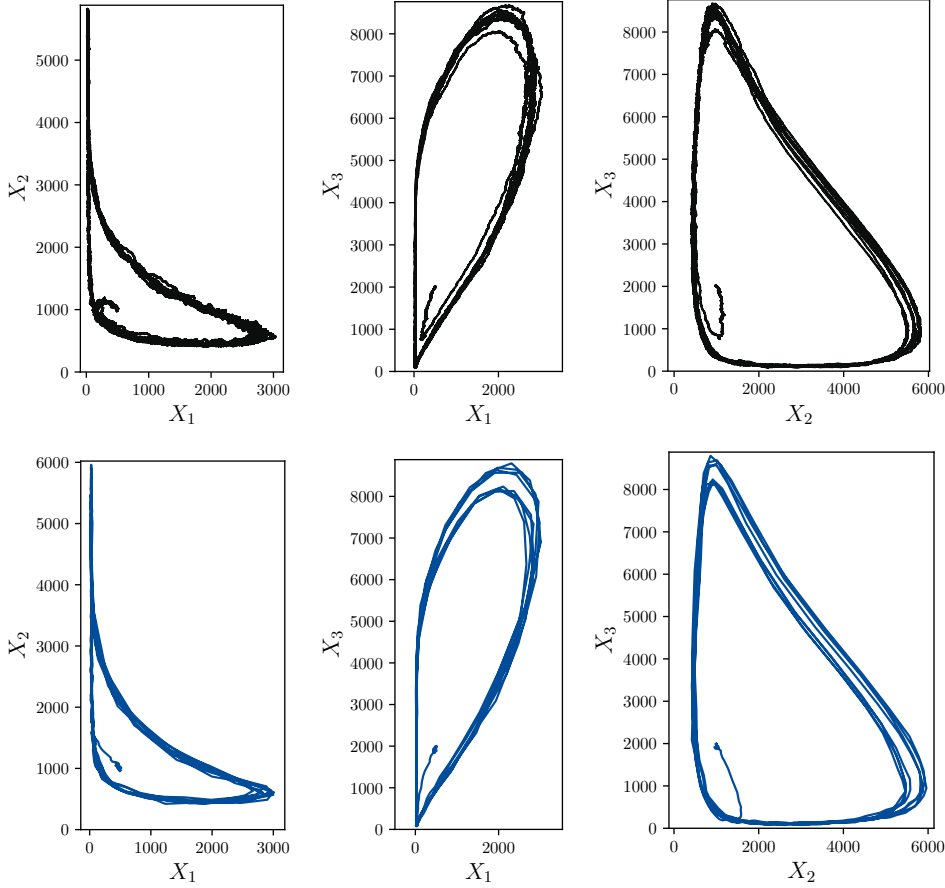


FIG. 14. Comparison of sample trajectories of Example 4.5 in phase space with initial condition $(500, 1000, 2000)$ and parameter $c_1\bar{Y}_1 = 2$, $c_2 = 0.1$, $c_3\bar{Y}_2 = 104$, $c_4 = 0.016$ and $c_5\bar{Y}_3 = 26$ up to $T = 6$. Upper: sample trajectory produced by SSA, lower: sample trajectory produced by trained model.

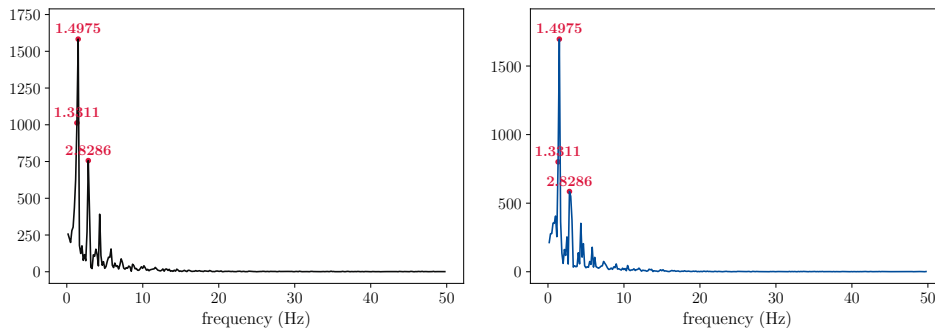


FIG. 15. Comparison of Fourier modes for two sample trajectories of Example 4.5 with initial condition (500, 1000, 2000) and parameter $c_1\bar{Y}_1 = 2$, $c_2 = 0.1$, $c_3\bar{Y}_2 = 104$, $c_4 = 0.016$ and $c_5\bar{Y}_3 = 26$ up to $T = 6$. Left: Fourier modes of SSA trajectory (left of Figure 13), right: Fourier modes of model trajectory (right of Figure 13).

Data Availability Statement. The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

- [1] D. F. ANDERSON, A. GANGULY, AND T. G. KURTZ, *Error analysis of tau-leap simulation methods*, The Annals of Applied Probability, 21 (2011), pp. 2226–2262.
- [2] D. F. ANDERSON AND T. G. KURTZ, *Stochastic Analysis of Biochemical Systems*, vol. 1.2 of Mathematical Biosciences Institute Lecture Series, Springer, 2015.
- [3] C. ARCHAMBEAU, D. CORNFORD, M. OPPER, AND J. SHAWE-TAYLOR, *Gaussian process approximations of stochastic differential equations*, in Gaussian Processes in Practice, N. D. Lawrence, A. Schwaighofer, and J. Quiñero Candela, eds., vol. 1 of Proceedings of Machine Learning Research, Bletchley Park, UK, 2007, PMLR, pp. 1–16.
- [4] L. BORTOLUSSI AND L. PALMIERI, *Deep abstractions of chemical reaction networks*, in Computational Methods in Systems Biology: 16th International Conference, CMSB 2018, Brno, Czech Republic, September 12–14, 2018, Proceedings 16, Springer, 2018, pp. 21–38.
- [5] S. L. BRUNTON, J. L. PROCTOR, AND J. N. KUTZ, *Discovering governing equations from data by sparse identification of nonlinear dynamical systems*, Proceedings of the National Academy of Sciences, 113 (2016), pp. 3932–3937.
- [6] F. CAIROLI, F. ANSELMi, A. D’ONOFRIO, AND L. BORTOLUSSI, *Generative abstraction of Markov population processes*, Theoretical Computer Science, 977 (2023), p. 114169.
- [7] Y. CAO, D. T. GILLESPIE, AND L. R. PETZOLD, *Avoiding negative populations in explicit poisson tau-leaping*, The Journal of Chemical Physics, 123 (2005), p. 054104.
- [8] Y. CAO, D. T. GILLESPIE, AND L. R. PETZOLD, *Efficient step size selection for the tau-leaping simulation method*, The Journal of Chemical Physics, 124 (2006), p. 044109.
- [9] Y. CAO, D. T. GILLESPIE, AND L. R. PETZOLD, *Adaptive explicit-implicit tau-leaping method with automatic tau selection*, The Journal of Chemical Physics, 126 (2007), p. 224101.
- [10] Y. CAO, H. LI, AND L. PETZOLD, *Efficient formulation of the stochastic simulation algorithm for chemically reacting systems*, The Journal of Chemical Physics, 121 (2004), pp. 4059–4067.
- [11] A. CHATTERJEE, D. G. VLACHOS, AND M. A. KATSOLAKIS, *Binomial distribution based tau-leap accelerated stochastic simulation*, The Journal of Chemical Physics, 122 (2005), p. 024112.
- [12] Q. CHEN, Z. XU, J. ZHANG, AND D. XIU, *Targeted digital twin via flow map learning and its application to fluid dynamics*, Journal of Computational Physics, (2026), p. 114935.
- [13] X. CHEN, J. DUAN, J. HU, AND D. LI, *Data-driven method to learn the most probable transition pathway and stochastic differential equation*, Physica D: Nonlinear Phenomena, 443 (2023), p. 133559.
- [14] X. CHEN, L. YANG, J. DUAN, AND G. E. KARNIADAKIS, *Solving inverse stochastic problems from discrete particle observations using the Fokker-Planck equation and physics-informed*

- neural networks*, SIAM Journal on Scientific Computing, 43 (2021), pp. B811–B830.
- [15] Y. CHEN AND D. XIU, *Data-driven effective modeling of multiscale stochastic dynamical systems*, Applied Mathematics for Modern Challenges, 2 (2024), pp. 348–364.
 - [16] Y. CHEN AND D. XIU, *Learning stochastic dynamical system via flow map operator*, Journal of Computational Physics, 508 (2024), p. 112984.
 - [17] Y. CHEN AND D. XIU, *Modeling unknown stochastic dynamical system subject to external excitation*, SIAM Journal on Scientific Computing, 48 (2026), pp. C216–C239.
 - [18] V. CHURCHILL AND D. XIU, *Flow map learning for unknown dynamical systems: Overview, implementation, and benchmarks*, Journal of Machine Learning for Modeling and Computing, 4 (2023), pp. 173–201.
 - [19] M. DARCY, B. HAMZI, G. LIVIERI, H. OWHADI, AND P. TAVALLALI, *One-shot learning of stochastic differential equations with data adapted kernels*, Physica D: Nonlinear Phenomena, 444 (2023), p. 133583.
 - [20] F. DIETRICH, A. MAKEEV, G. KEVREKIDIS, N. EVANGELOU, T. BERTALAN, S. REICH, AND I. G. KEVREKIDIS, *Learning effective stochastic differential equations from microscopic simulations: Linking stochastic numerics to deep learning*, Chaos: An Interdisciplinary Journal of Nonlinear Science, 33 (2023), p. 023121.
 - [21] M. A. GIBSON AND J. BRUCK, *Efficient exact stochastic simulation of chemical systems with many species and many channels*, The Journal of Physical Chemistry A, 104 (2000), pp. 1876–1889.
 - [22] D. T. GILLESPIE, *Exact stochastic simulation of coupled chemical reactions*, The Journal of Physical Chemistry, 81 (1977), pp. 2340–2361.
 - [23] D. T. GILLESPIE, *Approximate accelerated stochastic simulation of chemically reacting systems*, The Journal of Chemical Physics, 115 (2001), pp. 1716–1733.
 - [24] A. GRETTON, K. M. BORGWARDT, M. J. RASCH, B. SCHÖLKOPF, AND A. SMOLA, *A kernel two-sample test*, Journal of Machine Learning Research, 13 (2012), pp. 723–773.
 - [25] R. GRIMA, *A study of the accuracy of moment-closure approximations for stochastic chemical kinetics*, The Journal of Chemical Physics, 136 (2012), p. 154105.
 - [26] D. J. HIGHAM, *Modeling and simulating chemical reactions*, SIAM Review, 50 (2008), pp. 347–368.
 - [27] M. A. KATSOULAKIS AND A. J. MAJDA, *Coarse-grained stochastic processes for microscopic lattice systems*, Proceedings of the National Academy of Sciences, 100 (2003), pp. 782–787.
 - [28] M. A. KATSOULAKIS AND P. VILANOVA, *Data-driven, variational model reduction of high-dimensional reaction networks*, Journal of Computational Physics, 401 (2020), p. 108997.
 - [29] T. LI, *Analysis of explicit tau-leaping schemes for simulating chemically reacting systems*, Multiscale Modeling & Simulation, 6 (2007), pp. 417–436.
 - [30] Z. LI, N. B. KOVACHKI, K. AZIZZADENESHELI, B. LIU, K. BHATTACHARYA, A. STUART, AND A. ANANDKUMAR, *Fourier Neural Operator for Parametric Partial Differential Equations*, in International Conference on Learning Representations, 2021.
 - [31] Y. LIU, Y. CHEN, D. XIU, AND G. ZHANG, *A training-free conditional diffusion model for learning stochastic dynamical systems*, SIAM Journal on Scientific Computing, 47 (2025), pp. C1144–C1171.
 - [32] L. LU, P. JIN, G. PANG, Z. ZHANG, AND G. E. KARNIADAKIS, *Learning nonlinear operators via deeponet based on the universal approximation theorem of operators*, Nature Machine Intelligence, 3 (2021), pp. 218–229.
 - [33] Y. LU, X. LI, C. LIU, Q. TANG, AND Y. WANG, *Learning generalized diffusions using an energetic variational approach*, 2024.
 - [34] S. MATTHEW, F. CARTER, J. COOPER, M. DIPPEL, E. GREEN, S. HODGES, M. KIDWELL, D. NICKERSON, B. RUMSEY, J. REEVE, L. R. PETZOLD, K. R. SANFT, AND B. DRAWERT, *GillesPy2: A biochemical modeling framework for simulation driven biological discovery*, Letters in Biomathematics, 10 (2023), pp. 87–103.
 - [35] J. M. MCCOLLUM, G. D. PETERSON, C. D. COX, M. L. SIMPSON, AND N. F. SAMATOVA, *The sorting direct method for stochastic simulation of biochemical systems with varying reaction execution behavior*, Computational Biology and Chemistry, 30 (2006), pp. 39–49.
 - [36] M. OPPER, *Variational inference for stochastic differential equations*, Annalen der Physik, 531 (2019), p. 1800233.
 - [37] H. OWHADI, *Computational graph completion*, Research in the Mathematical Sciences, 9 (2022), p. 27.
 - [38] G. PAPAMAKARIOS, E. NALISNICK, D. J. REZENDE, S. MOHAMED, AND B. LAKSHMINARAYANAN, *Normalizing flows for probabilistic modeling and inference*, Journal of Machine Learning Research, 22 (2021), pp. 1–64.
 - [39] G. PAPAMAKARIOS, T. PAVLAKOU, AND I. MURRAY, *Masked autoregressive flow for density esti-*

- tion, in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017.
- [40] T. QIN, K. WU, AND D. XIU, *Data driven governing equations approximation using deep neural networks*, *Journal of Computational Physics*, 395 (2019), pp. 620–635.
 - [41] M. RAISSI, P. PERDIKARIS, AND G. KARNIADAKIS, *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*, *Journal of Computational Physics*, 378 (2019), pp. 686–707.
 - [42] M. RAISSI, P. PERDIKARIS, AND G. E. KARNIADAKIS, *Multistep neural networks for data-driven discovery of nonlinear dynamical systems*, arXiv preprint arXiv:1801.01236, (2018).
 - [43] M. RATHINAM, L. R. PETZOLD, Y. CAO, AND D. T. GILLESPIE, *Stiffness in stochastic chemically reacting systems: The implicit tau-leaping method*, *The Journal of Chemical Physics*, 119 (2003), pp. 12784–12794.
 - [44] D. SCHNOERR, G. SANGUINETTI, AND R. GRIMA, *Approximation and inference methods for stochastic biochemical kinetics—a tutorial review*, *Journal of Physics A: Mathematical and Theoretical*, 50 (2017), p. 093001.
 - [45] Y. SONG, J. SOHL-DICKSTEIN, D. P. KINGMA, A. KUMAR, S. ERMON, AND B. POOLE, *Score-based generative modeling through stochastic differential equations*, in *International Conference on Learning Representations*, 2021.
 - [46] A. SUKYS, K. ÓCAL, AND R. GRIMA, *Approximating solutions of the chemical master equation using neural networks*, *iScience*, 25 (2022), p. 105010.
 - [47] Y. TANG, J. WENG, AND P. ZHANG, *Neural-network solutions to stochastic reaction networks*, *Nature Machine Intelligence*, 5 (2023), pp. 376–385.
 - [48] M. XIA, X. LI, Q. SHEN, AND T. CHOU, *An efficient Wasserstein-distance approach for reconstructing jump-diffusion processes using parameterized neural networks*, *Machine Learning: Science and Technology*, 5 (2024), p. 045052.
 - [49] Z. XU, Y. CHEN, Q. CHEN, AND D. XIU, *Modeling unknown stochastic dynamical system via autoencoder*, *Journal of Machine Learning for Modeling and Computing*, 5 (2024), pp. 87–112.
 - [50] L. YANG, C. DASKALAKIS, AND G. E. KARNIADAKIS, *Generative ensemble regression: Learning particle dynamics from observations of ensembles with physics-informed deep generative models*, *SIAM Journal on Scientific Computing*, 44 (2022), pp. B80–B99.
 - [51] Ç. YILDIZ, M. HEINONEN, J. INTOSALMI, H. MANNERSTRÖM, AND H. LÄHDESMÄKI, *Learning stochastic differential equations with Gaussian processes without gradient matching*, in *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, IEEE, 2018, pp. 1–6.
 - [52] J. ZHANG, S. ZHANG, AND G. LIN, *MultiAuto-DeepONet: A multi-resolution autoencoder deep-onet for nonlinear dimension reduction, uncertainty quantification and operator learning of forward and inverse stochastic problems*, 2022.