Directed Chain Generative Adversarial Networks

Ming Min [*1] **Ruimeng Hu** [*1 2] **Tomoyuki Ichiba** [1]

Abstract

Real-world data can be multimodal distributed, e.g., data describing the opinion divergence in a community, the interspike interval distribution of neurons, and the oscillators' natural frequencies. Generating multimodal distributed real-world data has become a challenge to existing generative adversarial networks (GANs). For example, it is often observed that Neural SDEs have only demonstrated successful performance mainly in generating unimodal time series datasets. In this paper, we propose a novel time series generator, named directed chain GANs (DC-GANs), which inserts a time series dataset (called a neighborhood process of the directed chain or input) into the drift and diffusion coefficients of the directed chain SDEs with distributional constraints. DC-GANs can generate new time series of the same distribution as the neighborhood process, and the neighborhood process will provide the key step in learning and generating multimodal distributed time series. The proposed DC-GANs are examined on four datasets, including two stochastic models from social sciences and computational neuroscience, and two real-world datasets on stock prices and energy consumption. To our best knowledge, DC-GANs are the first work that can generate multimodal time series data and consistently outperforms state-of-the-art benchmarks with respect to measures of distribution, data similarity, and predictive ability.

1. Introduction

Generative models are important to overcome the limitation of data scarcity, privacy, and costs. In particular, medical data are not easy to get, use or share, due to privacy; and financial time series data are inadequate due to their non-stationarity nature. Times-series generative models, instead of seeking to learn the governing equations from real data, aim to discover and learn data automatically, and output new data that plausibly can be drawn from the original dataset. Some existing infinite-dimensional generative adversarial networks (GANs) (e.g., Kidger et al. (2021); Li et al. (2022)) showed successful performance in unimodal time series datasets. However, many real-world phenomena are multimodal distributed, e.g., data describing the opinion divergence in a community (Tsang & Larson, 2014), the interspike interval distribution (Sharma et al., 2018), and the oscillators' natural frequencies (Smith & Gottwald, 2019). All these bring the necessity of developing new generative models for multimodal time series data.

In this paper, we develop a novel time-series generator, named *directed chain GANs* (DC-GANs), motivated by the formulation of DC-SDEs (Detering et al., 2020). The drift and diffusion coefficients in DC-SDEs depend on another stochastic process, which we call the neighborhood process, with distribution required to be the same as the SDEs' distribution. Different from other GANs, which only use real data in discriminators, our proposed algorithm naturally takes the dataset as the neighborhood process, giving generators access to data information. This feature enables our model to outperform the state-of-the-art methods on many datasets, particularly for the situation of multimodal time-series data.

Contribution. We propose a generator for multimodal distributed time series based on DC-SDEs (cf. Definition 2.1), and prove that our model can handle any distribution that Neural SDEs are capable of generating (see Theorem 2.1). To train the generator, we propose to use a combination of two types of discriminators: Sig-WGAN (Ni et al., 2021) and Neural CDEs (Kidger et al., 2020).

We notice that data generated immediately from DC-GANs can be correlated, and propose an easy solution by walking along the directed chain in the path space for further steps (see Theorem 2.2). Combining branching the chain with different Brownian noises enables our model to generate unlimited independent fake data.

We test our algorithms in four different experiments and show that DC-GANs provide the best performance com-

[*]Equal contribution [1]Department of Statistics and Applied Probability, University of California, Santa Barbara, CA 93106-3110, USA. [2]Department of Mathematics, University of California, Santa Barbara, CA 93106-3080, USA.. Correspondence to: Ming Min <m_min@pstat.ucsb.edu>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

pared to existing popular models, including SigWGAN (Ni et al., 2021), CTFP (Deng et al., 2020), Neural SDEs (Kidger et al., 2021), TimeGAN (Yoon et al., 2019) and Transformer-based generator TTS-GAN (Li et al., 2022).

Related Literature. Neural ordinary differential equations (Neural ODEs), introduced by Chen et al. (2018), use neural networks to parameterize the vector fields of ODEs and bring a powerful tool for learning time series data. Later, significant effort has been put into improving Neural ODEs, e.g., Quaglino et al. (2019); Zhang et al. (2019); Massaroli et al. (2020); Hanshu et al. (2019). In fact, incorporating mathematical concepts into the Neural ODEs framework can provide the capability of analyzing and justifying its validity, leading to a deeper understanding of the framework itself. For example, Li et al. (2020) and Tzen & Raginsky (2019a) generalized the idea to neural stochastic differential equations (Neural SDEs), providing adjoint equations for efficient training. By integrating rough path theory (Lyons et al., 2007), Kidger et al. (2020) proposed neural controlled differential equations (Neural CDEs) and Morrill et al. (2021) proposed neural rough differential equations for modeling time series. Other examples integrating profound mathematical concepts include using higher order kernel mean embeddings to capture information filtration (Salvi et al., 2021), and solving high dimensional partial differential equations through backward stochastic differential equations (Han et al., 2018), to name a few.

The closely related model to ours is the Neural SDEs by Kidger et al. (2021), which uses the Wasserstein GAN method to train stochastic diffusion evolving in a hidden space and gains great success in simulating time series data. Other successful GANs models for time-series data include Cuchiero et al. (2020); Tzen & Raginsky (2019b); Deng et al. (2020); Kidger et al. (2021); Li et al. (2022); see Brophy et al. (2022) for a recent review. Note that we find in the numerical experiments that the performances of Neural SDEs are limited in simulating multimodal distributed time series, e.g., as shown in Figure 1 from the stochastic opinion dynamics (Example 1 in Section 4.2).

The directed chain is one of the simplest structures in random graph theory, where each node on the graph represents a stochastic process and has interactions only with its neighbor nodes (Figure 4). To our best knowledge, Detering et al. (2020) initiated the study of the SDE system on the directed chains, followed by Feng et al. (2021a;b) for the analysis of stochastic differential games on such chains with (deterministic and random) interactions. Later on, more complicated graph structures are studied beyond directed chains. For example, Lacker et al. (2021) analyzed particle behaviors where the interaction only happens between neighborhoods in an undirected graph, and proved Markov random fields property and constructed Gibbs measure on path space when interactions appear only in drift; Lacker & Soret (2022) considered stochastic differential games on transitive graphs; Carmona et al. (2022) studied games on a graphon which has infinitely many nodes. Despite numerous extensions, we find that the directed chain structure, although simple but rich enough for generating multimodal time series.

From another viewpoint, DC-SDEs can be understood as the reverse direction of mimicking theorems (Gyöngy, 1986). The idea of "mimicking" is that for a general SDE (even with path-dependence features), one can construct a Markovian one to mimic its marginal distribution; see Brunick & Shreve (2013) for details on mimicking aspects of Itô processes including the distributions of running maxima and running integrals. DC-SDEs work in the reverse direction: they can produce marginal distributions that are generated by Markovian SDEs (see Theorem 2.1 for a detailed statement). The benefit of using DC-SDEs, in particular in machine learning, is to have a more vital fitting ability by embedding data into a slightly more complicated system.

2. Directed Chan SDEs and Signatures

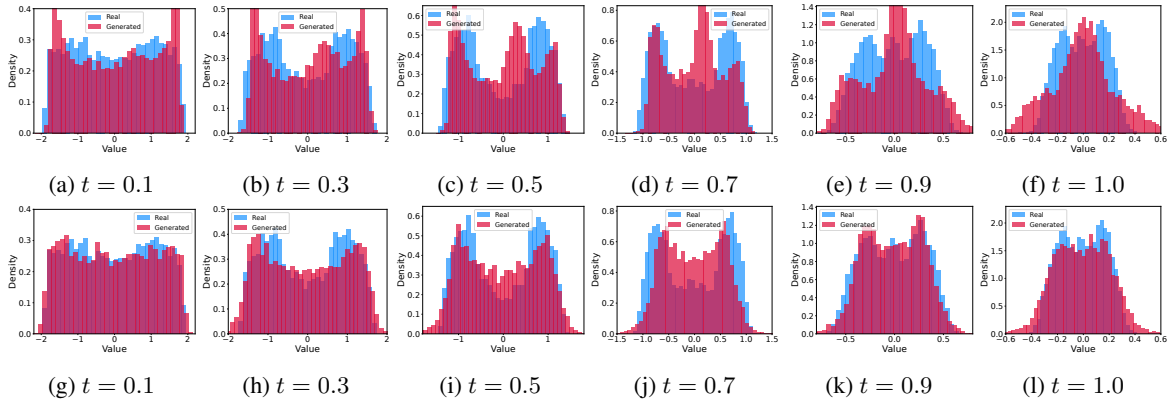
In this section, we introduce two mathematical concepts that serve as the backbones of our algorithm: directed chain SDEs and signatures. In Section 2.1, we identify the central issue of naively generating time series from true data using DC-SDEs: the non-independence of the true data and fake data (Problem 2.1). Then we overcome the non-independence issue by Decoorelating and Branching Phase in Section 3.1, and provide theoretical guarantees for this procedure (Theorem 2.2).

In the sequel, we shall use X_s, X_t to denote the state of X at time s and t, respectively. With no subscript, e.g., by X, we mean the whole path from $t = 0$ to T.

2.1. Directed Chain SDEs (DC-SDEs)

The DC-SDEs are the limit of a system of n-coupled SDEs interacting homogeneously on a directed chain when n goes to infinity. Below we will focus on DC-SDEs and defer the introduction of this limiting process to Appendix A. Under the general setup, DC-SDEs can be of McKean-Vlasov type where the coefficients have distributions as inputs, corresponding to the n-coupled system having mean-field interaction. In our proposed generator, it is sufficient to use the simple case mentioned above, DC-SDE without the mean-field interaction, as in the following definition.

Definition 2.1 (DC-SDEs). *Fix a filtered probability space* $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\geq 0}, \mathbb{P})$ *and a finite time horizon* $[0, T]$. *Let* $(X, \tilde{X})$ *with* $X, \tilde{X} \in L^2(\Omega \times [0, T], \mathbb{R}^N)$ *be a pair of square-*



(a) $t = 0.1$ (b) $t = 0.3$ (c) $t = 0.5$ (d) $t = 0.7$ (e) $t = 0.9$ (f) $t = 1.0$

(g) $t = 0.1$ (h) $t = 0.3$ (i) $t = 0.5$ (j) $t = 0.7$ (k) $t = 0.9$ (l) $t = 1.0$

Figure 1. Marginal distributions of real data (blue) and generated data (red) from Example 1 (Stochastic opinion dynamics) at $t \in \{0.1, 0.3, 0.5, 0.7, 0.9, 1\}$ in Section 4.2. Figures (a)–(f) are generated by Neural SDEs, and Figures (g)–(l) are generated by DC-GANs. One can see from Figures (e) and (f) that Neural SDEs fail to capture the bimodal distribution.

integrable stochastic processes satisfying

$$X_t = \xi + \int_0^t V_0(s, X_s, \tilde{X}_s)\,\mathrm{d}s + \int_0^t V_1(s, X_s, \tilde{X}_s)\,\mathrm{d}B_s, \tag{1}$$

for $t \in [0, T]$, with the distributional constraint

$$\mathrm{Law}(X_t, 0 \le t \le T) = \mathrm{Law}(\tilde{X}_t, 0 \le t \le T), \tag{2}$$

where Law$(\cdot)$ stands for the distribution, $V_0 \in \mathbb{R}^N$ and $V_1 \in \mathbb{R}^{N\times d}$ are smooth coefficients satisfying Lipschitz and linear growth conditions, B is a standard d-dimensional Brownian motion, and $X_0 := \xi$, $\tilde{X}$ and B are assumed to be independent.

The existence of the solution to (1) and the weak uniqueness in the sense of distribution have been proved under the Lipschitz and linear growth assumptions on the coefficients in Detering et al. (2020) for a simple case, and in Ichiba & Min (2022) for a more general case. Moreover, with the smoothness of the solution under certain additional conditions posed on the coefficients (cf. Ichiba & Min (2022)), we can derive a partial differential equation (PDE) for the marginal densities of the solution. Then, the associated PDEs lead to the following theorem: DC-SDEs have at least the same amount of flexibility as Neural SDEs.

Theorem 2.1. *Under proper assumptions, for any Y that satisfies a system of Markovian SDEs on $[0, T]$, there exists a unique solution to the DC-SDE (1) with constraints (2), some V_0 and non-degenerate coefficients V_1, such that they have the same marginal distributions for all $t \in [0, T]$. Here by degenerate, we mean that $V_i(t, x, \tilde{x}) := V_i(t, x)$, $i \in \{0, 1\}$, i.e., the coefficients have no dependence on neighborhood nodes at all.*

We defer the proof of Theorem 2.1 to Appendix B.2.

Naturally, if V_0 and V_1 are known (or learned from data), one can take real data paths as $\tilde{X}$ in (1) and straightforwardly generate paths of X that have the same distribution as $\tilde{X}$ by the constraint (2). However, naively implementing this idea will lead to the following potential problems.

Problem 2.1 (Lack of Independence)**.** *The distribution of the generated sequence crucially depends on the real data; Consequently, to avoid dependence, a single real path can only be used once as $\tilde{X}$ to generate one path of X, and thus the number of the generated sequence has to be the same as that of the training data set in one run.*

Note that a qualified generator should also be able to generate *unlimited* independent data that does not depend on the original one. Fortunately, both problems mentioned above can be overcome by the idea behind the following theorem.

Theorem 2.2. *Under mild non-degeneracy conditions, the correlation between training data and generated data in DC-SDEs decays exponentially fast, as the distance increases on the chain.*

Due to the page limit, we give the formal statement of Theorem 2.2 with detailed proof in Appendix B.3.

We shall explain how to beat the independence problem during the implementation described in Section 3.1. As shown in Appendix B.3, the introduction of independent Brownian motions to (1) is the key to solving the independence problem. We shall also provide an extreme example (cf. Remark B.1) showing that without $\int V_1\,\mathrm{d}B$, the system (1)–(2) has only trivial (deterministic) solution.

2.2. Signature

The proposed method utilizes signature (Lyons et al., 2007), a concept from rough path theory that we shall briefly introduce for completeness. As an infinitely graded sequence,

the signature can be understood as a feature extraction technique for time series data with certain regularity conditions. Let $x : \Omega \times [0, T] \to \mathbb{R}^N$ be a continuous random process, and denote the signature map by $S : x \mapsto S(x) \in T(\mathbb{R}^N)$, where $T(\mathbb{R}^N)$ is the tensor algebra defined on $\mathbb{R}^N$. Then,

$$S(x) := (1, x^1, \cdots, x^i, \dots), \quad \text{and}$$

$$x^i = \int_{0<t_1<\cdots<t_i<T} \mathrm{d}x_{t_1} \otimes \cdots \otimes \mathrm{d}x_{t_i}.$$

Signature characterizes paths uniquely up to the tree-like equivalence, and the equivalence is removed if at least one dimension of the path is strictly increasing (Boedihardjo et al., 2016). The concept has been applied to design machine learning methods, e.g., Chevyrev & Oberhauser (2022); Kidger et al. (2019); Min & Hu (2021); Ni et al. (2021); Min & Ichiba (2022); Dyer et al. (2021). In practice, one needs to truncate the signature up to a finite order M, denoted by $S^M(x) = (1, x^1, \cdots, x^M)$. We will justify the truncation by the factorial decay property and discuss other theoretical properties of signature in Appendix B.1.

In addition, signature induces another powerful tool to characterize the distribution of random processes: the *expected signatures*. It was proved by Chevyrev & Lyons (2016) that expected signatures characterize the distribution of random processes uniquely, i.e., if $\mathbb{E}[S(x)] = \mathbb{E}[S(y)]$ and $\mathbb{E}[S(x)]$ has an infinite radius of convergence, then x and y have the same distribution.

3. Proposed Method: DC-GANs

In this section, we describe DC-GANs for generating multimodal distributed time series. Our method builds on the DC-SDEs with a straightforward idea: To find the (sub-) optimal solution of the generator, we implement a GAN model with the Neural DC-SDEs as the generator. For the discriminator, we use Neural CDEs (Kidger et al., 2021) and Sig-Wasserstein GAN (Ni et al., 2020; 2021).

3.1. Generator

To overcome the independence issue explained in Problem 2.1, we design DC-GANs by two phases: 1) training and 2) decorrelating and branching. The second phase will be utilized during testing. Both V_0 and V_1 in (1) will be parameterized by multi-layers fully connected NNs.

Training Phase. We set aside the independence problem and focus on finding the optimal coefficients V_0 and V_1 (together with the discriminator). Denote the training data by $\{\tilde{X}(\omega_i)\}_{i=1}^M$, where each ω_i represents a realization of the randomness in the path space. We treat our training data $\{\tilde{X}(\omega_i)\}_{i=1}^M$ as the neighborhood process $\tilde{X}$ in (1). For each training path data $\tilde{X}(\omega_i)$, we generate a DC-SDE path $X(\omega_i)$, according to the Euler scheme of (1),

$$\begin{aligned} &X_{t_{j+1}}(\omega_i) \\ &= X_{t_j}(\omega_i) + V_0(t_j, X_{t_j}(\omega_i), \tilde{X}_{t_j}(\omega_i))(t_{j+1} - t_j) \\ &\quad + V_1(t_j, X_{t_j}(\omega_i), \tilde{X}_{t_j}(\omega_i))(B_{t_{j+1}}(\omega_i) - B_j(\omega_i)), \end{aligned} \tag{3}$$

where $0 = t_0 \le t_1 \le \ldots \le t_J = T$ is a partition on $[0, T]$, $\{B(\omega_i)\}_{i=1}^M$ are independent Brownian paths.

Both the generated paths $\{X(\omega_i)\}_{i=1}^M$ and the training paths $\{\tilde{X}(\omega_i)\}_{i=1}^M$ will be passed into the discriminator, where their Wasserstein distance needs to be minimized. To simplify the notations for later use, we define $G_\theta : (\xi, B, \tilde{X}) \mapsto X$ to represent the overall transformation in (3), with θ denoting all network parameters of V_0 and V_1.

Decorrelating and Branching Phase. During testing, we utilize a branching scheme to alleviate the independence problem; see Figure 2 for an illustrative example. Let q be the number of steps we "walk" along the directed chain. Here "walking" along the chain means: After we have finished the training (identified V_0 and V_1) phase, we start with the first chain (the grey one in Figure 2). We take real data as the first neighborhood X_1 to generate X_2 through the scheme (3), where X_1 takes the role of $\tilde{X}$ and X_2 takes the role of X. Then we use X_2 as the neighborhood to generate X_3, and repeat this procedure until we obtain X_q. By Theorem 2.2, X_q and X_1 are asymptotically uncorrelated, as $q \to \infty$. We describe the pseudo-code in Algorithm 1 below for this decorrelating step.

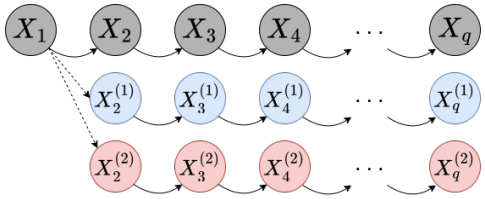


Figure 2. Branching Scheme. Let q be the number of steps we "walk" along the directed chain. We take real data as the first neighborhood X_1 to generate X_2 through the scheme (3), where X_1 takes the role of $\tilde{X}$ and X_2 takes the role of X. Then we use X_2 as the neighborhood to generate X_3, and repeat this procedure until we obtain X_q.

To generate more fake data, we can initiate more chains with the same starting node X_1 (where the real data are) and independent Brownian paths, and then "walk" along the chain to get $X_q^{(i)}, i = 2, 3, \ldots$. Again, $X_q^{(i)}$ is asymptotically uncorrelated to X_1. By the definition of DC-SDEs, X_q and $X_q^{(i)}$ are conditionally independent with conditioning on X_1. Therefore, we can claim that X_q and $X_q^{(i)}$ are asymptotically uncorrelated.

Architecture. Note that although the directed chain SDE

Algorithm 1 Generator in the Decorrelating and Branching

Input: real data $\{\tilde{X}(\omega_i)\}_{i=1}^M$, # of steps q, generator G_θ;
Set $\{X_1(\omega_i)\}_{i=1}^M := \{\tilde{X}(\omega_i)\}_{i=1}^M$;
for $k = 2$ **to** q **do**
  Generate M independent copies of initials positions and Brownian paths $\{\xi_k(\omega_i), B_k(\omega_i)\}_{i=1}^M$;
  Generate M paths $\{X_k(\omega_i)\}_{i=1}^M$ by

$$X_k(\omega_i) = G_\theta(\xi_k(\omega_i), B_k(\omega_i), X_{k-1}(\omega_i));$$

end for
Output: $\{X_q(\omega_i)\}_{i=1}^M$

pair $(X, \tilde{X})$ is Markovian, X itself can be non-Markovian as a standalone stochastic process. All the historical information can be embedded in the neighborhood process and fetched through V_0 and V_1. Such a property leads to one of the key differences between our method and Neural SDEs: there is no need to embed time series into a hidden space. In our implementations, V_0 and V_1 take standard feedforward neural networks; see Appendix C for details.

3.2. Discriminator

The purpose of the discriminator is to identify the optimal parameters in the V_0- and V_1- networks. We use the Wasserstein GAN framework (Goodfellow et al., 2020; Arjovsky et al., 2017) to train the generator, and two types of discriminators will be used here.

SigWGAN. Using the idea of expected signature, Ni et al. (2020; 2021) designed Sig-Wasserstein GAN by directly minimizing the signature Wasserstein-1 distance,

$$\text{Sig-W}_1(\mu, \nu) := |\mathbb{E}_{X\sim\mu}[S(X)] - \mathbb{E}_{X\sim\nu}[S(X)]|,$$

where μ and ν are two distributions of time series corresponding to real data and fake data, S is the signature map, and $|\cdot|$ is the l_2 norm. For practical use, we approximate the infinite sequence S by truncating signatures up to some finite order m, i.e.,

$$\text{Sig-W}_1^m(\mu, \nu) := |\mathbb{E}_\mu[S^m(X)] - \mathbb{E}_\nu[S^m(X)]|. \quad (4)$$

The higher the truncation order m, the more information the signature can capture. However, the number of terms in the truncated signature will grow exponentially and become costly when the time series data is high-dimensional.

Neural CDEs. Neural controlled differential equations are the second candidate for the discriminator when the underlying time series is of high dimension. This is also the discriminator used in (Kidger et al., 2021). Let $D_\phi : X \mapsto R$ be a Neural CDE discriminator where ϕ denotes the network parameters. The training goal is to solve the following optimization problem for the generator

$$\min_\theta \mathbb{E}_{\xi,B}\big[D_\phi\big(G_\theta(\xi, B, \tilde{X})\big)\big],$$

and the following one for the discriminator

$$\max_\phi \big\{\mathbb{E}_{\xi,B}\big[D_\phi\big(G_\theta(\xi, B, \tilde{X})\big)\big] - \mathbb{E}_{\tilde{X}}\big[D_\phi(\tilde{X})\big]\big\}. \quad (5)$$

Compared to only using the Neural CDEs as the discriminator, we notice that a combination of Neural CDEs and lower-order signature Wasserstein-1 distance as the discriminator works better for the third numerical example below. That is, the generator is optimized with respect to

$$\min_\theta \big\{\mathbb{E}_{\xi,B}\big[D_\phi\big(G_\theta(\xi, B, \tilde{X})\big)\big] + \text{Sig-W}_1^m\big(\text{Law}(\tilde{X}), \text{Law}\big(G_\theta(\xi, B, \tilde{X})\big)\big)\big\}. \quad (6)$$

Remark that DC-GANs can work with different discriminators, and here we choose to use neural CDEs and SigWGAN as the discriminators. The pseudo-algorithm of the overall training strategy is summarized in Algorithm 2.

Algorithm 2 The Training Phase

Input: real data $\{\tilde{X}(\omega_i)\}_{i=1}^M$, boolean variable cde, total epochs E, signature truncation order m;
for $e = 1$ **to** E **do**
  Generate independent copies of initials and Brownian motions $(\xi(\omega_i), B(\omega_i))_{i=1}^M$;
  Generate fake data $\{X(\omega_i)\}_{i=1}^M$ by

$$X(\omega_i) = G_\theta(\xi(\omega_i), B(\omega_i), \tilde{X}(\omega_i));$$

  if cde is True **then**
    Compute the loss (5) and its gradients w.r.t. ϕ;
    Compute the loss (6) and its gradients w.r.t. θ;
    Update θ by stochastic gradient descent optimiser;
    Update ϕ by stochastic gradient ascent optimiser;
  else
    Compute the loss (4) and its gradients w.r.t. θ;
    Update θ by stochastic gradient descent optimiser;
  end if
end for
Output: Generator G_θ.

4. Experiments

We present the performance of the proposed DC-GANs on four different datasets, including stochastic opinion dynamics, network dynamics from neural science, and real-world stock data and energy consumption data. In all cases, we set $q = 10$, i.e., "walk" along the chain for ten steps during the decorrelating phase. Other hyperparameters for neural network training can be found in Ap-

pendix C for details. The example implementation of DC-GAN is available at https://github.com/mmin0/DirectedChainSDE.

Benchmarks & Evaluation. The first two synthetic datasets are generated by SDEs, the third real-world data set of stock price time series was extracted from Yahoo Finance[1], and the fourth real-world energy consumption data were obtained from Ireland's open data portal[2]. We compare our results by DC-GANs with SigWGAN, CTFP, Neural SDEs and Transformer-based TTS-GANs, and DC-GANs give much better accuracy under discriminative, predictive, and maximum mean discrepancy (MMD) metrics detailed below. We also provide independence metrics to show that our decorrelating and branching scheme can resolve the independence problem. We also test over different discriminators, and show the flexibility of choosing the one that brings better performance or has a faster running time.

4.1. Metrics

Marginal Distribution & MMD. For the first two examples, we plot histograms to compare their marginal distributions at several time stamps. To measure the goodness of fitting for time series, we use maximum mean discrepancy (MMD) induced by the expected signature given in (4).

Discriminative Metric. To quantitively measure the similarity between the fake data generated by DC-GANs and real data, we train a post-hoc time series classifier by optimizing a two-layer LSTM to discriminate original and fake sequential data. The fake data is labeled *nonreal* and the original data is labeled *real*. The worse discriminative ability of the post-hoc time series classifier implies the better performance of the time series generator. Our discriminative score is calculated as the absolute difference between 0.5 and predicting accuracy on testing data, thus a smaller score indicates a better generator.

Predictive Metric. Typically, a useful time series dataset contains temporal evolution information, and we can predict the future given past data. We expect that DC-GANs can capture this temporal dynamic property accurately from the original data. To this end, we train an auxiliary two-layer LSTM sequential predictor on the generated time series and test this post-hoc predictor on the original time series. The predictive score is calculated as the L^1 distance between predicted sequences and true sequences on testing data (the real data), with smaller scores for better generators.

Independence Metric. It is crucial for success to show that our algorithm can address the independence problem. As an independence metric, we use

$$\rho(x, y) := \sup_{t\in[0,T]} \|\rho(x_t, y_t)\|_1, \tag{7}$$

where $x, y \in L^2(\Omega \times [0, T], \mathbb{R}^N)$ and $\rho(x_t, y_t)$ represents the cross-correlation matrix between random vectors x_t, y_t. Smaller $\rho(x, y)$ means less correlation between real data x and generated data y.

These metric scores are used to measure the algorithms' discriminative ability, predictive ability, and the distance and correlation between the generated data and the true distribution. A decrease in the metric score suggests an improvement in the quality of the generated data. Furthermore, the decrease in independence score is supported by Theorem 2.2 and Theorem B.5 in Appendix.

All experiments are run over ten different random seeds, and we report the mean and standard deviation (in the parentheses) for all metrics in Tables 1–4. We give more details on how all these metrics are implemented in Appendix C.1.

4.2. Example 1: Stochastic Opinion Dynamics

We first consider stochastic opinion dynamics modeled by the following MV-SDE

$$\mathrm{d}Y_t = -\left[\int_{\mathbb{R}} \varphi_\theta(\|Y_t - y\|)(Y_t - y)\ \mu_t(\mathrm{d}y)\right]\mathrm{d}t + \sigma\,\mathrm{d}W_t,$$

where φ_θ is a interaction kernel with $\theta_1, \theta_2 > 0$,

$$\varphi_\theta(r) = \begin{cases} \theta_1 \exp\left(-\frac{0.01}{1-(r-\theta_2)^2}\right), & r > 0, \\ 0, & r \le 0, \end{cases}$$

and $\mu_t = \mathrm{Law}(Y_t)$ denotes the distribution of Y_t. One can interpret θ_1 as a scale parameter that characterizes the intensity of the attraction between entities, and θ_2 as the range parameter that determines the distance, within which an entity must be of one another in order to interact. This model is widely used in many disciplines, from flocking and swarming behaviors in biology (where Y_t is the position) to public opinion evolution in social science (where Y_t is the opinion towards a topic). We refer to Motsch & Tadmor (2014) for further details.

We choose $\theta_1 = 6$, $\theta_2 = 0.2$, $\sigma = 0.1$, $T = 1$, $\Delta t = 0.01$, and generate 8192 paths. The distribution μ_t is approximated by the empirical distribution of 8192 samples. These samples are used to produce the blue density in Figure 1, where a clear shift in distribution from unimodality to bimodality is observed.

We first compare with the Neural SDEs method (Kidger et al., 2021). Figure 1 gives the comparison of the marginal distributions at $t = 0.1, 0.3, 0.5, 0.7, 0.9, 1.0$. One can see

[1] https://finance.yahoo.com/quote/GOOG?p=GOOG&.tsrc=fin-srch.

[2] https://data.gov.ie/dataset?theme=Energy.

that DC-GANs can accurately capture the bimodal distribution in general, but the Neural SDE method can not. Under the MMD metric (4), the discrepancy of DC-GANs is 0.07, while the Neural SDEs give 0.12. More comparisons with SigWGAN, CTFP, Neural SDEs and TTS-GAN under discriminative, MMD, and independence metrics are provided in Table 1. Our proposed DC-GANs have a smaller discriminative score, and an independence score comparable with the ones produced by the Neural SDE generator, SigWGAN, CTFP and TTS-GAN, all of which generate purely independent samples. Therefore, we conclude that DC-GANs can produce fake data closer to the real data without independence issues.

4.3. Example 2: Stochastic FitzHugh-Nagumo Model

FitzHugh-Nagumo model is a standard model from neuroscience (Baladron et al., 2012; Reisinger & Stockinger, 2022), used to describe the neurons' interacting spiking. Mathematically, for N neurons and P different neuron populations, and $i \in \{1, \dots, N\}$, we denote by $p(i) = \alpha, \alpha \in \{1, \dots, P\}$ the population of i-th particle that belongs to. The state vector of neural i, $(X_t^{i,N})_{t\in[0,T]} = (V_t^{i,N}, w_t^{i,N}, y_t^{i,N})_{t\in[0,T]}$, satisfies the SDE,

$$\begin{aligned} \mathrm{d}X_t^{t,N} &= f_\alpha(t, X_t^{t,N})\,\mathrm{d}t + g_\alpha(t, X_t^{t,N}) \begin{bmatrix} \mathrm{d}W_t^i \\ \mathrm{d}W_t^{i,y} \end{bmatrix} \\ &\quad + \sum_{\gamma=1}^{P} \frac{1}{N_\gamma} \sum_{j,p(j)=\gamma} \Big(b_{\alpha\gamma}(X_t^{i,N}, X_t^{j,N})\,\mathrm{d}t \\ &\quad + \beta_{\alpha\gamma}(X_t^{i,N}, X_t^{j,N})\,\mathrm{d}W_t^{i,\gamma} \Big), \end{aligned}$$

where V denotes a short, nonlinear elevation of membrane voltage, w denotes a slower, linear recovery variable, N_γ denotes the number of neurons in the population γ. We defer more details about model description and training data generation to Appendix C.2.

The FitzHugh-Nagumo system is an example of a relaxation oscillator, and exhibits a characteristic excursion in phase space, before the variables V and w relax back to their rest values. As a result, their distributions are typically multimodal distributed; see Figure 5 in Appendix C.2.

Figure 3 depicts the differences of their joint marginal densities between generated time series and training (real) time series on channels 1 and 3 at $t = 0.1, 0.3, 0.5, 0.7, 0.9, 1.0$. The darker the color the smaller the differences, thus the closer the distribution and indicating a better generator. It can be observed that DC-GANs produce less difference in joint marginal densities at multiple time stamps. Under discriminative, predictive, and MMD metrics, DC-GANs give better samples than SigWGAN, CTFP, Neural SDEs, and TTS-GAN consistently; see Table 2. In particular, fake samples produced by DC-GANs are almost indistinguishable for a two-layer LSTM classifier after exhaustive training. By the comparison using MMD, one can see that DC-GANs generate fake samples with distributions significantly closer to real data than the other three methods. The independence scores given by (7) are nearly indistinguishable.

4.4. Example 3: Stock Price Time Series (Real Data)

The third example is Google stock prices from 2004 to 2019, extracted from Yahoo Finance. Sequences of stock prices are known as continuous time series data with unknown distributions, and can even be non-Markovian. Our data have six channels, volume and high, low, opening, closing, and adjusted closing prices. Among all, the first five channels are multimodal. The combined discriminator (6) (Neural CDE and Sig-W_1) is used in GAN for this experiment, and we list the comparison results in Table 3. One can see that DC-GANs outperform SigWGAN, CTFP, TimeGAN, Neural SDEs and TTS-GAN under all three metrics.

4.5. Example 4: Energy Consumption Data (Real Data)

We download the Energy Consumption data from Ireland's open data portal, and choose four electric and gas consumption time series from 02/2011–02/2013, where channels 1,3, and 4 exhibit multimodal features. We list the comparison results in Table 4, which shows consistent advantages of DC-GANs compared with other methods under different metrics as in previous examples. Notice that DC-GANs can be used with both Neural CDEs (NCDE) and Signature Wasserstein (SigW) discriminators, and in this example, DC-GANs with SigW as the discriminator present better performance and have a faster running time.

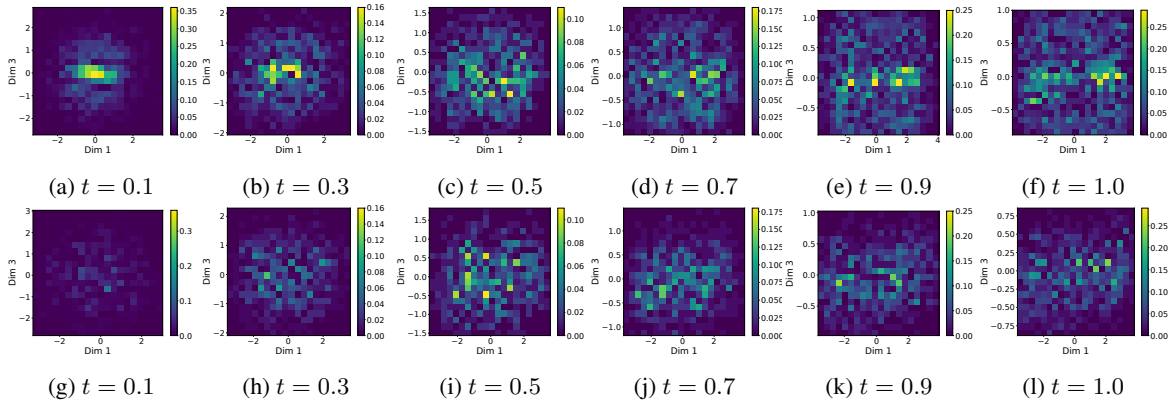
4.6. Dependence Elimination & Ablation Study.

To demonstrate the effectiveness of removing dependence in the *Decorrelating and Branching Phase* (Section 3.1), we compare the independence score (7) with choices of $q = 2$ (basic model) and $q = 10$ (DC-GANs), and present the results in Table 5. A smaller score indicates better independence. The large differences observed in all four experiments suggest that the proposed scheme significantly reduces correlation in the directed chain generator. We anticipate that this approach can be employed in other directed chain-related methods to effectively mitigate strong dependence on generated data.

We also conduct an ablation study on the discriminator by using an ordinary LSTM as the discriminator and implementing Wasserstein-GAN for comparison. The results, summarized in Table 6, indicate that DC-GANs (shown in Tables 1-4) outperform Wasserstein-GAN with an LSTM discriminator in both accuracy and speed.

Table 1. Stochastic Opinion Dynamics (Example 1). The scores are computed for SigWGAN, CTFP, Neural SDEs, TTS-GAN and DC-GANs under different metrics. The numbers in the parenthesis are the corresponding standard deviations of each score. Note that a smaller value means a better approximation, which indicates the DC-GANs provide more accurate fake data with compared independence and running time.

METHOD	DISCRIMINATIVE	MMD	INDEPENDENCE	TIME (MIN)
SIGWGAN	0.213 (0.01)	0.328 (0.004)	0.009(0.004)	6.55
CTFP	0.131 (0.02)	0.281 (0.005)	0.010(0.003)	5.58
NEURAL SDES	0.045 (0.025)	0.122 (0.003)	0.007 (0.005)	7.07
TTS-GAN	0.127(0.014)	0.176(0.003)	0.008(0.003)	15.6
DC-GANS	**0.028 (0.019)**	**0.07 (0.003)**	0.009 (0.004)	6.82



(a) $t = 0.1$ (b) $t = 0.3$ (c) $t = 0.5$ (d) $t = 0.7$ (e) $t = 0.9$ (f) $t = 1.0$

(g) $t = 0.1$ (h) $t = 0.3$ (i) $t = 0.5$ (j) $t = 0.7$ (k) $t = 0.9$ (l) $t = 1.0$

Figure 3. Stochastic FitzHugh-Nagumo Model (Example 2). Figures (a)-(f) are generated by Neural SDEs, and Figures (g)-(l) are generated by DC-GANs. They show their joint marginal densities differences between estimated time series and real-time series on channels 1 (Dim 1) and 3 (Dim 3) at $t \in \{0.1, 0.3, 0.5, 0.7, 0.9, 1.0\}$. Darker color means a smaller difference, and thus a better fitting. One can observe that DC-GANs produce less difference in joint marginal densities at multiple time stamps.

5. Conclusion

We propose a novel time series generator, DC-GANs, motivated by the study of Detering et al. (2020); Ichiba & Min (2022) on directed chain SDEs (DC-SDEs). Compared to more complicated graph systems, we find from numerical examples that the directed chain systems exhibit promising ability in fitting time series of multimodal probability distributions. We prove in theory that DC-GANs have the same flexibility as the Neural SDEs in capturing marginal distributions, and DC-GANs naturally embrace the non-Markovian property in the topological structure, if needed. We also prove that the correlation of the generated path decays exponentially fast as the graph distance of the generated path from the original data becomes large under some mild assumptions, and hence, the lack-of-independence problem can be overcome by walking along the directed chain. We present four numerical examples, two synthetic datasets generated by the SDEs, and two real-world data of stock price and energy consumption, and show that DC-GANs have a better performance than SigWGAN, CTFP, Neural SDEs, TimeGAN and TTS-GAN, with the comparable independence property. We remark that the DC-GANs algorithm can also work with irregular data (i.e., the sample paths may have data sampled on different time grids), which may happen in healthcare applications.

Potential Societal Impact. The proposed DC-GANs in this paper offer a fast and flexible generative adversarial network method for machine learning research areas, such as biology, economics, environmental science, finance, medicine, and more, where path-dependent analysis is critical. They have the potential to contribute to research areas where sequential data is scarce or missing. Furthermore, the strong predictive power of DC-GANs demonstrated in the energy consumption example (Example 4) can aid in the development of energy management and help reduce energy waste. As DC-GANs are primarily used to generate time-series data or sequential data, rather than fake faces, the usual negative social impact associated with creating fake social media accounts to spam would not be a concern in this context.

Acknowledgements

R.H. was partially supported by the NSF grant DMS-1953035, and the Early Career Faculty Acceleration funding

Table 2. Stochastic FitzHugh-Nagumo Model (Example 2). The scores are computed for SigWGAN, CTFP, Neural SDEs, TTS-GAN and DC-GANs under different metrics. Note that a smaller value means a better approximation. Parenthesized numbers are standard deviations.

METHOD	DISCRIMATIVE	PREDICTIVE	MMD	INDEPENDENCE	TIME (MIN)
SIGWGAN	0.126 (0.04)	0.44 (0.001)	0.737 (0.01)	0.0083(0.0024)	9.63
CTFP	0.275 (0.05)	0.501 (0.004)	1.095 (0.02)	0.0088(0.0023)	6.88
NEURAL SDES	0.20 (0.003)	0.44 (0.000)	0.97 (0.02)	0.0085 (0.0023)	8.25
TTS-GAN	0.258(0.02)	0.45(0.001)	0.96(0.04)	0.0082(0.002)	16.3
DC-GANS	**0.01 (0.009)**	**0.439 (0.000)**	**0.47 (0.02)**	0.0085 (0.0027)	8.13

Table 3. Stocks Price Time Series (Example 3). The scores are computed for SigWGAN, CTFP, TimeGAN, Neural SDEs, TTS-GAN and DC-GANs under different metrics. Note that a smaller value means a better approximation. Parenthesized numbers are standard deviations.

MODEL	DISCRIMINATIVE	PREDICTIVE	MMD	INDEPENDENCE	TIME (MIN)
SIGWGAN	0.183 (0.03)	0.060 (0.004)	0.121 (0.011)	0.012(0.004)	4.13
CTFP	0.256 (0.05)	0.138 (0.006)	0.187 (0.009)	0.013(0.005)	6.40
TIMEGAN	0.102 (0.021)	0.038 (0.001)	0.0220 (0.007)	0.011 (0.005)	>660
NEURAL SDES	0.085 (0.028)	0.048 (0.001)	0.0193 (0.008)	0.011 (0.006)	9.93
TTS-GAN	0.093(0.022)	0.041(0.001)	0.023(0.007)	0.010(0.004)	19.2
DC-GANS	**0.045 (0.015)**	**0.036 (0.000)**	**0.0133 (0.005)**	0.013 (0.006)	9.53

Table 4. Energy Consumption Data from Ireland's open data portal (Example 4). The scores are computed for SigWGAN, CTFP, Neural SDEs, and DC-GANs under different metrics. Note that a smaller value means a better approximation. Parenthesized numbers are standard deviations.

METHOD	DISCRIMINATIVE	PREDICTIVE	MMD	INDEPENDENCE	TIME(MIN)
SIGWGAN	0.368 (0.09)	0.159 (0.002)	0.135 (0.006)	0.022(0.007)	9.47
CTFP	0.487 (0.01)	0.185 (0.001)	0.558 (0.006)	0.021(0.008)	8.52
NEURAL SDES	0.413 (0.06)	0.172 (0.004)	0.126 (0.004)	0.022(0.006)	9.73
TTS-GAN	0.394(0.04)	0.167(0.003)	0.183(0.008)	0.022(0.006)	17.4
DC-GANS (W/ NCDE)	0.322 (0.12)	0.155 (0.006)	0.077 (0.003)	0.029(0.007)	23.44
DC-GANS (W/ SIGW)	**0.310 (0.09)**	**0.151 (0.008)**	**0.075 (0.003)**	0.033(0.008)	9.38

Table 5. The comparison of independence scores between a basic model (with $q = 2$) and DC-GANs (with $q = 10$). A smaller score indicates better independence.

EXPERIMENTS	OPINION (EXP.1)	FITZ-NAG (EXP.2)	STOCK (EXP.3)	ENERGY(EXP.4)
$q = 2$ (BASIC)	0.078(0.013)	0.052(0.009)	0.130(0.024)	0.119(0.017)
$q = 10$ (DC-GANS)	0.009(0.004)	0.0085(0.0027)	0.013(0.006)	0.033(0.008)

Table 6. The scores computed when the discriminator is replaced by an ordinary WGAN. Parenthesized numbers are standard deviations. Compared to Tables 1-4, DC-GANs outperform Wasserstein-GAN with an LSTM discriminator in both accuracy and speed.

EXPERIMENTS	DISCRIMINATIVE	PREDICTIVE	MMD	INDEPENDENCE	TIME (MIN)
STOCH. OPINION	0.041(0.016)	-	0.123(0.004)	0.008(0.003)	19.5
FITZHUGH-NAGUMO	0.09(0.01)	0.441(0.000)	0.66 (0.07)	0.0083(0.0022)	26.2
GOOGLE STOCK	0.082 (0.024)	0.059 (0.002)	0.0176 (0.006)	0.010 (0.005)	20.85
ENERGY CONSUMPTION	0.343 (0.13)	0.161(0.004)	0.093 (0.004)	0.024(0.004)	21.17

and the Regents' Junior Faculty Fellowship at the University of California, Santa Barbara. T.I. was partially supported by NSF grant DMS-2008427. The authors are grateful to the reviewers for their valuable and constructive comments.

References

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *International conference on machine learning*, pp. 214–223. PMLR, 2017.

Baladron, J., Fasoli, D., Faugeras, O., and Touboul, J. Mean-field description and propagation of chaos in networks of hodgkin-huxley and fitzhugh-nagumo neurons. *The Journal of Mathematical Neuroscience*, 2(1):1–50, 2012.

Boedihardjo, H., Geng, X., Lyons, T., and Yang, D. The signature of a rough path: uniqueness. *Advances in Mathematics*, 293:720–737, 2016.

Brophy, E., Wang, Z., She, Q., and Ward, T. Generative adversarial networks in time series: A systematic literature review. *ACM Computing Surveys (CSUR)*, 2022.

Brunick, G. and Shreve, S. Mimicking an itô process by a solution of a stochastic differential equation. *The Annals of Applied Probability*, 23(4):1584–1628, 2013.

Carmona, R., Cooney, D. B., Graves, C. V., and Lauriere, M. Stochastic graphon games: I. the static case. *Mathematics of Operations Research*, 47(1):750–778, 2022.

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Chevyrev, I. and Lyons, T. Characteristic functions of measures on geometric rough paths. *The Annals of Probability*, 44(6):4049–4082, 2016.

Chevyrev, I. and Oberhauser, H. Signature moments to characterize laws of stochastic processes. *Journal of Machine Learning Research*, 23(176):1–42, 2022. URL http://jmlr.org/papers/v23/20-1466.html.

Cuchiero, C., Khosrawi, W., and Teichmann, J. A generative adversarial network approach to calibration of local stochastic volatility models. *Risks*, 8(4):101, 2020.

Deng, R., Chang, B., Brubaker, M. A., Mori, G., and Lehrmann, A. Modeling continuous stochastic processes with dynamic normalizing flows. *Advances in Neural Information Processing Systems*, 33:7805–7815, 2020.

Detering, N., Fouque, J.-P., and Ichiba, T. Directed chain stochastic differential equations. *Stochastic Processes and their Applications*, 130(4):2519–2551, 2020.

dos Reis, G., Engelhardt, S., and Smith, G. Simulation of McKean–Vlasov SDEs with super-linear growth. *IMA Journal of Numerical Analysis*, 42(1):874–922, 2021.

Dyer, J., Cannon, P., and Schmon, S. M. Approximate bayesian computation with path signatures. *arXiv preprint arXiv:2106.12555*, 2021.

Feng, Y., Fouque, J.-P., and Ichiba, T. Linear-quadratic stochastic differential games on random directed networks. *Journal of mathematics and statistical science*, 7 (3), 2021a.

Feng, Y., Fouque, J.-P., and Ichiba, T. Linear-quadratic stochastic differential games on directed chain networks. *Journal of mathematics and statistical science*, 7(2), 2021b.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Gyöngy, I. Mimicking the one-dimensional marginal distributions of processes having an itô differential. *Probability theory and related fields*, 71(4):501–516, 1986.

Han, J., Jentzen, A., and E, W. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, 115(34): 8505–8510, 2018.

Hanshu, Y., Jiawei, D., Vincent, T., and Jiashi, F. On robustness of neural ordinary differential equations. In *International Conference on Learning Representations*, 2019.

Ichiba, T. and Min, M. Smoothness of directed chain stochastic differential equations. *arXiv preprint arXiv:2202.09354*, 2022.

Karatzas, I. and Shreve, S. *Brownian motion and stochastic calculus*, volume 113. Springer Science & Business Media, 2012.

Kidger, P., Bonnier, P., Perez Arribas, I., Salvi, C., and Lyons, T. Deep signature transforms. *Advances in Neural Information Processing Systems*, 32, 2019.

Kidger, P., Morrill, J., Foster, J., and Lyons, T. Neural controlled differential equations for irregular time series. *Advances in Neural Information Processing Systems*, 33: 6696–6707, 2020.

Kidger, P., Foster, J., Li, X., and Lyons, T. J. Neural sdes as infinite-dimensional gans. In *International Conference on Machine Learning*, pp. 5453–5463. PMLR, 2021.

Lacker, D. and Soret, A. A case study on stochastic games on large graphs in mean field and sparse regimes. *Mathematics of Operations Research*, 47(2):1530–1565, 2022.

Lacker, D., Ramanan, K., and Wu, R. Locally interacting diffusions as markov random fields on path space. *Stochastic Processes and their Applications*, 140:81–114, 2021.

Li, X., Wong, T.-K. L., Chen, R. T., and Duvenaud, D. Scalable gradients for stochastic differential equations. In *International Conference on Artificial Intelligence and Statistics*, pp. 3870–3882. PMLR, 2020.

Li, X., Metsis, V., Wang, H., and Ngu, A. H. H. Tts-gan: A transformer-based time-series generative adversarial network. In Michalowski, M., Abidi, S. S. R., and Abidi, S. (eds.), *Artificial Intelligence in Medicine*, pp. 133–143, Cham, 2022. Springer International Publishing.

Lyons, T. and Qian, Z. *System control and rough paths*. Oxford University Press, 2002.

Lyons, T. J., Caruana, M., and Lévy, T. *Differential equations driven by rough paths*. Springer, 2007.

Massaroli, S., Poli, M., Park, J., Yamashita, A., and Asama, H. Dissecting neural odes. *Advances in Neural Information Processing Systems*, 33:3952–3963, 2020.

Min, M. and Hu, R. Signatured deep fictitious play for mean field games with common noise. In *International Conference on Machine Learning*, pp. 7736–7747. PMLR, 2021.

Min, M. and Ichiba, T. Convolutional signature for sequential data. *Digital Finance*, pp. 1–26, 2022.

Morrill, J., Salvi, C., Kidger, P., and Foster, J. Neural rough differential equations for long time series. In *International Conference on Machine Learning*, pp. 7829–7838. PMLR, 2021.

Motsch, S. and Tadmor, E. Heterophilious dynamics enhances consensus. *SIAM review*, 56(4):577–621, 2014.

Ni, H., Szpruch, L., Wiese, M., Liao, S., and Xiao, B. Conditional sig-wasserstein gans for time series generation. *arXiv preprint arXiv:2006.05421*, 2020.

Ni, H., Szpruch, L., Sabate-Vidales, M., Xiao, B., Wiese, M., and Liao, S. Sig-wasserstein gans for time series generation. In *Proceedings of the Second ACM International Conference on AI in Finance*, pp. 1–8, 2021.

Quaglino, A., Gallieri, M., Masci, J., and Koutník, J. Snode: Spectral discretization of neural odes for system identification. In *International Conference on Learning Representations*, 2019.

Reisinger, C. and Stockinger, W. An adaptive euler–maruyama scheme for mckean–vlasov sdes with superlinear growth and application to the mean-field fitzhugh–nagumo model. *Journal of Computational and Applied Mathematics*, 400:113725, 2022.

Salvi, C., Lemercier, M., Liu, C., Horvath, B., Damoulas, T., and Lyons, T. Higher order kernel mean embeddings to capture filtrations of stochastic processes. *Advances in Neural Information Processing Systems*, 34:16635–16647, 2021.

Sharma, S. K., Kumar, S., et al. Suppression of multimodality in inter-spike interval distribution: Role of external damped oscillatory input. *IEEE Transactions on NanoBioscience*, 17(3):329–341, 2018.

Smith, L. D. and Gottwald, G. A. Chaos in networks of coupled oscillators with multimodal natural frequency distributions. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(9):093127, 2019.

Tsang, A. and Larson, K. Opinion dynamics of skeptical agents. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pp. 277–284, 2014.

Tzen, B. and Raginsky, M. Neural stochastic differential equations: Deep latent gaussian models in the diffusion limit. *arXiv preprint arXiv:1905.09883*, 2019a.

Tzen, B. and Raginsky, M. Theoretical guarantees for sampling and inference in generative models with latent diffusions. In *Conference on Learning Theory*, pp. 3084–3114. PMLR, 2019b.

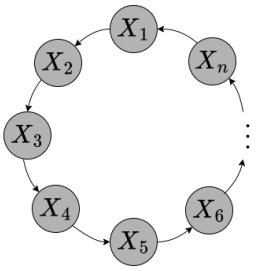
Yoon, J., Jarrett, D., and Van der Schaar, M. Time-series generative adversarial networks. *Advances in neural information processing systems*, 32, 2019.

Zhang, T., Yao, Z., Gholami, A., Gonzalez, J. E., Keutzer, K., Mahoney, M. W., and Biros, G. Anodev2: A coupled neural ode framework. *Advances in Neural Information Processing Systems*, 32, 2019.

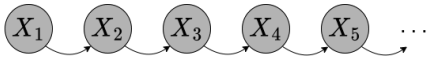
A. Preliminaries on Directed Chain SDEs

In this appendix, we give an intuitive explanation of how the limit of an n-coupled SDE system leads to the DC-SDE. As before, we use X_s, X_t to denote the state of X at time s and t, respectively. With no subscript, e.g., by X, we mean the whole path from $t = 0$ to T. With a little abuse of notations, we use X_i, X_{i+1} to denote the i-th or $(i + 1)$-th node, and with two subscripts, e.g., $X_{i,t}$, it represents the state value of the i-th node at time t.

Let us start with a system of n-coupled SDEs, which approximates a generic directed chain SDE when n goes infinity. An illustration of their chain-like coupling is given in Figure 4(a). Each node X_i satisfies an SDE, which also depends on the node pointing to it. More specifically, for $i \leq n - 1$, X_{i+1} depends on X_i, and we say the i-th node is the neighborhood of the $(i + 1)$-th node; the n-th node affects the first one, yielding a circular structure. The dependence is determined in a homogeneous manner over the whole system of particles. Such a circular chain structure forces every node to be identically distributed. When n goes to infinity, the circle chain is equivalent to a non-circular chain with the *distribution constraint* (see Definition 2.1), and we get the abstract DC-SDE. We refer interested readers to Detering et al. (2020); Ichiba & Min (2022) for more details.



(a) Circular structure in the n-coupled SDEs



(b) None-Circular structure in the DC-SDEs

Figure 4. Illustrative Directed Chain Structure. Each node X_i satisfies an SDE, which also depends on the node pointing to it.

B. Additional Theorems and Proofs

B.1. Property of Signatures

We provide some related properties and theorems of signatures. Firstly, we justify the validity of truncating signatures in Section 2.2. The signature of a path is an infinite series of iterated integrals, which can be used to represent the path. Practically, one can only deal with finite sequences, thus requiring truncating signatures up to a finite order (called the signature depth). This truncation order is, in general, determined by its factorial decay property, which is indicated in the following extension theorem. For the formal definition of control ω and β, we refer interested readers to the book by Lyons & Qian (2002).

Theorem B.1 (Extension Theorem, Lyons & Qian (2002, Theorem 3.7))**.** *Let $p \geq 1$ be a real number and $n \geq 1$ an integer with $n \geq \lfloor p \rfloor$. Denote $\mathbb{X} : \Delta_T \to T^n(\mathbb{R}^d)$ as a multiplicative functional with finite p-variation controlled by a control ω. Then there exists a unique extension of $\mathbb{X}$ to a multiplicative functional $\Delta_T \to T((\mathbb{R}^d))$ which possesses finite p-variation.*

More precisely, for every $m \geq \lfloor p \rfloor + 1$, there exists a unique continuous function $\mathbb{X}^m : \Delta_T \to (\mathbb{R}^d)^{\otimes m}$, such that

$$(s,t) \to \mathbb{X}_{s,t} = \left(1, \mathbb{X}^1_{s,t}, \ldots, \mathbb{X}^{\lfloor p \rfloor}_{s,t}, \ldots, \mathbb{X}^m_{s,t}, \ldots\right) \in T((\mathbb{R}^d))$$

is a multiplicative functional with finite p-variation controlled by ω, i.e.,

$$\|\mathbb{X}^i_{s,t}\| \leq \frac{\omega(s,t)^{\frac{i}{p}}}{\beta(\frac{i}{p})!} \quad \forall i \geq 1, \quad \forall (s,t) \in \Delta_T. \tag{8}$$

The extension theorem states that, for any multiplicative functional with finite p-variation, we can extend it to an infinite sequence. In particular, the signature is one of such objects for some special multiplicative functionals (what we call geometric rough paths). In most real-world applications, time series data are interpolated linearly and hence fall into the case of $p = 1$, i.e., paths with bounded variation. In certain financial applications or the first two examples in this paper, we have semi-martingales that fall into the cases of $p \in (2, 3)$, that is, the geometric rough paths with finite p-variation. Their signatures are all well-defined. The factorial decay property is implied by equation (8).

As a feature map of sequential data, the signature has a universality detailed in the following theorem.

Theorem B.2 (Universality)**.** *Let $p \geq 1$ and $f : \mathcal{V}^p([0, T], \mathbb{R}^d) \to \mathbb{R}$ be a continuous function in paths. For any compact set $K \subset \mathcal{V}^p([0, T], \mathbb{R}^d)$, if $S(x)$ is a geometric rough path for any $x \in K$, then for any $\epsilon > 0$ there exist $M > 0$ and a linear functional $l \in T((\mathbb{R}^d))^*$ such that*

$$\sup_{x \in K} |f(x) - \langle l, S(x) \rangle| < \epsilon. \tag{9}$$

Given that signature is well-defined and with finite expectation, we call $\mathbb{E}[S(X)]$ the expected signature of X. Intuitively, the expected signature serves the moment-generating function, which can characterize the law induced by a stochastic process under some regularity conditions. More precisely, an immediate consequence of Proposition 6.1 in Chevyrev & Oberhauser (2022) on the uniqueness of the expected signature is summarized in the below theorem:

Theorem B.3. *Let X, Y be two random variables of geometric rough paths such that $\mathbb{E}[S(X)]] = \mathbb{E}[S(Y)]$ and $\mathbb{E}[S(X)]$ has an infinite radius of convergence, then X, Y have the same distribution.*

B.2. Proof of Theorem 2.1

We first restate Theorem 2.1 formally. Without loss of generality, we treat the time-homogeneous case, i.e., μ and σ are independent of t. Our proof relies on constructing the forward equations characterizing marginal distributions of both SDEs and directed chain SDEs, thus can be easily generalized to time-dependence cases. The forward equation associated with directed chain SDEs has been constructed by Ichiba & Min (2022) and will be used directly in our proof.

Theorem B.4. *Let $Y \in L^2(\Omega \times [0, T], \mathbb{R}^N)$ be an N-dimensional stochastic process with the following dynamics*

$$\mathrm{d}Y_t = \mu(Y_t)\,\mathrm{d}t + \sigma(Y_t)\,\mathrm{d}B_t^y, \quad Y_0 = \xi^y,$$

where B^y is a standard d-dimensional Brownian motion, and $\mu : \mathbb{R}^N \to \mathbb{R}^N, \sigma : \mathbb{R}^N \to \mathbb{R}^{N \times d}$ are Borel measurable functions with Lipschitz and linear growth conditions. Then, there exist functions V_0 and V_1 such that the process X has the same marginal distribution as Y for all $t \in [0, T]$, where X is described by the following directed chain SDEs with an initial position ξ as an independent copy of ξ^y,

$$\begin{aligned} &\mathrm{d}X_t = V_0(X_t, \tilde{X}_t)\,\mathrm{d}t + V_1(X_t, \tilde{X}_t)\,\mathrm{d}B_t^y, \quad X_0 = \xi, \\ &\textit{subject to: } \mathrm{Law}(X_t, 0 \leq t \leq T) = \mathrm{Law}(\tilde{X}_t, 0 \leq t \leq T). \end{aligned}$$

Proof. Let $g \in \mathcal{C}^2(\mathbb{R}^N)$ be a twice continuously differentiable function. To characterize marginal distributions of the SDE solution Y for all $t \in [0, T]$, we use the Kolmogorov forward equations. Define $u(t, x) := \mathbb{E}[g(Y_t)|Y_0 = x]$, it is the solution of the following Cauchy problem

$$(\partial_t - \mathcal{L})u(t, x) = 0, \tag{10}$$

$$u(0, x) = g(x). \tag{11}$$

The derivation relies on Itô's formula and can be found in stochastic calculus textbooks, e.g., in Karatzas & Shreve (2012). Here the *infinitesimal operator* $\mathcal{L}$ is given by

$$\mathcal{L}g(x) = \mu(x) \cdot \nabla_x g(x) + \frac{1}{2}\mathrm{Tr}(\sigma\sigma^T(x)\mathrm{Hess}_x g(x)),$$

where $\mathrm{Hess}_x(\cdot)$ denotes the Hessian matrix, and $\mathrm{Tr}(\cdot)$ denotes the matrix trace. In Ichiba & Min (2022, Section 4.5), a similar partial differential equation for the directed chain SDEs is derived, and we here summarize a simpler version without

the mean-field interaction term. Define $v(t,x) := \mathbb{E}[g(X_t)|X_0 = x]$, then v solves

$$(\partial_t - \mathcal{L}^{dc})v(t,x) = 0, \tag{12}$$
$$v(0,x) = g(x). \tag{13}$$

Let $\tilde{\xi}$ be an independent copy of ξ, and the differential operator $\mathcal{L}^{dc}$ is given by

$$\mathcal{L}^{dc}g(x) = \mathbb{E}_{\tilde{\xi}}\left[V_0(x,\tilde{\xi})\cdot\nabla_x g(x) + \frac{1}{2}\mathrm{Tr}(V_1 V_1^T(x,\tilde{\xi})\mathrm{Hess}_x g(x))\right], \tag{14}$$

where $\mathbb{E}_{\tilde{\xi}}$ is the expectation with respect to the distribution of $\tilde{\xi}$. As long as we can match these two operators $\mathcal{L}$ and $\mathcal{L}^{dc}$ with some non-degenerate choices of V_0, V_1, then (10)-(11) and (12)-(13) agree with each other and so do their solutions u and v. To this end, it suffices to choose V_0, V_1 such that

$$\mathbb{E}_{\tilde{\xi}}[V_0(x,\tilde{\xi})] = \mu(x),$$
$$\mathbb{E}_{\tilde{\xi}}[V_1 V_1^\top(x,\tilde{\xi})] = \sigma\sigma^\top(x).$$

A toy example of non-degenerate V_0, V_1 can be $V_0(x,\tilde{\xi}) = \mu(x) + \varphi_1(\tilde{\xi}) - \mathbb{E}_{\tilde{\xi}}[\varphi_1(\tilde{\xi})]$ and $V_1(x,\tilde{\xi})$ such that $V_1 V_1^\top(x,\tilde{\xi}) = \sigma\sigma^\top(x) + \varphi_2(\tilde{\xi}) - \mathbb{E}_{\tilde{\xi}}[\varphi_2(\tilde{\xi})]$ with measurable and integrable functions φ_1, φ_2. □

B.3. Proof of Theorem 2.2

From Ichiba & Min (2022, Proposition 2.1), we have the existence and weak uniqueness of directed chain SDEs. Denote this unique measure flow by

$$m := \mathrm{Law}(X_t, 0 \le t \le T) = \mathrm{Law}(\tilde{X}_t, 0 \le t \le T).$$

This measure can also be understood as a probability distribution on $C([0,T],\mathbb{R}^N)$. Given the Brownian motion path and the neighborhood path, we define a map $\Phi : C([0,T],\mathbb{R}^N) \times C([0,T],\mathbb{R}^d) \to C([0,T],\mathbb{R}^N)$ such that

$$X = \Phi(\tilde{X}; B) \in C([0,T],\mathbb{R}^N)$$

and Φ_t as the projection of Φ onto any specific time stamp, i.e. $X_t \equiv \Phi_t(\tilde{X}; B)$. Then, on a chain-like structure depicted in Figure 4(b) or 2, we write

$$X_q = \Phi(X_{q-1}; B^q) = \Phi(\Phi(X_{q-2}; B^{q-1}); B^q) = \Phi \circ \Phi(X_{q-2}; B^{q-1}, B^q).$$

Namely, X_q is obtained as an output of the composite map $\Phi \circ \Phi$ from the inputs X_{q-2}, B^{q-1} and B^q. Repeating the above equation until tracing back to the first node produces

$$X_q = \Phi \circ \cdots \circ \Phi(X_1; B^2, \ldots, B^q) := \Phi^q(X_1; \mathbf{B}),$$

where $\mathbf{B} = (B^2, \ldots, B^q)$ and $B^2, \ldots, B^q$ are independent d-dimensional Brownian motions. Such a chain-like structure possesses *local Markov property* as pointed out in Proposition 4.6 in Ichiba & Min (2022). Let us denote $X_{t,q} = \Phi_t^q(X_1; \mathbf{B})$. In the proof below, we impose Lipschitz and linear growth conditions on coefficients V_0 and V_1.

Assumption B.1. *For both coefficients V_0 and V_1, there exists a positive constant C_T such that,*

1. *(Lipschitz conditions) for $i = 0, 1$,*

$$|V_i(x_1,y_1) - V_i(x_2,y_2)| \le C_T(|x_1 - x_2| + |y_1 - y_2|);$$

2. *(Linear growth conditions) for $i = 0, 1$,*

$$V_i(x,y) \le C_T(1 + |x| + |y|).$$

The following lemma gives the necessity of having the Brownian motion noises B^j, $j = 2, \ldots, q$ in Φ_t^q, in order to have dependence decay properties.

Lemma B.1. *Suppose Assumption B.1 holds. In the degenerate case, i.e., $V_1 \equiv 0$ and $X_{t,q} = \Phi_t^q(X_1)$, if all the initial conditions $X_{0,1} = X_{0,2} = \cdots = X_{0,q} = \xi$ are identical, then the directed chain SDE satisfy $X_1 = X_2 = \cdots X_q$ in the L^2 sense.*

Proof. We first write our directed chain dynamics in the integral form,

$$X_{t,q} = \xi + \int_0^t V_0(X_{s,q}, X_{s,q-1})\,\mathrm{d}s. \tag{15}$$

Note that the current directed chain system with degenerate V_1 also has unique solutions. By the Lipschitz property on V_0, we compute

$$\begin{aligned}
\mathbb{E}[\sup_{0\le s\le t}|X_{s,q} - X_{s,q-1}|^2] &\le \mathbb{E}\bigg[\sup_{0\le s\le t} 2C_T\int_0^s (|X_{v,q} - X_{v,q-1}|^2 + |X_{v,q-1} - X_{v,q-2}|^2)\,\mathrm{d}v\bigg] \\
&\le C\cdot\mathbb{E}\bigg[\int_0^t \sup_{0\le v\le s}\Big(|X_{v,q} - X_{v,q-1}|^2 + |X_{v,q-1} - X_{v,q-2}|^2\Big)\,\mathrm{d}v\bigg] \\
&\le C\cdot\int_0^t \mathbb{E}[\sup_{0\le v\le s}|X_{v,q} - X_{v,q-1}|^2]\,\mathrm{d}v + C\cdot\int_0^t \mathbb{E}[\sup_{0\le v\le s}|X_{v,q-1} - X_{v,q-2}|^2]\,\mathrm{d}v \\
&\le C\cdot e^{CT}\int_0^t \mathbb{E}[\sup_{0\le v\le s}|X_{v,q-1} - X_{v,q-2}|^2]\,\mathrm{d}v,
\end{aligned}$$

where the third inequality comes from Fubini's theorem and Proposition 2.2 in Ichiba & Min (2022), and the last inequality follows from Gronwall's inequality. Iterating back to the beginning of the chain, we deduce

$$\mathbb{E}[\sup_{0\le s\le T}|X_{s,q} - X_{s,q-1}|^2] \le \frac{TC^{q-1}e^{(q-1)CT}}{(q-1)!}\mathbb{E}[\sup_{0\le s\le T}|X_{s,2} - X_{s,1}|^2].$$

According to the invariance of (joint) distribution (see Detering et al. (2020); Ichiba & Min (2022)), we get

$$\mathbb{E}[\sup_{0\le s\le T}|X_{s,2} - X_{s,1}|^2] \le \frac{TC^{q-1}e^{(q-1)CT}}{(q-1)!}\mathbb{E}[\sup_{0\le s\le T}|X_{s,2} - X_{s,1}|^2].$$

The constant q can be arbitrarily large and hence the above inequality forms a contraction, which implies

$$\mathbb{E}[\sup_{0\le s\le T}|X_{s,2} - X_{s,1}|^2] = 0.$$

We then conclude $X_1 = X_2 = \cdots = X_q$ in the L^2 sense. □

Although the assumption of identical initials in Lemma B.1 is different from the general setting of directed chain SDEs, where initials should be i.i.d, it is consistent in the case that initials are deterministic. Therefore, the existence of non-degenerate V_1 becomes crucial, and we give the following necessary assumptions for factorial dependence decay property.

Definition B.1 ($\mathcal{C}^{k,k}_{b,\mathrm{Lip}}$)**.** *We have the following definition for $\mathcal{C}^{k,k}_{b,\mathrm{Lip}}$:*

(a) We use ∂_x, ∂_y to denote the derivative with respect to the first and second Euclidean variables in V_0, V_1.

(b) Let $V : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}^N$ with components $V^1, \ldots, V^N : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$. We say $V \in \mathcal{C}^{1,1}_{b,\mathrm{Lip}}(\mathbb{R}^N \times \mathbb{R}^N; \mathbb{R}^N)$ if the following is true: for each $i = 1, \ldots, N$, $\partial_x V^i, \partial_y V^i$ exist. Moreover, assume the boundedness of the derivatives for all $(x, y) \in \mathbb{R}^N \times \mathbb{R}^N$,

$$|\partial_x V^i(x, y)| + |\partial_y V(x, y)| \le C.$$

In addition, suppose that $\partial_x V^i, \partial_y V^i$ are all Lipschitz in the sense that for all $(x, y) \in \mathbb{R}^N \times \mathbb{R}^N$,

$$\begin{aligned}
|\partial_x V^i(x, y) - \partial_x V^i(x', y')| &\le C(|x - x'| + |y - y'|), \\
|\partial_y V^i(x, y) - \partial_y V^i(x', y')| &\le C(|x - x'| + |y - y'|),
\end{aligned}$$

and $V^i, \partial_x V^i, \partial_y V^i$ all have linear growth property,

$$|V^i(x,y)| + |\partial_x V^i(x,y)| + |\partial_y V^i(x,y)| < C_T(1 + |x| + |y|),$$

where C_T is a constant depending only on T.

(c) We write $V \in \mathcal{C}^{k,k}_{b,\mathrm{Lip}}(\mathbb{R}^N \times \mathbb{R}^N; \mathbb{R}^N)$, if the following holds: for each $1, \dots, N$, and all multi-indices α, β on $\{1, \dots, N\}$ satisfying $|\alpha| + |\beta| \le k$, the derivative $\partial_x^\alpha \partial_y^\beta$ exists and is bounded, Lipschitz continuous, and satisfies linear growth condition.

(d) We say $V_0 \in \mathcal{C}^{k,k}_{b,\mathrm{Lip}}(\mathbb{R}^N \times \mathbb{R}^N)$ for short if $V_0 : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}^N$ satisfies (c). Let $V_1 : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}^{N \times d}$ with components $V_1^1, \dots, V_1^d : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}^N$. We say $V_1 \in \mathcal{C}^{k,k}_{b,\mathrm{Lip}}(\mathbb{R}^N \times \mathbb{R}^N)$ for short if $V_1^j \in \mathcal{C}^{k,k}_{b,\mathrm{Lip}}(\mathbb{R}^N \times \mathbb{R}^N)$ for every $j = 1, \dots, d$.

Assumption B.2. *We emphasize two assumptions used for the existence and smoothness of the marginal densities of directed chain SDEs:*

1. *(Uniform ellipticity on V_1) Assume that there exists $\epsilon > 0$ such that for all $\eta, x, \tilde{x} \in \mathbb{R}^N$,*
$$\eta^\top V_1(x,\tilde{x}) V_1(x,\tilde{x})^\top \eta \ge \epsilon |\eta|^2.$$
2. *(Smoothness on V_0, V_1) Assume that $V_0, V_1 \in \mathcal{C}^{k,k}_{b,\mathrm{Lip}}(\mathbb{R}^N, \mathbb{R}^N)$ with $k \ge N + 2$, where $V_0, V_1 \in \mathcal{C}^{k,k}_{b,\mathrm{Lip}}(\mathbb{R}^N, \mathbb{R}^N)$ is defined in Definition B.1.*

Under Assumption B.2, one can prove the existence of the density function of directed chain SDEs (Ichiba & Min, 2022, Theorem 4.3).

Theorem B.5. *Suppose Assumption B.2 is satisfied. For every Lipschitz function $\varphi : \mathbb{R}^N \to \mathbb{R}$ with Lipschitz constant K, there exists a constant $c > 0$ such that the difference between the conditional expectation of $\varphi(X_{t,q})$, given X_1 and the unconditional expectation $\varphi(X_{t,q})$ for all $t \in [0,T]$ is bounded, i.e.,*

$$\mathbb{E}\big[\sup_{0 \le t \le T} \big|\mathbb{E}[\varphi(X_{t,q})|X_1] - \mathbb{E}[\varphi(X_{t,q})]\big|^2\big] \le \frac{c^{q-1}}{(q-1)!}. \tag{16}$$

We shall first provide some interpretations for Theorem B.5 before giving the proof. For random variables in space $C([0,T], \mathbb{R}^N)$, there is no unique choice on how to measure their correlation or covariance. Here, we measure the difference between conditional expectation and unconditional expectation over a family of testing functions φ. Thus, we use the left-hand side in inequality (16) to measure the dependence between X_q and X_1.

Proof. Note that Assumption B.2 is a stronger version of Assumption B.1, and it not only ensures the existence and weak uniqueness of the solution, but also guarantees the existence of a smooth density which excludes the case of deterministic $X_{t,q}$. If X_q and X_1 are independent, the left-hand side is zero for every Lipschitz function φ. The vice versa is also correct because of the exclusion of the deterministic case.

Let us start from the left-hand side in (16), the difference between the conditional expectation of φ and unconditional expectation can be bounded by

$$\begin{aligned}
&\mathbb{E}\big[\sup_{0 \le t \le T} \big|\mathbb{E}[\varphi(X_{t,q})|X_1] - \mathbb{E}[\varphi(X_{t,q})]\big|^2\big] \\
&= \int_{C([0,T],\mathbb{R}^N)} \sup_{0 \le t \le T} \left| \int_{C([0,T],\mathbb{R}^N)} \Big(\mathbb{E}_{\mathbf{B}}\big[\varphi(\Phi_t^q(\omega;\mathbf{B}))\big] - \mathbb{E}_{\mathbf{B}}\big[\varphi(\Phi_t^q(\tilde{\omega};\mathbf{B}))\big] \Big) m(\,\mathrm{d}\tilde{\omega}) \right|^2 m(\,\mathrm{d}\omega) \\
&\le \int_{C([0,T],\mathbb{R}^N)^2} \sup_{0 \le t \le T} \mathbb{E}_{\mathbf{B}}\big[\big|\varphi(\Phi_t^q(\omega;\mathbf{B})) - \varphi(\Phi_t^q(\tilde{\omega};\mathbf{B}))\big|^2\big] m(\,\mathrm{d}\tilde{\omega}) m(\,\mathrm{d}\omega) \\
&\le K^2 \int_{C([0,T],\mathbb{R}^N)^2} \mathbb{E}_{\mathbf{B}}\big[\sup_{0 \le t \le T} \big|\Phi_t^q(\omega;\mathbf{B}) - \Phi_t^q(\tilde{\omega};\mathbf{B})\big|^2\big] m(\,\mathrm{d}\tilde{\omega}) m(\,\mathrm{d}\omega) \\
&\le \frac{c^{q-1}}{(q-1)!},
\end{aligned}$$

for some positive constant c, where the proof of the last inequality is verbatim to the procedures in Lemma B.1. □

Remark B.1. *We shall emphasize that the assumption $X_{0,1} = X_{0,2} = \cdots = X_{0,q} = \xi$ is not allowed under directed chain framework except for $\xi \equiv x \in \mathbb{R}^N$ (the deterministic initial condition). This is quite common in practice, for instance, the investment returns usually start from 1. Given results from Lemma B.1 and equation (16), we are able to conclude that*

$$\mathbb{E}[\sup_{0\le t\le T} |\varphi(X_1) - \mathbb{E}[\varphi(X_1)]|^2] \le \frac{c^{q-1}}{(q-1)!}.$$

Here q can be arbitrarily large, hence we conclude that $\mathbb{E}[\sup_{0\le t\le T} |\varphi(X_1) - \mathbb{E}[\varphi(X_1)]|^2] = 0$. The only possible solution for a directed-chain system is the deterministic case where we have deterministic initial conditions and degenerate V_1 (or we should call it "ODE"). Brownian motion is the key ingredient to enrich the representability of our directed-chain systems.

C. Experimental Details

Both discriminative and predictive metrics involve training tasks, and we shall first list all implementing details of these metrics, which is universal for all experiments. Then, we provide training hyper-parameters and training details used in different experiments.

C.1. Metrics

Discriminative Metric. We first generate the same amount of fake data paths as true data paths to avoid imbalance, and choose 80% from both real and fake data as training data, leaving the rest 20% as testing data. We use a two-layer LSTM classifier with `channels/2` as the size of the hidden state, where `channels` is the dimension of generated and real series. We will minimize the cross-entropy loss, and the optimization is done by Adam optimizer with a learning rate of 0.001 for 5000 iterations. The discriminative score is calculated by the difference between 0.5 and the prediction accuracy on testing data.

Predictive Metric. We first generate the same amount of fake data as true data, and use it as training data for the predictive metric, whereas true data is for testing. We use a two-layer LSTM sequential predictor with `channels/2` as the size of the hidden state, where `channels` is the dimension of generated and real series. Our objective function is L^1 distance between predicted sequences and true sequences. The predictor generates one-step future predictions in the last feature with the others as input. Optimization is done by Adam optimizer with a learning rate of 0.001 for 5000 iterations. The predictive score is reported as the L^1 distance (also interpreted as mean absolute error (MAE)) between the predictive sequences and true sequences on testing data.

Independence Metric. The independence score is computed by the maximum of the L^1 distance of cross-correlation matrices over the time period $[0, T]$. In practice, we consider the maximum over the time stamps $t \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$.

C.2. Experiments

In all four experiments, we use feed-forward neural networks with two hidden layers of sizes [128,128] to parameterize the drift V_0 and diffusion coefficient V_1. For the purpose of fair comparisons, we use the same GAN structure for both neural SDEs and DC-GANs, i.e., the same Sig-Wasserstein GAN setup (4) or the combination of neural CDE and Sig-WGAN scheme (6) as discriminators. We remark that DC-GANs can be adapted to `torchsde`[3] framework and use their adjoint method for back-propagation.

Stochastic Opinion Dynamics. In this experiment, we only use Sig-Wasserstein GAN approach for the discriminator, and choose $m = 8$ as the truncation depth in (4). We choose $N = 1$ and $d = 3$ dimensional standard Brownian motion in the DC-GANs generator (3), a batch size of 1024, a learning rate of 0.001 decaying to one-tenth for every 500 steps, and train a total of 2000 steps. Training data and testing data are sampled by the Euler scheme (3) with a sample size of 8192, and their initial distributions ξ are drawn independently from a uniform distribution on $[-2, 2]$.

Stochastic FitzHugh-Nagumo Model. The stochastic FitzHugh-Nagumo model is widely used in neuroscience for describing the neurons' interacting spiking, in particular, to capture the multimodality of neurons' interspike interval

[3] See the Python package `https://github.com/google-research/torchsde`.

distribution. For N neurons and P different neuron populations, we denote by $p(i) = \alpha, \alpha \in \{1, \dots, P\}$, the population of i-th particle belongs to, for $i \in \{1, \dots, N\}$. The state vector $(X_t^{t,N})_{t\in[0,T]} = (V_t^{i,N}, w_t^{i,N}, y_t^{i,N})_{t\in[0,T]}$ of neural i follows a three-dimensional SDE:

$$\begin{aligned}\mathrm{d}X_t^{t,N} &= f_\alpha(t, X_t^{t,N})\,\mathrm{d}t + g_\alpha(t, X_t^{t,N}) \begin{bmatrix} \mathrm{d}W_t^i \\ \mathrm{d}W_t^{i,y} \end{bmatrix} \\ &\quad + \sum_{\gamma=1}^{P} \frac{1}{N_\gamma} \sum_{j,p(j)=\gamma} \Big(b_{\alpha\gamma}(X_t^{i,N}, X_t^{j,N})\,\mathrm{d}t + \beta_{\alpha\gamma}(X_t^{i,N}, X_t^{j,N})\,\mathrm{d}W_t^{i,\gamma} \Big),\end{aligned}$$

where N_γ denotes the number of neurons in population γ. For all γ and $\alpha \in \{1, \dots, P\}$, $I^\alpha(t) := I, \forall t \in [0, T], \forall \alpha$ for some constant value I, f_α, g_α, $b_{\alpha\gamma}$ and $\beta_{\alpha\gamma}$ are given by

$$f_\alpha(t, X_t^{i,N}) = \begin{bmatrix} V_t^{i,N} - \frac{(V_t^{i,N})^3}{3} - w_t^{i,N} + I^\alpha(t) \\ c_\alpha(V_t^{i,N} + a_\alpha - b_\alpha w_t^{i,N}) \\ a_r^\alpha S_\alpha(V_t^{i,N})(1 - y_t^{i,N}) - a_d^\alpha y_t^{i,N} \end{bmatrix}, \quad g_\alpha(t, X_t^{i,N}) = \begin{bmatrix} \sigma_{\text{ext}}^\alpha & 0 \\ 0 & 0 \\ 0 & \sigma_\alpha^y(V_t^{i,N}, y_t^{i,N}) \end{bmatrix},$$

and

$$b_{\alpha\gamma}(X_t^{i,N}, X_t^{j,N}) = \begin{bmatrix} -\bar{J}_{\alpha\gamma}(V_t^{i,N} - V_{\text{rev}}^{\alpha\gamma}) y_t^{i,N} \\ 0 \\ 0 \end{bmatrix}, \quad \beta_{\alpha\gamma}(X_t^{i,N}, X_t^{j,N}) = \begin{bmatrix} -\sigma_{\alpha\gamma}^J(V_t^{i,N} - V_{\text{rev}}^{\alpha\gamma}) y_t^{i,N} \\ 0 \\ 0 \end{bmatrix}.$$

The functions S_α, $\mathcal{X}$ and σ_α^y are defined as

$$\begin{aligned} S_\alpha(V_t^{i,N}) &= \frac{T_{\max}^\alpha}{1 + e^{-\gamma_\alpha(V_t^{i,N} - V_T^{i,N})}}, \\ \mathcal{X}(y_t^{i,N}) &= \mathbb{1}_{y_t^{i,N} \in (0,1)} \Gamma e^{-\Lambda/(1-(2y_t^{i,N}-1)^2)}, \\ \sigma_\alpha^y(V_t^{i,N}, y_t^{i,N}) &= \sqrt{a_r^\alpha S_\alpha(V_t^{i,N})(1 - y_t^{i,N}) + a_d^\gamma y_t^{i,N}} \times \mathcal{X}(y_t^{i,N}), \end{aligned}$$

where $(W^i, W^{i,y}, W^{i,\gamma}), i = 1, \dots, N$ are standard three-dimensional Brownian motions that are mutually independent. For sample paths produced by this model, we follow the parameter choices in line with dos Reis et al. (2021),

$$\begin{array}{ccccccc} V_0 = 0, & \sigma_{V_0} = 0.4, & a = 0.7, & b = 0.8, & c = 0.08, & I = 0.5, & \sigma_{ext} = 0.5, \\ w_0 = 0.5, & \sigma_{w_0} = 0.4, & V_{rev} = 1, & a_r = 1, & a_d = 1, & T_{max} = 1, & \lambda = 0.2, \\ y_0 = 0.3, & \sigma_{y_0} = 0.05, & J = 1, & \sigma_j = 0.2, & V_T = 2, & \Gamma = 0.1, & \Lambda = 0.5. \end{array}$$

The above choice produces the joint multimodal distribution of V and w; see the figure below.

All training and testing data are generated through the Euler scheme with the above parameters. In the training phase, we choose the Sig-Wasserstein GAN approach again for our discriminator and choose $m = 6$ as the truncation depth in (4). We take $N = 3$ and $d = 5$ in our DC-GANs generator, and use a batch size of 1024 for training 2000 steps with a learning rate of 0.001 decaying to one-tenth every 500 steps. Training and testing data are generated by the Euler scheme, where the initial positions ξ are drawn from a 3-dimensional Gaussian random variable with means $(0, 0.5, 0.3)$ and standard deviations $(0.4, 0.4, 0.05)$.

Stock Price Time Series. In this real-world example, we use the six-dimensional stock price data of Google from 2004 to 2019. We segment them into sequences of length 24, which results in 3773 sequences as our time series data set. The combination of Neural CDEs and Signature MMD (6) is used as the discriminator. For the purpose of a fair comparison, we use the same noise size d and discriminator setup for both Neural SDEs and DC-GANs generators. In particular, for the Neural CDEs discriminator, we set the dimension of the hidden process to be 16, and their coefficients are approximated by a feed-forward neural network with two hidden layers of size [128, 128]. For both DC-GANs and Neural SDEs generator, the Brownian motion's dimension is set at $d = 10$; and for the Neural SDEs which embed stock prices data into a hidden space, we set its (the hidden space) dimension at 12. The batch size is chosen to be 128. Both generators and discriminators are trained using Adam optimizer. Both learning rates start at 0.0001 and decay to one-tenth after 2000 steps, the signature depth is chosen at $m = 4$ in (6) to alleviate the dimensional burden, and training steps are set as 4000. Our CTFP implementation follows the setup in Deng et al. (2020), SigWGAN follows from Ni et al. (2021) and TimeGAN implementation follows the setup in Yoon et al. (2019).

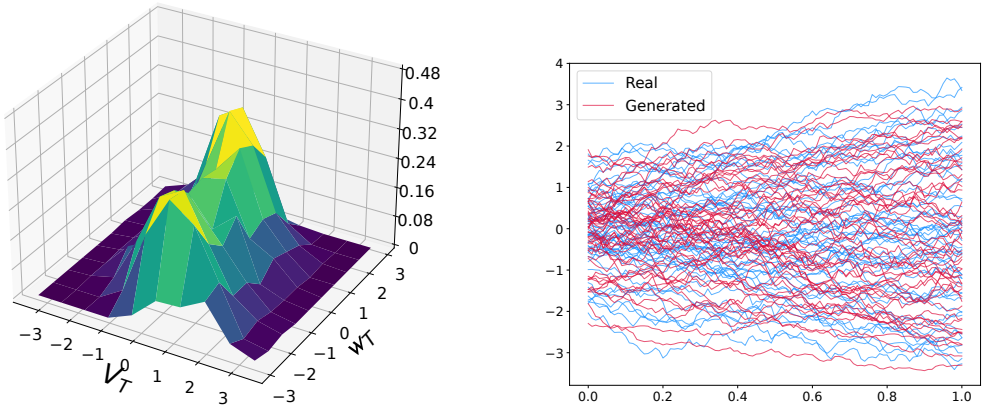


Figure 5. Stochastic FitzHugh-Nagumo Model (Example 2). Left subfigure shows the multimodal joint density of V_T and w_T, and right subfigure shows the sample paths from the time-dependent model (blue) and from the DC-GANs (red).

Energy Consumption. In the real-world energy consumption example, we choose four electric and gas consumption time series from 02/2011-02/2013 and use daily data as a single time series, bringing 694 sequences with a length of 96. For both neural SDEs and DC-GANs, we use a ten-dimensional Brownian motion and neural nets with two hidden layers of size [128, 128] to estimate drift and diffusion coefficients. The batch size is 128, the training step is 4000, and the learning rate for the generator starts at 0.0001. In the case of using Neural CDEs as the discriminator, we use hidden size 16, [128, 128] as the hidden layers of neural nets estimating coefficients and 0.0001 as the starting learning rate of the discriminator. In the case of using SigWGAN, we consider signature depth 6. All learning rates decay to one-tenth after 2000 steps. Our CTFP implementation follows the setup in Deng et al. (2020), SigWGAN follows from Ni et al. (2021) and TimeGAN follows from Yoon et al. (2019).