SIAM J. SCI. COMPUT.
Vol. 47, No. 5, pp. C1144–C1171

A TRAINING-FREE CONDITIONAL DIFFUSION MODEL FOR LEARNING STOCHASTIC DYNAMICAL SYSTEMS*

YANFANG LIU†, YUAN CHEN‡, DONGBIN XIU‡, AND GUANNAN ZHANG§

Abstract. This study introduces a training-free conditional diffusion model for learning unknown SDEs using data. The proposed approach addresses key challenges in computational efficiency and accuracy for modeling SDEs by utilizing a score-based diffusion model to approximate their stochastic flow map. Unlike the existing methods, this technique is based on an analytically derived closed-form exact score function, which can be efficiently estimated by Monte Carlo methods using the trajectory data, and eliminates the need for neural network training to learn the score function. By generating labeled data through solving the corresponding reverse ODE, the approach enables supervised learning of the flow map. Extensive numerical experiments across various SDE types, including linear, nonlinear, and multidimensional systems, demonstrate the versatility and effectiveness of the method. The learned models exhibit significant improvements in predicting both short-term and long-term behaviors of unknown stochastic systems, often surpassing baseline methods such as generative adversarial networks in estimating drift and diffusion coefficients.

Key words. diffusion model, surrogate modeling, generative model, stochastic flow map, SDEs

MSC codes. 68Q25, 68R10, 68U05

DOI. 10.1137/24M1699589

See reproducibility of computational results at end of the article.

1. Introduction.

SDEs play a pivotal role in scientific and engineering simulations, particularly when modeling large ensembles of particles. These equations find diverse applications across fields such as plasma physics, where they describe electron and ion interactions with electromagnetic fields, and fluid mechanics, where they model the dispersion of pollutants in oceans and the atmosphere. SDEs also prove valuable in simulating biological and chemical systems. From a mathematical standpoint, SDEs offer a discrete particle representation that corresponds to the continuous Fokker–Planck equation, underpinning Monte Carlo methods for solving PDEs. While highly valuable, the use of SDEs faces a significant challenge: For many complex systems, it is difficult, if not impossible, to derive accurate SDE models that capture all the important physics embedded in these systems.

Recent years have seen a growing interest in developing data-driven approaches

*Submitted to the journal's Machine Learning Methods for Scientific Computing section October 4, 2024; accepted for publication (in revised form) July 10, 2025; published electronically October 13, 2025.

https://doi.org/10.1137/24M1699589

Funding: The work of the first and fourth authors was partially supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Applied Mathematics program, under contracts ERKJ388 and ERKJ443. ORNL is operated by UT-Battelle, LLC, for the U.S. Department of Energy under contract DE-AC05-00OR22725. The work of the second and third authors was partially supported by AFOSR FA9550-24-1-0237.

†Department of Mathematics, Middle Tennessee State University, Murfreesboro, TN 37132 USA (yanfang.liu@mtsu.edu).

‡Department of Mathematics, Ohio State University, Columbus, OH 43210 USA (chen.11050@osu.edu, xiu.16@osu.edu).

§Corresponding author. Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831 USA (zhangg@ornl.gov).

for uncovering unknown dynamical systems. The primary objective is to unearth the core principles or mathematical formulations underlying observational data, enabling the creation of effective predictive models for unexplored dynamics. When dealing with stochastic systems, the presence of data noise and the inability to directly observe the system's inherent randomness present considerable obstacles to understanding the system. Most existing approaches concentrate on analyzing Itô-type SDEs. These methods utilize various techniques, including Gaussian processes [1, 10, 26], polynomial approximations [20, 37], sparse regression such as stochastic SINDy [36], and deep neural networks [6, 7, 39]. A recent development involves the extension of the deterministic flow map approach to stochastic scenarios, incorporating generative models such as generative adversarial networks (GANs) [8, 27], autoencoders [9, 38], and normalizing flows [41]. The performance of polynomial approximation or sparse regression methods such as SINDy [36] is highly dependent on the choice of the dictionary. These methods are therefore most effective when reliable prior knowledge about the drift and diffusion terms of the dynamical system is available. In contrast, generative model–based approaches offer more robust performance in scenarios where such prior information is limited or unavailable.

Despite the success of generative models in learning SDEs, there is a significant challenge in training the generative models to achieve the desired accuracy. Due to the lack of labeled data, training generative models is usually classified as unsupervised learning. Various unsupervised loss functions have been defined to train generative models, including adversarial loss for GANs [14], the maximum likelihood loss for normalizing flows [17], and the score matching losses for diffusion models [16, 32, 35]. There are several computational issues resulting from the unsupervised training nature of generative models. For example, the training of GANs may suffer from mode collapse, vanishing gradients, and training instability [30]. The maximum likelihood loss used in normalizing flows requires efficient computation of the determinant of the Jacobian matrix, which places restrictions on the architectures of networks [17].

In this work, we propose a training-free conditional diffusion model that enables supervised learning of unknown SDEs under diverse noise distributions, encompassing both Gaussian and non-Gaussian cases. Diffusion models have been successfully used in a variety of applications, including image processing [15, 29, 31, 33], natural language processing [5, 13, 19, 42], and uncertainty quantification [2, 3, 4, 12, 18, 21, 22, 23]. Unlike existing diffusion models [24, 33, 40], our method does not need to train neural networks to learn the conditional score function. Specifically, we derived the closed form of the exact score function for the conditional diffusion model used to approximate the stochastic flow map of the target SDE. In fact, the score function can be represented as an expectation with respect to the conditional distribution defined by the flow map. However, since we use trajectories of the target SDE as observation data, we do not have a large number of samples from each conditional distribution. Instead, the trajectory data can be treated as samples from the joint distribution of the input and the output of the flow map. Thus, we propose an approximate score function that involves a weighted expectation with respect to the joint distribution and use Monte Carlo estimators (based on the trajectory data) to approximate the expectation. This approach allows us to directly generate samples of the target flow map. With the estimated score function, we then generate labeled data by solving the corresponding reverse ODE instead of the reverse SDE for the diffusion model. Following this, we employ the generated labeled data to train a simple fully connected neural network to learn the flow map via supervised learning. Our method shows remarkable accuracy in predicting both short-term and long-term behaviors of sto-

chastic systems, often outperforming baseline methods such as GANs [8] in terms of drift and diffusion coefficient estimation, as well as in predicting mean and standard deviation at termination times.

The rest of this paper is organized as follows. In section 2, we briefly introduce the learning of unknown stochastic dynamical systems under consideration. In section 3, we provide a comprehensive discussion of a conditional score–based generative diffusion model to learn the flow map. Finally, in section 4, we demonstrate the performance of our method by applying it to a set of benchmark stochastic dynamical systems.

2. Problem setting. Let $\{\Omega, \mathcal{F}, \mathbb{P}, \{\mathcal{F}_t\}_{0\le t\le T}\}$ be a complete, filtered probability space on which a standard m-dimensional Brownian motion W_t is defined such that $\{\mathcal{F}_t\}_{0\le t\le T}$ is the natural filtration of the Brownian motion W_t and all the $\mathbb{P}$-null sets are augmented to each σ-field $\mathcal{F}_t$. In the probability space $\{\Omega, \mathcal{F}, \mathbb{P}, \{\mathcal{F}_t\}_{0\le t\le T}\}$, we define the following d-dimensional SDE:

$$dX_t = a(X_t)dt + b(X_t)\, dW_t, \tag{2.1}$$

where $a : \mathbb{R}^d \to \mathbb{R}^d$ is referred to as the drift coefficient and $b : \mathbb{R}^d \to \mathbb{R}^{d\times m}$ with $m \le d$ is referred to as the diffusion coefficient. Assuming that $a(x)$ and $b(x)$ satisfy the Lipschitz condition and the linear growth condition, the SDE in (2.1) has a unique solution,

$$X^{t,x}_{t+\Delta t} = x + \int_t^{t+\Delta t} a(X^{t,x}_s)ds + \int_t^{t+\Delta t} b(X^{t,x}_s)dW_s \ \text{ with } \ X_t = x, \tag{2.2}$$

for any $0 \le t \le T$ and $\Delta t > 0$, where $X^{t,x}_{t+\Delta t}$ is the solution of the SDE at $t + \Delta t$ conditional on $X_t = x$ and $\int_t^{t+\Delta t} b(X^{t,x}_s)dW_s$ is an Itô integral. Because both the drift and diffusion terms do not explicitly depend on the time t, the probability distribution of increment $X^{t,x}_{t+\Delta t} - x$ only depends on the starting location x and the time step Δt regardless of the time instant t in (2.2). Therefore, the representation in (2.2) defines a stochastic flow map,

$$F_{\Delta t}(x, \omega) := X^x_{\Delta t} - x, \tag{2.3}$$

where we omit the dependence of $X^x_{\Delta t}$ on t, ω represents the sample from the probability space $\{\Omega, \mathcal{F}, \mathbb{P}, \{\mathcal{F}_t\}_{0\le t\le T}\}$, and $F_{\Delta t}(x, \omega)$ is independent of $t \in [0, T]$.

The objective of this work is to build a conditional generative model, denoted by $G_\theta(x, z)$, to approximate the exact flow map $F_{\Delta t}(x, \omega)$ in (2.3), i.e.,

$$G_\theta(x, z) \approx F_{\Delta t}(x, \omega), \tag{2.4}$$

where x denotes a sample of X_t for $t \in [0, T]$, z denotes a sample from the standard normal distribution, and θ denotes the set of trainable parameters of the conditional generative model. For notational simplicity, we omit the subscript Δt in $G_\theta(x, z)$ in the rest of this paper. Also, we introduce a uniform temporal mesh,

$$\mathcal{T} := \{t_n : t_n = n\Delta t \ \text{ for } \ n = 0, 1, \ldots, N_T\}, \tag{2.5}$$

where $\Delta t = T/N_T$, and we focus on approximating the flow map $F_{\Delta t}(x, \omega)$ defined by the time step Δt associated with the temporal mesh $\mathcal{T}$. Once the generative model $G_\theta(x, z)$ is well trained, it can be used to efficiently generate unlimited trajectories of the SDE in (2.1) in an autoregressive manner.

The observation dataset of the target SDE includes $H \geq 1$ solution trajectories of X_t at discrete time instants on the uniform temporal mesh $\mathcal{T}$ in (2.5), denoted by

$$X_{t_0}^{(i)}, X_{t_1}^{(i)}, \cdots, X_{t_L}^{(i)}, \qquad i = 1, \cdots, H, \tag{2.6}$$

where $\{t_0, t_1, \ldots, t_L\} \in \mathcal{T}$, $L+1$ denotes the length of the ith trajectory, and the initial state $X_{t_0}^{(i)}$ is a sample from a given initial distribution $p_{X_0}(x)$. The trajectory data in (2.6) can be separated and regrouped to obtain data pairs that can be used to describe the input-output relationship of the flow map $F_{\Delta t}(x, \omega)$, i.e.,

$$x_m := X_{t_l}^{(i)} \text{ and } \Delta x_m := X_{t_{l+1}}^{(i)} - X_{t_l}^{(i)} \tag{2.7}$$

for $m = l \times H + i$ with $l = 0, \ldots, L-1, i = 1, \ldots, H$, which leads to a total of $M = HL$ adjacent data pairs. We denote the collection of the paired samples as the observation dataset for the flow map, i.e.,

$$\mathcal{D}_{\text{obs}} := \{(x_m, \Delta x_m) \,|\, m = 1, \cdots, M = HL\}. \tag{2.8}$$

The next section will describe how to use the observation dataset $\mathcal{D}_{\text{obs}}$ in (2.8) to perform a supervised training of the desired generative model in (2.3).

3. The conditional diffusion model for supervised learning of the flow map. We now describe the details of the proposed training-free conditional diffusion model for learning the flow map of interest. Section 3.1 introduces the conditional score–based diffusion model, section 3.2 introduces the definition of the conditional score function as well as its training-free approximation scheme, and section 3.3 describes how to use the proposed conditional diffusion model to generate labeled data for supervised training of the desired conditional generative model $G_\theta(\cdot, \cdot)$ in (2.4).

3.1. The conditional score–based diffusion model. We intend to define a score-based diffusion model to represent a transport map from a standard normal random variable, denoted by $Z \sim \mathcal{N}(0, \mathbf{I}_d)$, to the random variable $X_{\Delta t}^x - x$ in (2.3) conditional on $X_t = x$ for any $t \in [0, T]$. It is referred to as a conditional diffusion model because the generated probability distribution also depends on the conditional state x. The diffusion model includes a forward SDE and a reverse SDE, both of which are defined in a bounded temporal domain $\tau \in [0, 1]$. The forward SDE is defined by

$$dZ_\tau^x = b(\tau) Z_\tau^x d\tau + \sigma(\tau) dW_\tau \text{ with } Z_0^x = X_{\Delta t}^x - x, \tag{3.1}$$

where Z_0^x is the target random variable $X_{\Delta t}^x - x$ for a fixed x. Note that the entire forward SDE for the diffusion model is conditional on $X_t = x$ for any $t \in [0, T]$. When properly choosing the drift and diffusion coefficients, the forward SDE can transport the initial distribution $p_{Z_0^x}(z_0^x)$ to the standard normal distribution at $\tau = 1$. In this work, we use the following definitions of $b(\tau)$ and $\sigma(\tau)$ in (3.1):

$$b(\tau) = \frac{\mathrm{d} \log \alpha_\tau}{\mathrm{d}\tau} \quad \text{and} \quad \sigma^2(\tau) = \frac{\mathrm{d}\beta_\tau^2}{\mathrm{d}\tau} - 2\frac{\mathrm{d} \log \alpha_\tau}{\mathrm{d}\tau}\beta_\tau^2, \tag{3.2}$$

where α_τ and β_τ are set as

$$\alpha_\tau = 1 - \tau, \ \beta_\tau^2 = \tau \quad \text{for} \quad \tau \in [0, 1). \tag{3.3}$$

Because the forward process in (3.1) is a *linear* SDE, its solution is

$$Z_\tau^x = Z_0^x \exp\left[\int_0^\tau b(s) ds\right] + \int_0^\tau \exp\left[\int_s^\tau b(r) dr\right] \sigma(s) dW_s. \tag{3.4}$$

Substituting $b(\tau)$ and $\sigma(\tau)$ defined in (3.2) into (3.4), we have the conditional PDF $q_{Z_\tau^x|Z_0^x}(z_\tau^x|z_0^x)$ for a fixed z_0^x being a Gaussian distribution,

$$q_{Z_\tau^x|Z_0^x}(z_\tau^x|z_0^x) = \psi_{(\alpha_\tau z_0^x, \beta_\tau^2 \mathbf{I}_d)}(z_\tau^x|z_0^x), \tag{3.5}$$

where $\psi_{(\alpha_\tau z_0^x, \beta_\tau^2 \mathbf{I}_d)}(\cdot)$ is the standard normal PDF with mean $\alpha_\tau z_0^x$ and covariance matrix $\beta_\tau^2 \mathbf{I}_d$. Then the asymptotic limit of $q_{Z_\tau^x|Z_0^x}(z_\tau^x|z_0^x)$ as $\tau \to 1$ is the standard normal distribution, which means that the forward SDE can transport any initial distribution to the standard normal distribution within a bounded pseudotemporal domain $[0, 1)$.

The corresponding reverse SDE, which is used to generate new samples of the target distribution, is defined by

$$dZ_\tau^x = \left[b(\tau) Z_\tau^x - \sigma^2(\tau) S(Z_\tau^x, \tau)\right] d\tau + \sigma(\tau) dB_\tau \quad \text{with } Z_1^x = Z \sim \mathcal{N}(0, \mathbf{I}_d), \tag{3.6}$$

where B_τ is the reverse-time Brownian motion and $S(z_\tau^x, \tau)$ is the score function defined by

$$S(z_\tau^x, \tau) := \nabla_z \log p_{Z_\tau^x}(z_\tau^x), \tag{3.7}$$

where $p_{Z_\tau^x}(z_\tau^x)$ is the PDF of the state Z_τ^x in the forward SDE in (3.1). The reverse SDE in (3.6) performs as a denoiser which can transform the standard normal distribution of $p_{Z_1^x}(z_1^x)$ to the target distribution $p_{Z_0^x}(z_0^x)$ for $Z_0^x = X_{\Delta t}^x - x$. If the score function is known, then we can solve the reverse SDE in (3.6) to generate an unlimited amount of samples of the flow map in (2.3). The standard score-based diffusion model uses a neural network to learn the score function, which is computationally intensive because it requires solving the reverse SDE to generate each sample of the target distribution. Moreover, when estimating the score function, the neural network is trained in an unsupervised manner due to the lack of labeled data, which requires storing a large number of trajectories of the forward SDE to be used to train the score function approximation.

3.2. Training-free score estimation. We introduce the score estimation approach in this subsection. To proceed, we first write out the PDF $p_{Z_\tau^x}(z_\tau^x)$ in (3.7) as the following integral form:

$$q_{Z_\tau^x}(z_\tau^x) = \int_{\mathbb{R}^d} q_{Z_\tau^x, Z_0^x}(z_\tau^x, z_0^x) dz_0^x = \int_{\mathbb{R}^d} q_{Z_\tau^x|Z_0^x}(z_\tau^x|z_0^x) q_{Z_0^x}(z_0^x) dz_0^x, \tag{3.8}$$

where $q_{Z_\tau^x|Z_0^x}(z_\tau^x|z_0^x)$ is defined in (3.5) and $q_{Z_0^x}(z_0^x)$ is represented by

$$q_{Z_0^x}(z_0^x) = p_{X_{\Delta t}^x - x}(z_0^x), \tag{3.9}$$

according to the definition $Z_0^x = X_{\Delta t}^x - x$ given in (3.1). Substituting (3.8) and (3.9) into (3.7), we have

$$\begin{aligned} S(z_\tau^x, \tau) &= \nabla_z \log \left(\int_{\mathbb{R}^d} q_{Z_\tau^x|Z_0^x}(z_\tau^x|z_0^x) q_{Z_0^x}(z_0^x) dz_0^x \right) \\ &= \frac{1}{\int_{\mathbb{R}^d} q_{Z_\tau^x|Z_0^x}(z_\tau^x|\bar{z}_0^x) q_{Z_0^x}(\bar{z}_0^x) d\bar{z}_0^x} \int_{\mathbb{R}^d} -\frac{z_\tau^x - \alpha_\tau z_0^x}{\beta_\tau^2} q_{Z_\tau^x|Z_0^x}(z_\tau^x|z_0^x) q_{Z_0^x}(z_0^x) dz_0^x \\ &= \int_{\mathbb{R}^d} -\frac{z_\tau^x - \alpha_\tau z_0^x}{\beta_\tau^2} w(z_\tau^x, z_0^x) q_{Z_0^x}(z_0^x) dz_0^x, \end{aligned} \tag{3.10}$$

where the weight function $w(z_\tau^x, z_0^x)$ is defined by

$$w(z_\tau^x, z_0^x) := \frac{q_{Z_\tau^x|Z_0^x}(z_\tau^x|z_0^x)}{\displaystyle\int_{\mathbb{R}^d} q_{Z_\tau^x|Z_0^x}(z_\tau^x|\bar{z}_0^x) q_{Z_0^x}(\bar{z}_0^x) d\bar{z}_0^x}, \tag{3.11}$$

satisfying $\int_{\mathbb{R}^d} w(z_\tau^x, z_0^x) q_{Z_0^x}(z_0^x) dz_0^x = 1$.

Instead of training a neural network to learn the score function, we use Monte Carlo estimation to directly approximate the integrals in (3.10). Unlike unconditional diffusion models where we have samples from $q_{Z_0^x}(z_0^x)$ as the observation data, we do *not* have a large number of samples from $Z_0^x = X_{\Delta t}^x - x$ for any fixed x due to the way the observation dataset $\mathcal{D}_{\text{obs}}$ in (2.8) is constructed. On the other hand, there may be many samples in $\mathcal{D}_{\text{obs}}$ that are located within a small neighborhood of x. Hence, we need to develop an approximation to the initial distribution $q_{Z_0^x}(z_0^x)$. Specifically, we can rewrite $q_{Z_0^x}(z_0^x)$ as

$$q_{Z_0^x}(z_0^x) = \int_{\mathbb{R}^d} q_{Z_0^{\tilde{x}}}(z_0^{\tilde{x}}) \delta(\tilde{x} - x) d\tilde{x}, \tag{3.12}$$

where $\delta(\cdot)$ is the Dirac delta function. If we treat $\delta(\tilde{x} - x)$ as a posterior distribution of $\tilde{x}$ conditional on x, then we can define an approximation scheme of $\delta(\tilde{x} - x)$ using the Bayesian formula

$$\delta(\tilde{x} - x) \approx \psi_{(x, \nu^2 \mathbf{I}_d)}(\tilde{x})\, p_X(\tilde{x}), \tag{3.13}$$

where $\psi_{(x, \nu^2 \mathbf{I}_d)}(\tilde{x})$ is the Gaussian likelihood with mean x and standard deviation $\nu > 0$ and $p_X(\tilde{x})$ is the prior distribution. It is easy to see that the posterior distribution $\psi_{(x, \nu^2 \mathbf{I}_d)}(\tilde{x})\, p_X(\tilde{x})$ converges to $\delta(\tilde{x} - x)$ as $\nu \to 0$ in (3.13). Substituting (3.13) into (3.12), we have an approximation of $q_{Z_0^x}(z_0^x)$, i.e.,

$$q_{Z_0^x}(z_0^x) \approx \widehat{q}_{Z_0^x}(z_0^x) := \int_{\mathbb{R}^d} \psi_{(x, \nu^2 \mathbf{I}_d)}(\tilde{x})\, q_{Z_0^{\tilde{x}}}(z_0^{\tilde{x}})\, p_X(\tilde{x}) d\tilde{x}, \tag{3.14}$$

where the product $q_{Z_0^{\tilde{x}}}(z_0^{\tilde{x}})\, p_X(\tilde{x})$ is treated as a joint PDF of $(\tilde{x}, z_0^{\tilde{x}})$, denoted by

$$q_{X, Z_0}(\tilde{x}, z_0^{\tilde{x}}) := q_{Z_0^{\tilde{x}}}(z_0^{\tilde{x}})\, p_X(\tilde{x}). \tag{3.15}$$

Substituting (3.14) into (3.10), we have an approximate score function as follows:

$$\begin{aligned}
\widehat{S}(z_\tau^x, \tau) &:= \nabla_z \log \left(\int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \left[q_{Z_\tau^{\tilde{x}}|Z_0^{\tilde{x}}}(z_\tau^x|z_0^{\tilde{x}})\, \psi_{(x, \nu^2 \mathbf{I}_d)}(\tilde{x}) \right] q_{X, Z_0}(\tilde{x}, z_0^{\tilde{x}})\, dz_0^{\tilde{x}} d\tilde{x} \right) \\
&= \frac{\displaystyle\int_{\mathbb{R}^d} \int_{\mathbb{R}^d} -\frac{z_\tau^x - \alpha_\tau z_0^{\tilde{x}}}{\beta_\tau^2} \left[q_{Z_\tau^{\tilde{x}}|Z_0^{\tilde{x}}}(z_\tau^x|z_0^{\tilde{x}})\, \psi_{(x, \nu^2 \mathbf{I}_d)}(\tilde{x}) \right] q_{X, Z_0}(\tilde{x}, z_0^{\tilde{x}})\, dz_0^{\tilde{x}} d\tilde{x}}{\displaystyle\int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \left[q_{Z_\tau^{\bar{x}}|Z_0^{\bar{x}}}(z_\tau^x|\bar{z}_0^{\bar{x}})\, \psi_{(x, \nu^2 \mathbf{I}_d)}(\bar{x}) \right] q_{X, Z_0}(\bar{x}, \bar{z}_0^{\bar{x}})\, d\bar{z}_0^{\bar{x}} d\bar{x}} \\
&= \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} -\frac{z_\tau^x - \alpha_\tau z_0^{\tilde{x}}}{\beta_\tau^2} \widehat{w}(z_\tau^x, z_0^{\tilde{x}})\, q_{X, Z_0}(\tilde{x}, z_0^{\tilde{x}})\, dz_0^{\tilde{x}}\, d\tilde{x},
\end{aligned} \tag{3.16}$$

where the weight function $\widehat{w}(z_\tau^x, z_0^{\tilde{x}})$ is defined by

$$\widehat{w}(z_\tau^x, z_0^{\tilde{x}}) := \frac{q_{Z_\tau^{\tilde{x}}|Z_0^{\tilde{x}}}(z_\tau^x|z_0^{\tilde{x}})\, \psi_{(x, \nu^2 \mathbf{I}_d)}(\tilde{x})}{\displaystyle\int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \left[q_{Z_\tau^{\bar{x}}|Z_0^{\bar{x}}}(z_\tau^x|\bar{z}_0^{\bar{x}})\, \psi_{(x, \nu^2 \mathbf{I}_d)}(\bar{x}) \right] q_{X, Z_0}(\bar{x}, \bar{z}_0^{\bar{x}})\, d\bar{z}_0^{\bar{x}} d\bar{x}}, \tag{3.17}$$

satisfying that $\int_{\mathbb{R}^d}\int_{\mathbb{R}^d}\widehat{w}(z_\tau^x, z_0^{\tilde{x}})q_{X,Z_0}(\tilde{x}, z_0^{\tilde{x}})dz_0^{\tilde{x}}d\tilde{x} = 1$.

Note that we have not yet specified the prior distribution $p_X(\tilde{x})$ in (3.13). In this work, we assume that the samples $\{x_m\}_{m=1}^M$ of $\mathcal{D}_{\text{obs}}$ are samples from the prior distribution $p_X(\tilde{x})$. Then $\{(x_m, \Delta x_m)\}_{m=1}^M$ are samples from the joint distribution defined in (3.15). Therefore, we can use the samples in $\mathcal{D}_{\text{obs}}$ to define the Monte Carlo estimator of $\widehat{S}(z_\tau^x, \tau)$ in (3.16) as follows:

$$\widehat{S}^{\text{MC}}(z_\tau^x, \tau) := \frac{1}{M}\sum_{m=1}^{M} -\frac{z_\tau^x - \alpha_\tau z_0^{x_m}}{\beta_\tau^2}\widehat{w}^{\text{MC}}(z_\tau^x, z_0^{x_m}), \tag{3.18}$$

where the weight $\widehat{w}^{\text{MC}}(z_\tau^x, z_0^{x_m})$ is defined by

$$\widehat{w}^{\text{MC}}(z_\tau^x, z_0^{x_m}) := \frac{\exp\left\{\dfrac{-(z_\tau^x - \alpha_\tau \Delta x_m)^2}{2\beta_\tau^2}\right\}\exp\left\{-\dfrac{\|x - x_m\|_2^2}{2\nu^2}\right\}}{\displaystyle\sum_{m'=1}^{M}\exp\left\{\dfrac{-(z_\tau^x - \alpha_\tau \Delta x_{m'})^2}{2\beta_\tau^2}\right\}\exp\left\{-\dfrac{\|x - x_{m'}\|_2^2}{2\nu^2}\right\}} \tag{3.19}$$

for $z_0^{x_m} = \Delta x_m$ and $m = 1, \ldots, M$. We observe that $\exp\{-\|x - x_m\|_2^2/(2\nu^2)\}$ in (3.19) determines how much contribution each sample in $\mathcal{D}_{\text{obs}}$ makes to the calculation of the approximate score function $\widehat{S}^{\text{MC}}$. The expressions in (3.18) and (3.19) are mathematically rigorous in the sense that $\widehat{S}^{\text{MC}}$ converges to the exact score as $\nu \to 0$ and $M \to \infty$. However, (3.18) and (3.19) may be computationally expensive, especially when M is very large. Therefore, in the numerical experiments in section 4, we select a subset[1] of $\mathcal{D}_{\text{obs}}$ that contains the closest neighboring samples around x and only compute the weight $\exp\{-\|x - x_m\|_2^2/(2\nu^2)\}$ for the subset with $\nu = 1$. In this way, a much smaller number of samples are involved in the computation of the Monte Carlo estimator $\widehat{S}^{\text{MC}}$. Moreover, because $\exp\{-\|x - x_m\|_2^2/(2\nu^2)\}$ is independent of the pseudotime τ, we do not need to update the subset during the solving of the reverse SDE.

There are several advantages of the proposed training-free score estimators compared to learning the score function using neural networks. First, our method does not require solving the forward SDE in (3.1) and storing a large number of trajectories because it does not need to use the trajectory data to train a neural network to learn the score function. Instead, our method can directly solve the reverse SDE in (3.6) and use the Monte Carlo estimators to approximate the score function at any state z_τ^x. Second, it is easy to parallelize the solution process of the reverse SDE on a large number of GPUs. In our previous work [28], we implemented a parallel-in-time method to solve the reverse process of the unconditional version of our diffusion model, but the parallelization approach can be easily extended to the conditional setting.

3.3. Supervised learning of the generative model using labeled data. In this section, we describe how to leverage the score function estimated in section 3.2 to generate labeled data for the target stochastic flow map $F_{\Delta t}(x, \omega)$ in (2.3) such that we can train the desired generator G_θ in (2.4) using supervised learning. We observe that the flow map $G_\theta(x, z)$ is a deterministic map from the terminal state Z_1^x to the initial state Z_0^x. However, the stochastic nature of the reverse SDE in (3.6) leads to a random map from Z_1^x to Z_0^x. This means that the reverse SDE cannot be used to generate labeled data for supervised training of $G_\theta(x, z)$. Fortunately, we can convert the reverse SDE to a reverse ODE using the property that

[1]The size of the subset is set to 1% of the total number of samples M.

$$\nabla q_{Z_\tau^x}(z_\tau^x) = q_{Z_\tau^x}(z_\tau^x)\nabla \log(q_{Z_\tau^x}(z_\tau^x)) = q_{Z_\tau^x}(z_\tau^x)S(z_\tau^x,\tau), \tag{3.20}$$

which holds true for all differentiable PDFs $q_{Z_\tau^x}$. Specifically, the PDF $q_{Z_\tau^x}(z_\tau^x)$ of the reverse SDE's state is the solution of a reverse Fokker–Planck equation,

$$\frac{\partial q_{Z_\tau^x}(z_\tau^x)}{\partial \tau} = \nabla \cdot \left[(b(\tau)z_\tau^x - \sigma^2(\tau)S(z_\tau^x,\tau))q_{Z_\tau^x}(z_\tau^x) + \frac{\sigma^2(\tau)}{2}\nabla q_{Z_\tau^x}(z_\tau^x) \right], \tag{3.21}$$

where $b(\tau)$, $\sigma(\tau)$, and $S(z_\tau^x,\tau)$ are the same as in (3.6). Substituting (3.20) into (3.21), we can convert the reverse Fokker–Planck equation to a reverse convection equation, i.e.,

$$\begin{aligned} \frac{\partial q_{Z_\tau^x}(z_\tau^x)}{\partial \tau} &= \nabla \cdot \left[g(z_\tau^x,\tau) q_{Z_\tau^x}(z_\tau^x) \right], \\ g(z_\tau^x,\tau) &= (b(\tau)z_\tau^x - \sigma^2(\tau)S(z_\tau^x,\tau)) + \frac{\sigma^2(\tau)}{2}\nabla \log(q_{Z_\tau^x}(z_\tau^x)) \\ &= b(\tau)z_\tau^x - \frac{\sigma^2(\tau)}{2}S(z_\tau^x,\tau), \end{aligned} \tag{3.22}$$

which corresponds to the reverse ODE

$$dZ_\tau^x = \left[b(\tau)Z_\tau^x - \frac{1}{2}\sigma^2(\tau)S(Z_\tau^x,\tau) \right] d\tau \qquad \text{with } Z_1^x = Z \sim \mathcal{N}(0,\mathbf{I}_d), \tag{3.23}$$

whose state has the same distribution as the reverse SDE. In addition, this ODE has a unique solution and thus provides a smoother function relationship between the initial state Z_0^x and the terminal state Z_1^x. Therefore, we adopt the ODE in (3.23) to generate labeled data for supervised learning of the flow map $F_{\Delta t}(x,\omega)$ in (2.3). The equivalence of marginal distributions can be rigorously established through martingale problem theory [11, 25, 34]. In this work, we use the simple Euler scheme to solve the reverse ODE, i.e.,

$$z_{\tau_{k-1}}^x = z_{\tau_k}^x - \left[b(\tau_k)z_{\tau_k}^x - \frac{1}{2}\sigma^2(\tau_k)S(z_{\tau_k}^x,\tau_k) \right] \Delta\tau, \qquad k = K,\dots,1, \tag{3.24}$$

where $\Delta\tau = 1/K$ and $\tau_k = k\Delta\tau$ for $k = 0,\dots,K$. We denote the labeled data by

$$\mathcal{D}_{\text{label}} := \left\{ (x_j, z_j, y_j) \,\middle|\, x_j \in \mathcal{D}_{\text{obs}} \text{ and } z_j \sim \mathcal{N}(0,\mathbf{I}_d),\ j = 1,\dots,J \right\}, \tag{3.25}$$

where $J \le M$ is the size of the labeled dataset, x_j is a sample from $\{x_m\}_{m=1}^M \subset \mathcal{D}_{\text{obs}}$, z_j is a sample from the standard normal distribution $\mathcal{N}(0,\mathbf{I}_d)$, and y_j is the generated labeled data by solving the reverse ODE using the training-free score estimator in section 3.2. For $j = 1,\dots,J$, we solve the reverse ODE from $\tau = 1$ to $\tau = 0$ by setting $x = x_j$ and $z_1^{x_j} = z_j$ in (3.23). After solving the ODE, we collect the state $z_0^{x_j}$ and let $y_j = z_0^{x_j}$. Once the entire labeled dataset $\mathcal{D}_{\text{label}}$ is generated, we train a fully connected neural network $G_\theta(x,z)$ to approximate the flow map $F_{\Delta t}(x,\omega)$ in (2.3) using the standard mean squared error (MSE) loss. The workflow of the proposed method is summarized in Algorithm 3.1.

4. Numerical experiments. In this section, we present several numerical examples to demonstrate the accuracy and efficiency of the proposed training-free diffusion model for learning the flow map of stochastic dynamical systems. The examples include the following:

Algorithm 3.1. The training-free conditional diffusion model.

Input: Observation dataset $\mathcal{D}_{\text{obs}} = \{(x_m, \Delta x_m)\}_{m=1}^{M}$
Procedure:
for $j = 1, 2, \ldots, J$ **do** ▷ `Generating labeled data`
  Random select a sample x_j from $\{x_m\}_{m=1}^{M} \subset \mathcal{D}_{\text{obs}}$;
  Generate a sample z_j from $\mathcal{N}(0, \mathbf{I}_d)$, and set $z_{\tau_K}^{x_j} = z_j$ in (3.24);
  for $k = K, \ldots, 1$ **do**
    Compute the weight $\{\widehat{w}^{\text{MC}}(z_{\tau_k}^{x_j}, z_0^{x_m})\}_{m=1}^{M}$ using (3.19);
    Compute the score function $\widehat{S}^{\text{MC}}(z_{\tau_k}^{x_j}, \tau_k)$ using (3.18);
    Solve $z_{\tau_{k-1}}^{x_j}$ using the Euler scheme in (3.24)
  end for
  Set $y_j = z_{\tau_0}^{x_j}$, and assemble (x_j, z_j, y_j)
end for
Train $G_\theta(x, z)$ using the labeled data $\mathcal{D}_{\text{label}} = \{(x_j, z_j, y_j)\}_{j=1}^{J}$;
Output: The trained generative model $G_\theta(x, z)$.

- linear SDEs: Ornstein–Uhlenbeck (OU) process and geometric Brownian motion;
- nonlinear SDEs: SDEs with exponential and trigonometric drift or diffusion as well as SDEs featuring a double-well potential;
- SDEs with non-Gaussian noise of exponential and lognormal distributions;
- multidimensional SDEs: two-dimensional and five-dimensional OU processes.

Remark 4.1 (reproducibility). Our method is implemented using PyTorch with GPU acceleration. The source code has been made publicly accessible at https://github.com/YanfangLiu11/Conditional-Diffusion-Model-for-SDE-Learning. All numerical results presented in this section are fully reproducible using the provided GitHub repository.

Training dataset $\mathcal{D}_{\text{obs}}$ generation: For each example, we generate H trajectories by solving the exact SDEs using the Euler–Maruyama method, with initial conditions uniformly distributed within specified regions. These trajectories are then reorganized as data pairs $\{(x_m, \Delta x_m)\}$ as described in section 2 to form observation dataset $\mathcal{D}_{\text{obs}}$. The time step in the Euler–Maruyama method is set to $\Delta t = 0.01$ up to $T = 1.0$ for all examples (except for geometric Brownian motion with $T = 0.5$). Then each trajectory will contribute $L = 100$ (for geometric Brownian motion $L = 50$) samples to $\mathcal{D}_{\text{obs}}$. Accordingly, the size of observation data for each example is equal to $M = H \times L$, as shown in (2.8). For the training-free diffusion model, the reverse-time ODE in (3.23) is solved using the explicit Euler scheme with 10,000 time steps, i.e., $K = 10,000$ in (3.24). Each ODE solution generates a labeled data (x_j, z_j, y_j) for the labeled dataset $\mathcal{D}_{\text{label}}$. This labeled dataset $\mathcal{D}_{\text{label}}$ is then divided into training and validation sets, with 80% of the data for training and the remaining 20% for validation.

Supervised training of $G_\theta(x, z)$: We define the generative model $G_\theta(x, z)$ as a fully connected feed-forward neural network with one hidden layer to learn the flow map $F_{\Delta t}(x, \omega)$ in (2.3). Because we have labeled data $\mathcal{D}_{\text{label}}$, we use the standard MSE loss to train the generative model $G_\theta(x, z)$. For each example, the neural network is trained for 2,000 epochs using the Adam optimizer with a learning rate of 0.01. During the training period, the best neural network parameters are saved based on validation

performance to ensure optimal training loss. Since the neural network structure is simple, we use the simple grid search method to tune the number of neurons in the hidden layer.

Metrics for testing the generative model $G_\theta(x, z)$: The trained conditional generative model $G_\theta(x, z)$ is applied to predict each target SDE system for a time horizon typically up to $T = 5 \sim 500$, which is a much longer time horizon than the time horizon covered by the observation data $\mathcal{D}_{\text{obs}}$, where the time horizon is $T = 1$ or 0.5. For all the numerical examples, we simulate 500,000 trajectories using both the approximate flow map and the ground truth flow map. The prediction is then compared with the ground truth solution using the following metrics:

- the mean and standard deviation of the simulated trajectories for visual comparison of the accuracy;
- comparison of the approximate drift and diffusion coefficients, obtained from simulated trajectories, against the true drift and diffusion coefficients;
- comparison of the approximate flow map G_θ to the ground truth flow map $F_{\Delta t}$.

4.1. Linear SDEs. We first consider the learning of an OU process and a geometric Brownian motion.

4.1.1. One-dimensional OU process. The one-dimensional OU process under consideration is defined by

$$dX_t = \theta(\mu - X_t)dt + \sigma dW_t, \tag{4.1}$$

where $\theta = 1.0$, $\mu = 1.2$, and $\sigma = 0.3$. The observation dataset $\mathcal{D}_{\text{obs}}$ consists of $M = 1,500K$ data pairs from $H = 15,000$ trajectories, obtained by solving the SDE in (4.1) with initial values uniformly sampled from $\mathcal{U}(0, 2.5)$ up to the time horizon $T = 1.0$. We randomly choose $50,000$ initial states from $\mathcal{D}_{\text{obs}}$ to generate the corresponding labeled training data. After training the generative model G_θ, we simulate 500,000 prediction trajectories for the time horizon up to $T = 5.0$.

The mean and standard deviation of the predicted solutions when $X_0 = 1.5$ by the generative model, compared with those of the exact solutions, are displayed in Figure 1. Good agreements between the mean and standard deviation of the predicted solutions and those of the exact solution can be observed for time up to $T = 5.0$ despite

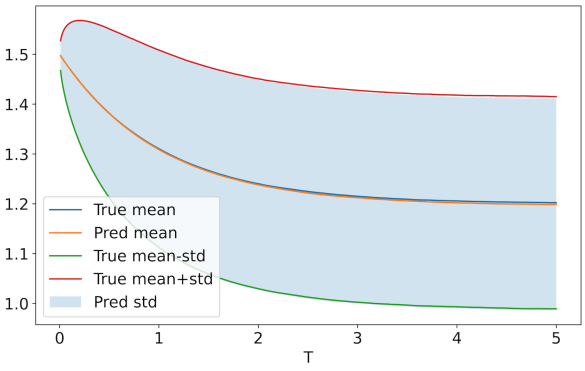

FIG. 1. *One-dimensional OU process. Comparison of the mean and standard deviation of solutions with the initial state being $X_0 = 1.5$, obtained by the generative model and the ground truth.*

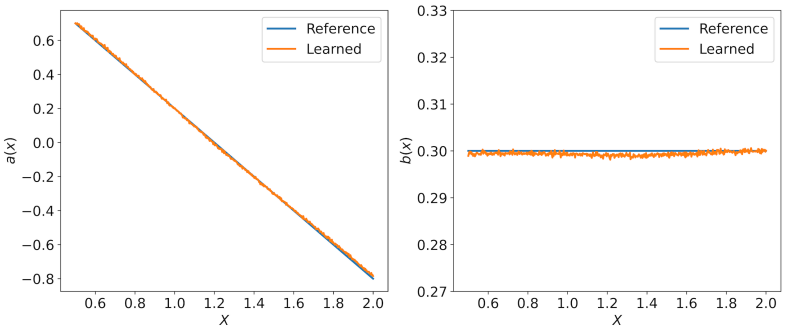

FIG. 2. *One-dimensional OU process. Comparison of effective drift and diffusion functions obtained by the simulated trajectories using the generative model and the exact SDE. Left: drift* $a(x) = \mu - x$. *Right: diffusion* $b(x) = \sigma$.

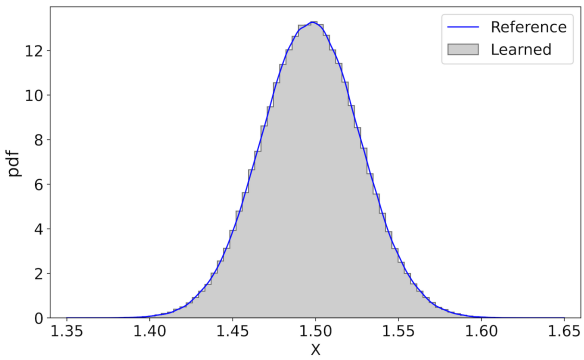

FIG. 3. *One-dimensional OU process. Comparison of conditional PDF* $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = 1.5)$ *determined by the generative model* G_θ *and the exact flow map* $F_{\Delta t}$.

the training dataset being limited to $T = 1.0$. In Figure 2, we present the effective drift and diffusion determined by the generative model, which are closely aligned with the true drift and diffusion. The relative errors for drift and diffusion functions are on the order of 10^{-2} and 10^{-3}, respectively. Figure 3 illustrates the one-step conditional distribution $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = 1.5)$ determined by the generative model G_θ and the exact flow map $F_{\Delta t}$. The predicted conditional distribution accurately approximates the exact one.

4.1.2. Geometric Brownian motion. The one-dimensional geometric Brownian motion is defined by

$$dX_t = \mu X_t dt + \sigma X_t dW_t, \tag{4.2}$$

where $\mu = 2$ and $\sigma = 1$. The observation dataset $\mathcal{D}_{\text{obs}}$ consists of $M = 5,000K$ data pairs from $H = 100,000$ trajectories of the SDE, obtained by solving the SDE in (4.2) with initial values uniformly sampled from $\mathcal{U}(0, 2)$ up to $T = 0.5$. We randomly choose $120,000$ initial states from the samples in $\mathcal{D}_{\text{obs}}$ to generate the corresponding labeled training data. After training the generative model G_θ, the prediction trajectories are simulated over a time horizon up to $T = 1.0$.

The mean and standard deviation of the predicted solutions for $X_0 = 0.5$ by the generative model, compared with those of the exact solutions, are displayed in

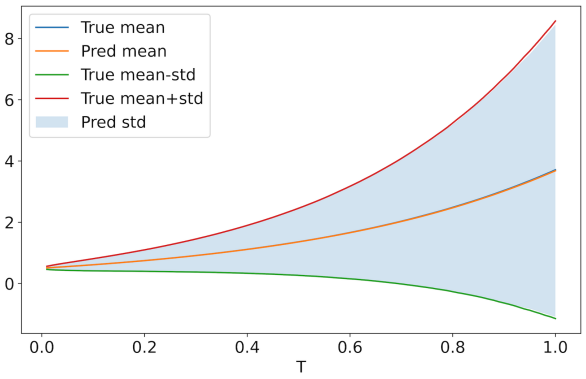

FIG. 4. *One-dimensional geometric Brownian motion. Comparison of the mean and standard deviation of solutions with the initial state being $X_0 = 0.5$, obtained by the generative model and the ground truth.*

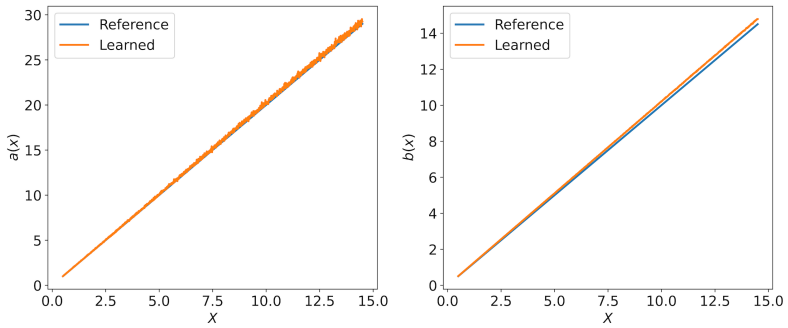

FIG. 5. *One-dimensional geometric Brownian motion. Comparison of effective drift and diffusion functions obtained by the simulated trajectories using the generative model and the exact SDE. Left: drift $a(x) = \mu x$. Right: diffusion $b(x) = \sigma x$.*

Figure 4. Good agreements between the mean and standard deviation of the predicted solutions and those of the exact solution can be observed for time up to $T = 1.0$ despite the training dataset being limited to $T = 0.5$. In Figure 5, we present the effective drift and diffusion determined by the generative model, which are closely aligned with the true drift and diffusion. The relative errors for drift and diffusion functions are on the order of 10^{-2}. Figure 6 illustrates the one-step conditional distribution $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = 5.0)$ for both the generative model and the exact SDE. The predicted conditional distribution accurately approximates the exact one.

4.2. Nonlinear SDEs. This section considers two SDEs with exponential and trigonometric drift or diffusion functions as well as SDEs featuring a double-well potential.

4.2.1. SDE with nonlinear diffusion. The SDE with a nonlinear diffusion is defined by

$$dX_t = -\mu X_t dt + \sigma e^{-X_t^2} dW_t, \tag{4.3}$$

where $\mu = 5$ and $\sigma = 0.5$. The observation dataset $\mathcal{D}_{\text{obs}}$ consists of $M = 15,000K$ data pairs from $H = 150,000$ trajectories, obtained by solving the SDE (4.3) with initial

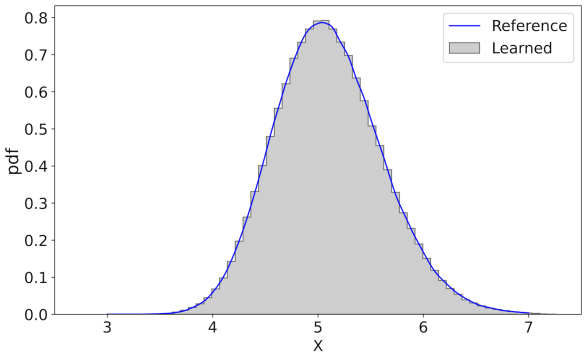

FIG. 6. *One-dimensional geometric Brownian motion. Comparison of conditional PDF* $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = 5.0)$ *determined by the generative model* G_θ *and the exact flow map* $F_{\Delta t}$.

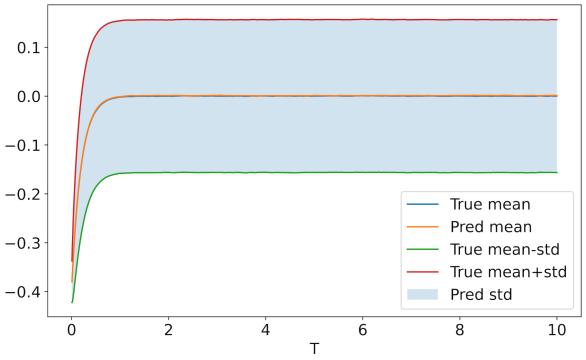

FIG. 7. *SDE with nonlinear diffusion. Comparison of the mean and tandard deviation of solutions with the initial state being* $X_0 = -0.4$*, obtained by the generative model and the ground truth.*

values uniformly sampled from $\mathcal{U}(-1,1)$ up to $T = 1.0$. We randomly choose $60,000$ initial states from $\mathcal{D}_{\text{obs}}$ to generate the corresponding labeled training data. After training the generative model G_θ, we simulate 500,000 prediction trajectories up to $T = 10.0$.

The mean and standard deviation of the predicted solutions for $X_0 = -0.4$ by the generative model, compared with those of the exact solutions, are displayed in Figure 7. Good agreements between the mean and standard deviation of the predicted solutions and those of the exact solution can be observed for time up to $T = 10.0$ despite the training dataset being limited to $T = 1.0$. In Figure 8, we present the effective drift and diffusion determined by the generative model, which are closely aligned with the true drift and diffusion coefficients. The relative errors for drift and diffusion functions are on the order of 10^{-3}. Figure 9 illustrates the one-step conditional distribution $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = -0.3)$ for both the generative model and the exact SDE. The predicted conditional distribution accurately approximates the exact one.

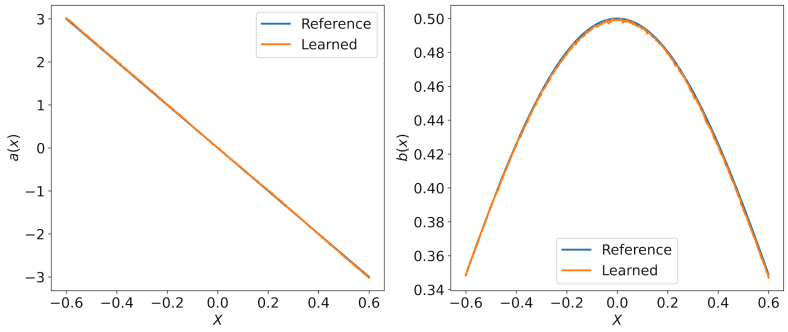

FIG. 8. *SDE with nonlinear diffusion. Comparison of effective drift and diffusion functions obtained by the simulated trajectories using the generative model and the exact SDE. Left: drift* $a(x) = -\mu x$. *Right: diffusion* $b(x) = \sigma e^{-x^2}$.

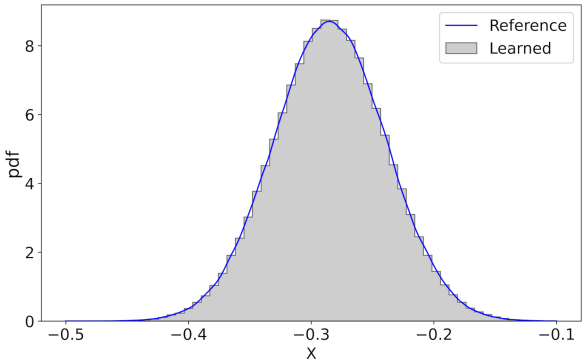

FIG. 9. *SDE with nonlinear diffusion. Comparison of conditional PDF* $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = -0.3)$ *determined by the generative model* G_θ *and the exact flow map* $F_{\Delta t}$.

4.2.2. Trigonometric SDE. The SDE with trigonometric drift and diffusion is defined by

$$dX_t = \sin(2k\pi X_t)dt + \sigma\cos(2k\pi X_t)dW_t, \tag{4.4}$$

where $k = 1$ and $\sigma = 0.5$. The observation dataset $\mathcal{D}_{\text{obs}}$ consists of $M = 20,000K$ data pairs from $H = 200,000$ trajectories, obtained by solving the SDE in (4.4) with initial values uniformly sampled from $\mathcal{U}(0.35, 0.7)$ up to $T = 1.0$. We randomly choose $60,000$ initial states from $\mathcal{D}_{\text{obs}}$ to generate the corresponding labeled training data. After training the generative model G_θ, we simulate 500,000 prediction trajectories up to $T = 10.0$.

The mean and standard deviation of the predicted solutions for $X_0 = 0.6$ by the generative model, compared with those of the exact solutions, are displayed in Figure 10. Good agreements between the mean and standard deviation of the predicted solutions and those of the exact solution can be observed for time up to $T = 10.0$ despite the training dataset being limited to $T = 1.0$. In Figure 11, we present the effective drift and diffusion determined by the generative model, which are closely aligned with the true drift and diffusion. The relative errors for drift and diffusion functions are on the order of 10^{-2} and 10^{-3}, respectively. Figure 12 illustrates the one-step conditional distribution $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = 0.5)$ for both the generative model

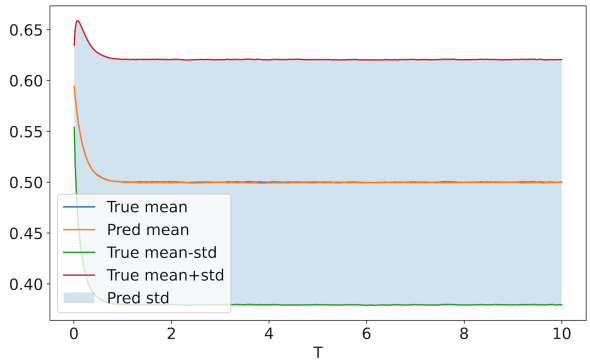

FIG. 10. *Trigonometric SDE. Comparison of the mean and standard deviation of solutions with the initial state being* $X_0 = 0.6$*, obtained by the generative model and the ground truth.*

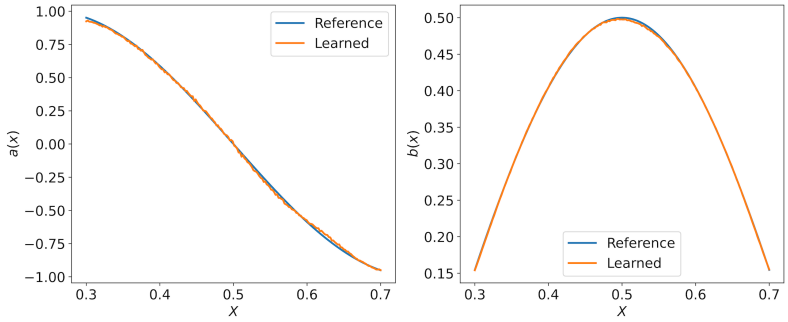

FIG. 11. *Trigonometric SDE. Comparison of effective drift and diffusion functions obtained by the simulated trajectories using the generative model and the exact SDE. Left: drift* $a(x) = \sin(2k\pi x)$*. Right: diffusion* $b(x) = \sigma\cos(2k\pi x)$*.*

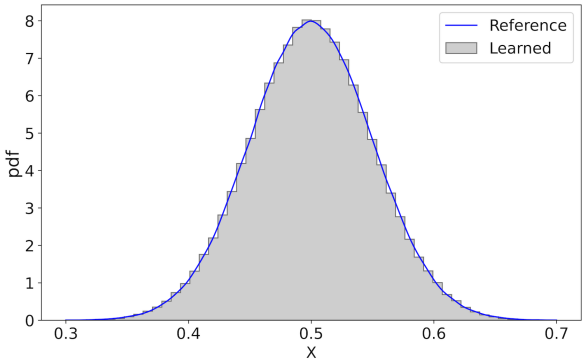

FIG. 12. *Trigonometric SDE. Comparison of conditional PDF* $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = 0.5)$ *determined by the generative model* G_θ *and the exact flow map* $F_{\Delta t}$*.*

and the exact SDE. The predicted conditional distribution accurately approximates the exact one.

4.2.3. SDE with double-well potential. The SDE with a double-well potential is defined by

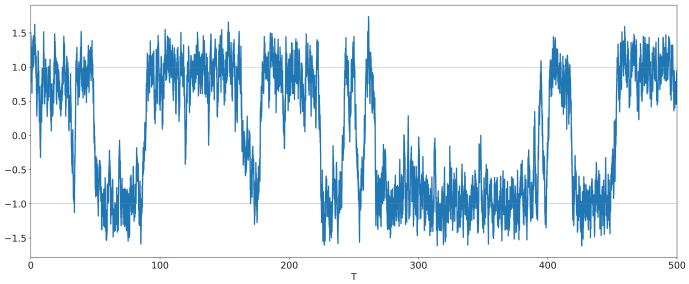

FIG. 13. *SDE with a double-well potential. Solution trajectory for time up to* $T = 500$ *when* $X_0 = 1.5$.

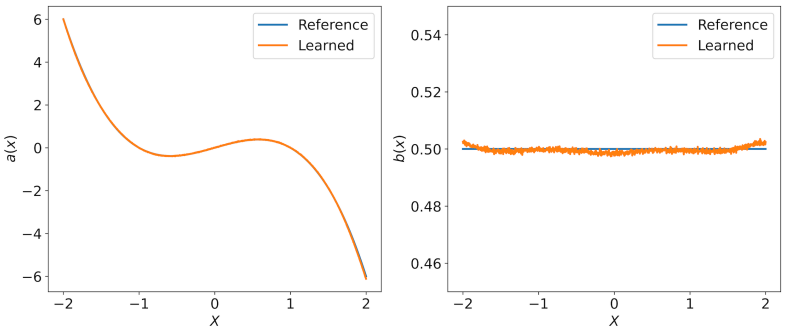

FIG. 14. *SDE with a double-well potential. Comparison of effective drift and diffusion functions obtained by the simulated trajectories using the generative model and the exact SDE. Left: drift* $a(x) = x - x^3$. *Right: diffusion* $b(x) = \sigma$.

$$dX_t = (X_t - X_t^3)dt + \sigma dW_t, \tag{4.5}$$

where $\sigma = 0.5$. There are two stable states at $x = \pm 1$, and the system can randomly transition between these two stable states over time. The observation dataset $\mathcal{D}_{\text{obs}}$ consists of $M = 10,000K$ data pairs from $H = 100,000$ trajectories, obtained by solving the SDE in (4.5) with initial values uniformly sampled from $\mathcal{U}(-2.5, 2.5)$ up to $T = 1.0$. We randomly choose 60,000 initial states from $\mathcal{D}_{\text{obs}}$ to generate the corresponding labeled training data. After training the generative model G_θ, we simulate 500,000 prediction trajectories up to $T = 500$.

A predicted solution at $X_0 = 1.5$ for time up to $T = 500$ by the generative model is displayed in Figure 13. Despite the training dataset being limited to $T = 1.0$, where transitions between the two stable states are not observed, the generative model is able to predict the transitions over longer time periods. In Figure 14, we present the effective drift and diffusion determined by the generative model, which are closely aligned with the true drift and diffusion. The relative errors for drift and diffusion functions are on the order of 10^{-2} and 10^{-3}, respectively. Figure 15 illustrates the one-step conditional distribution $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = 1.5)$ for both the generative model and the exact SDE. The predicted conditional distribution accurately approximates the exact one.

4.3. SDEs with non-Gaussian noise. This section considers the learning of SDEs driven by a non-Gaussian stochastic process.

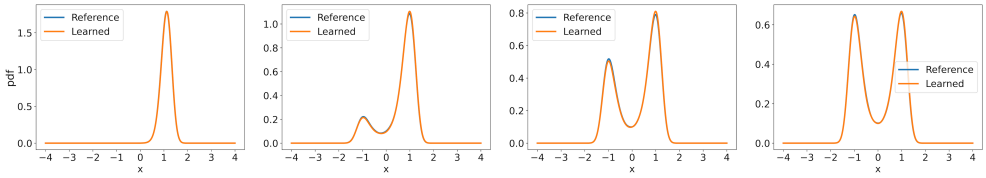

FIG. 15. *SDE with a double-well potential. Temporal evolution of the solution probability distribution when* $X_0 = 1.5$ *at time* $T = 0.5$, 10, 30, *and* 100 *(from left to right).*

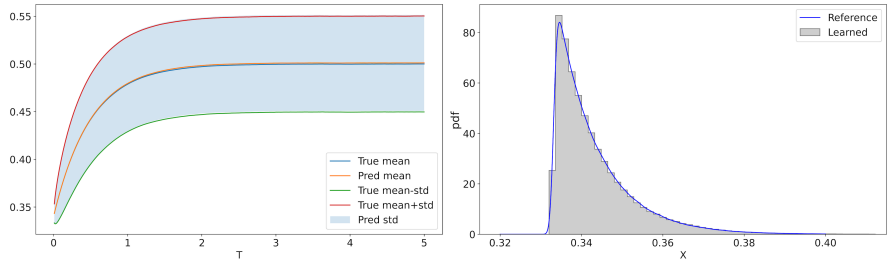

FIG. 16. *SDE with exponentially distributed noise. Left: comparison of mean and standard deviation of solutions with* $x_0 = 0.34$. *Right: comparison of conditional distribution* $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = 0.34)$.

4.3.1. Noise with exponential distribution. The SDE with exponentially distributed noise is defined by

$$dX_t = \mu X_t dt + \sigma\sqrt{dt}\eta_t, \qquad \eta_t \sim \exp(1), \tag{4.6}$$

where η_t has an exponential PDF $f_\eta(x) = e^{-x}$, $x \geq 0$, $\mu = -2.0$, and $\sigma = 0.1$. The observation dataset $\mathcal{D}_{\text{obs}}$ consists of $M = 15,000K$ data pairs from $H = 150,000$ trajectories, obtained by solving the SDE in (4.6) with initial values uniformly sampled from $\mathcal{U}(0.2, 0.9)$ up to $T = 1.0$. We randomly choose 60,000 initial states from $\mathcal{D}_{\text{obs}}$ to generate the corresponding labeled training data. After training the generative model G_θ, we simulate 500,000 prediction trajectories up to $T = 5.0$.

The mean and standard deviation of the predicted solutions for $X_0 = 0.34$ by the generative model, compared with those of the exact solutions, are shown on the left side of Figure 16. The right side of Figure 16 illustrates the one-step conditional distribution $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = 0.34)$ for both the generative model and the exact SDE. The predicted conditional distribution accurately approximates the exact one. Good agreements between the mean and standard deviation of the predicted solutions and those of the exact solution can be observed for time up to $T = 5.0$ despite the training dataset being limited to $T = 1.0$. The exact effective drift and diffusion functions are

$$\begin{aligned} a(x) &= \mathbb{E}\left(\frac{X_{t+\Delta t} - X_t}{\Delta t}\middle| X_t = x\right) = \mu x + \frac{\sigma}{\sqrt{\Delta t}}, \\ b(x) &= \text{Std}\left(\frac{X_{t+\Delta t} - X_t}{\sqrt{\Delta t}}\middle| X_t = x\right) = \sigma. \end{aligned} \tag{4.7}$$

The corresponding effective drift and diffusion by the generative model $G_\theta(x, z)$ can be given by

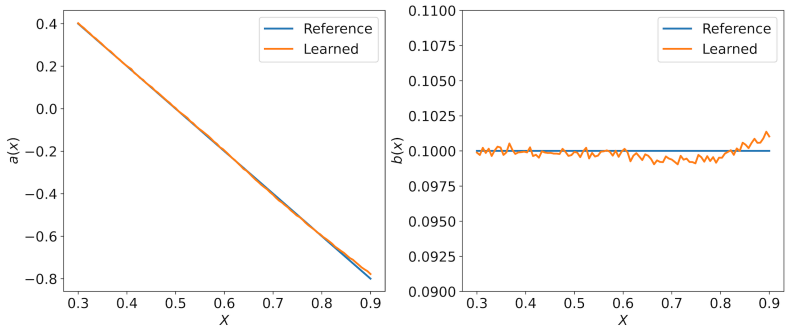

FIG. 17. *SDE with exponentially distributed noise. Comparison of effective drift and diffusion functions obtained by the simulated trajectories using the generative model and the exact SDE. Left: drift* $a(x) = \mu x + \sigma/\sqrt{\Delta t}$. *Right: diffusion* $b(x) = \sigma$.

$$\hat{a}(x) = \mathbb{E}_z\left(\frac{G_\theta(x,z)}{\Delta t}\right), \qquad \hat{b}(x) = \mathrm{Std}_z\left(\frac{G_\theta(x,z)}{\sqrt{\Delta t}}\right). \tag{4.8}$$

In Figure 17, we present the effective drift and diffusion determined by the generative model, which are closely aligned with the true drift and diffusion. The relative errors for drift and diffusion functions are on the order of 10^{-2} and 10^{-3}, respectively.

4.3.2. Noise with lognormal distribution. The SDE with lognormal noise is defined by

$$d\log X_t = (\log m - \theta \log X_t)dt + \sigma dW_t, \tag{4.9}$$

where $m = 1/\sqrt{e}$, $\theta = 1.0$, and $\sigma = 0.3$. The observation dataset $\mathcal{D}_{\text{obs}}$ consists of $M = 20,000K$ data pairs from $H = 200,000$ trajectories, obtained by using the Euler–Maruyama method with the scheme

$$X_{t+\Delta t} = m^{\Delta t} X_t^{1-\theta\Delta t} \eta_t^{\sigma\sqrt{\Delta t}}, \qquad \eta_t \sim \text{lognormal}(0,1), \tag{4.10}$$

starting from initial values that follow $\mathcal{U}(0.1, 2.0)$ up to $T = 1.0$. We randomly choose $60,000$ initial states from $\mathcal{D}_{\text{obs}}$ to generate the corresponding labeled training data. After training the generative model G_θ, we simulate 500,000 prediction trajectories up to $T = 5.0$.

The mean and standard deviation of the predicted solutions for $X_0 = 0.4$ by the generative model, compared with those of the exact solutions, are displayed on the left side of Figure 18. The right side of Figure 18 illustrates the one-step conditional distribution $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = 0.4)$ for both the generative model and the exact SDE. The predicted conditional distribution accurately approximates the exact one. Good agreements between the mean and standard deviation of the predicted solutions and those of the exact solution can be observed for time up to $T = 5.0$ despite the training dataset being limited to $T = 1.0$.

The SDE is rewritten in the form of the classical SDE, and its effective drift and diffusion functions are

$$\begin{aligned} a(x) &= \ln\left[\left(\mathbb{E}\left(\frac{X_{t+\Delta t}}{X_t}\middle| X_t = x\right)\right)^{1/\Delta t}\right] = \ln\left(mx^{-\theta}\right) + \frac{\sigma^2}{2}, \\ b(x) &= \mathrm{Std}\left(X_{t+\Delta t}|X_t = x\right) = \sqrt{e^{\sigma^2\Delta t} - 1}\left(me^{\sigma^2/2}\right)^{\Delta t} x^{1-\theta\Delta t}. \end{aligned} \tag{4.11}$$

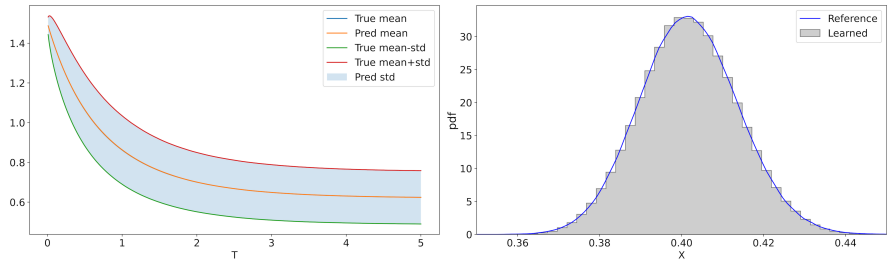

FIG. 18. *SDE with lognormal noise. Left: comparison of mean and standard deviation of solutions with* $x_0 = 0.4$. *Right: comparison of conditional distribution* $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = 0.4)$.

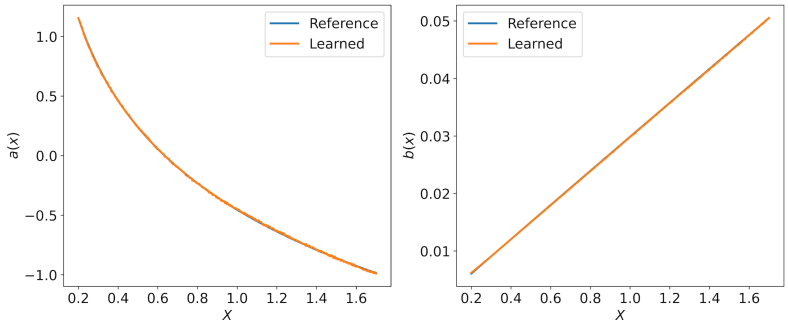

FIG. 19. *SDE with lognormal noise. Comparison of effective drift and diffusion functions obtained by the simulated trajectories using the generative model and the exact SDE. Left: drift* $a(x)$. *Right: diffusion* $b(x)$. *(See* (4.11) *for the exact drift and diffusion functions and* (4.12) *for the learned ones.)*

The corresponding effective drift and diffusion by the generative model $G_\theta(x, z)$ can be given by

$$\hat{a}(x) = \ln\left[\left(\mathbb{E}_z\left(\frac{G_\theta(x,z)+x}{x}\right)\right)^{1/\Delta t}\right], \qquad \hat{b}(x) = \mathrm{Std}_z\left(G_\theta(x,z)\right). \tag{4.12}$$

In Figure 19, we present the effective drift and diffusion determined by the generative model, which are closely aligned with the true drift and diffusion. The relative errors for drift and diffusion functions are on the order of 10^{-3}.

4.4. Multidimensional examples. In this section, we consider the learning of multidimensional SDE systems.

4.4.1. Two-dimensional OU process. The first example is the two-dimensional OU process

$$dX_t = BX_t dt + \Sigma\, dW_t, \tag{4.13}$$

where $X_t = (x_1, x_2) \in \mathbb{R}^2$ are the state variables and B and Σ are the following 2×2 matrices:

$$B = \begin{pmatrix} -1 & -0.5 \\ -1 & -1 \end{pmatrix}, \qquad \Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 0.5 \end{pmatrix}. \tag{4.14}$$

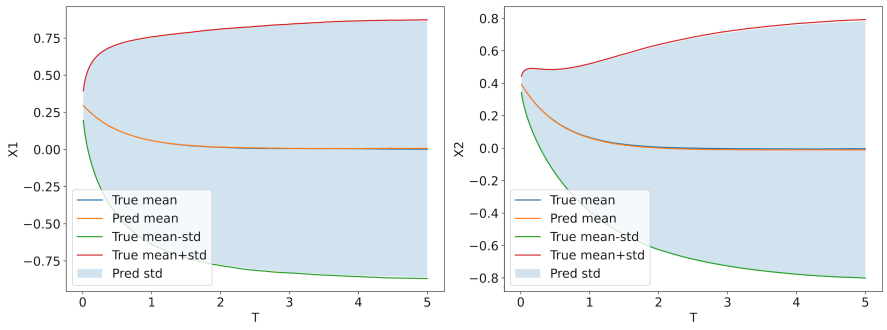

FIG. 20. *Two-dimensional OU. Comparison of mean and standard deviation of solutions with* $X_0 = (0.3, 0.4)$.

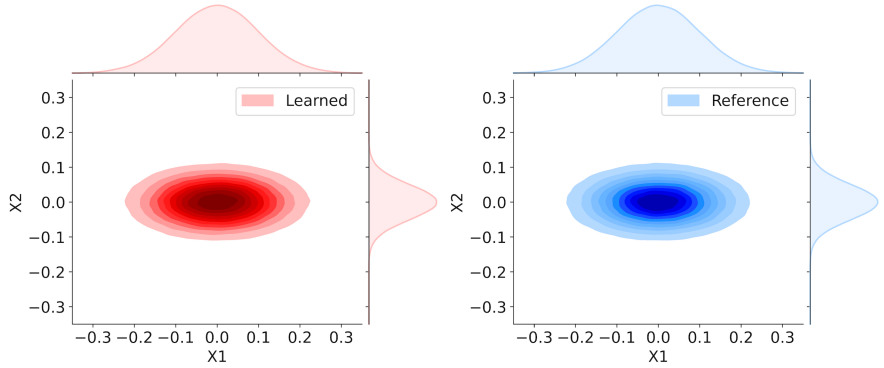

FIG. 21. *Two-dimensional OU. Left: comparison of conditional PDF* $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = (0,0))$ *determined by the generative model* G_θ *and the exact flow map* $F_{\Delta t}$. *Left: learned. Right: reference.*

The observation dataset $\mathcal{D}_{\text{obs}}$ consists of $M = 35,000K$ data pairs from $H = 350,000$ trajectories, obtained by solving the SDE in (4.13) under (4.14) with initial values uniformly sampled from $(-4,4) \times (-3,3)$ up to $T = 1.0$. We randomly choose 120,000 initial states from $\mathcal{D}_{\text{obs}}$ to generate the corresponding labeled training data. After training the generative model G_θ, we simulate 500,000 prediction trajectories up to $T = 5.0$.

The mean and standard deviation of the predicted solutions for $X_0 = (0.3, 0.4)$ by the generative model, compared with those of the exact solutions, are displayed in Figure 20. Figure 21 illustrates the one-step conditional probability distribution $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = (0,0))$ for both the generative model and the exact SDE. The predicted conditional distribution accurately approximates the exact one. Good agreements between the mean and standard deviation of the predicted solutions and those of the exact solution can be observed for time up to $T = 5.0$ despite the training dataset being limited to $T = 1.0$.

4.4.2. Two-dimensional stochastic oscillator. The second example is the stochastic oscillator with

$$B = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \qquad \Sigma = \begin{pmatrix} 0 & 0 \\ 0 & 0.1 \end{pmatrix} \tag{4.15}$$

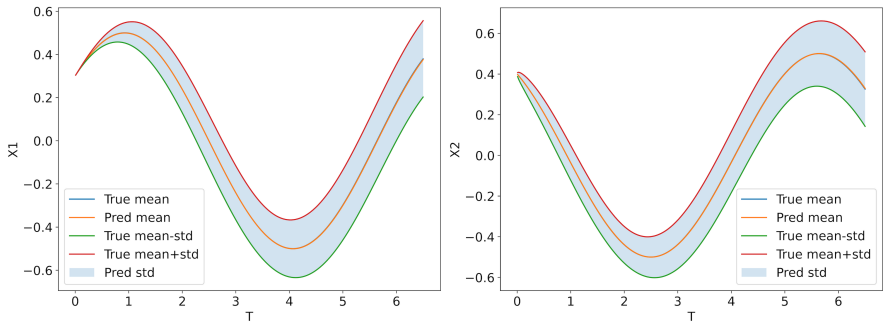

FIG. 22. *Two-dimensional stochastic oscillator. Comparison of mean and standard deviation of solutions at* $X_0 = (0.3, 0.4)$. *Left:* x_1. *Right:* x_2.

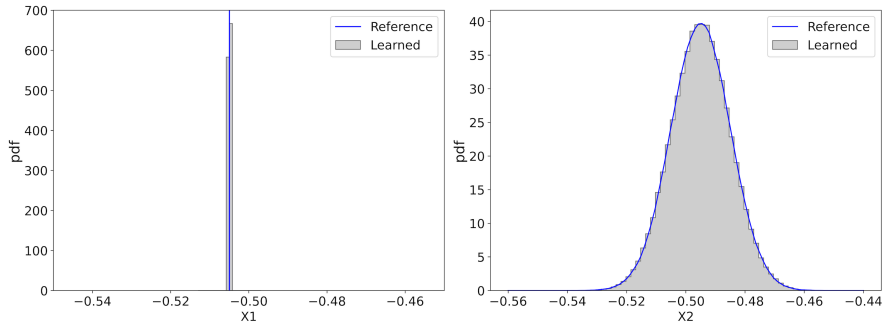

FIG. 23. *Two-dimensional stochastic oscillator. Comparison of marginal probability distribution at* $X_0 = (-0.5, -0.5)$. *Left:* x_1. *Right:* x_2.

in (4.13). The observation dataset $\mathcal{D}_{\text{obs}}$ consists of $M = 300,000K$ data pairs from $H = 3,000,000$ trajectories, obtained by solving the SDE in (4.13) under (4.15) with initial values uniformly sampled from $(-1.5, 1.5) \times (-1.5, 1.5)$ up to $T = 1.0$. We randomly choose 50,000 initial states from $\mathcal{D}_{\text{obs}}$ to generate the corresponding labeled training data. After training the generative model G_θ, we simulate 500,000 prediction trajectories up to $T = 6.5$.

The mean and standard deviation of the predicted solutions for $X_0 = (0.3, 0.4)$ by the generative model, compared with those of the exact solutions, are displayed in Figure 22. Figure 23 illustrates the one-step conditional probability distribution $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = (-0.5, -0.5))$ for both the generative model and the exact SDE. The predicted conditional distribution accurately approximates the exact one. Good agreements between the mean and standard deviation of the predicted solutions and those of the exact solution can be observed for time up to $T = 6.5$ despite the training dataset being limited to $T = 1.0$.

4.4.3. Five-dimensional OU process. This example is the five-dimensional OU process, driven by Wiener processes of varying dimensions,

$$dX_t = BX_t dt + \Sigma\, dW_t, \tag{4.16}$$

where $X_t = (x_1, \ldots, x_5) \in \mathbb{R}^5$ are the state variables and B and Σ are the following 5×5 matrices. The matrix B is defined as follows:

$$B = \begin{pmatrix} 0.2 & 1.0 & 0.2 & 0.4 & 0.2 \\ -1.0 & 0.0 & 0.2 & 0.8 & -1.0 \\ 0.2 & 0.2 & -0.8 & -1.2 & 0.2 \\ -0.6 & 0.0 & 1.2 & -0.2 & 0.6 \\ 0.2 & 0.2 & 0.6 & 0.4 & 0.0 \end{pmatrix}. \tag{4.17}$$

We consider the following five different scenarios for Σ, which has rank varying from 1 to 5. The matrices for Σ are

$$\Sigma_1 = \mathrm{diag}(0,0,1,0,0),$$

$$\Sigma_2 = \mathrm{diag}(0,0.8,0,0,-0.8),\ \Sigma_3 = \begin{pmatrix} 0.8 & 0.2 & 0.0 & 0.0 & 0.0 \\ -0.4 & 0.6 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.7 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{pmatrix},$$

$$\Sigma_4 = \begin{pmatrix} 0.7 & 0.0 & -0.4 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.1 & 0.0 & 0.6 & 0.2 & -0.1 \\ 0.0 & 0.0 & 0.1 & -0.6 & 0.2 \\ 0.0 & 0.0 & 0.0 & 0.3 & 0.8 \end{pmatrix},\ \Sigma_5 = \begin{pmatrix} 0.8 & 0.2 & 0.1 & -0.3 & 0.1 \\ -0.3 & 0.6 & 0.1 & 0.0 & -0.1 \\ 0.2 & -0.1 & 0.9 & 0.1 & 0.2 \\ 0.1 & 0.1 & -0.2 & 0.7 & 0.0 \\ -0.1 & 0.1 & 0.1 & -0.1 & 0.5 \end{pmatrix}.$$

The observation dataset $\mathcal{D}_{\text{obs}}$ consists of $M = 300,000K$ data pairs from $H = 3,000,000$ trajectories, obtained by solving the SDE in (4.16) with initial values uniformly sampled from the hypercube $(-1.0,1.0)^5$ up to $T = 1.0$. We randomly choose 50,000 initial states from $\mathcal{D}_{\text{obs}}$ to generate the corresponding labeled training data. After training the generative model G_θ, we simulate 500,000 prediction trajectories up to $T = 5.0$.

The mean and standard deviation of the predicted solutions for initial value $X_0 = (0.3, -0.2, -0.7, 0.5, 0.6)$ by the generative model, compared with those of the exact solutions, are displayed in Figure 24. Figure 25 illustrates the one-step conditional distribution $p_{X_{t+\Delta t}|X_t}(x_{t+\Delta t}|x_t = X_0)$ for both the generative model and the exact SDE. The predicted conditional distribution accurately approximates the exact one. Good agreements between the mean and standard deviation of the predicted solutions and those of the exact solution can be observed for time up to $T = 6.5$ despite the training dataset being limited to $T = 1.0$.

4.5. Comparison with baseline methods. In this section, we compare the numerical performance of the proposed method with the baseline: the GANs model proposed in [8]. The following metrics are considered for comparison of model accuracy:

- (relative) error of (one-dimensional) effective drift function: $E_a = \|a(x) - \hat{a}(x)\|/\|a(x)\|$;
- (relative) error of (one-dimensional) effective diffusion function: $E_b = \|b(x) - \hat{b}(x)\|/\|b(x)\|$;
- error of the predictive mean at termination time: $E_T^m = \|\mathbb{E}(\mathbf{x}_T) - \mathbb{E}(\hat{\mathbf{x}}_T)\|$;
- error of the predictive Standard deviation at termination time: $E_T^s = \|STD(\mathbf{x}_T) - STD(\hat{\mathbf{x}}_T)\|$.

The comparison results are summarized in Tables 1 and 2. In these tables, we highlight the better results. We observe that for all four metrics, our method outperforms the baseline in most examples.

It is also worth noting that our method is more efficient than GANs in terms of training cost. Table 3 displays the training and evaluation times (in seconds) for

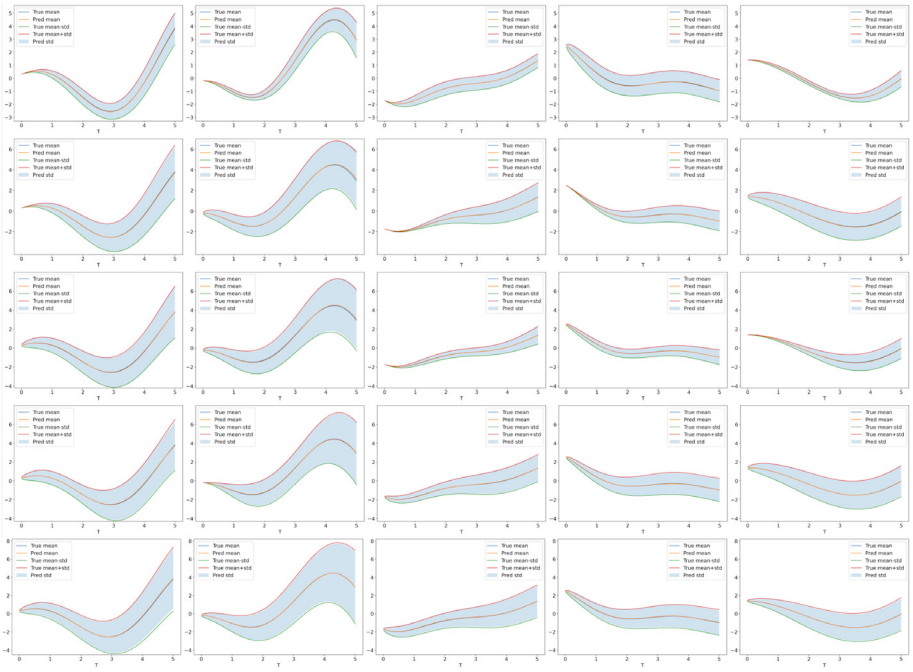

FIG. 24. *Five-dimensional OU process. Comparison of mean and standard deviation of solutions at* $X_0 = (0.3, -0.2, -0.7, 0.5, 0.6)$. *Left to right:* $x_1, x_2, \ldots, x_5$. *Top to bottom:* $\Sigma_1, \Sigma_2, \ldots, \Sigma_5$.

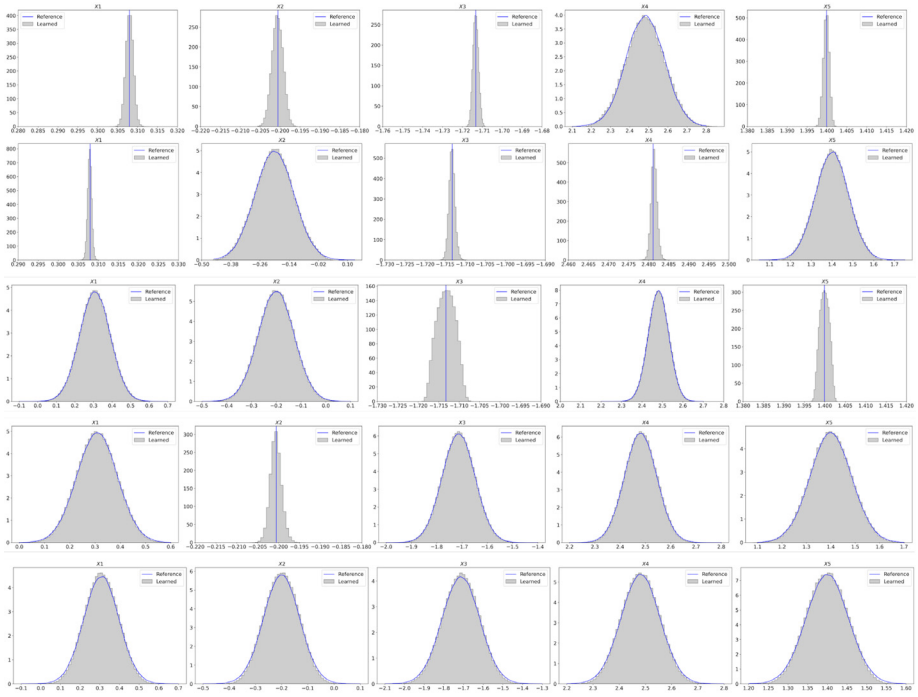

FIG. 25. *Five-dimensional OU process. Comparison of marginal probability distribution at* $X_0 = (0.3, -0.2, -0.7, 0.5, 0.6)$. *Left to right:* $x_1, x_2, \ldots, x_5$. *Top to bottom:* $\Sigma_1, \Sigma_2, \ldots, \Sigma_5$.

TABLE 1
Comparison of effective drift and diffusion functions estimation for GANs and our method.

Example	E_a of GANs	E_a of our method	E_b of GANs	E_b of our method
OU process	3.7615E-02	**2.0264E-02**	3.4912E-03	**2.5218E-03**
Geometric Brownian motion	3.8360E-02	**1.3084E-02**	3.0818E-02	**2.1089E-02**
Exponential diffusion SDE	2.9407E-02	**2.0723E-03**	1.4196E-02	**9.5213E-03**
Trigonometric SDE	3.8318E-02	**2.4540E-02**	3.8146E-02	**3.4354E-03**
Double-well potential	5.6902E-02	**1.2980E-02**	1.8266E-02	**2.0697E-03**
Exponential noise SDE	2.4217E-02	**1.6449E-02**	3.5840E-02	**4.9811E-03**
Lognormal noise SDE	6.7650E-02	**9.5213E-03**	1.9651E-02	**2.0723E-03**

TABLE 2
Comparison of SDE end-time mean and Standard deviation estimation for GANs and our method.

Example	T	E_T^m of GANs	E_T^m of our method	E_T^s of GANs	E_T^s of our method
OU process	4	**1.3538E-03**	3.6823E-03	7.3188E-04	**5.4972E-04**
Geometric Brownian motion	1	1.0389E-01	**3.1599E-02**	2.3274E-01	**8.9173E-02**
Exponential diffusion SDE	10	4.0665E-03	**1.8914E-04**	**4.3357E-04**	7.4537E-04
Trigonometric SDE	10	6.6718E-04	**2.8130E-04**	1.7917E-03	**2.2179E-04**
Double-wellpotential	300	1.2037E-01	**1.7988E-02**	2.5084E-02	**2.2183E-03**
Exponential noise SDE	5	1.8862E-03	**1.1758E-03**	4.4275E-04	**2.6745E-05**
Lognormal noise SDE	5	9.2052E-03	**1.8914E-04**	3.0957E-03	**7.4537E-04**
Two-dimensional OU process	5	3.8093E-02	**8.8535E-03**	2.4988E-02	**5.1506E-03**
Stochastic oscillator	6.5	1.7828E-02	**1.8628E-03**	1.1120E-02	**3.7311E-04**

TABLE 3
Training and evaluation cost (in seconds) for our method of OU process (one, two, and five dimensions).

Models	OU dimensions	Training		Evaluation	
		Data labeling	Training $G(\cdot)$	500K samples	1,000K samples
Ours	One dimension	204.26	5.14	5.40	10.00
	Two dimensions	495.84	5.38	22.25	41.66
	Five dimensions	928.83	8.16	44.21	87.09

our method on the OU process example in one, two, and five dimensions with Σ_5 settings. The training phase includes generating 50,000 labeled data and training a neural network to learn the flow map. In the evaluation phase, we show the total time required to generate 500,000 and 1,000,000 trajectories up to $T = 5$. The GANs modeling for stochastic Flow Map suffers from many training difficulties, which are inherited from vanilla GANs, such as the requirement of a huge number of epochs (estimated $\mathcal{O}(10^5)$), a lack of reliable metrics indicating when to terminate, and so on. In contrast, in our method, the time-consuming step is the generation of labeled data, which still takes only a few minutes for the one- and two-dimensional cases (204.26 and 495.84 seconds, respectively) and about 15 minutes for the five-dimensional case (928.83 seconds), all of which are at most $\mathcal{O}(10^3)$. Once the flow map is trained, generating 500,000 trajectories up to $T = 5$ takes less than 1 minute in all cases (e.g., 5.40 seconds for one dimension, 22.25 seconds for two dimensions, and 44.21 seconds for five dimensions).

5. Conclusion. In this paper, we have introduced a novel training-free conditional diffusion model for learning the flow map of stochastic dynamical systems. Our approach offers a promising alternative to traditional methods for solving SDEs, addressing key challenges in computational efficiency and accuracy. The proposed

method leverages the power of score-based diffusion models to generate samples from complex distributions without the need for extensive training data or computationally intensive solving procedures. By introducing a conditional framework, we have enabled our model to capture the dynamics of SDEs across a wide range of initial conditions and system parameters. Our numerical experiments, spanning various types of SDEs including linear, nonlinear, and multidimensional systems, demonstrate the effectiveness and versatility of our approach. The model shows remarkable accuracy in predicting both short-term and long-term behaviors of stochastic systems, often outperforming baseline methods such as GANs in terms of drift and diffusion function estimation as well as in predicting mean and standard deviation at termination times.

Despite the effectiveness and efficiency of our training-free conditional diffusion model as demonstrated across various examples, several limitations exist, particularly in the Monte Carlo estimator, which directly affect the quality of labeled data generated for training the flow map. First, for extremely high-dimensional systems, the cost of the Monte Carlo estimator under the analytic form increases dramatically, requiring substantial computational resources, potentially limiting the method's application to very complex systems. Second, for SDEs with highly complicated or discontinuous drift and diffusion terms, the score function estimation becomes less accurate at capturing sharp transitions or irregular behaviors. The Monte Carlo estimator may fail to provide sufficient resolution near discontinuities, leading to suboptimal approximations of the flow map. Third, the score function estimation depends significantly on the availability and quality of trajectory data; when available data are limited or sparse in certain regions of the state space, the estimation may struggle to provide accurate approximations. The estimation of score function directly affects the quality of labeled data generated for training the flow map. Obtaining an accurate estimation of the score function is therefore critical to the success of our method.

While our proposed method has shown promising results, there are several avenues for future research that could further enhance its capabilities and applications. First, extending the model to handle more complex stochastic processes, such as jump diffusions or fractional Brownian motion, could broaden its applicability in fields such as finance and environmental science. Second, investigating the theoretical properties of the proposed score estimation technique, including convergence rates and error bounds, would provide a stronger mathematical foundation for the method. Third, exploring the integration of this approach with other machine learning techniques, such as transfer learning or meta-learning, could potentially improve its performance on new, unseen stochastic systems. Fourth, applying this method to real-world problems in areas such as climate modeling, epidemiology, or quantum systems could demonstrate its practical value and potentially lead to new insights in these fields. Finally, optimizing the computational efficiency of the algorithm, particularly for high-dimensional systems, remains an important area for future work. This could involve developing advanced parallelization strategies or leveraging specialized hardware accelerators to further reduce computation time.

Reproducibility of computational results. This paper has been awarded the "SIAM Reproducibility Badge: Code and data available" as a recognition that the authors have followed reproducibility principles valued by SISC and the scientific computing community. Code and data that allow readers to reproduce the results in this paper are available at https://github.com/YanfangLiu11/Conditional-Diffusion-Model-for-SDE-Learning and in the supplementary materials (Conditional-Diffusion-Model-for-SDE-Learning-main.zip [local/web 235KB]).

REFERENCES

[1] C. Archambeau, D. Cornford, M. Opper, and J. Shawe-Taylor, *Gaussian process approximations of stochastic differential equations*, in Gaussian Processes in Practice, Proceedings of Machine Learning Research 1, N. D. Lawrence, A. Schwaighofer, and J. Quiñonero Candela, eds., 2007, pp. 1–16, https://proceedings.mlr.press/v1/archambeau07a.html.

[2] F. Bao, H. G. Chipilski, S. Liang, G. Zhang, and J. S. Whitaker, *Nonlinear ensemble filtering with diffusion models: Application to the surface quasi-geostrophic dynamics*, Monthly Weather Rev., 153 (2025), pp. 1155–1169, https://doi.org/10.1175/MWR-D-24-0069.1.

[3] F. Bao, Z. Zhang, and G. Zhang, *An ensemble score filter for tracking high-dimensional nonlinear dynamical system*, Comput. Methods Appl. Mech. Engrg., 432 (2024), 117447, https://doi.org/10.1016/j.cma.2024.117447.

[4] F. Bao, Z. Zhang, and G. Zhang, *A score-based filter for nonlinear data assimilation*, J. Comput. Phys., 514 (2024), 113207, https://doi.org/10.1016/j.jcp.2024.113207.

[5] T. Chen, J. Sohl-Dickstein, and D. Schuurmans, *Analog bits: Generating discrete data using diffusion models with self-conditioning*, in The Eleventh International Conference on Learning Representations, 2023, https://openreview.net/forum?id=3itjR9QxFw.

[6] X. Chen, J. Duan, J. Hu, and D. Li, *Data-driven method to learn the most probable transition pathway and stochastic differential equation*, Phys. D, 443 (2023), 133559, https://doi.org/10.1016/j.physd.2022.133559.

[7] 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 J. Sci. Comput., 43 (2021), pp. B811–B830, https://doi.org/10.1137/20M1360153.

[8] Y. Chen and D. Xiu, *Learning stochastic dynamical system via flow map operator*, J. Comput. Phys., 508 (2024), 112984, https://doi.org/10.1016/j.jcp.2024.112984.

[9] N. Cheng, O. A. Malik, S. De, S. Becker, and A. Doostan, *Bi-fidelity variational auto-encoder for uncertainty quantification*, Comput. Methods Appl. Mech. Engrg., 421 (2024), 116793, https://doi.org/10.1016/j.cma.2024.116793.

[10] M. Darcy, B. Hamzi, G. Livieri, H. Owhadi, and P. Tavallali, *One-shot learning of stochastic differential equations with data adapted kernels*, Phys. D, 444 (2023), 133583, https://doi.org/10.1016/j.physd.2022.133583.

[11] S. N. Ethier and T. G. Kurtz, *Markov Processes: Characterization and Convergence*, Wiley Ser. Probab. Stat. 282, John Wiley, New York, 1986.

[12] M. Fan, Z. Zhang, D. Lu, and G. Zhang, *GenAI4UQ: A software for forward and inverse uncertainty quantification using conditional generative AI,* SoftwareX, 31 (2025), 102232, https://doi.org/10.1016/j.softx.2025.102232.

[13] S. Gong, Y. Shen, L. Huang, Y. Mao, and X. Zhu, *DiffuSeq: Sequence to sequence text generation with diffusion models*, in The Eleventh International Conference on Learning Representations, 2023, ICLR, Kigali, Rwanda, https://openreview.net/pdf?id=jQj-_rLVXsj.

[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial nets*, in Advances in Neural Information Processing Systems, Vol. 27, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, eds., Curran Associates, Red Hook, NY, 2014, pp. 2672–2680, https://proceedings.neurips.cc/paper_files/paper/2014/file/f033ed80deb0234979a61f95710dbe25-Paper.pdf.

[15] J. Ho, A. Jain, and P. Abbeel, *Denoising diffusion probabilistic models*, in Advances in Neural Information Processing Systems, Vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, eds., Curran Associates, 2020, pp. 6840–6851, https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

[16] A. Hyvärinen, *Estimation of non-normalized statistical models by score matching*, J. Mach. Learn. Res., 6 (2005), pp. 695–709, http://jmlr.org/papers/v6/hyvarinen05a.html.

[17] I. Kobyzev, S. J. Prince, and M. A. Brubaker, *Normalizing flows: An introduction and review of current methods*, IEEE Trans. Pattern Anal. Mach. Intell., 43 (2021), pp. 3964–3979, https://doi.org/10.1109/TPAMI.2020.2992934.

[18] F. Li and Y. Marzouk, *Diffusion map particle systems for generative modeling*, Found. Data Sci., 7 (2025), pp. 814–837, https://doi.org/10.3934/fods.2024054.

[19] X. L. Li, J. Thickstun, I. Gulrajani, P. Liang, and T. B. Hashimoto, *Diffusion-LM improves controllable text generation*, in Advances in Neural Information Processing Systems, Vol. 35, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, eds., Curran Associates, Inc., 2022, pp. 4328–4343, https://proceedings.neurips.cc/paper_files/paper/2022/file/1be5bc25d50895ee656b8c2d9eb89d6a-Paper-Conference.pdf.

[20] Y. Li and J. Duan, *A data-driven approach for discovering stochastic dynamical systems with non-Gaussian Lévy noise*, Phys. D, 417 (2021), 132830, https://doi.org/10.1016/j.physd.2020.132830.

[21] Y. Liu, M. Yang, Z. Zhang, F. Bao, Y. Cao, and G. Zhang, *Diffusion-model-assisted supervised learning of generative models for density estimation*, J. Mach. Learn. Model. Comput., 5 (2024), pp. 25–38, https://doi.org/10.1615/JMachLearnModelComput.2024051346.

[22] D. Lu, Y. Liu, Z. Zhang, F. Bao, and G. Zhang, *A diffusion-based uncertainty quantification method to advance E3SM land model calibration*, J. Geophys. Res. Mach. Learn. Comput., 1 (2024), e2024JH000234, https://doi.org/10.1029/2024JH000234.

[23] N. Mimikos-Stamatopoulos, B. Zhang, and M. Katsoulakis, *Score-based generative models are provably robust: An uncertainty quantification perspective*, in Advances in Neural Information Processing Systems, Vol. 37, Curran Associates, Inc., 2024, pp. 63154–63183, https://proceedings.neurips.cc/paper_files/paper/2024/file/7371b5f5fab9fd401c4ebd352f29dc48-Paper-Conference.pdf.

[24] A. Q. Nichol and P. Dhariwal, *Improved denoising diffusion probabilistic models*, in International Conference on Machine Learning, 2021, pp. 8162–8171.

[25] B. Øksendal, *Stochastic Differential Equations: An Introduction with Applications*, 6th ed., Springer-Verlag, Berlin, 2003.

[26] M. Opper, *Variational inference for stochastic differential equations*, Ann. Phys., 531 (2019), 1800233, https://doi.org/10.1002/andp.201800233.

[27] T. Qin, K. Wu, and D. Xiu, *Data driven governing equations approximation using deep neural networks*, J. Comput. Phys., 395 (2019), pp. 620–635, https://doi.org/10.1016/j.jcp.2019.06.042.

[28] A. H. M. Rafid, J. Yin, Y. Geng, S. Liang, F. Bao, L. Ju, and G. Zhang, *A scalable training-free diffusion model for uncertainty quantification*, in Proceedings of the SC '24 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis, 2024, pp. 380–386, https://doi.org/10.1109/SCW63240.2024.00057.

[29] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, *High-resolution image synthesis with latent diffusion models*, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 10684–10695.

[30] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, *Improved techniques for training GANs*, in Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett, eds., 2016, pp. 2234–2242, https://proceedings.neurips.cc/paper/2016/hash/8a3363abe792db2d8761d6403605aeb7-Abstract.html.

[31] Y. Song and S. Ermon, *Generative modeling by estimating gradients of the data distribution*, in Advances in Neural Information Processing Systems, Vol. 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds., Curran Associates, Inc. 2019, pp. 11918–11930, https://proceedings.neurips.cc/paper_files/paper/2019/file/3001ef257407d5a371a96dcd947c7d93-Paper.pdf.

[32] Y. Song, S. Garg, J. Shi, and S. Ermon, *Sliced score matching: A scalable approach to density and score estimation*, in Proceedings of the 35th Uncertainty in Artificial Intelligence Conference, Proceedings of Machine Learning Research 115, R. P. Adams and V. Gogate, eds., 2020, pp. 574–584, https://proceedings.mlr.press/v115/song20a.html.

[33] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, *Score-based generative modeling through stochastic differential equations*, in 9th International Conference on Learning Representations, ICLR, Virtual Event, Austria, 2021, https://openreview.net/forum?id=PxTIG12RRHS.

[34] D. W. Stroock and S. R. Varadhan, *Multidimensional Diffusion Processes*, Grundlehren Math. Wiss. 233, Springer Science & Business Media, New York, 1979.

[35] P. Vincent, *A connection between score matching and denoising autoencoders*, Neural Comput., 23 (2011), pp. 1661–1674, https://doi.org/10.1162/NECO_a_00142.

[36] M. Wanner and I. Mezić, *On higher order drift and diffusion estimates for stochastic SINDy*, SIAM J. Appl. Dyn. Syst., 23 (2024), pp. 1504–1539, https://doi.org/10.1137/23M1567011.

[37] S. W. R. Werner and B. Peherstorfer, *On the sample complexity of stabilizing linear dynamical systems from data*, Found. Comput. Math., 24 (2024), pp. 955–987, https://doi.org/10.1007/s10208-023-09605-y.

[38] Z. Xu, Y. Chen, Q. Chen, and D. Xiu, *Modeling unknown stochastic dynamical system via autoencoder*, J. Mach. Learn. Model. Comput., 5 (2024), pp. 87–112.

[39] 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 J. Sci. Comput., 44 (2022), pp. B80–B99, https://doi.org/10.1137/21M1413018.

[40] L. Yang, Z. Zhang, Y. Song, S. Hong, R. Xu, Y. Zhao, Y. Shao, W. Zhang, B. Cui, and M.-H. Yang, *Diffusion models: A comprehensive survey of methods and applications*, ACM Comput. Surv., 56 (2023), pp. 1–39, https://doi.org/10.1145/3626235.

[41] M. Yang, P. Wang, D. del-Castillo-Negrete, Y. Cao, and G. Zhang, *A pseudoreversible normalizing flow for stochastic dynamical systems with various initial distributions*, SIAM J. Sci. Comput., 46 (2024), pp. C508–C533, https://doi.org/10.1137/23M1585635.

[42] P. Yu, S. Xie, X. Ma, B. Jia, B. Pang, R. Gao, Y. Zhu, S. Zhu, and Y. Wu, *Latent diffusion energy-based model for interpretable text modeling*, in Proceedings of the 39th International Conference on Machine Learning, PMLR, 2022, pp. 25702–25720, https://proceedings.mlr.press/v162/yu22h.html.