EFFICIENT TRAINING OF NEURAL STOCHASTIC DIFFERENTIAL EQUATIONS BY MATCHING FINITE DIMENSIONAL DISTRIBUTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural Stochastic Differential Equations (Neural SDEs) have emerged as powerful mesh-free generative models for continuous stochastic processes, with critical applications in fields such as finance, physics, and biology. Previous state-of-the-art methods have relied on adversarial training, such as GANs, or on minimizing distance measures between processes using signature kernels. However, GANs suffer from issues like instability, mode collapse, and the need for specialized training techniques, while signature kernel-based methods require solving linear PDEs and backpropagating gradients through the solver, whose computational complexity scales quadratically with the discretization steps. In this paper, we identify a novel class of strictly proper scoring rules for comparing continuous Markov processes. This theoretical finding naturally leads to a novel approach called Finite Dimensional Matching (FDM) for training Neural SDEs. Our method leverages the Markov property of SDEs to provide a computationally efficient training objective. This scoring rule allows us to bypass the computational overhead associated with signature kernels and reduces the training complexity from $O(D^2)$ to $O(D)$ per epoch, where D represents the number of discretization steps of the process. We demonstrate that FDM achieves superior performance, consistently outperforming existing methods in terms of both computational efficiency and generative quality.

1 INTRODUCTION

Stochastic differential equations (SDEs) are a modeling framework used to describe systems influenced by random forces, with applications spanning finance, physics, biology, and engineering. They incorporate stochastic terms to allow the modeling of complex systems under uncertainties.

A *neural stochastic differential equation* (Neural SDE) (Kidger et al., 2021; Issa et al., 2023; Tzen & Raginsky, 2019; Jia & Benson, 2019; Hodgkinson et al., 2021; Li et al., 2020; Morrill et al., 2020) is an SDE where neural networks parameterize the drift and diffusion terms. This model acts as a mesh-free generative model for time-series data and has shown a significant impact in financial applications (Arribas et al., 2021; Gierjatowicz et al., 2020; Choudhary et al., 2023; Hoglund et al., 2023).

Training Neural SDEs typically involves minimizing a distance measure between the distribution of generated paths and the distribution of observed data paths. State-of-the-art performance has been achieved using signature kernels to define a distance measure on path space (Issa et al., 2023). Although effective, this approach requires solving a linear partial differential equation (PDE) whose computational complexity scales quadratically with the discretization step, which becomes impractical for long time series. An alternative is training these models adversarially as Generative Adversarial Networks (GANs) (Kidger et al., 2021). However, GAN-based training can be fraught with issues such as instability, mode collapse, and the need for specialized techniques.

In this paper, we present a theoretical result that extends scoring rules for comparing distributions in finite-dimensional spaces to those for continuous Markov processes. This extension forms the basis of a novel algorithm, *Finite Dimensional Matching* (FDM), designed for training generative models of stochastic processes. FDM exploits the Markovian nature of SDEs by leveraging the

two-time joint distributions of the process, providing an efficient training objective that bypasses the complexities of signature kernels. Notably, FDM reduces the computational complexity from $O(D^2)$ to $O(D)$ per training step, where D represents the number of discretization steps.

The key contributions of this paper are as follows:

- Our main theorem shows that scoring rules to compare continuous Markov processes can be easily built upon scoring rules on finite-dimensional space.
- Our main theorem suggests an efficient training method, FDM, for neural SDE.
- Our experiments show that FDM outperforms prior methods on multiple benchmarks.

The rest of the paper is organized as follows: Section 2 provides a review of the relevant literature. In Section 3, we present preliminary results that lay the foundation for our main contributions. Section 4 introduces our main theorem, which extends scoring rules for finite dimensions to continuous Markov processes and leads to the development of our novel *Finite Dimensional Matching* (FDM) algorithm. Section 5 details the experimental setup and results, demonstrating the superiority of FDM in terms of both computational efficiency and generative performance across several benchmark datasets. Finally, Section 6 concludes the paper by summarizing the contributions, limitations, and directions for future research.

2 Related Work

We begin by reviewing prior applications of scoring rules in generative modeling followed by an exploration of the literature on training neural SDEs.

2.1 Scoring Rules

Scoring rules offer a method to measure discrepancies between distributions (Gneiting & Raftery, 2007) and are especially appealing for generative modeling and have been employed in training various generative models (Bouchacourt et al., 2016; Gritsenko et al., 2020; Pacchiardi et al., 2024; Pacchiardi & Dutta, 2022; Issa et al., 2023; Bonnier & Oberhauser, 2024). Notably, Pacchiardi et al. (2024) apply scoring rules to discrete Markov chains, although their extension to continuous-time processes has not yet been explored. Issa et al. (2023) and Bonnier & Oberhauser (2024) construct scoring rules for continuous processes by utilizing signature kernels.

2.2 Neural SDEs

Several methods have been proposed for training Neural SDEs as generative models, each differing in how they define the divergence or distance between distributions on pathspace. In Table 1, we compare different methods for training Neural SDEs, highlighting their divergence measures and the corresponding discriminator or training objectives. Our approach, **Finite Dimensional Matching** (FDM), introduces a novel scoring rule specifically designed for continuous Markov processes.

Table 1: Methods for training neural SDEs. **SigKer** stands for signature kernel (Issa et al., 2023), **TruncSig** is for truncated signature (Bonnier & Oberhauser, 2024), **SDE-GAN** is proposed by Kidger et al. (2021), and **Latent SDE** is proposed by Li et al. (2020).

Methods	Divergence or distance	Discriminator or training objective
Latent SDE	KL-divergence	Monte-Carlo simulation of free energy
SDE-GAN	1-Wasserstein distance	Optimizing discriminator nets
SigKer	Signature kernel score	Solving Goursat PDEs
TruncSig	Truncated signature kernel score	Truncated approximation of signature
FDM (Ours)	A novel class of scoring rule dedicated to continuous Markov processes	Standard ERM of the expected scores

One method to train neural SDE is the latent SDE model introduced by Li et al. (2020), which trains a Neural SDE using variational inference principles (Opper, 2019). In this framework, training involves optimizing the free energy (Opper, 2019) that includes the Kullback-Leibler (KL) divergence

between the original SDE and an auxiliary SDE. These two SDEs share the same diffusion term but have different drift terms. The KL divergence between their laws can be computed using Girsanov's change of measure theorem. However, the performance of latent SDEs is generally inferior to SDE-GANs due to lower model capacity(Kidger et al., 2021; Issa et al., 2023).

A prominent method is the SDE-GAN introduced by Kidger et al. (2021), which employs adversarial training to fit a Neural SDE, as in Wasserstein-GANs (Arjovsky et al., 2017). This approach relies on the 1-Wasserstein distance, with the discriminator parameterized by neural controlled differential equations (Kidger et al., 2020; Morrill et al., 2021). However, SDE-GANs are notoriously difficult to train due to their high sensitivity to hyperparameters. Another major challenge is the need for a Lipschitz discriminator, which requires techniques like weight clipping and gradient penalties to enforce this constraint (Kidger, 2022). Adversarial training for time-series generative models has also been explored in the context of discrete data (Ni et al., 2022; Yoon et al., 2019).

Another key contribution to training Neural SDEs is the signature kernel method (Issa et al., 2023; Bonnier & Oberhauser, 2024), which minimizes a distance measure based on signature kernels (Lee & Oberhauser, 2023) of paths. However, evaluation of the signature kernel solving Goursat partial differential equations (PDEs) and backpropagating gradients through the PDE solvers (Salvi et al., 2021; Issa et al., 2023). The computational complexity of solving Goursat PDEs scales quadratically with the number of discretization steps, which can be prohibitive for long time series data. Bonnier & Oberhauser (2024) approximates the signature kernel as inner products of truncated signature transforms, called truncated signature. However, the scoring rule based on truncated signature is not strictly proper and its memory complexity is of $\mathcal{O}(d^N)$ where d is the number of features and N is the truncation size.

3 Preliminaries

In this section, we set up the notations and introduce the following preliminary concepts: neural SDEs, Markov processes, and scoring rules.

3.1 Background and Notations

Let $\{\Omega, \mathcal{F}, \mathbb{P}\}$ be a probability space where $\Omega, \mathcal{F}, \mathbb{P}$ denote the sample space, sigma-algebra, and probability measure, respectively. For a random variable ξ, the function $\mathbb{P}_\xi = \mathbb{P} \circ \xi^{-1}$ is the induced measure on its range space. In particular, for a random process X, $\mathbb{P}_X$ denotes its law. We use the superscript $^\top$ for the transposition of a matrix or vector.

3.2 Neural SDE

Let $B : [0, T] \times \Omega \to \mathbb{R}^{d_{noise}}$ be a Brownian motion on $\mathbb{R}^{d_{noise}}$, where $d_{noise} \in \mathbb{N}$. We define a neural SDE as in Issa et al. (2023) and Kidger et al. (2021):

$$Z_0 = \xi^\theta(a),\ dZ_t = \mu^\theta(t, Z_t)dt + \sigma^\theta(t, Z_t)dW_t,\ X_t^\theta = A^\theta Z_t + b^\theta$$

where a is sampled from a $d_{initial}$-dimensional standard Gaussian distribution,

$$\xi^\theta : \mathbb{R}^{d_{initial}} \to \mathbb{R}^{d_z},\ \mu^\theta : [0, T] \times \mathbb{R}^{d_z} \to \mathbb{R}^{d_z},\ \sigma^\theta : [0, T] \times \mathbb{R}^{d_z} \to \mathbb{R}^{d_z \times d_{noise}}$$

along with $A^\theta \in \mathbb{R}^{d_x \times d_z}$, $b^\theta \in \mathbb{R}^{d_x}$, are functions parameterized by neural networks, and $d_{initial}, d_x, d_z \in \mathbb{N}$. We assume additionally that μ^θ and σ^θ are Lipschitz continuous in both arguments and $\xi^\theta(a)$ has finite second-order momentum. These conditions ensure that the SDE for Z_t has a unique strong solution. Suppose Y_t is the data process, we'd like to train the neural networks θ on data sampled from $\mathbb{P}_Y$ so that $\mathbb{P}_{X^\theta}$ matches $\mathbb{P}_Y$.

3.3 Markov Process

We say a continuous process X_t with filtration $\{\mathcal{F}_t\}$ is Markov if X_u is independent of $\mathcal{F}_t$ for all $u \geq t$ given X_t(Kallenberg, 2021). For an SDE of the form $dX_t = \mu(t, X_t)\, dt + \sigma(t, X_t)\, dB_t$, with the filtration generated by the Brownian motion B_t, X_t is Markov as long as the SDE has a unique strong solution (Theorem 9.1, Mao (2007)).

3.4 Scoring Rules

Given a measurable space $(\Omega_0, \mathcal{F}_0)$ and $\omega_0 \in \Omega_0$, a scoring rule (Gneiting & Raftery, 2007) $s(P, \omega_0)$ maps a probability measure P on Ω_0 and a sample ω_0 to $\mathbb{R}$. The expected score is defined as $S(P, Q) = \mathbb{E}_Q[s(P, \omega_0)] = \int_{\Omega_0} s(P, \omega_0)\, dQ(\omega_0)$, where P is the predictive distribution and Q is the true distribution. The scoring rule s is said to be *proper* if the expected score satisfies $S(P, Q) \leq S(Q, Q)$. It is *strictly proper* if $S(P, Q) = S(Q, Q) \iff P = Q$. For example, let $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be the RBF kernel defined as $k(x, y) = \exp\left(-\gamma \|x - y\|^2\right)$, where $\gamma > 0$ is a parameter that determines the width of the kernel, then $s(P, z) = \frac{1}{2}\mathbb{E}_{Z,Z' \sim P} k(Z, Z') - \mathbb{E}_{Z \sim P} k(Z, z)$ is a strictly proper scoring rule for distribution on $\mathbb{R}^d$ (Gneiting & Raftery, 2007).

Let P^θ be a distribution controlled by a generative model θ, and let Q be the true distribution accessed through data. Given a strictly proper scoring rule s, sufficient model capacity of θ, and suffcient data points from Q, P^θ can be trained by maximizing $S(P^\theta, Q)$ over θ, leading to $P^\theta = Q$. While many scoring rules for finite-dimensional spaces have been proposed, we lack strictly proper scoring rules for random processes that can be evaluated efficiently. In our main claim, we show that a strictly proper scoring rule for a two-time joint distribution, *i.e.*, the distributions $\{(X_{t_1}, X_{t_2}), \forall t_1, t_2 \in [0, T]\}$, for a random process X, can be converted into a strictly proper scoring rule for continuous Markov processes.

4 Finite Dimensional Matching

In this section, we present our main theorem, which converts a scoring rule for a two-time joint distribution into a scoring rule for a Markov process. Specifically, if we have a scoring rule for $\Omega_0 = \mathbb{R}^{2d}$, then Theorem 2 allows us to convert it into a scoring rule for Markov processes $X, Y : [0, T] \to \mathbb{R}^d$, where $d \in \mathbb{N}$ and $T \in \mathbb{R}_{>0}$.

4.1 Scoring Rule for Markov Process

In this section, we present our main theorem which shows that a strictly proper scoring rule for the two-time joint distributions can be converted to a scoring rule for two Markov processes. Let continuous Markov processes X, Y on $[0, T]$ take values in a Polish space $\mathcal{E}$ endowed with its Borel σ-algebra. Let s be any strictly proper scoring rule defined on $\mathcal{E} \times \mathcal{E}$. Let $S(P, Q) = \mathbb{E}_Q[s(P, \omega)] < \infty, \forall$ measures P, Q on $\mathcal{E} \times \mathcal{E}$. We define the scoring rule $\bar{s}$ for continuous Markov processes as following:

Definition 1. $\bar{s}(\mathbb{P}_X, y) = \mathbb{E}_{(t_1,t_2) \sim U([0,T]^2)} s(\mathbb{P}_{(X_{t_1}, X_{t_2})}, (y_{t_1}, y_{t_2}))$, *where* $\mathbb{P}_{(X_{t_1}, X_{t_2})}$ *is the joint marginal distributions at times* t_1, t_2 *of* X*, and* $U([0, T]^2)$ *is the uniform distribution on* $[0, T]^2$.

Let $\bar{S}(\mathbb{P}_X, \mathbb{P}_Y) = \mathbb{E}_{y \sim \mathbb{P}_Y}[\bar{s}(\mathbb{P}_X, y)]$. Now we present our main claim, with its proofs deferred to the appendix.

Theorem 2. *If* s *is a strictly proper scoring rule for distributions on* $\mathcal{E} \times \mathcal{E}$*,* $\bar{s}$ *is a strictly proper scoring rule for* $\mathcal{E}$*-valued continuous Markov processes on* $[0, T]$ *where* $T \in \mathbb{R}_{>0}$*. That is, for any* $\mathcal{E}$*-valued continuous Markov processes* X, Y *with laws* $\mathbb{P}_X, \mathbb{P}_Y$*, respectively,* $\bar{S}(\mathbb{P}_X, \mathbb{P}_Y) \leq \bar{S}(\mathbb{P}_Y, \mathbb{P}_Y)$ *with equality achieved only if* $\mathbb{P}_X = \mathbb{P}_Y$.

In the appendix, we present a more generalized version of Theorem 2 that does not require the timestamps t_1 and t_2 to be sampled from $U([0, T]^2)$ in the definition of $\bar{s}$. Nonetheless, to maintain clarity and simplicity, we focus our discussion on the uniform sampling case in the main paper.

Suppose X^θ is a Markov process parameterized by neural net parameters θ with sufficient capacity. Therefore, maximizing $\bar{S}(\mathbb{P}_{X^\theta}, \mathbb{P}_Y) = \mathbb{E}_{Y \sim \mathbb{P}_Y}[\bar{s}(\mathbb{P}_{X^\theta}, Y)]$, which can be achieved by maximizing the corresponding empirical average, will result in $\mathbb{P}_{X^\theta} = \mathbb{P}_Y$.

We present a concrete example on the application of Theorem 2. Consider continuous Markov processes X, Y on $[0, T]$ taking values in $\mathbb{R}^d$. Let $k : \mathbb{R}^{2d} \times \mathbb{R}^{2d} \to \mathbb{R}$ be the RBF kernel, recall that $s(P, z) = \frac{1}{2}\mathbb{E}_{Z,Z' \sim P} k(Z, Z') - \mathbb{E}_{Z \sim P} k(Z, z)$ is a strictly proper scoring rule for distribution

on $\mathbb{R}^{2d}$ (Gneiting & Raftery, 2007). By Theorem 2,

$$\bar{s}(\mathbb{P}_X, y) = \mathbb{E}_{(t_1,t_2)\sim U([0,T]^2)}\left[\frac{1}{2}\mathbb{E}_{X,X'}k([X_{t_1}, X_{t_2}], [X'_{t_1}, X'_{t_2}]) - \mathbb{E}_X k([X_{t_1}, X_{t_2}], [y_{t_1}, y_{t_2}])\right] \tag{1}$$

is strictly proper, where $[\cdot,\cdot]$ is the concatenation of two vectors. $\bar{S}(\mathbb{P}_{X^\theta}, \mathbb{P}_Y) = \mathbb{E}_{Y\sim\mathbb{P}_Y}[\bar{s}(\mathbb{P}_{X^\theta}, Y)]$ can be estimated through empirical average and optimized efficiently.

4.2 Algorithm

We consider an expected score $\bar{S}(\mathbb{P}_{X^\theta}, \mathbb{P}_Y) = \mathbb{E}_{Y\sim\mathbb{P}_Y}[\bar{s}(\mathbb{P}_{X^\theta}, Y)]$, which can be estimated using an empirical average $\hat{S}$. For example, an unbiased estimator of $\bar{S}$ for $\bar{s}$ defined in (1) can be constructed using batches of generated paths $\mathcal{B}_X = \{x^i\}_{i=1}^B$ and data paths $\mathcal{B}_Y = \{y^i\}_{i=1}^B$. For each i, independently sample two timestamps t_i and t'_i. The empirical estimator is then given by:

$$\hat{S}(\mathcal{B}_X, \mathcal{B}_Y) = \frac{1}{2B(B-1)}\sum_{i\neq j} k\left([x^i_{t_j}, x^i_{t'_j}], [x^j_{t_j}, x^j_{t'_j}]\right) - \frac{1}{B^2}\sum_{i=1}^B\sum_{j=1}^B k\left([x^i_{t_j}, x^i_{t'_j}], [y^j_{t_j}, y^j_{t'_j}]\right).$$

Note that the above estimator $\hat{S}$ only requires each data path to be (potentially irregularly) observed at two distinct timestamps, and we can observe the x^i's at any timestamps since they are generated by the neural SDE model. Alternative empirical objectives are provided in the appendix.

In Algorithm 1, we present the concrete *finite dimensional matching* (FDM) algorithm derived from Theorem 2 to train a neural SDE X^θ.

Algorithm 1: Finite Dimensional Matching (FDM)

Input: Neural SDE X^θ, data paths $\{y^i : i \in [N]\}$, strictly proper scoring rule s , batch size B

repeat
- Generate a batch of simulated paths $\mathcal{B}_X = \{x^i : i \in [B]\}$ using the neural SDE model θ;
- Randomly sample a batch of data paths $\mathcal{B}_Y \subset \{y^i : i \in [N]\}$ with $|\mathcal{B}_Y| = B$;
- Compute the empirical estimate $\hat{S}(\mathcal{B}_X, \mathcal{B}_Y)$ of $\bar{S}(\mathbb{P}_{X^\theta}, \mathbb{P}_Y)$;
- Maximize $\hat{S}$ with respect to θ using an optimizer of the user's choice;

until *stopping criterion is met*;

5 Experiments

[1]We evaluate our method, **FDM**, by comparing it to three existing methods for training neural SDEs: the signature kernel method (**SigKer**, Issa et al. (2023)), the truncated signature method (**TruncSig**, Bonnier & Oberhauser (2024)), and **SDE-GAN** (Kidger et al., 2021). Our experiments are conducted across five real-world datasets: energy prices, bonds, metal prices, U.S. stock indices, and exchange rates, as well as one synthetic dataset, the Rough Bergomi model[2]. The real-world datasets are historical price data for variety of financial instruments. The rough Bergomi model is a widely used stochastic volatility model and has been extensively described in Issa et al. (2023).

Consistent with Issa et al. (2023), we use the Kolmogorov-Smirnov (KS) test to assess the marginal distributions for each dimension. Specifically, we compare a batch of generated paths against an unseen batch from the real data distribution and calculate the KS scores and the chance of rejecting the null hypothesis, which states that the two distributions are identical. This process is repeated for all the test batches and we report the averaged KS scores and the chance of rejecting the null hypothesis across all the batches.

For all experiments, we use fully connected neural networks to parameterize the drift and diffusion terms, with hyperparameters and preprocessings suggested in Issa et al. (2023). We choose s to be

[1]code available at `https://anonymous.4open.science/r/Efficient-Training-of-Neural-SDEs-by-Matching-Finite-Dimensional-Distributions-E12B`

[2]All real-world datasets are obtained from `https://www.dukascopy.com/swiss/english/marketwatch/historical/`

$s(P, z) = \frac{1}{2}\mathbb{E}_{Z,Z'\sim P}k(Z, Z') - \mathbb{E}_{Z\sim P}k(Z, z)$ where k is the rbf kernel with unit kernel bandwidth. In particular, following Issa et al. (2023), we let our method and **TruncSig** train for 10000 steps, while **SDE-GAN** trains for 5000 steps and **SigKer** for 4000 steps, to normalize the training time. Despite the differences in training steps, our method remains the fastest in terms of wall-clock time. All models are trained and evaluated on a single NVIDIA H100 GPU.

For our experiments, we first follow Issa et al. (2023) to train and evaluate the models on three datasets with 64 timestamps using random train-test splits: metal prices, stock indices, and exchange rates, with results reported in Tables 2, 3, and 4, respectively. For the bonds and energy price datasets, we held out the latest 20% of the data as test data and evaluated the models by running the KS test on the generated sequences and unseen future sequences, as reported in Tables 5 and 6. Additionally, for the exchange rates dataset, we trained and tested the models using sequences with 256 and 1024 timestamps, reporting KS test results in Tables 7 and 8, and training time in Table 11. For the synthetic rough Bergomi model, we generated sequences with 64 timestamps across both 16 and 32 dimensions, with results reported in Tables 9 and 10, where we report the average KS scores and the chance of rejecting the null hypothesis across different dimensions. We also compare the computational efficiency of the models in terms of training time for different dimensions of the Rough Bergomi model, with detailed results summarized in Table 12. We highlight the best-performing model across all tables.

We include qualitative studies in Figures 1 and 2, which compare the dynamics of joint distributions of real and generated data points. Due to space constraints, we provide further qualitative studies in the appendix. Our results demonstrate that our method outperforms competitors in an overwhelming majority of cases in terms of KS test results, qualitative results, and computational efficiency.

Table 2: Average KS test scores (**lower is better**) and chance of rejecting the null hypothesis (%) at 5%-significance level on marginals (**lower is better**) of metal prices, trained on paths evenly sampled at 64 timestamps.

Dim	Model	t=6	t=19	t=32	t=44	t=57
SILVER	**SigKer**	.137, 14.8	.137, 14.4	.136, 14.6	.133, 12.7	.131, 11.5
	TruncSig	.231, 97.5	.250, 98.8	.270, 99.5	.287, 99.6	.309, 99.7
	SDE-GAN	.365, 99.7	.536, 100.	.624, 100.	.675, 100.	.711, 100.
	FDM	**.112**, **5.70**	**.111**, **5.80**	**.110**, **5.50**	**.112**, **6.10**	**.111**, **6.20**
GOLD	**SigKer**	.129, 10.8	.132, 11.7	.134, 13.0	.132, 12.3	.128, 10.5
	TruncSig	.196, 77.8	.218, 93.7	.245, 98.7	.269, 99.3	.285, 99.6
	SDE-GAN	.239, 90.7	.380, 99.9	.476, 100.	.555, 100.	.627, 100.
	FDM	**.118**, **8.50**	**.116**, **7.90**	**.114**, **6.60**	**.114**, **6.20**	**.113**, **6.60**

6 Conclusion, Limitations, and Future works

Our main theorem demonstrates that any strictly proper scoring rule for comparing distributions on finite dimensions can be extended to strictly proper scoring rules for comparing the laws of continuous Markov processes. This theorem naturally leads to the **FDM** algorithm for training neural SDEs. We empirically show that **FDM** outperforms current state-of-the-art methods for training neural SDEs, both in terms of generative quality and computational efficiency.

However, the applicability of our main theorem is currently constrained by the assumptions of continuity and the Markov property. Although this lies beyond the scope of Neural SDEs, we provide a straightforward extension of the main theorem to Càdlàg Markov processes in the appendix. This extension broadens the applicability of **FDM** to a wider range of models, including jump processes. Furthermore, an intriguing direction for future work would be to relax the Markov assumptions, for instance, by incorporating hidden Markov models.

Table 3: Average KS test scores and chance of rejecting the null hypothesis (%) at 5%-significance level on marginals of U.S. stock indices, trained on paths evenly sampled at 64 timestamps. "DOLLAR", "USA30", "USA500", "USATECH", and "USSC2000" stand for US Dollar Index, USA 30 Index, USA 500 Index, USA 100 Technical Index, and US Small Cap 2000, respectively.

Dim	Model	t=6	t=19	t=32	t=44	t=57
DOLLAR	**SigKer**	.164, 40.2	.178, 56.4	.183, 62.6	.186, 66.3	.186, 66.5
	TruncSig	.223, 95.1	.252, 99.0	.275, 99.4	.294, 99.7	.312, 99.7
	SDE-GAN	.189, 65.5	.232, 91.4	.263, 97.6	.289, 99.0	.304, 99.2
	FDM	**.121**, **7.50**	**.130**, **11.3**	**.133**, **12.9**	**.135**, **13.8**	**.137**, **15.9**
USA30	**SigKer**	.168, 44.8	.204, 86.2	.220, 94.9	.219, 94.6	.215, 93.1
	TruncSig	**.118**, **6.20**	.129, **8.30**	.146, 20.4	.160, 35.1	.181, 62.6
	SDE-GAN	.233, 87.2	.349, 99.7	.443, 100.	.526, 100.	.595, 100.
	FDM	.142, 21.9	**.128**, 13.1	**.125**, **10.9**	**.124**, **9.80**	**.121**, **7.50**
USA500	**SigKer**	.217, 93.2	.226, 96.2	.225, 95.8	.217, 93.7	.211, 90.5
	TruncSig	.234, 97.8	.257, 99.1	.276, 99.3	.285, 99.4	.292, 99.7
	SDE-GAN	.231, 90.2	.329, 99.5	.409, 99.9	.483, 100.	.550, 100.
	FDM	**.115**, **5.80**	**.116**, **5.50**	**.119**, **5.90**	**.118**, **6.30**	**.119**, **6.50**
USATECH	**SigKer**	.197, 80.4	.217, 93.2	.221, 94.5	.216, 93.0	.211, 89.8
	TruncSig	.197, 79.7	.227, 96.5	.247, 98.9	.264, 99.3	.276, 99.5
	SDE-GAN	.469, 100.	.756, 100.	.894, 100.	.953, 100.	.982, 100.
	FDM	**.121**, **8.60**	**.118**, **6.80**	**.118**, **6.50**	**.118**, **7.20**	**.119**, **7.30**
USSC2000	**SigKer**	.168, 44.9	.202, 83.6	.221, 94.9	.218, 93.8	.214, 92.6
	TruncSig	**.121**, **6.60**	.137, **11.9**	.162, 36.5	.185, 67.2	.211, 90.4
	SDE-GAN	.168, 46.1	.253, 93.1	.378, 99.9	.517, 100.	.644, 100.
	FDM	.140, 20.2	**.128**, 12.7	**.122**, **9.40**	**.124**, **8.90**	**.122**, **6.80**

Table 4: Average KS test scores and the chance of rejecting the null hypothesis (%) at 5%-significance level on marginals for different currency pairs (EUR/USD and USD/JPY), trained on paths evenly sampled at 64 timestamps.

Dim	Model	t=6	t=19	t=32	t=44	t=57
EUR/USD	**SigKer**	**.122**, **8.30**	.139, 19.1	.140, 19.9	.134, 15.9	.130, 14.0
	TruncSig	.221, 94.8	.272, 99.4	.304, 99.6	.322, 99.7	.343, 99.9
	SDE-GAN	.170, 45.9	.200, 78.9	.198, 80.1	.219, 88.3	.292, 99.0
	FDM	.129, 13.7	**.112**, **6.60**	**.110**, **5.80**	**.112**, **6.60**	**.113**, **6.50**
USD/JPY	**SigKer**	**.114**, **4.60**	.124, 8.00	.128, 10.8	.127, 11.3	.126, 12.1
	TruncSig	.206, 86.3	.247, 98.9	.279, 99.3	.301, 99.7	.322, 99.9
	SDE-GAN	.143, 19.7	.165, 40.5	.179, 56.5	.185, 62.2	.192, 69.0
	FDM	.120, 10.4	**.111**, **5.50**	**.109**, **5.00**	**.110**, **5.70**	**.109**, **5.40**

Table 5: Average KS test scores and the chance of rejecting the null hypothesis (%) at 5%-significance level on marginals of energy prices, trained on paths evenly sampled at 64 timestamps. We reserve the latest 20% data as test dataset and measure how well the model predicts into future. "BRENT", "DIESEL", "GAS", and "LIGHT" stand for U.S. Brent Crude Oil, Gas oil, Natural Gas, and U.S. Light Crude Oil, respectively.

Dim	Model	t=6	t=19	t=32	t=44	t=57
BRENT	**SigKer**	.145, 21.8	.145, 22.1	.142, 19.9	.140, 18.3	**.130**, **13.1**
	TruncSig	.210, 88.7	.223, 94.8	.234, 97.1	.254, 98.8	.270, 99.4
	SDE-GAN	.493, 100.	.883, 100.	.984, 100.	.998, 100.	1.00, 100.
	FDM (ours)	**.129**, **14.1**	**.129**, **13.8**	**.136**, **17.6**	**.135**, **17.0**	.134, 16.2
DIESEL	**SigKer**	.141, 17.6	.155, 29.7	.158, 32.5	.158, 32.6	.160, 35.4
	TruncSig	.181, 61.8	.206, 86.7	.223, 94.9	.234, 97.5	.262, 99.0
	SDE-GAN	.169, 44.9	.233, 89.3	.319, 99.4	.422, 100.	.523, 100.
	FDM (ours)	**.124**, **11.3**	**.117**, **7.60**	**.117**, **8.50**	**.122**, **9.90**	**.120**, **9.70**
GAS	**SigKer**	.178, 56.3	.190, 69.9	.185, 61.5	.180, 56.7	.168, 44.2
	TruncSig	.275, 99.6	.313, 99.7	.338, 99.8	.370, 99.9	.383, 99.9
	SDE-GAN	.263, 98.9	.273, 99.2	.281, 99.4	.394, 99.9	.565, 100.
	FDM (ours)	**.116**, **6.40**	**.123**, **10.1**	**.130**, **15.2**	**.134**, **19.3**	**.135**, **20.2**
LIGHT	**SigKer**	.161, 35.9	.156, 31.4	.146, 23.6	**.138**, **17.4**	**.124**, **9.70**
	TruncSig	.247, 98.8	.266, 99.4	.279, 99.4	.298, 99.6	.315, 99.8
	SDE-GAN	.199, 81.4	.218, 89.7	.370, 99.6	.580, 100.	.761, 100.
	FDM (ours)	**.125**, **11.5**	**.127**, **12.9**	**.138**, **18.9**	.140, 20.8	.145, 24.0

Table 6: Average KS test scores and chance of rejecting the null hypothesis (%) at 5%-significance level on marginals of bonds, trained on paths evenly sampled at 64 timestamps. We reserve the most latest 20% data as test dataset and measure how well the model predicts into future. "BUND", "UKGILT", and "USTBOND" stand for Euro Bund, UK Long Gilt, and US T-BOND, respectively.

Dim	Model	t=6	t=19	t=32	t=44	t=57
BUND	**SigKer**	.137, 17.0	.156, 30.6	.169, 45.9	.179, 59.0	.178, 58.1
	TruncSig	.216, 91.7	.252, 99.0	.286, 99.5	.311, 99.9	.331, 99.9
	SDE-GAN	.291, 98.9	.442, 100.	.546, 100.	.618, 100.	.695, 100.
	FDM	**.111**, **6.30**	**.112**, **5.40**	**.123**, **7.70**	**.132**, **12.8**	**.130**, **12.5**
UKGILT	**SigKer**	**.113**, **7.40**	.114, **5.80**	.122, 8.20	.127, 10.9	.128, 11.6
	TruncSig	.164, 40.0	.211, 89.9	.256, 99.0	.286, 99.6	.311, 99.7
	SDE-GAN	.283, 97.6	.671, 100.	.949, 100.	.993, 100.	.997, 100.
	FDM	.129, 13.0	**.112**, 6.30	**.108**, **4.90**	**.109**, **5.20**	**.108**, **5.20**
USTBOND	**SigKer**	.134, 15.3	.131, 10.8	.142, 18.3	.146, 21.7	.150, 25.9
	TruncSig	.253, 99.0	.274, 99.6	.312, 99.8	.337, 99.9	.358, 100.
	SDE-GAN	.462, 100.	.867, 100.	.974, 100.	.999, 100.	1.00, 100.
	FDM	**.128**, **12.3**	**.120**, **8.80**	**.113**, **5.70**	**.113**, **5.70**	**.111**, **5.10**

Table 7: Average KS test scores and the chance of rejecting the null hypothesis (%) at 5%-significance level on marginals for different currency pairs (EUR/USD and USD/JPY), trained on paths evenly sampled at 256 timestamps.

Dim	Model	$t = 25$	$t = 76$	$t = 128$	$t = 179$	$t = 230$
EUR/USD	**SigKer**	.535, 100.	.535, 100.	.536, 100.	.546, 100.	.540, 100.
	TruncSig	.137, **18.9**	.184, 67.1	.252, 99.6	.290, 100.	.318, 100.
	SDE-GAN	**.134**, 21.6	.411, 100.	.569, 100.	.548, 100.	.338, 99.9
	FDM	.136, 23.0	**.112**, **5.70**	**.123**, **12.7**	**.132**, **17.8**	**.141**, **26.6**
USD/JPY	**SigKer**	.535, 100.	.534, 100.	.535, 100.	.538, 100.	.541, 100.
	TruncSig	**.114**, **7.10**	.152, 28.3	.199, 82.8	.232, 97.9	.242, 99.2
	SDE-GAN	.201, 72.6	.334, 99.9	.407, 100.	.405, 100.	.338, 100.
	FDM	.124, 13.8	**.112**, **6.30**	**.118**, **6.90**	**.122**, **9.00**	**.115**, **6.20**

Table 8: Average KS test scores and the chance of rejecting the null hypothesis (%) at 5%-significance level on marginals for different currency pairs (EUR/USD and USD/JPY), trained on paths evenly sampled at 1024 timestamps. A thread limit error is encountered during the training of the **SigKer (Issa et al., 2023)**, which relies on a dedicated parallel PDE solver.

Dim	Model	$t = 102$	$t = 307$	$t = 512$	$t = 716$	$t = 921$
EUR/USD	**SigKer**	-	-	-	-	-
	TruncSig	.476, 100.	.718, 100.	.993, 100.	.996, 100.	.887, 100.
	SDE-GAN	.280, 98.4	.818, 100.	.963, 100.	.846, 100.	.805, 100.
	FDM	**.117**, **11.1**	**.117**, **9.00**	**.138**, **25.1**	**.153**, **36.2**	**.191**, **66.5**
USD/JPY	**SigKer**	-	-	-	-	-
	TruncSig	.766, 100.	.743, 100.	.670, 100.	.998, 100.	1.00, 100.
	SDE-GAN	.528, 100.	.291, 100.	.389, 100.	.530, 100.	.655, 100.
	FDM	**.138**, **20.9**	**.124**, **14.3**	**.150**, **32.1**	**.199**, **74.9**	**.260**, **97.9**

Table 9: Average KS test scores and the chance of rejecting the null hypothesis (%) at 5%-significance level on marginals across all dimensions, trained on paths evenly sampled at 64 timestamps from a 16-dimension rough Bergomi model.

Model	$t = 6$	$t = 19$	$t = 32$	$t = 44$	$t = 57$
SigKer	**.112**, **6.60**	.118, **7.80**	.124, 10.8	.132, 16.3	.144, 25.5
TruncSig	.450, 100.	.458, 100.	.462, 100.	.461, 100.	.460, 100.
SDE-GAN	.308, 99.8	.374, 99.4	.393, 99.5	.406, 99.6	.430, 99.7
FDM	.113, 7.20	**.116**, **7.80**	**.119**, **8.80**	**.124**, **11.8**	**.131**, **15.8**

Table 10: Average KS test scores and the chance of rejecting the null hypothesis (%) at 5%-significance level on marginals across all dimensions, trained on paths evenly sampled at 64 timestamps from a 32-dimension rough Bergomi model. TruncSig runs out of GPU memory.

Model	t=6	t=19	t=32	t=44	t=57
SigKer	.120, 11.1	.137, 18.5	.149, 26.5	.157, 35.3	.168, 45.2
TruncSig	-	-	-	-	-
SDE-GAN	.284, 99.8	.288, 99.7	.298, 99.8	.311, 99.9	.326, 100.
FDM	**.117**, **9.10**	**.119**, **10.2**	**.122**, **11.4**	**.124**, **13.0**	**.128**, **15.4**

Table 11: Training time of different methods on forex data with different lengths in terms of hours. SDE-GAN hits the max wall times of 20 hours while the training progress is nearly 25%.

Method	**64 Timestamps**	**256 Timestamps**	**1024 Timestamps**
Signature Kernel	0.66	7.80	thread limit error
Truncated Signature	0.31	1.34	5.61
SDE-GAN	0.64	4.21	> 80
FDM	**0.27**	**1.21**	**5.43**

Table 12: Training time of Rough Bergomi model with different data dimensions in terms of hours.

Method	**16 Dim**	**32 Dim**
SDE-GAN	1.41	1.58
FDM	**0.40**	**0.54**
Signature Kernel	4.11	6.74
Truncated Signature	6.86	GPU out of RAM

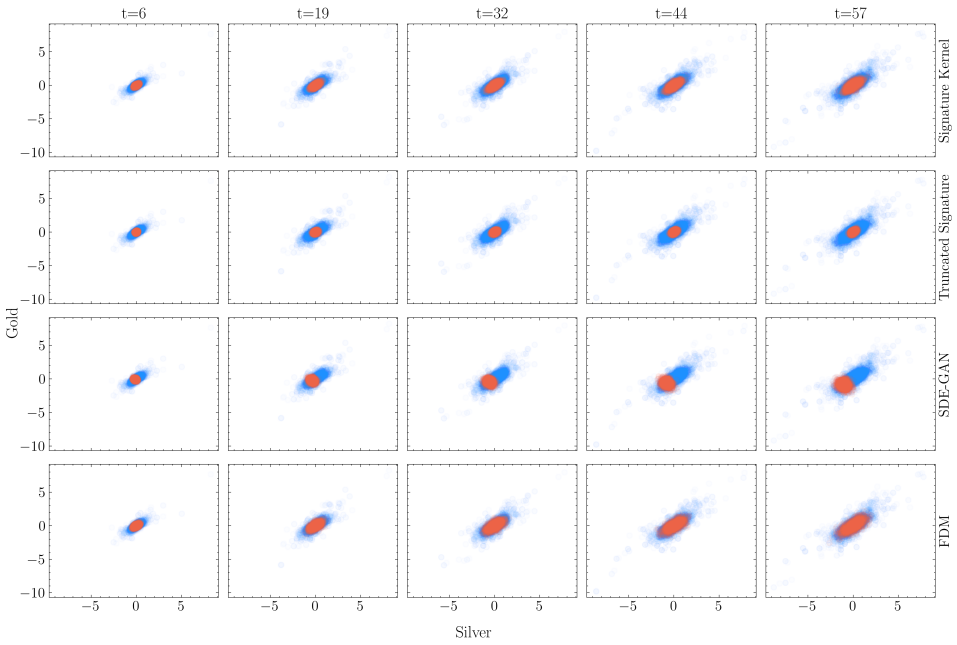

Figure 1: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of gold and silver prices in the metal price data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is the silver price and the vertical axis is the gold price.

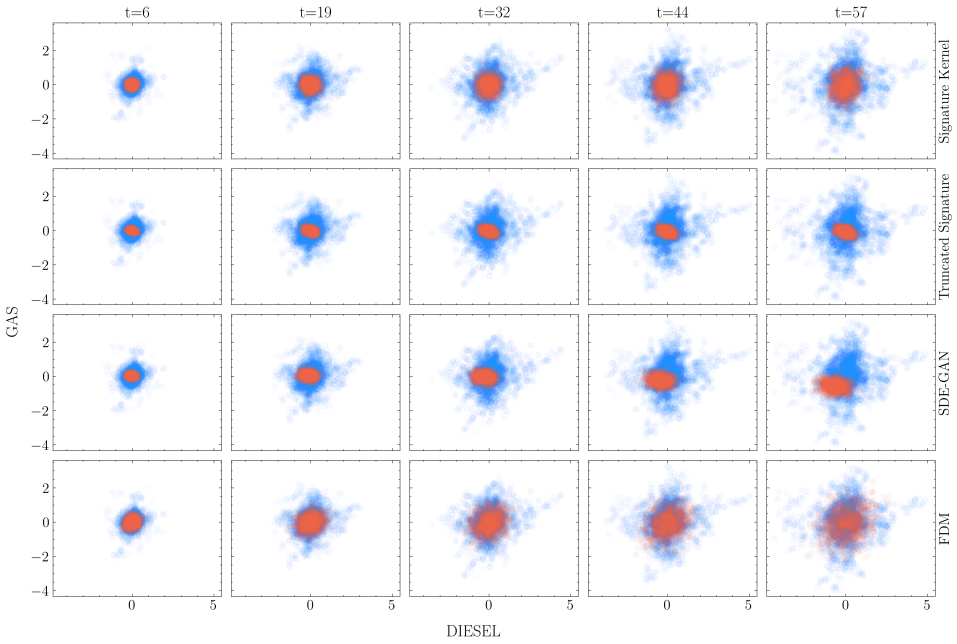

Figure 2: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of gas oil and natural gas prices in the metal price data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is DIESEL (Gas oil price), and the vertical axis is GAS (Natural Gas price).

REFERENCES

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 214–223. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/arjovsky17a.html.

Imanol Perez Arribas, Cristopher Salvi, and Lukasz Szpruch. Sig-sdes model for quantitative finance. In *Proceedings of the First ACM International Conference on AI in Finance*, ICAIF '20, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450375849. doi: 10.1145/3383455.3422553. URL https://doi.org/10.1145/3383455.3422553.

Patric Bonnier and Harald Oberhauser. Proper Scoring Rules, Gradients, Divergences, and Entropies for Paths and Time Series. *Bayesian Analysis*, pp. 1 – 32, 2024. doi: 10.1214/24-BA1435. URL https://doi.org/10.1214/24-BA1435.

Diane Bouchacourt, M. Pawan Kumar, and Sebastian Nowozin. Disco nets: dissimilarity coefficient networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pp. 352–360, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819.

Vedant Choudhary, Sebastian Jaimungal, and Maxime Bergeron. Funvol: A multi-asset implied volatility market simulator using functional principal components and neural sdes, 2023. URL https://arxiv.org/abs/2303.00859.

Patryk Gierjatowicz, Marc Sabate-Vidales, David Šiška, Lukasz Szpruch, and Žan Žurič. Robust pricing and hedging via neural sdes, 2020.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. doi: 10.1198/016214506000001437. URL https://doi.org/10.1198/016214506000001437.

Alexey A. Gritsenko, Tim Salimans, Rianne van den Berg, Jasper Snoek, and Nal Kalchbrenner. A spectral energy distance for parallel speech synthesis. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Liam Hodgkinson, Chris van der Heide, Fred Roosta, and Michael W. Mahoney. Stochastic continuous normalizing flows: training SDEs as ODEs. In Cassio de Campos and Marloes H. Maathuis (eds.), *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pp. 1130–1140. PMLR, 27–30 Jul 2021. URL https://proceedings.mlr.press/v161/hodgkinson21a.html.

Melker Hoglund, Emilio Ferrucci, Camilo Hernandez, Aitor Muguruza Gonzalez, Cristopher Salvi, Leandro Sanchez-Betancourt, and Yufei Zhang. A neural rde approach for continuous-time non-markovian stochastic control problems, 2023. URL https://arxiv.org/abs/2306.14258.

Zacharia Issa, Blanka Horvath, Maud Lemercier, and Cristopher Salvi. Non-adversarial training of neural sdes with signature kernel scores. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 11102–11126. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/2460396f2d0d421885997dd1612ac56b-Paper-Conference.pdf.

Junteng Jia and Austin R Benson. Neural jump stochastic differential equations. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 9847–9858. Curran Associates, Inc., 2019. URL http://papers.nips.cc/paper/9177-neural-jump-stochastic-differential-equations.pdf.

Olav Kallenberg. *Foundations of Modern Probability*. Number 75 in Probability Theory and Stochastic Modelling. Springer Cham, 3 edition, 2021. ISBN 978-3-030-61871-1. doi: 10.1007/978-3-030-61871-1. URL https://doi.org/10.1007/978-3-030-61871-1.

Patrick Kidger. On neural differential equations, 2022. URL https://arxiv.org/abs/2202.02435.

Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural Controlled Differential Equations for Irregular Time Series. *Advances in Neural Information Processing Systems*, 2020.

Patrick Kidger, James Foster, Xuechen Li, and Terry J Lyons. Neural sdes as infinite-dimensional gans. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5453–5463. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/kidger21b.html.

Darrick Lee and Harald Oberhauser. The signature kernel, 2023. URL https://arxiv.org/abs/2305.04625.

Xuechen Li, Ting-Kam Leonard Wong, Ricky T. Q. Chen, and David K. Duvenaud. Scalable gradients and variational inference for stochastic differential equations. In Cheng Zhang, Francisco Ruiz, Thang Bui, Adji Bousso Dieng, and Dawen Liang (eds.), *Proceedings of The 2nd Symposium on Advances in Approximate Bayesian Inference*, volume 118 of *Proceedings of Machine Learning Research*, pp. 1–28. PMLR, 08 Dec 2020. URL https://proceedings.mlr.press/v118/li20a.html.

Xuerong Mao. *Stochastic Differential Equations and Applications*. Horwood Publishing, Chichester, UK, 2nd edition, 2007. ISBN 978-1-904275-29-5.

James Morrill, Cristopher Salvi, Patrick Kidger, James Foster, and Terry Lyons. Neural rough differential equations for long time series. In *International Conference on Machine Learning*, 2020. URL https://api.semanticscholar.org/CorpusID:234358733.

James Morrill, Cristopher Salvi, Patrick Kidger, James Foster, and Terry Lyons. Neural rough differential equations for long time series. *International Conference on Machine Learning*, 2021.

Hao Ni, Lukasz Szpruch, Marc Sabate-Vidales, Baoren Xiao, Magnus Wiese, and Shujian Liao. Sig-wasserstein gans for time series generation. In *Proceedings of the Second ACM International Conference on AI in Finance*, ICAIF '21, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391481. doi: 10.1145/3490354.3494393. URL https://doi.org/10.1145/3490354.3494393.

Manfred Opper. Variational inference for stochastic differential equations. *Annalen der Physik*, 531(3):1800233, 2019. doi: https://doi.org/10.1002/andp.201800233. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/andp.201800233.

Lorenzo Pacchiardi and Ritabrata Dutta. Likelihood-free inference with generative neural networks via scoring rule minimization, 2022. URL https://arxiv.org/abs/2205.15784.

Lorenzo Pacchiardi, Rilwan A. Adewoyin, Peter Dueben, and Ritabrata Dutta. Probabilistic forecasting with generative networks via scoring rule minimization. *Journal of Machine Learning Research*, 25(45):1–64, 2024. URL http://jmlr.org/papers/v25/23-0038.html.

Cristopher Salvi, Thomas Cass, James Foster, Terry Lyons, and Weixin Yang. The signature kernel is the solution of a goursat pde. *SIAM Journal on Mathematics of Data Science*, 3(3):873–899, 2021. doi: 10.1137/20M1366794. URL https://doi.org/10.1137/20M1366794.

Belinda Tzen and Maxim Raginsky. Neural stochastic differential equations: Deep latent gaussian models in the diffusion limit, 2019. URL https://arxiv.org/abs/1905.09883.

Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. Time-series generative adversarial networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/c9efe5f26cd17ba6216bbe2a7d26d490-Paper.pdf.

A PROOFS

Suppose random processes X, Y on $\mathcal{T}$ take values in a Polish space $\mathcal{E}$ endowed with its Borel σ-algebra $\mathcal{A}$ [3], let their transition kernels be $\mu^X_{u,v}(X_v, B) = \mathbb{P}(X_v \in B|X_u)$ and $\mu^Y_{u,v}(Y_v, B) = \mathbb{P}(Y_v \in B|Y_u)$ for $u, v \in \mathcal{T}$. For convenience, we use the kernel operations introduced in Chapter 3 of Kallenberg (2021). Let $B_1, B_2 \in \mathcal{A}$ and $t, u, v \in \mathcal{T}$, recall that $\mu^X_{t,u} \otimes \mu^X_{u,v}$ is given by

$$(\mu^X_{t,u} \otimes \mu^X_{u,v})(x, B_1 \times B_2) = \int \mu^X_{t,u}(x, dz_1) \int \mu^X_{u,v}(z_1, dz_2) \mathbb{1}_{B_1 \times B_2}(z_1, z_2)$$

We need the following lemma to prove the main claim.

Lemma 3. *Let $\mathcal{T}$ be an index set. Let X, Y be $\mathcal{E}$-valued Markov processes on $\mathcal{T}$. Then $X \overset{d}{=} Y \iff \forall t_1, t_2 \in \mathcal{T}$, $(X_{t_1}, X_{t_2}) \overset{d}{=} (Y_{t_1}, Y_{t_2})$, where $\overset{d}{=}$ stands for equal in distribution.*

Proof. The $\implies$ direction is straightforward; we prove the other direction. Fix $t_1 \leq t_2 \in T$. Since S is Borel, Theorem 8.5 of Kallenberg (2021) implies that the conditional distribution $\mu^X_{t_1,t_2}(z, \cdot) = \mu^Y_{t_1,t_2}(z, \cdot)$ for almost all z under $\mathbb{P}_{X_{t_1}}$. By Proposition 11.2 of Kallenberg (2021), for any $t_0 \leq t_1 \cdots \leq t_n$ in T,

$$\begin{aligned}
\mathbb{P}_{X_{t_0}, X_{t_1}, \ldots, X_{t_n}} &= \mathbb{P}_{X_{t_0}} \otimes \mu^X_{t_0,t_1} \otimes \cdots \otimes \mu^X_{t_{n-1},t_n} \\
&= \mathbb{P}_{Y_{t_0}} \otimes \mu^Y_{t_0,t_1} \otimes \cdots \otimes \mu^Y_{t_{n-1},t_n} \\
&= \mathbb{P}_{Y_{t_0}, Y_{t_1}, \ldots, Y_{t_n}},
\end{aligned}$$

i.e. $(X_{t_0}, X_{t_1}, \ldots, X_{t_n}) \overset{d}{=} (Y_{t_0}, Y_{t_1}, \ldots, Y_{t_n})$. Then $X \overset{d}{=} Y$ as their finite-dimensional distributions agree. □

Recall that $\{\Omega, \mathcal{F}, \mathbb{P}\}$ is a probability space where $\Omega, \mathcal{F}, \mathbb{P}$ denote the sample space, sigma-algebra, and probability measure, respectively. Random processes X, Y on $\mathcal{T} = [0, T]$ take values in a Polish space $\mathcal{E}$ endowed with its Borel σ-algebra $\mathcal{A}$. For a random variable ξ, the function $\mathbb{P}_\xi = \mathbb{P} \circ \xi^{-1}$ is the induced measure on its range space. In particular, for a random process X, $\mathbb{P}_X$ denotes its law. Let s be any strictly proper scoring rule defined on $\mathcal{E} \times \mathcal{E}$ and $S(P, Q) = \mathbb{E}_Q[s(P, \omega)] < \infty, \forall$ measures P, Q on $\mathcal{E} \times \mathcal{E}$ equipped with σ-algebra $\mathcal{A} \otimes \mathcal{A}$.

Here we present a more general version of Theorem 2 where t_1 and t_2 do not need to be uniformly sampled from $\mathcal{T}$. Let μ be the Lebesgue measure on $\mathcal{T}^2$. Let ν be a measure that is equivalent to μ. That is, there exists the function $\lambda : \mathcal{T}^2 \to \mathbb{R}$ such that $\lambda(t_1, t_2) > 0$ μ-*a.e.* and $\nu(A) = \int_A \lambda(t_1, t_2) d\mu$ for any measurable set A. We define the scoring rule $\bar{s}_\nu$ for continuous Markov processes with respect to the sampling measure ν:

Definition 4. $\bar{s}_\nu(\mathbb{P}_X, y) = \mathbb{E}_{(t_1,t_2)\sim\nu} s(\mathbb{P}_{(X_{t_1}, X_{t_2})}, (y_{t_1}, y_{t_2}))$, *where $\mathbb{P}_{(X_{t_1}, X_{t_2})}$ is the joint marginal distributions at times t_1, t_2 of X.*

Let $\bar{S}_\nu(\mathbb{P}_X, \mathbb{P}_Y) = \mathbb{E}_{y\sim\mathbb{P}_Y}[\bar{s}_\nu(\mathbb{P}_X, y)]$. We present a generalized version of the main statement:

Theorem 5. *If s is a strictly proper scoring rule for distributions on $\mathcal{E} \times \mathcal{E}$, $\bar{s}_\nu$ is a strictly proper scoring rule for $\mathcal{E}$-valued continuous Markov processes on $[0, T]$ where $T \in \mathbb{R}_{>0}$. That is, for any $\mathcal{E}$-valued continuous Markov processes X, Y with laws $\mathbb{P}_X, \mathbb{P}_Y$, respectively, $\bar{S}_\nu(\mathbb{P}_X, \mathbb{P}_Y) \leq \bar{S}_\nu(\mathbb{P}_Y, \mathbb{P}_Y)$ with equality achieved only if $\mathbb{P}_X = \mathbb{P}_Y$.*

[3] S is Borel isomorphic to a Borel set in $[0, 1]$. A Polish space with its Borel σ-algebra is Borel [p14, Kallenberg]

Proof for Theorem 5.

$$
\begin{aligned}
\bar{S}_\nu(\mathbb{P}_X, \mathbb{P}_Y) &= \int \mathbb{E}_{(t_1,t_2)\sim\nu} s(\mathbb{P}_{(X_{t_1},X_{t_2})}, (y_{t_1}, y_{t_2})) \quad \mathbb{P}_Y(dy) \\
&= \mathbb{E}_{(t_1,t_2)\sim\nu} \int s(\mathbb{P}_{(X_{t_1},X_{t_2})}, (y_{t_1}, y_{t_2})) \quad \mathbb{P}_Y(dy) && (2)\\
&= \mathbb{E}_{(t_1,t_2)\sim\nu} \int s(\mathbb{P}_{(X_{t_1},X_{t_2})}, (y_{t_1}, y_{t_2})) \quad \mathbb{P}_{(Y_{t_1},Y_{t_2})}(d(y_{t_1}, y_{t_2})) && (3)\\
&= \mathbb{E}_{(t_1,t_2)\sim\nu} S(\mathbb{P}_{(X_{t_1},X_{t_2})}, \mathbb{P}_{(Y_{t_1},Y_{t_2})}) && (4)\\
&\leq \mathbb{E}_{(t_1,t_2)\sim\nu} S(\mathbb{P}_{(Y_{t_1},Y_{t_2})}, \mathbb{P}_{(Y_{t_1},Y_{t_2})}) && (5)\\
&= \int \mathbb{E}_{(t_1,t_2)\sim\nu} s(\mathbb{P}_{(Y_{t_1},Y_{t_2})}, (y_{t_1}, y_{t_2})) \quad \mathbb{P}_{(Y_{t_1},Y_{t_2})}(d(y_{t_1}, y_{t_2})) && (6)\\
&= \int \mathbb{E}_{(t_1,t_2)\sim\nu} s(\mathbb{P}_{(Y_{t_1},Y_{t_2})}, (y_{t_1}, y_{t_2})) \quad \mathbb{P}_Y(dy) && (7)\\
&= \bar{S}_\nu(\mathbb{P}_Y, \mathbb{P}_Y),
\end{aligned}
$$

We apply Fubini's theorem for the (2) and use the substitution rule (Lemma 1.24, Kallenberg (2021)) (3). (4) and (5) follow from the definition of S and the properness of the scoring rule s, respectively. Fubini's theorem and the substitution rule (Lemma 1.24, Kallenberg (2021)) are used again for the (6) and (7), respectively.

We then show strictness. Let $\bar{S}_\nu(\mathbb{P}_X, \mathbb{P}_Y) = \bar{S}_\nu(\mathbb{P}_Y, \mathbb{P}_Y)$. Then

$$
\begin{aligned}
&\mathbb{E}_{(t_1,t_2)\sim\nu} S(\mathbb{P}_{(X_{t_1},X_{t_2})}, \mathbb{P}_{(Y_{t_1},Y_{t_2})}) = \mathbb{E}_{(t_1,t_2)\sim\nu} S(\mathbb{P}_{(Y_{t_1},Y_{t_2})}, \mathbb{P}_{(Y_{t_1},Y_{t_2})}) \\
\iff\ &\mathbb{E}_{(t_1,t_2)\sim\mu} \lambda(t_1,t_2) S(\mathbb{P}_{(X_{t_1},X_{t_2})}, \mathbb{P}_{(Y_{t_1},Y_{t_2})}) = \mathbb{E}_{(t_1,t_2)\sim\mu} \lambda(t_1,t_2) S(\mathbb{P}_{(Y_{t_1},Y_{t_2})}, \mathbb{P}_{(Y_{t_1},Y_{t_2})}).
\end{aligned}
$$

So $S(\mathbb{P}_{(X_{t_1},X_{t_2})}, \mathbb{P}_{(Y_{t_1},Y_{t_2})}) = S(\mathbb{P}_{(Y_{t_1},Y_{t_2})}, \mathbb{P}_{(Y_{t_1},Y_{t_2})})$ μ-$a.e.$. This implies $(X_{t_1}, X_{t_2}) \overset{d}{=} (Y_{t_1}, Y_{t_2})$ μ-$a.e.$. Next, we show that this statement can be extended to all (t_1, t_2).

Without loss of generality, let $(u_0, u_0') \in [0,T]^2$ and $u_0 < u_0'$. We can inductively select $u_1, u_2, \ldots, u_n, \ldots$ and $u_1', u_2', \ldots, u_n', \ldots$ such that $u_1 \in (u_0, \frac{u_0+u_0'}{2}]$, $u_1' \in [\frac{u_0+u_0'}{2}, u_0')$, $u_{n+1} \in (u_0, \frac{u_0+u_n}{2}]$, $u_{n+1}' \in [\frac{u_n'+u_0'}{2}, u_0')$, and $(X_{u_n}, X_{u_n'}) \overset{d}{=} (Y_{u_n}, Y_{u_n'}) \forall n$. This is possible because $(u_0, \frac{u_0+u_n}{2}] \times [\frac{u_n'+u_0'}{2}, u_0')$ has positive measure. Recall that X and Y are continuous processes. $(X_{u_n}, X_{u_n'})$ converges to $(X_{u_0}, X_{u_0'})$ and $(Y_{u_n}, Y_{u_n'})$ converges to $(Y_{u_0}, Y_{u_0'})$ almost surely as $u_n \to u_0$ and $u_n' \to u_0'$. Recall that $\mathcal{E} \times \mathcal{E}$ is also Polish. Then the convergence also holds in distribution and $(X_{u_0}, X_{u_0'}) \overset{d}{=} (Y_{u_0}, Y_{u_0'})$ (Lemma 5.2 and 5.7, Kallenberg (2021)).

By Lemma 3, $X \overset{d}{=} Y$. □

Theorem 2 is a straightforward result of Theorem 5:

Proof for Theorem 2. Theorem 2 is a direct consequence of Theorem 5 by letting $\nu = \mu$. □

B Extension to Càdlàg Markov Process

We show that the proof can be extended to Càdlàg Markov processes, where the paths $t \mapsto X_t$ are right-continuous with left limits everywhere, with probability one. Although this extension goes beyond the scope of neural SDEs, such processes encompass a wide range of applications. Consider Càdlàg Markov processes X, Y on $\mathcal{T}' = [0, T)$ that take values in a Polish space $\mathcal{E}$ endowed with its Borel σ-algebra $\mathcal{A}$. Let s be any strictly proper scoring rule defined on $\mathcal{E} \times \mathcal{E}$, and let $S(P, Q) = \mathbb{E}_Q[s(P, \omega)] < \infty$ for all measures P, Q on $\mathcal{E} \times \mathcal{E}$ equipped with σ-algebra $\mathcal{A} \otimes \mathcal{A}$.

We generalize Theorem 5 to Càdlàg Markov processes. Let μ denote the Lebesgue measure on $\mathcal{T}' \times \mathcal{T}'$, and let ν be a measure equivalent to μ. We define $\bar{s}_\nu$ as in Definition 4. Let $\bar{S}_\nu(\mathbb{P}_X, \mathbb{P}_Y) = \mathbb{E}_{y\sim\mathbb{P}_Y}[\bar{s}_\nu(\mathbb{P}_X, y)]$. The main statement is presented below in its Càdlàg form:

Theorem 6. *If s is a strictly proper scoring rule for distributions on $\mathcal{E} \times \mathcal{E}$, $\bar{s}_\nu$ is a strictly proper scoring rule for $\mathcal{E}$-valued Càdlàg Markov processes on $[0, T)$ where $T \in \mathbb{R}_{>0}$. That is, for any $\mathcal{E}$-valued Càdlàg Markov processes X, Y with laws $\mathbb{P}_X, \mathbb{P}_Y$, respectively, $\bar{S}_\nu(\mathbb{P}_X, \mathbb{P}_Y) \leq \bar{S}_\nu(\mathbb{P}_Y, \mathbb{P}_Y)$ with equality achieved only if $\mathbb{P}_X = \mathbb{P}_Y$.*

Proof. Following the proof of theorem 5, we can show that $(X_{t_1}, X_{t_2}) \overset{d}{=} (Y_{t_1}, Y_{t_2})$ μ-$a.e.$. We show that this statement can be extended to all $(t_1, t_2) \in \mathcal{T}' \times \mathcal{T}'$ using the right continuity.

Without loss of generality, let $(u_0, u_0') \in [0, T)^2$ and $u_0 < u_0'$. We can inductively select $u_1, u_2, \ldots, u_n, \ldots$ and $u_1', u_2', \ldots, u_n', \ldots$ such that $u_1 \in (u_0, \frac{u_0+u_0'}{2}]$, $u_1' \in [u_0', T)$, $u_{n+1} \in (u_0, \frac{u_0+u_n}{2}]$, $u_{n+1}' \in [u_0', \frac{u_n'+u_0'}{2})$, and $(X_{u_n}, X_{u_n'}) \overset{d}{=} (Y_{u_n}, Y_{u_n'}) \forall n$. This is possible because $(u_0, \frac{u_0+u_n}{2}] \times [u_0', \frac{u_n'+u_0'}{2})$ has positive measure. Recall that X and Y are Càdlàg processes. $(X_{u_n}, X_{u_n'})$ converges to $(X_{u_0}, X_{u_0'})$ and $(Y_{u_n}, Y_{u_n'})$ converges to $(Y_{u_0}, Y_{u_0'})$ almost surely as $u_n \to u_0$ and $u_n' \to u_0'$. Recall that $\mathcal{E} \times \mathcal{E}$ is also Polish. Then the convergence also holds in distribution and $(X_{u_0}, X_{u_0'}) \overset{d}{=} (Y_{u_0}, Y_{u_0'})$ (Lemma 5.2 and 5.7, Kallenberg (2021)). □

C Reduction in Complexity

In this section, we clarify and explain the reduction in computational complexity achieved by our proposed method.

The $O(D^2)$ complexity arises from the previous state-of-the-art Neural SDE training method proposed in Issa et al. (2023), which involves solving a partial differential equation (PDE):

$$f(s,t) = 1 + \int_0^s \int_0^t f(u,v)\langle dx_u, dy_v\rangle_1 dv du,$$

as shown in Equation (2) of their paper. Backpropagation through the PDE solver introduces significant computational cost.

To approximate the double integral numerically, a rectangular rule with D discretization steps is typically employed:

$$\int_0^T \int_0^T f(u,v)\langle dx_u, dy_v\rangle_1 dv du \approx \sum_{i=1}^{D}\sum_{j=1}^{D} f(u_i, v_j)\langle dx_{u_i}, dy_{v_j}\rangle \Delta u \Delta v,$$

where $\Delta u = T/D$, $\Delta v = T/D$, and $u_i = i\Delta u$, $v_j = j\Delta v$ for $i, j = 1, \ldots, D$. This double sum results in $O(D^2)$ complexity.

Furthermore, their method involves a double sum over the batch size B in the objective function (Equation (4) in their paper). Our B corresponds to their m, and the double integral appears in their k_{sig} term. Consequently, their overall complexity is $O(D^2B^2)$.

Our proposed method reduces the complexity from $O(D^2)$ to $O(D)$, or from $O(D^2B^2)$ to $O(DB^2)$ when considering the batch size. This improvement is achieved because our approach eliminates the need to solve the PDE with the double integral, avoiding the computationally expensive operations required by the previous method.

D Alternative Empirical Objectives

Empirical Objective: Multiple Observations Concatenated. An unbiased estimator can be constructed using batches of generated paths $\mathcal{B}_X = \{x^i\}_{i=1}^B$ and data paths $\mathcal{B}_Y = \{y^i\}_{i=1}^B$, where each path is observed at more than two timestamps. Suppose for each i, we select multiple (potentially

irregular) observations at timestamps $t_i^1, t_i^2, \ldots, t_i^N$ where N itself can be random or a tuning parameter. We concatenate these multiple observations to form vectors $[x^i_{t_i^1}, x^i_{t_i^2}, \ldots, x^i_{t_i^N}]$. The empirical estimator is then given by:

$$\begin{aligned}\hat{S}_1(\mathcal{B}_X, \mathcal{B}_Y) =& \frac{1}{2B(B-1)} \sum_{i \neq j} k\left([x^i_{t_j^1}, \ldots, x^i_{t_j^N}],\ [x^j_{t_j^1}, \ldots, x^j_{t_j^N}]\right) \\ &- \frac{1}{B^2} \sum_{i=1}^{B} \sum_{j=1}^{B} k\left([x^i_{t_j^1}, \ldots, x^i_{t_j^N}],\ [y^j_{t_j^1}, \ldots, y^j_{t_j^N}]\right).\end{aligned}$$

Empirical Objective: Adjacent Timestamps as IID Samples. Alternatively, we consider every pair of adjacent timestamps as independent and identically distributed (i.i.d.) samples. Suppose each data path is observed at timestamps $t_i^1 < t_i^2 < \ldots < t_i^M$. For each pair of adjacent timestamps (t_i^m, t_i^{m+1}), we treat the (potentially irregular) observations as i.i.d. samples. The empirical estimator is then:

$$\begin{aligned}\hat{S}_2(\mathcal{B}_X, \mathcal{B}_Y) =& \frac{1}{2B(M-1)(B-1)} \sum_{m=1}^{M-1} \sum_{i \neq j} k\left([x^i_{t_j^m}, x^i_{t_j^{m+1}}],\ [x^j_{t_j^m}, x^j_{t_j^{m+1}}]\right) \\ &- \frac{1}{B^2(M-1)} \sum_{m=1}^{M-1} \sum_{i=1}^{B} \sum_{j=1}^{B} k\left([x^i_{t_j^m}, x^i_{t_j^{m+1}}],\ [y^j_{t_j^m}, y^j_{t_j^{m+1}}]\right).\end{aligned}$$

Note that both estimators only require each data path to be observed at multiple timestamps, which can be irregular and path-dependent. All three empirical objectives, including the one presented in the main paper, perform similarly well in our preliminary experiments.

E Additional Qualitative Results

Here we present additional qualitative results showing the pairwise joint dynamics generated by models trained on different datasets. Results for the metal price dataset are shown in Figure 3. Results for the U.S. stock indices dataset are presented in Figures 4, 5, 6, 7, 8, 9, 10, 11, 12, and 13. Results for the exchange rates data are presented in Figure 14. Results for the energy price data are shown in Figures 15, 16, 17, 18, 19, and 20. Finally, results for the bonds data are presented in Figures 21, 22, and 23.

We present additional qualitative results comparing real and generated sample paths. Results for the exchange rates data are presented in Figures 24 and 25. Similarly, Figures 26 through 30 show the sample paths for five features from the stock indices dataset: "DOLLAR," "USA30," "USA500," "USATECH," and "USSC2000". Finally, Figures 31 and 32 depict the sample paths for silver and gold prices from the metal dataset. These plots demonstrate the ability of neural SDEs to capture dynamics across diverse datasets.

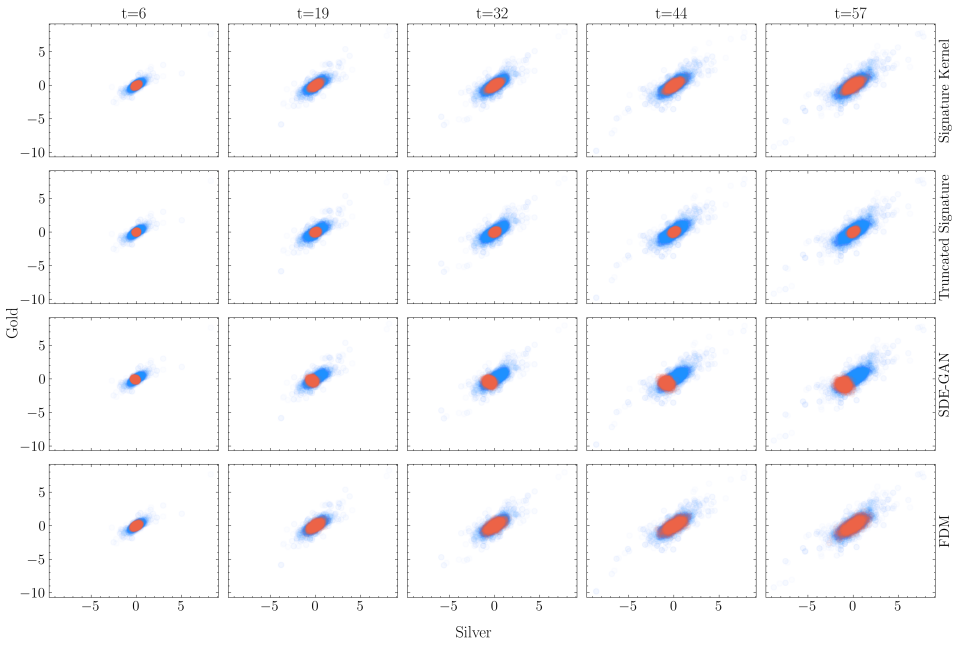

Figure 3: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of gold and silver prices in the metal price data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is the silver price and the vertical axis is the gold price.

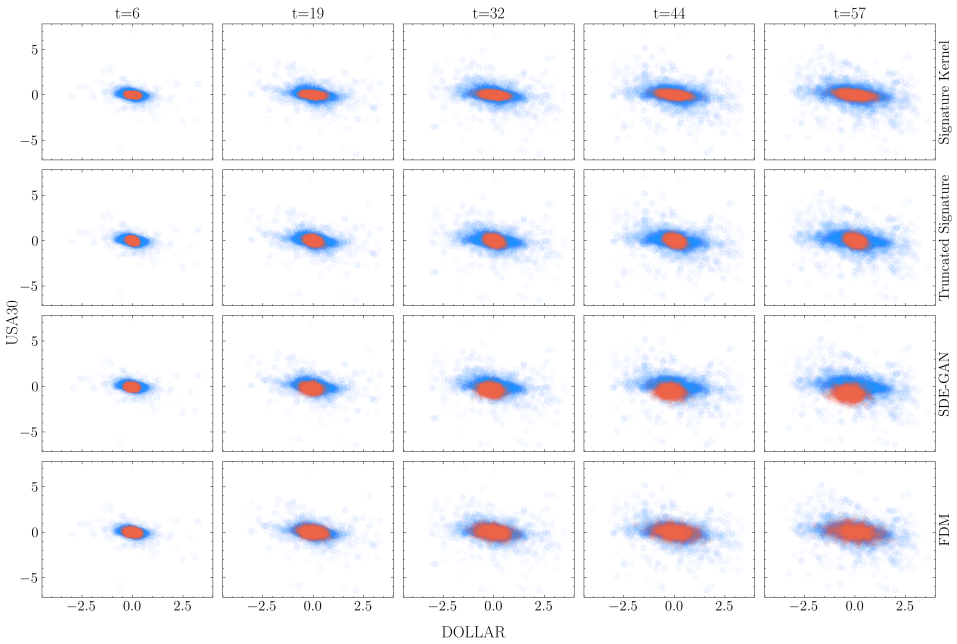

Figure 4: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of Dollar and USA30 in the U.S. stock indices data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is Dollar (US Dollar Index) and the vertical axis is USA30 (USA 30 Index).

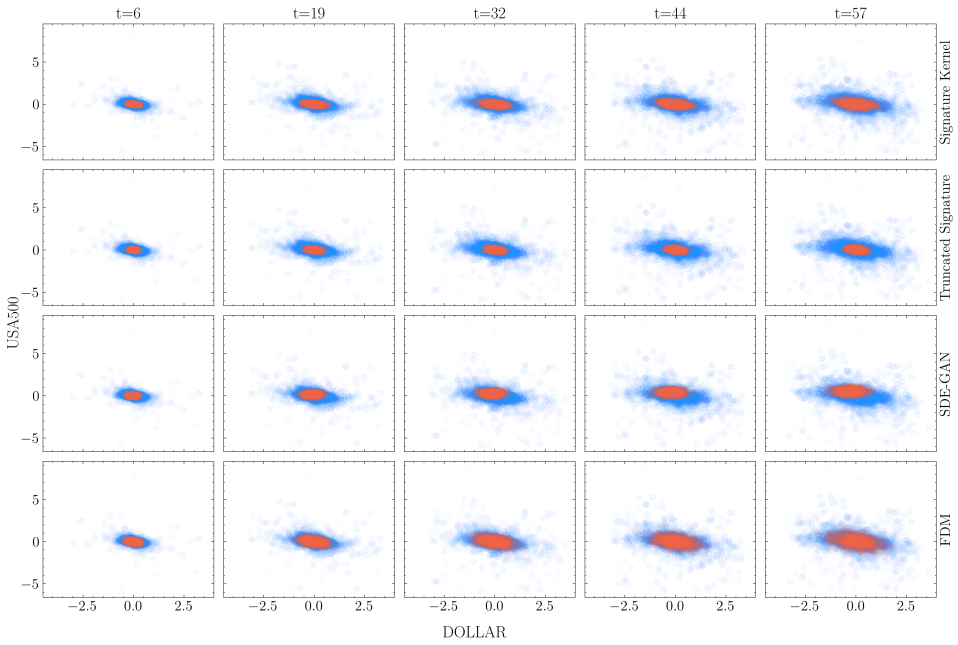

Figure 5: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of Dollar and USA500 in the U.S. stock indices data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is Dollar (US Dollar Index) and the vertical axis is USA500 (USA 500 Index).

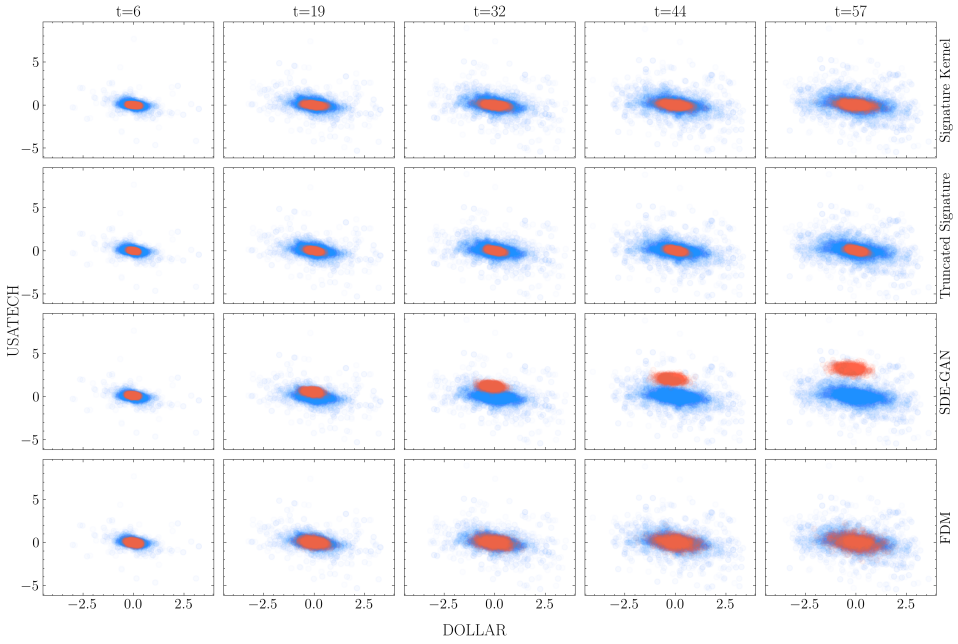

Figure 6: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of Dollar and USATECH in the U.S. stock indices data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is Dollar (US Dollar Index) and the vertical axis is USATECH (USA 100 Technical Index).

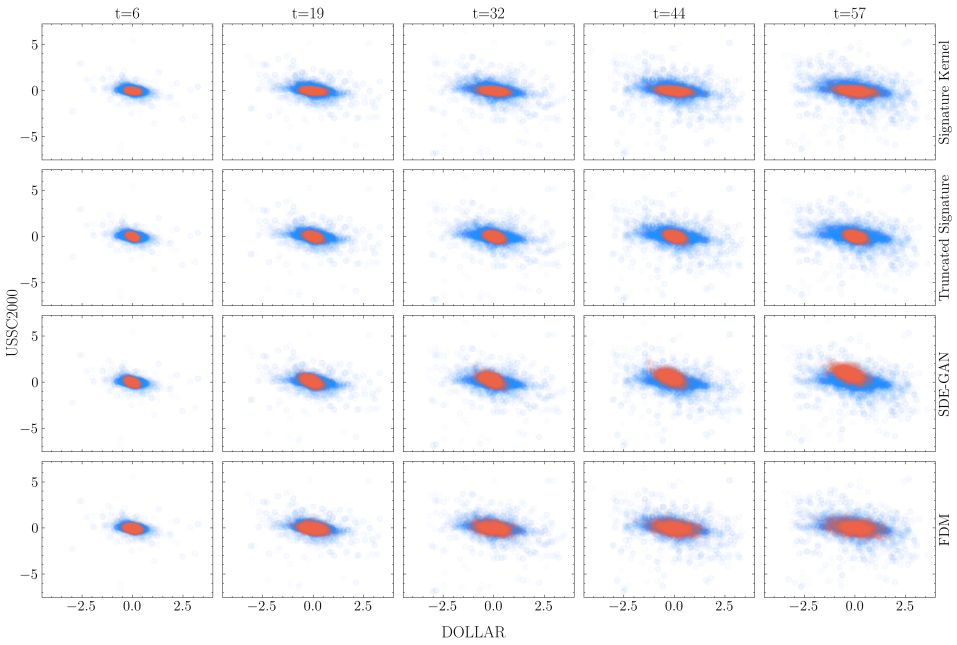

Figure 7: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of Dollar and USSC2000 in the U.S. stock indices data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is Dollar (US Dollar Index) and the vertical axis is USSC2000 (US Small Cap 2000).

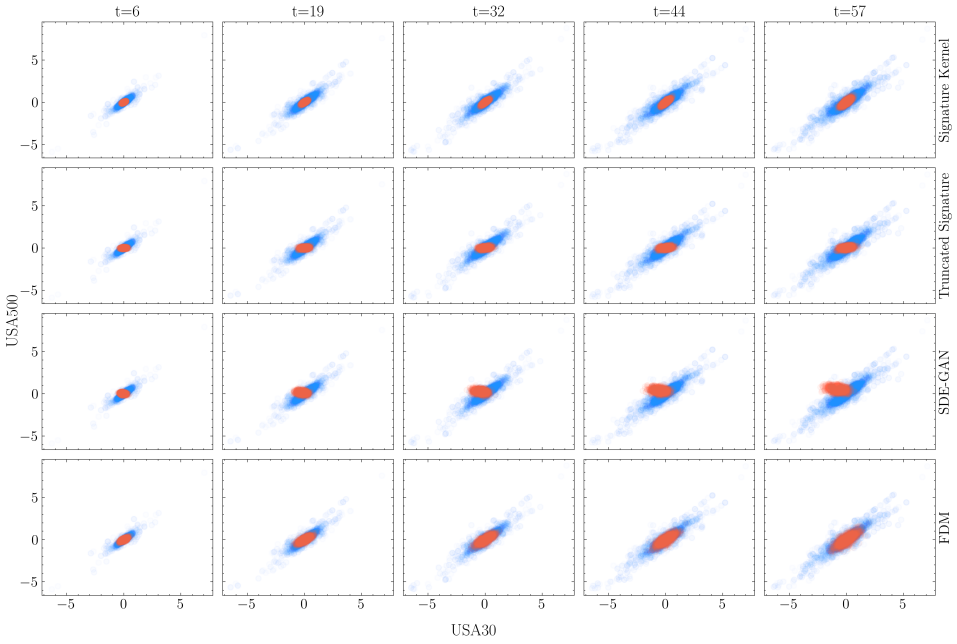

Figure 8: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of USA30 and USA500 in the U.S. stock indices data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is USA30 (USA 30 Index) and the vertical axis is USA500 (USA 500 Index).

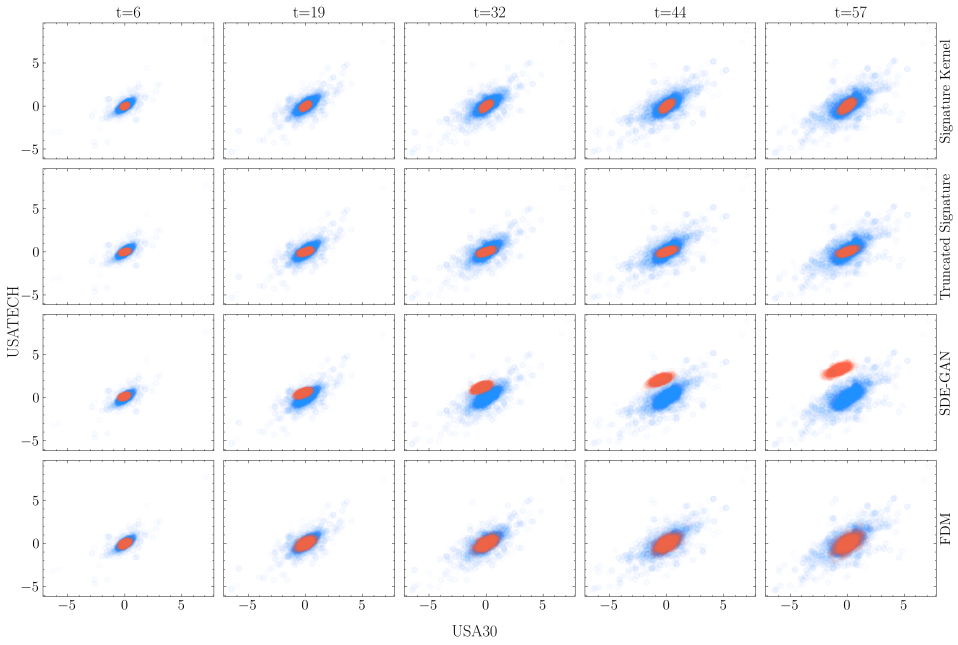

Figure 9: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of USA30 and USATECH in the U.S. stock indices data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is USA30 (USA 30 Index) and the vertical axis is USATECH (USA 100 Technical Index).

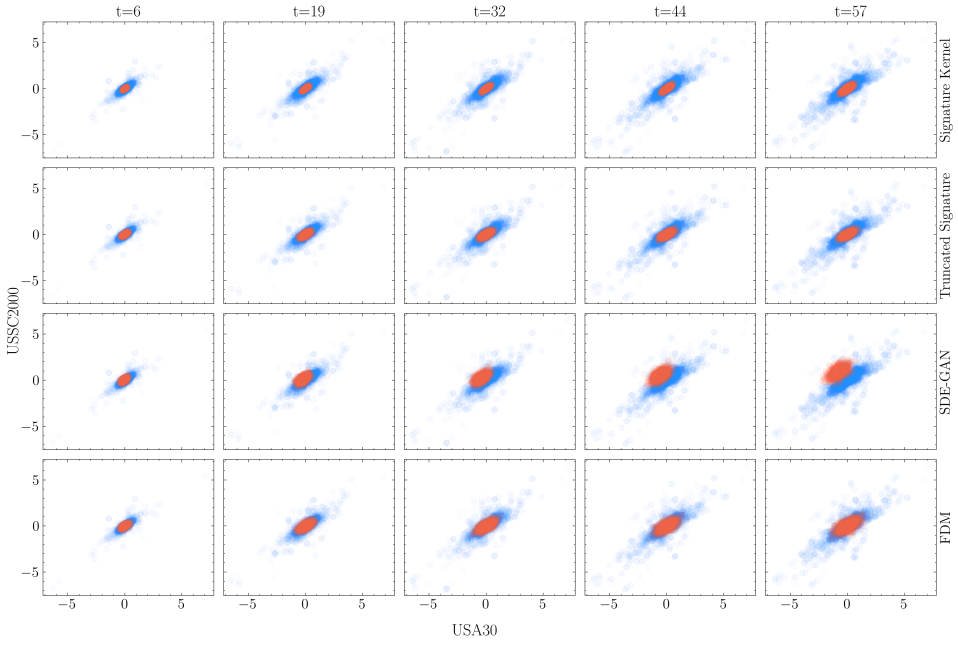

Figure 10: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of USA30 and USSC2000 in the U.S. stock indices data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is USA30 (USA 30 Index) and the vertical axis is USSC2000 (US Small Cap 2000).

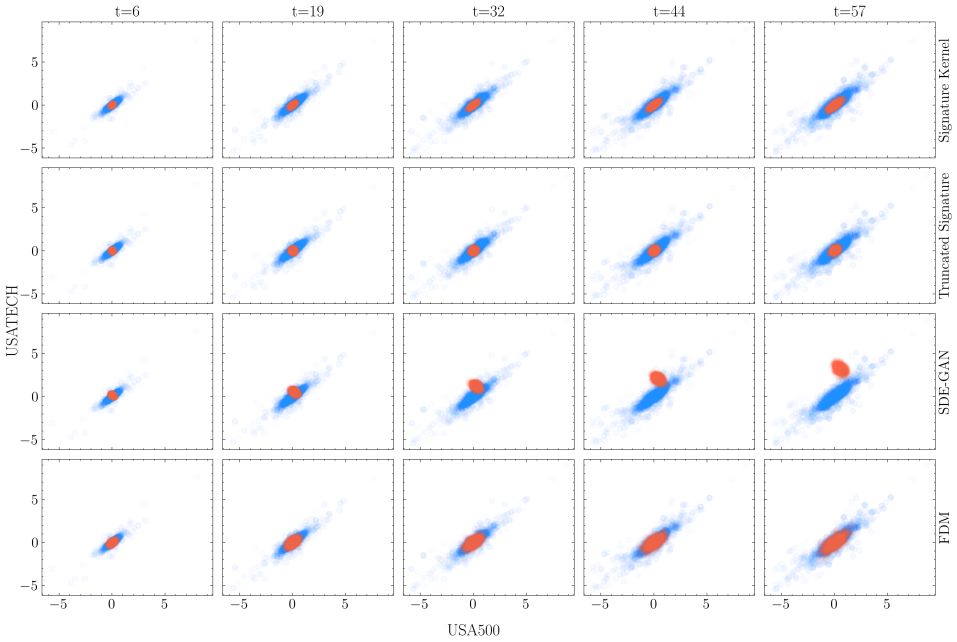

Figure 11: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of USA500 and USATECH in the U.S. stock indices data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is USA500 (USA 500 Index) and the vertical axis is USATECH (USA 100 Technical Index).

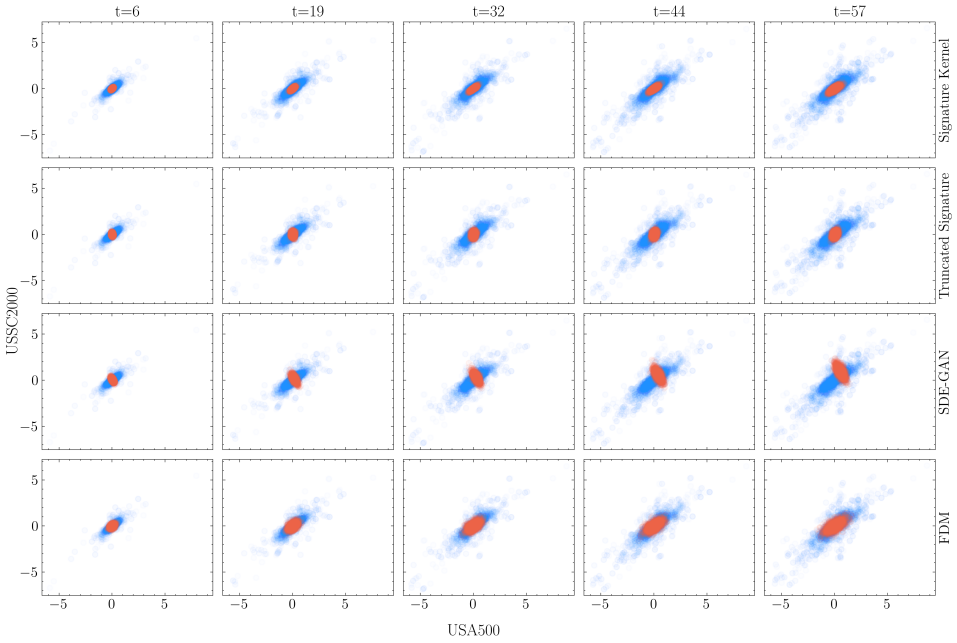

Figure 12: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of USA500 and USSC2000 in the U.S. stock indices data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is USA500 (USA 500 Index) and the vertical axis is USSC2000 (US Small Cap 2000).

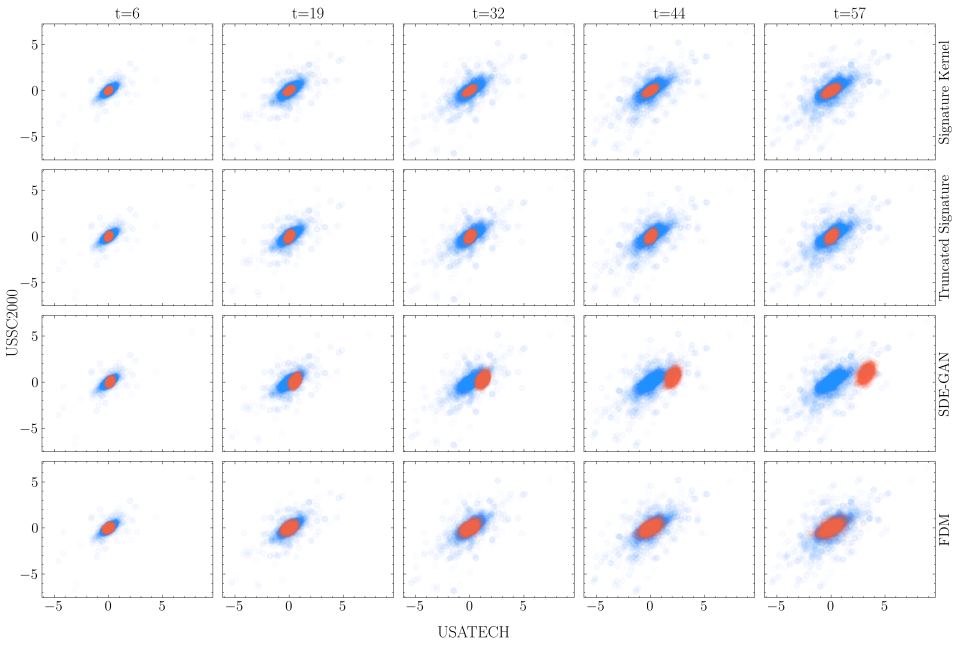

Figure 13: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of USATECH and USSC2000 in the U.S. stock indices data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is USATECH (USA 100 Technical Index) and the vertical axis is USSC2000 (US Small Cap 2000).

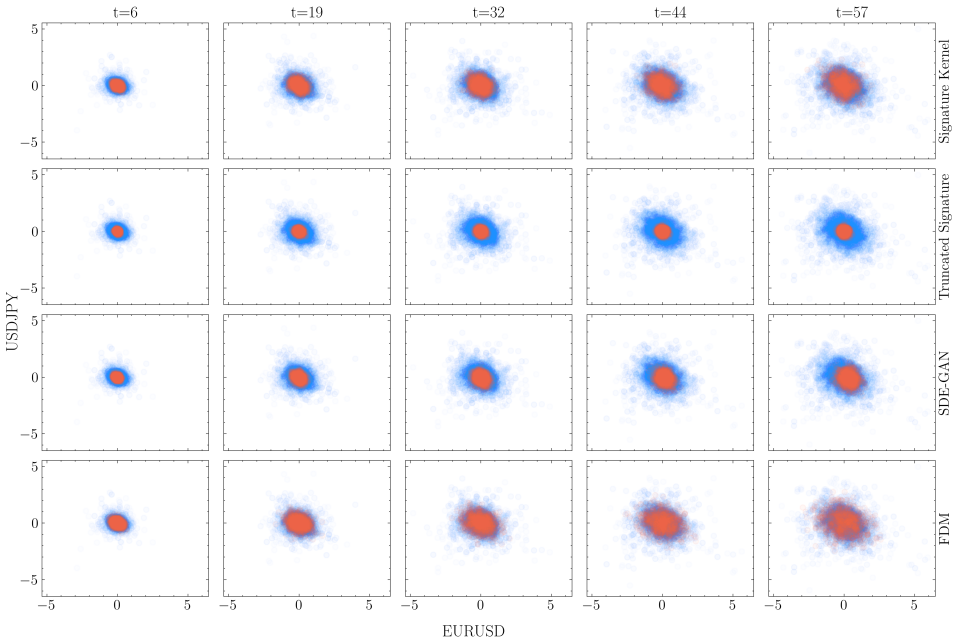

Figure 14: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of EUR/USD and USD/JPY in exchange rate data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is EUR/USD and the vertical axis is USD/JPY.

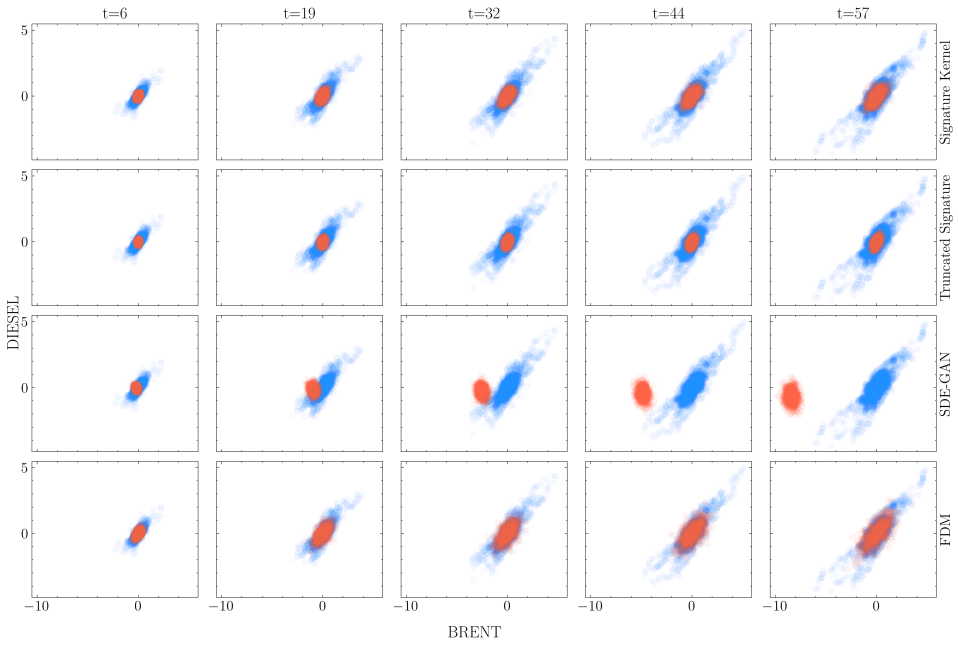

Figure 15: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of BRENT and DIESEL in energy data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is BRENT (U.S. Brent Crude Oil) and the vertical axis is DIESEL (Gas Oil).

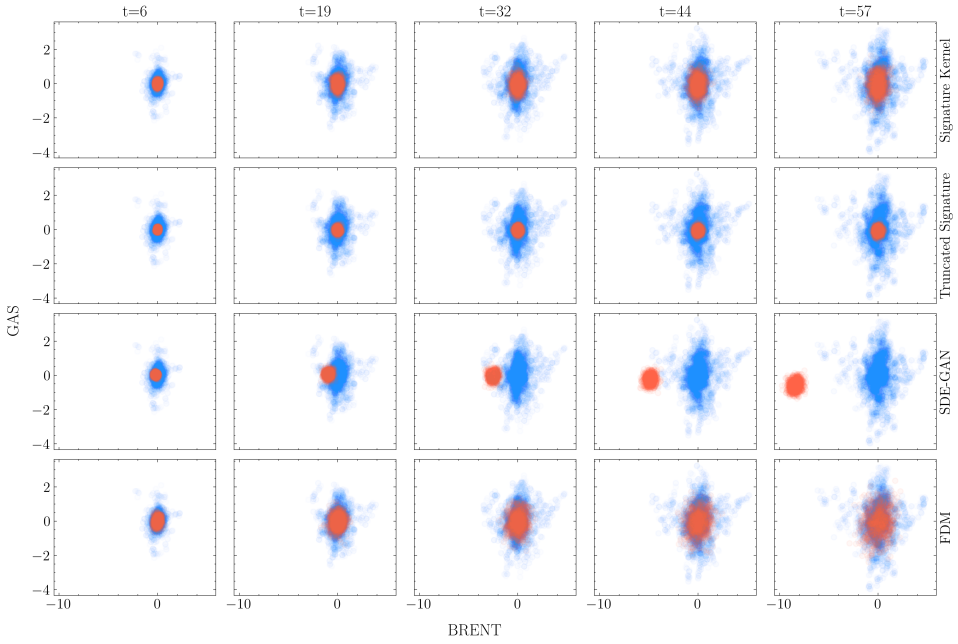

Figure 16: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of BRENT and GAS in energy data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is BRENT (U.S. Brent Crude Oil) and the vertical axis is GAS (Natural Gas).

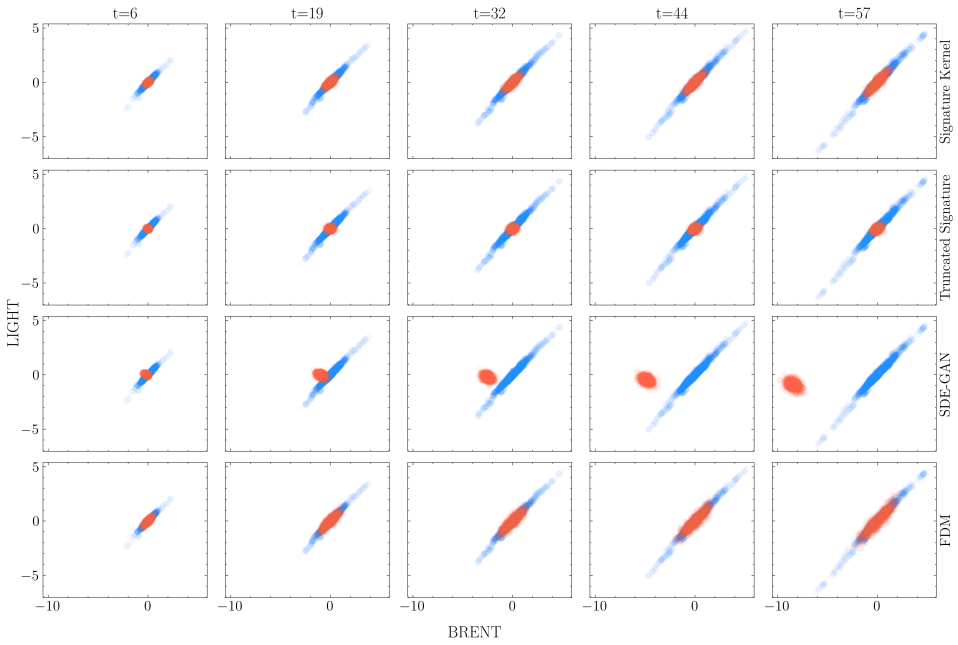

Figure 17: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of BRENT and LIGHT in energy data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is BRENT (U.S. Brent Crude Oil) and the vertical axis is LIGHT (U.S. Light Crude Oil).

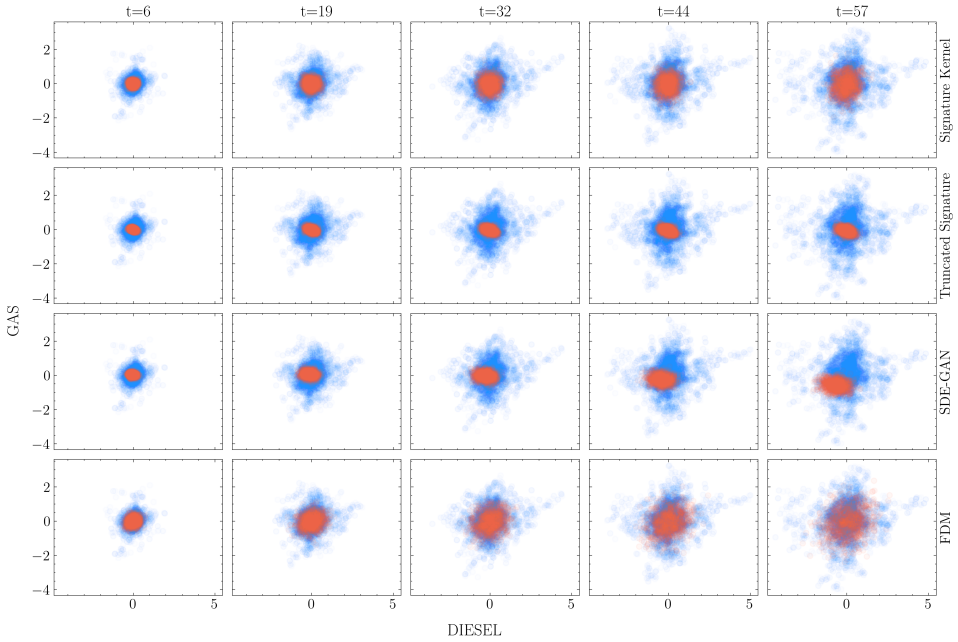

Figure 18: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of DIESEL and GAS in energy data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is DIESEL (Gas Oil) and the vertical axis is GAS (Natural Gas).

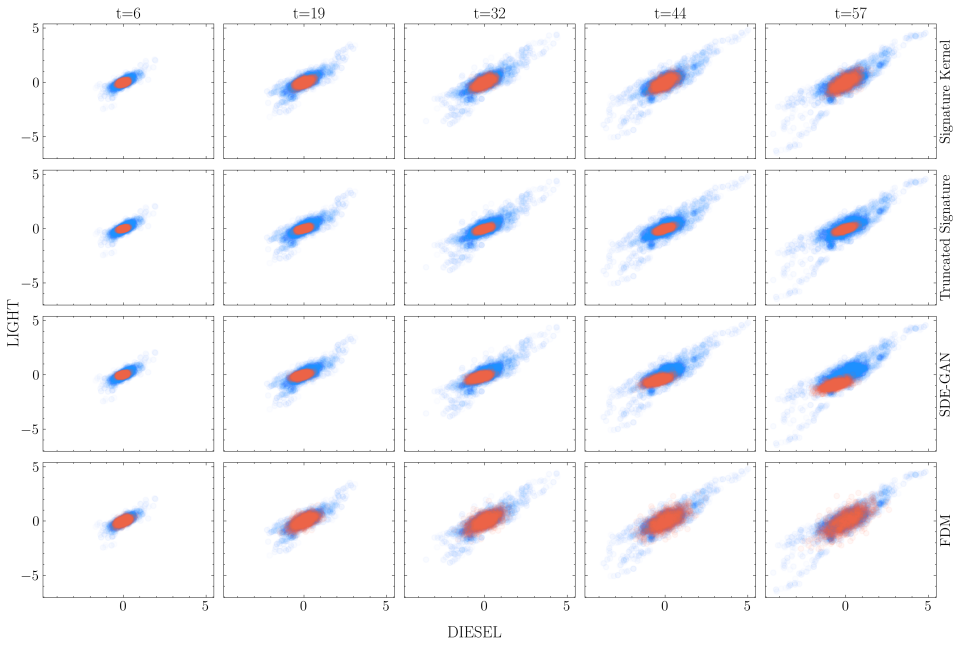

Figure 19: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of DIESEL and LIGHT in energy data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is DIESEL (Gas Oil) and the vertical axis is LIGHT (U.S. Light Crude Oil).

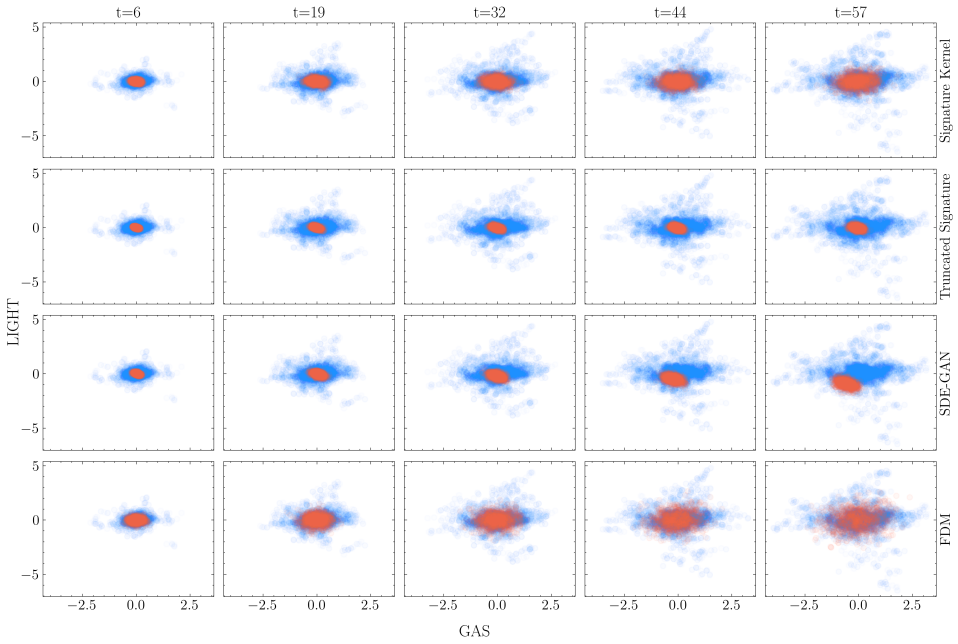

Figure 20: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of GAS and LIGHT in energy data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is GAS (Natural Gas) and the vertical axis is LIGHT (U.S. Light Crude Oil).

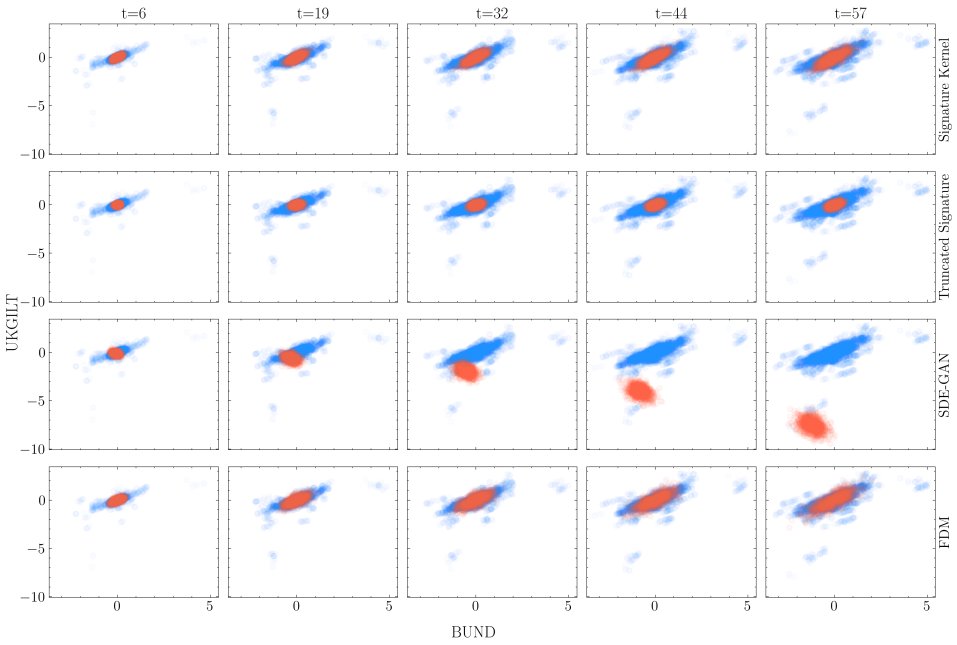

Figure 21: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of BUND and UKGILT in bunds data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is BUND (Euro Bund) and the vertical axis is UKGILT (UK Long Gilt).

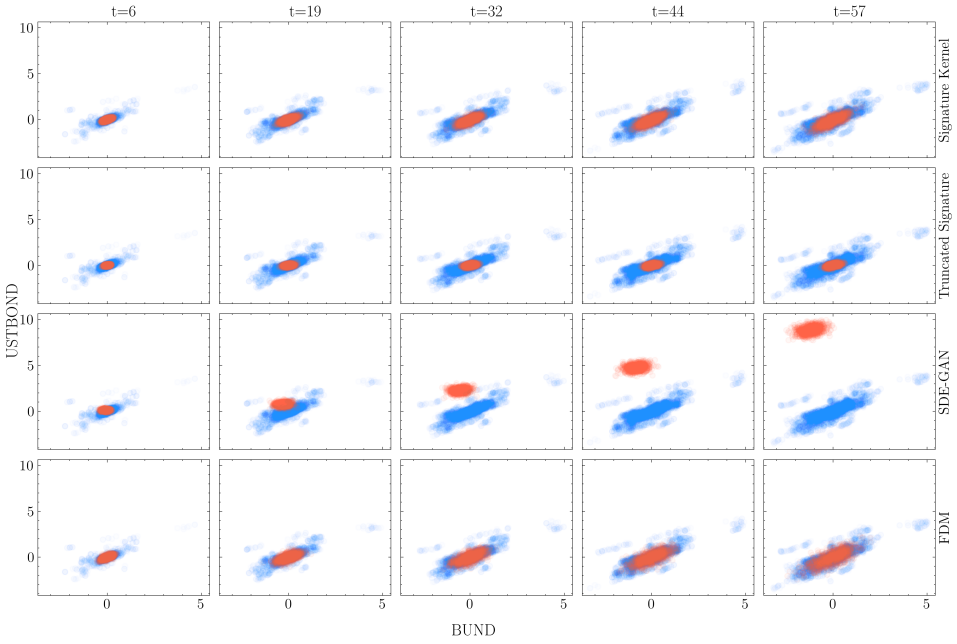

Figure 22: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of BUND and USTBOND in bunds data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is BUND (Euro Bund) and the vertical axis is USTBOND (US T-BOND).

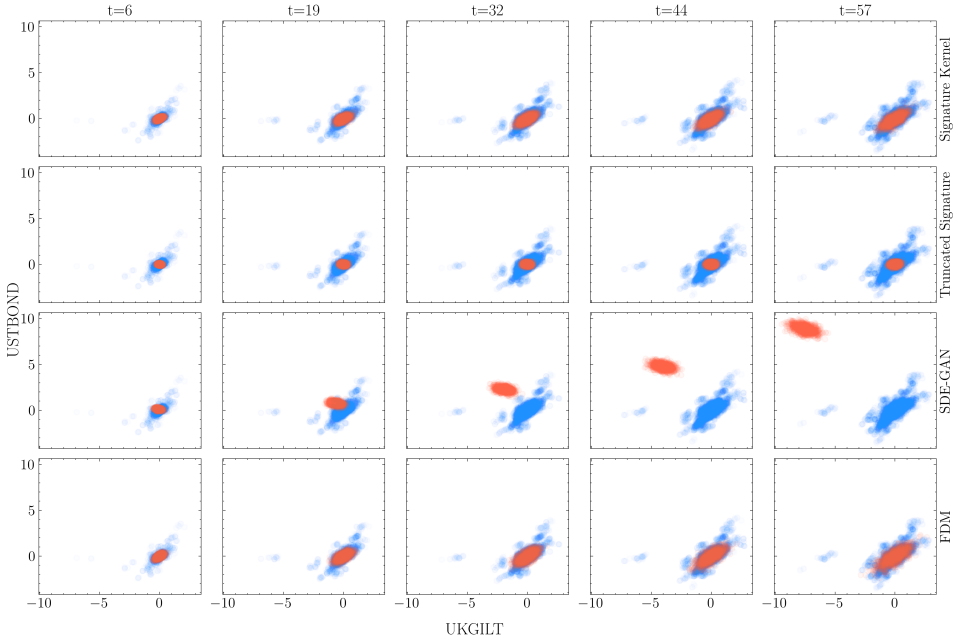

Figure 23: Blue points are real samples and orange points are generated by neural SDEs. The dynamics of the joint distribution of UKGILT and USTBOND in bunds data. Each row of plots corresponds to a method and each row corresponds to a timestamp. For each plot, the horizontal axis is UKGILT (UK Long Gilt) and the vertical axis is USTBOND (US T-BOND).

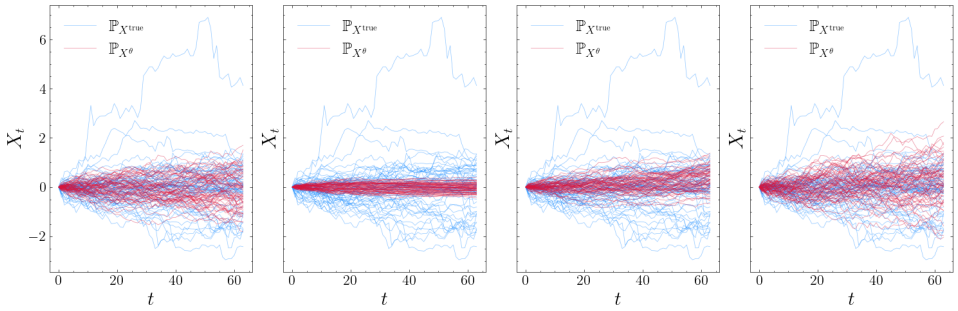

Figure 24: Sample paths for EUR/USD exchange rates from the exchange rate dataset. Blue lines represent real samples, while red lines represent those generated by neural SDEs. From left to right, the plots correspond to signature kernels, truncated signature, SDE-GAN, and FDM, respectively. The horizontal axis represents time, and the vertical axis represents the EUR/USD exchange rate.

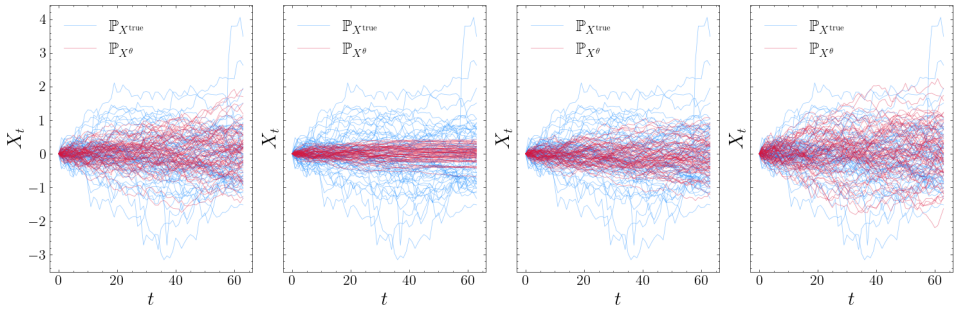

Figure 25: Sample paths for USD/JPY exchange rates from the exchange rate dataset. Blue lines represent real samples, while red lines represent those generated by neural SDEs. From left to right, the plots correspond to signature kernels, truncated signature, SDE-GAN, and FDM, respectively. The horizontal axis represents time, and the vertical axis represents the USD/JPY exchange rate.

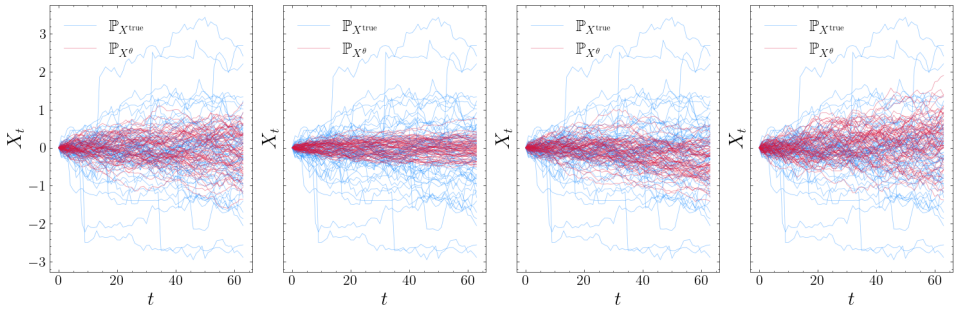

Figure 26: Sample paths for ”DOLLAR” index from the stock indices dataset. Blue lines represent real samples, while red lines represent those generated by neural SDEs. From left to right, the plots correspond to signature kernels, truncated signature, SDE-GAN, and FDM, respectively. The horizontal axis represents time, and the vertical axis represents the ”DOLLAR” index value.

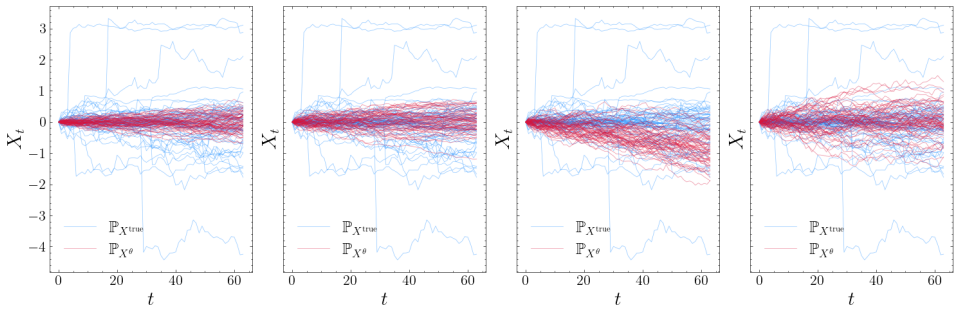

Figure 27: Sample paths for ”USA30” index from the stock indices dataset. Blue lines represent real samples, while red lines represent those generated by neural SDEs. From left to right, the plots correspond to signature kernels, truncated signature, SDE-GAN, and FDM, respectively. The horizontal axis represents time, and the vertical axis represents the ”USA30” index value.

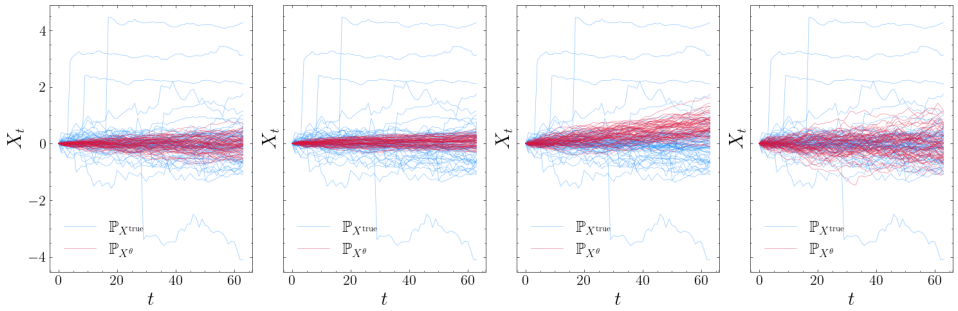

Figure 28: Sample paths for "USA500" index from the stock indices dataset. Blue lines represent real samples, while red lines represent those generated by neural SDEs. From left to right, the plots correspond to signature kernels, truncated signature, SDE-GAN, and FDM, respectively. The horizontal axis represents time, and the vertical axis represents the "USA500" index value.

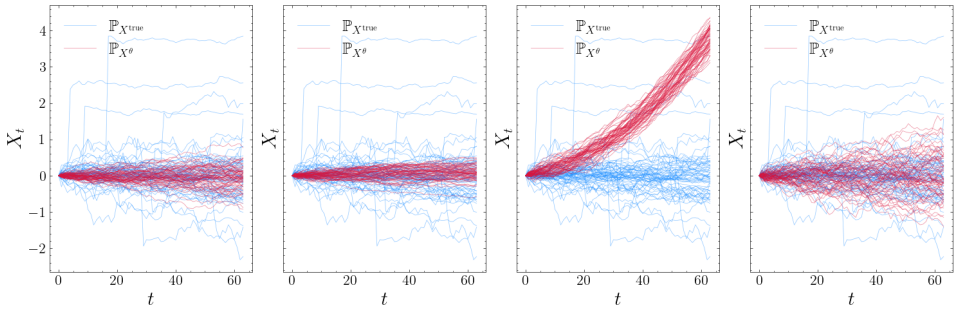

Figure 29: Sample paths for "USATECH" index from the stock indices dataset. Blue lines represent real samples, while red lines represent those generated by neural SDEs. From left to right, the plots correspond to signature kernels, truncated signature, SDE-GAN, and FDM, respectively. The horizontal axis represents time, and the vertical axis represents the "USATECH" index value.

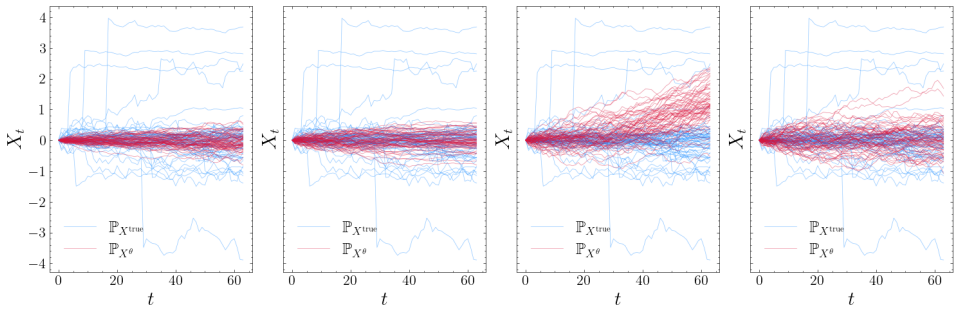

Figure 30: Sample paths for "USSC2000" index from the stock indices dataset. Blue lines represent real samples, while red lines represent those generated by neural SDEs. From left to right, the plots correspond to signature kernels, truncated signature, SDE-GAN, and FDM, respectively. The horizontal axis represents time, and the vertical axis represents the "USSC2000" index value.

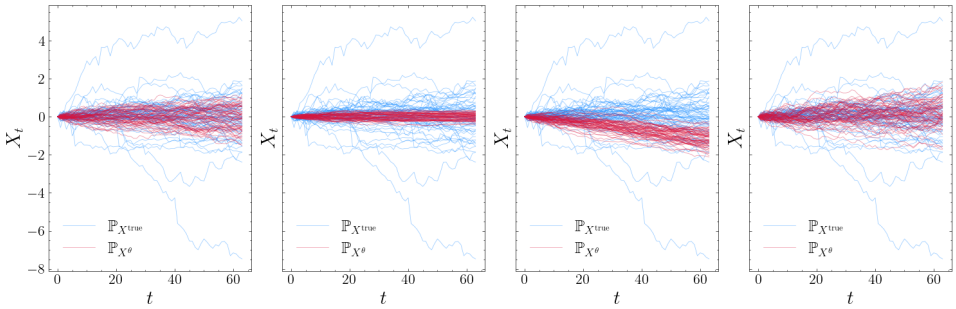

Figure 31: Sample paths for silver prices from the metal dataset. Blue lines represent real samples, while red lines represent those generated by neural SDEs. From left to right, the plots correspond to signature kernels, truncated signature, SDE-GAN, and FDM, respectively. The horizontal axis represents time, and the vertical axis represents silver prices.

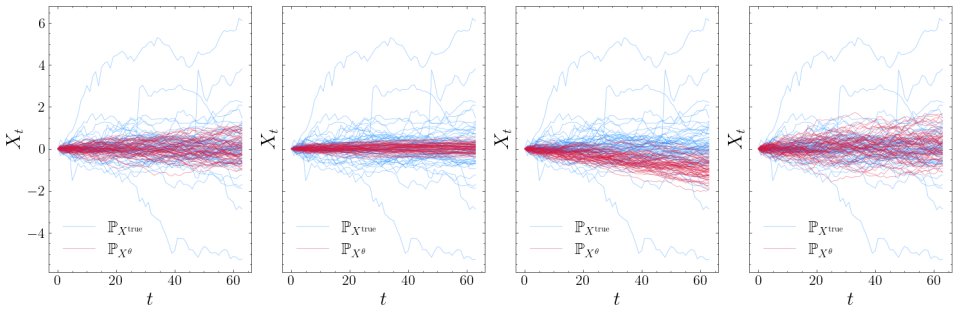

Figure 32: Sample paths for gold prices from the metal dataset. Blue lines represent real samples, while red lines represent those generated by neural SDEs. From left to right, the plots correspond to signature kernels, truncated signature, SDE-GAN, and FDM, respectively. The horizontal axis represents time, and the vertical axis represents gold prices.