

TSGGuide: Recommendation Guide for Multivariate Time Series Generation

Beining Zhang
Guangzhou Institute of Technology,
Xidian University
Guangzhou, Guangdong, China
22171214736@stu.xidian.edu.cn

Kai Wu*
School of Artificial Intelligence,
Xidian University
Xi'an, Shannxi, China
kwu@xidian.edu.cn

Xiaoyu Zhang
School of Cyber Engineering, Xidian
University
Xi'an, Shannxi, China
xiaoyuzhang@xidian.edu.cn

Handing Wang
School of Artificial Intelligence,
Xidian University
Xi'an, Shannxi, China
hdwang@xidian.edu.cn

Jing Liu
Guangzhou Institute of Technology,
Xidian University
Guangzhou, Guangdong, China
neouma@mail.xidian.edu.cn

Bei Wang
NIO
Shanghai, China
bei.wang@nio.com

Kai Wang
NIO
Shanghai, China
kai.wang@nio.com

Abstract

Multivariate Time Series Generation (MTSG) plays a crucial role in time series analysis, supporting tasks such as data augmentation and anomaly detection. While several methods exist for MTSG, recommending the most suitable method for new scenarios remains a significant challenge. Although prior work by [2] provides guidance for selecting MTSG methods, it lacks coverage of recent diffusion-based methods and has limited exploration of channel-independent frameworks. We address these gaps by improving the recommendation guide, highlighting the effectiveness of a central discriminator within the channel-independent framework. Our revised guide makes three key recommendations: 1) VAE-based methods excel on small-scale datasets; 2) a channel-independent framework with the newly designed central discriminator is optimal in most cases; and 3) a diffusion-based method is preferable when ample data and computational resources are available.

CCS Concepts

• Computing methodologies → Neural networks.

Keywords

Time Series Generation, Diffusion Model, Channel Independence

*Corresponding Author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
WWW 2025 AI4TS Workshop, April 28–29, 2025, Sydney, Australia
© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-XXXX-X/18/06
<https://doi.org/XXXXXXX.XXXXXXX>

ACM Reference Format:

Beining Zhang, Kai Wu, Xiaoyu Zhang, Handing Wang, Jing Liu, Bei Wang, and Kai Wang. 2025. TSGGuide: Recommendation Guide for Multivariate Time Series Generation. In *Proceedings of WWW 2025 AI4TS Workshop*. ACM, New York, NY, USA, 17 pages. <https://doi.org/XXXXXXX.XXXXXXX>

1 Introduction

Time series generation (TSG) involves producing synthetic sequences of temporally ordered data that mimic the statistical properties of real-world time series. Multivariate time series generation (MTSG) has gained prominence due to its applications in tasks like data augmentation [40], time series prediction [9, 50], and anomaly detection [3, 5].

Numerous methods have been developed for MTSG [2], typically falling into two categories: 1) channel-mixing frameworks, which merge time series features using models like generative adversarial networks (GANs) [12, 53, 45, 32, 31], variable autoencoders (VAEs) [11, 29, 30, 34], flow-based methods [7, 25, 42, 23, 56], and diffusion-based methods [27, 18, 10, 54]; and 2) channel-independent frameworks with a central discriminator [43], which independently capture single-channel information while the discriminator handles inter-channel correlations.

Choosing an appropriate TSG method depends on specific factors such as data volume, periodicity, and trends. For instance, Fourier flow [1] is well-suited for tasks involving autocorrelation, while TimeVAE [11] performs better on smaller datasets. Therefore, a robust recommendation guide is essential for users. However, existing analyses, like [2], may be incomplete—favoring channel-mixing frameworks, particularly VAE-based methods, while neglecting advanced diffusion models and offering limited exploration of channel-independent frameworks. For example, the periodicity of the multivariate time series and the correlation between the channels are important features. However, the central discriminator designed in [43] does not adequately address this problem. In addition, [2] has not tested the newly proposed diffusion model-based methods and

needs to be upgraded to form a reliable recommendation guide for time series generation [27, 18, 10, 54].

In light of the insufficient comparison with advanced diffusion-based generative frameworks and limited exploration of channel-independent frameworks, we have updated the recommendation guidelines from [2] to better aid users in selecting TSG frameworks. Specifically, we advocate for an increased priority recommendation for both channel-independent and diffusion-based generative frameworks, forming an enhanced TSGGuide.

The key contributions of this work are summarized as follows:

- We updated the recommended guidelines for time series generation methods, increased the priority of channel-independence-based MTSG, and added results and recommended guidelines for time series generation methods based on diffusion models.
- In the channel-independence-based strategy, we designed the Central Convolution Discriminator (CCD), which significantly improved the performance of MTSG based on the channel-independence-based strategy by better capturing the features within and between channels.

2 Evaluation of current methods and TSGGuide

2.1 Channel Mixing-based MTSG

There is a predominant focus on channel-mixing time series generation framework. Some methods [33, 14, 48, 52, 21, 22] employ GAN-based [15] architecture combined with neural networks like LSTM [17], GRU [8], and Transformer [46], which excel at capturing sequential data. Certain methods leverage VAEs [26] to effectively capture temporal features through variational inference, resulting in efficient models with potential interpretability. Recent advancements have explored hybrid methods, combining flow-based models with techniques such as ODE [7], or integrating them with GANs or VAEs. Moreover, several diffusion-based methods [54, 27, 10, 18] have been designed to handle MTSG. Due to their stable training and high-quality generation, diffusion models exhibit promising performance on numerous datasets. Employing channel-mixing framework intuitively allows for improved consideration of correlations between channels in multivariate time series data, thereby enhancing model performance.

2.2 Channel Independence-based MTSG

COSCIGAN [43] employs channel GANs and a central discriminator to generate multivariate time series, but it fails to effectively capture key characteristics like periodicity and inter-channel correlations, limiting the performance of the central discriminator. Compared to the MLP-based discriminator in COSCIGAN, more advanced architectures such as attention mechanisms, TimesNet [49], and ModernTCN [13] offer better potential for performance improvement. Our findings suggest that the central discriminator operates primarily as a time series classifier in small-sample contexts. However, MLP-based methods, as well as attention-based approaches like TimesNet, exhibit limited effectiveness in handling such scenarios.

3 Problem Definition

Dataset Setup. Suppose that an MTS has length L with N channels. The general length of these time series is often long, making it difficult to input the entire time series data at once and extract its features. To generate time series within a short period and extract features from the time series data, we need to choose an appropriate subsequence length, denoted as l , and use a step size of 1 when performing the partition. Then we transform MTS into $T \in \mathcal{R}^{K \times l \times N}$, where $K = L - l + 1$. Additionally, we normalize the dataset to the range of $[0, 1]$ to enhance efficiency and numerical stability.

Time Series Generation. We define $p(T)$ as the true distribution of the time series. Our goal is to create a synthetic time series $\hat{T} \in \mathcal{R}^{K \times l \times N}$ in which its distribution $q(\hat{T})$ is similar to $p(T)$.

4 Background

4.1 Channel Mixing-based MTSG

Channel-mixing frameworks in time series generation models aim to learn the joint distribution of all channels in a multivariate time series, regardless of the number of channels. The goal of these frameworks can be defined as follows: $\min_q D(p(T) \| q(\hat{T}))$, where D is any suitable measure of the distance between two distributions. Irrespective of the utilization of GANs, VAEs, or flow-based methods, most of them employing a channel-mixing framework strive to learn an improved $q(\hat{T})$.

4.2 Channel Independence-based MTSG

Figure 1 illustrates the structure of the channel-independent framework with a central discriminator, comprising two main components: 1) Channel-independent TSGs; 2) Central discriminator.

4.2.1 Channel-independent TSGs. Here, each single-channel TSG module synthesizes one single-channel time series. The input T is divided into N single-variable sequences $t_i \in \mathcal{R}^{K \times l \times 1}$, where $i \in [1, N]$. Subsequently, following the channel-independent setup, the i -th channel's time series t_i is input into the i -th single-channel TSG module, yielding the generated single-channel time series data $\hat{t}_i = G_{i, \theta_i}(z)$, where G is a TSG module and θ_i is the parameters of the i th module. This framework allows flexibility in selecting channel-independent generators, such as GANs and VAEs.

In the channel objective, we aim to find a distribution $q(\hat{t}_i)$ that closely approximates the true channel distribution $p(t_i)$ of the dataset. Subsequently, by concatenating N instances of \hat{t}_i , where $i = 1, \dots, N$, along the second dimension, the generated time series \hat{T} is obtained. For each channel t_i , we optimize $\min_q D(p(t_i) \| q(\hat{t}_i))$. When channel-independent generators are trained, incorporating central discriminator loss suffices.

4.2.2 Central Discriminator. In the center objective, we consider the overall characteristics of multivariate time series, which involves integrating the time series signals from all channels. This part designs powerful binary classifiers to distinguish between \hat{T} and T . Certainly, the more difficult it is to distinguish between \hat{T} and T under this classifier, the better. We should optimize $\min_q D(p(T) \| q(\hat{T}))$.

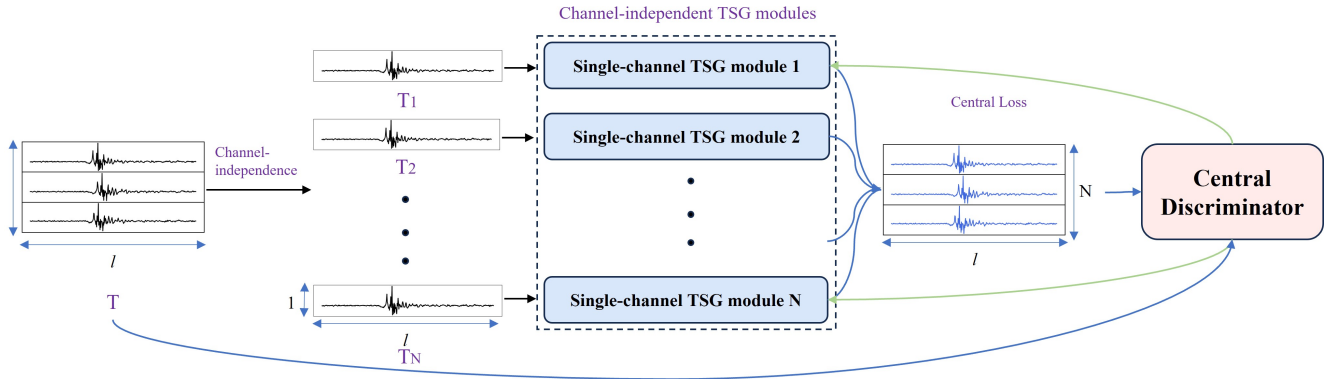


Figure 1: Structure of channel independence-based MTSG.

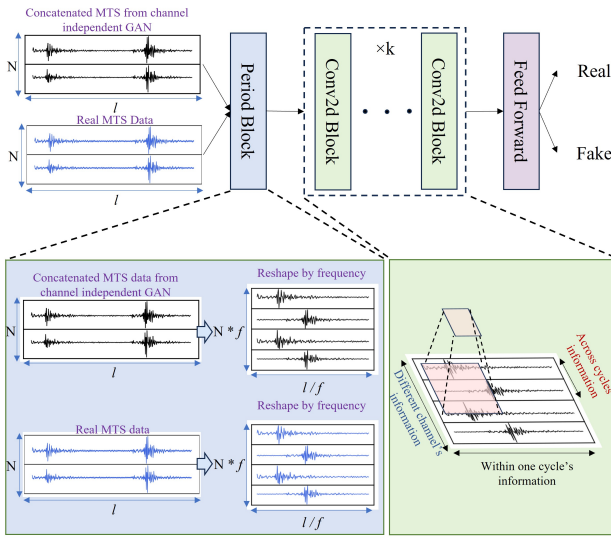


Figure 2: The structure of CCD.

5 Central Convolution Discriminator

CCD uses Fast Fourier Transform (FFT) or Wavelet Transform (WT) to perform periodic segmentation of the samples, reducing the classification difficulty while revealing inter-period relationships within the samples to enhance key features that are beneficial for classification. It then uses 2D convolution to capture both intra-channel and inter-channel relationships. The details of CCD are shown in Appendix A. The designed CCD is shown in Figure 2.

5.1 Period Block

For a multivariate time series T , we utilize Frequency to obtain its periodicity in the frequency domain.

$$A = \text{Avg}(\text{Amp}(\text{Frequency}(T))) \quad (1)$$

where $\text{Frequency}(\cdot)$ represents the solution to obtain the frequency. Here, we use FFT or WT. A represents the calculated amplitude of each frequency, which is averaged from the dimensions N using $\text{Avg}(\cdot)$.

The true periodicity of T is computed using $l_p = \lceil \frac{l}{f} \rceil$. Depending on the selected frequency and the corresponding period length, we can transform the dimension of the multivariate time series T and \hat{T} into $(K, l_p, N \times f)$.

$$T_p \in \mathcal{R}^{K \times l_p \times f}, \hat{T}_p \in \mathcal{R}^{K \times l_p \times f} = \text{Reshape}(T, \hat{T}) \quad (2)$$

5.2 Conv2d Block

After passing through the Period module, we utilize multiple conv2d blocks to capture the three types of local information mentioned earlier, distinguishing between real and synthesized multivariate time series. The formulation is as follows:

$$t_c, \hat{t}_c = \text{Conv2d Block}(T_p, \hat{T}_p) \quad (3)$$

where we transform 2D representations $T_p, \hat{T}_p \in \mathcal{R}^{K \times l_p \times f}$ into 1D space $t_c, \hat{t}_c \in \mathcal{R}^{K \times d}$. Considering both performance and efficiency, we opt for experiments using the `nn.conv2d()` block based on PyTorch for our main experiments.

Finally, the features extracted through the Conv2d blocks are further processed by the feed forward module, which consists of a linear layer followed by a sigmoid activation function, yielding the ultimate classification results.

6 Evaluation of CCD

6.1 Validation of TSGGuide

This section explores the performance of current typical time series generation models on different domain data and forms a user-oriented guide for recommending time series generation models.

6.1.1 Baselines. We present a detailed analysis of five representative time series generation (TSG) methods that are based on three foundational generative models. To ensure experimental fairness, all experiments are conducted on a machine with Intel® Core® i9 12900K CPU @ 5.20 GHz, 64 GB memory, and NVIDIA GeForce RTX 3090.

Pure GAN-based Methods. Early studies [33, 14] incorporated vanilla GAN architectures originally designed for image generation and combined them with neural networks such as RNN and LSTM, specifically tailored for sequential data. Subsequent research has

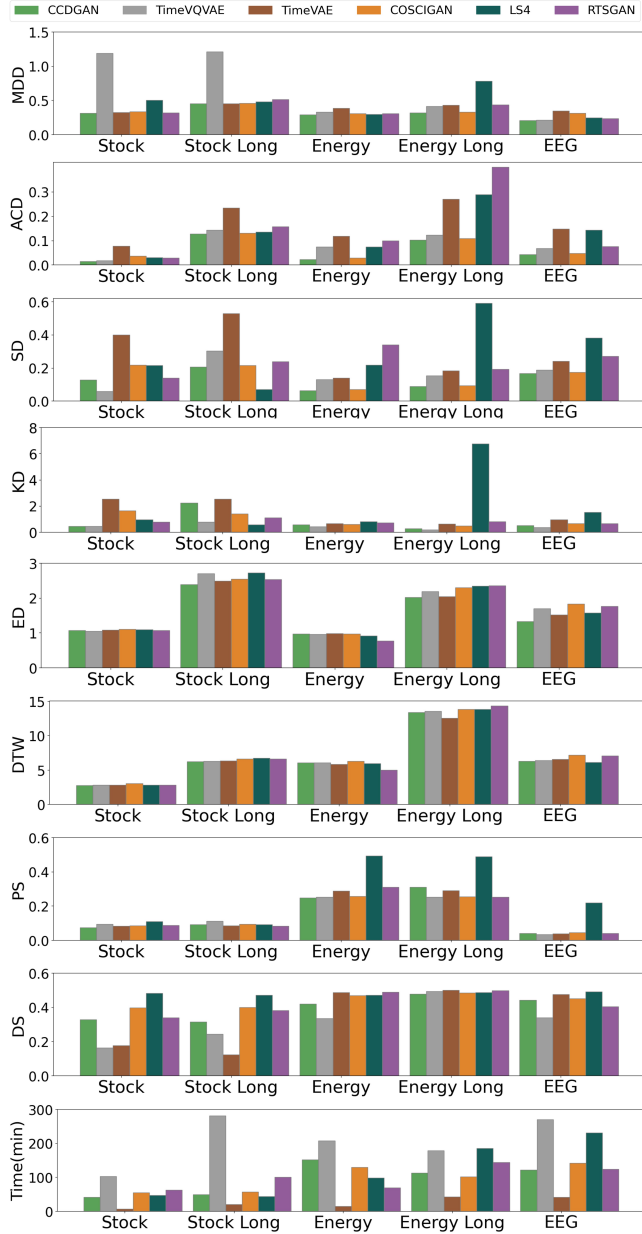


Figure 3: Results of six methods on real datasets.

been dedicated to pioneering techniques that adapt to time series data and enhance performance.

- RTSGAN [39]. RTSGAN integrates an autoencoder into GANs and focuses on generating time series with variable lengths while effectively handling missing data.
- COSCI-GAN [43]. COSCI-GAN is specifically designed to explicitly capture the complex dynamical patterns within each series, with a focus on preserving the relationships among channels or features.

Pure VAE-based Methods. VAE-based methods commonly leverage variational inference to effectively capture temporal features. These methods are known for their efficiency and potential interpretability.

- TimeVAE [11]. TimeVAE extends the application of Variational Autoencoders (VAEs) to general-purpose time series generation. It incorporates convolutional operations and enhances interpretability through time series decomposition techniques.
- TimeVQVAE [29]. TimeVQVAE incorporates the short time Fourier Transform (STFT) to decompose input time series into low-frequency and high-frequency components. It further enhances the modeling of these components by integrating Vector Quantization with VAEs [29], ensuring the preservation of both the general shape and specific details of the time series.

Mixed-Type Methods. Recent advancements in time series generation (TSG) have explored mixed-type methods, which involve combining flow-based models with techniques such as Discrete Fourier Transform (DFT) or Ordinary Differential Equations (ODEs). Additionally, these methods have been integrated with Generative Adversarial Networks (GANs) or Variational Autoencoders (VAEs) to further enhance their capabilities.

- LS4 [56]. LS4 is derived from deep state-space models and integrates stochastic latent variables to augment the model’s capacity while leveraging the training objectives of Variational Autoencoders (VAEs).

6.1.2 Parameters. For RTSGAN, we adhere to its complete time series generation [39] and set $\beta_1 = 0.9$ and $\beta_2 = 0.999$. For COSCI-GAN, we set $\gamma = 5$, employ MLP-based networks for the central discriminator, and follow other hyper-parameters from [43]. For TimeVAE, we set the latent dimension to 8 and the hidden layer sizes to 50, 100, and 200. For TimeVQVAE, we adopt the settings from [29], with $n_fft = 8$ and varying $max_epochs \in \{2000, 10000\}$ for two training stages. For LS4, we set the latent space dimension to 5 and configured the batch size to 1024.

6.1.3 Datasets. Dataset Selection. To ensure reproducibility and mitigate biases or oversimplification in our evaluations, we exclusively employ publicly available, real-world datasets. It is crucial to emphasize that our objective is not to accumulate an exhaustive collection of datasets, but rather to curate a diverse set encompassing multiple domains, showcasing varied data statistics and distributions. Table 1 summarizes their statistics. Below, we provide a brief description of each dataset.

- Stock [53]. It comprises daily historical Google stock data from 2004 to 2019, including volume and high, low, opening, closing, and adjusted closing prices.
- Stock Long [53]. It is identical to the Stock dataset but with a sequence length of 125.
- Energy [6]. It includes information on appliance’s energy use in a low-energy building.
- Energy Long [6]. It is identical to the Energy dataset but with a sequence length of 125.
- EEG [41]. It is with the measurements derived from ElectroEncephaloGraphy (EEG) data captured by Emotiv EEG

Neuroheadset. It helps to understand brainwave patterns, especially those under specific cognitive conditions or stimuli.

- Dodgers Loop Game (DLG) [19]. It consists of loop sensor data from the Glendale on-ramp for the 101 North freeway in Los Angeles.
- D9: Air [55]. It has air quality, meteorological, and weather forecast data from 4 major Chinese cities: Beijing, Tianjin, Guangzhou, and Shenzhen from 2014/05/01 to 2015/04/30.

This study selected five datasets from different real-world domains: the Stock dataset from the financial domain, the Energy dataset from the energy sector, the EEG dataset from biological signals, the DLG dataset from traffic, and the Air dataset. To better investigate the impact of subsequence length on the model, we set different values of l for the Stock and Energy datasets. Therefore, our method is applicable to real cases or practical applications.

Table 1: The statistics of seven datasets.

Datasets	R	l	N	Domain
Stock [53]	3,294	24	6	Financial
Stock Long [53]	3,204	125	6	Financial
Energy [6]	17,739	24	28	Appliances
Energy Long [6]	17,649	125	28	Appliances
EEG [41]	13,366	128	14	Medical
DLG [19]	246	14	20	Traffic
Air [55]	7731	168	6	Sensor

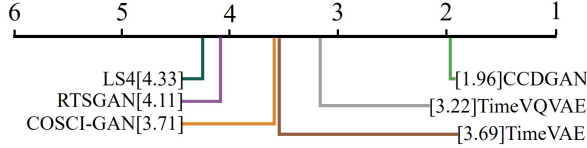


Figure 4: The critical difference diagram illustrates the performance ranking of six algorithms across five datasets employing Wilcoxon-Holm analysis [20] at a significance level of $p = 0.05$. Algorithm positions are indicative of their mean ranks across multiple datasets, with higher ranks signifying a method consistently outperforming competitors. Thick horizontal lines denote scenarios where there is no statistically significant difference in algorithm performance.

6.1.4 Evaluation Metrics. We adopt the rigorous evaluation framework of TSGBench [2], utilizing a comprehensive set of metrics to assess both global and local properties of the generated time series (See Appendix D for details): 1) *Global Similarity Metrics*: MDD (Mean Distribution Distance), ACD (Auto-Correlation Distance), SD (Spectral Distance), and KD (Kernel Distance). 2) *Local Similarity Metrics*: ED (Euclidean Distance) and DTW (Dynamic Time Warping). Smaller values of these metrics are preferred.

6.2 Results

The results shown in Figures 3 and 4 provide evidence for the effectiveness of CCDGAN in synthesizing MTS data. We also conducted

experiments based on DLG and Air, which are from the traffic flow dataset and air dataset respectively. For more dataset details, see Appendix C. The experimental results are shown in Tables 2 and 3. The performance of CCDGAN surpasses that of COSCI-GAN, validating the effectiveness of the proposed CCD module. As shown in Table 12, we choose the LSTM, which exhibits moderate generation performance, as the channel-independent generator. However, its performance exceeds that of the current state-of-the-art channel-mixing methods, TimeVQVAE and TimeVAE. Particularly on datasets with a higher number of channels such as Energy and Energy Long, the superiority of CCDGAN is more pronounced. Relative to COSCI-GAN, the proposed framework does not consume additional time. Furthermore, in comparison to channel-mixing methods, the runtime of the channel-independent approach is competitive.

6.3 CCDGAN v.s. Diffusion-based Methods

In recent years, diffusion models have been increasingly applied to MTSG, achieving notable results. From the above experimental results, CCDGAN performs better when the amount of data is sufficient. We replicate the experimental setup from [53], detailed in Appendix Section H.

Baselines. To facilitate a fair comparison among different approaches and provide updated recommendations for MTSG methods, we compare CCDGAN with three diffusion-based TSG methods: Diffwave [27], DiffTime [10], and Diffusion-TS [54]. These methods, including Diffusion-TS, DiffWave, and DiffTime, are all based on DDPM, with modifications tailored to the characteristics of time series data.

Parameters. For Diffusion-TS, DiffWave and DiffTime, to ensure the fairness of the experiments, we strived to maintain consistency in parameter settings. We chose 4 attention heads, each with a dimension of 16, and selected 2 encoder and decoder layers.

Datasets. For dataset selection, we chose the previously mentioned Stock and Energy datasets. For the Sine dataset, we opted for the sine wave dataset provided in [53], which is channel independent and exhibits more diverse variations.

The results, presented in Table 4, indicate that diffusion-based methods generally exhibit superior performance. However, they also require significant time and computational resources. For those prioritizing stable training and high-quality results, diffusion-based methods are recommended. Nonetheless, CCDGAN outperforms certain diffusion-based methods in terms of performance while demanding less time and computational resources, highlighting its competitive advantage in both effectiveness and efficiency.

6.4 TSGGuide: Recommendation Guidelines

Finally, combining the guidelines from the study by TSGBench [2], we offer guidance to assist users in effectively selecting suitable TSG methods. In contrast to TSGBench, the updated sections are shown in Appendix Section J.

- (1) In Figure 3, VAE-based methods demonstrate faster training times and rank above average on several metrics. This makes VAE-based methods suitable for initial attempts due to their lower time requirements. As a foundational step, we recommend that users start with the VAE-based method. Its

Table 2: Results on DLG dataset.

Method	CCDGAN	TimeVQVAE	TimeVAE	COSCI-GAN	LS4	RTSGAN
MDD↓	0.284	0.293	0.301	0.241	0.227	0.238
ACD↓	0.121	0.162	0.164	0.137	0.117	0.178
SD↓	0.214	0.235	0.209	0.257	0.216	0.227
KD↓	12.341	12.25	12.175	12.275	12.377	12.375
ED↓	1.12	1.363	1.315	1.641	1.363	1.177
DTW↓	2.346	2.367	2.476	2.316	2.375	2.438
PS↓	0.415	0.488	0.446	0.461	0.425	0.427
DS↓	0.245	0.221	0.237	0.257	0.227	0.288

Table 3: Results on Air dataset.

Method	CCDGAN	TimeVQVAE	TimeVAE	COSCI-GAN	LS4	RTSGAN
MDD↓	0.139	0.127	0.121	0.142	0.148	0.117
ACD↓	0.109	0.117	0.109	0.117	0.115	0.126
SD↓	0.356	0.328	0.387	0.382	0.361	0.371
KD↓	8.147	8.278	8.178	8.187	8.169	8.171
ED↓	0.816	0.828	0.827	0.826	0.829	0.877
DTW↓	2.044	2.28	2.091	2.081	2.062	2.027
PS↓	0.404	0.483	0.426	0.433	0.429	0.416
DS↓	0.107	0.186	0.131	0.124	0.139	0.121

Table 4: Results between CCDGAN and Diffusion-based methods. Bold indicates best performance.

Metric	Methods	Sines	Stocks	Energy
Context-FID↓	CCDGAN	0.008±.001	0.158±.022	0.134±.019
	Diffusion-TS	0.006±.000	0.147±.025	0.089±.024
	Diffwave	0.014±.002	0.232±.032	1.031±.131
	DiffTime	0.006±.001	0.236±.074	0.279±.045
Correlational Score↓	CCDGAN	0.016±.000	0.023±.012	0.823±.108
	Diffusion-TS	0.015±.004	0.004±.001	0.856±.147
	Diffwave	0.022±.005	0.030±.020	5.001±.154
	DiffTime	0.017±.004	0.006±.002	1.158±.095
Discriminative Score↓	CCDGAN	0.009±.000	0.138±.042	0.131±.000
	Diffusion-TS	0.006±.007	0.067±.015	0.122±.003
	Diffwave	0.017±.008	0.232±.061	0.493±.004
	DiffTime	0.013±.006	0.097±.016	0.445±.004
Predictive Score↓	CCDGAN	0.093±.000	0.041±.000	0.250±.000
	Diffusion-TS	0.093±.000	0.036±.000	0.250±.000
	Diffwave	0.093±.000	0.047±.000	0.251±.000
	DiffTime	0.093±.000	0.038±.001	0.252±.000
Training Time(min)↓	CCDGAN	11	10	37
	Diffusion-TS	17	15	60
	Diffwave	19	16	68
	DiffTime	19	16	69
2000 Data Sampling Time(s)↓	CCDGAN	5	7	13
	Diffusion-TS	23	26	65
	Diffwave	24	30	70
	DiffTime	24	29	69

consistently excellent computational efficiency makes it the preferred choice for initial exploration and handling of small data sets.

- (2) If we emphasize autocorrelation or forecasting, such as predictive maintenance or stock market analysis, the ACD measure becomes crucial. CCDGAN is highly suitable for these scenarios.
- (3) In Figure 3, CCD shows a leading performance on the majority of datasets, proving that CCDGAN performs well on real-world datasets, even though they come from different domains. When the dataset originates from a novel domain or exhibits complex multivariate relationships, CCDGAN is a recommended choice. It shows excellent performance across data sets from various domains and outperforms other methods, particularly on datasets with a large number of variables.
- (4) In Table 4, diffusion model-based methods achieve certain advantages in metrics. However, these methods require significant training and sampling time, leading to higher computational costs. Therefore, if ample computational resources are available, diffusion model-based approaches should be considered. When computational resources and dataset are abundant and optimal results are desired, it is recommended to utilize Diffusion-TS. These methods typically exhibit stable training and yield superior performance.
- (5) Users can further adjust method selection based on specific application requirements, including identifying the appropriate channel-independent TSG module (see Section I).

Table 5: The results of Transformer, Timesnet, COSCIGAN, and CCD on the Stock dataset and the 10% Stock dataset. Due to poor downstream task performance with 10% of the Stock data, comparative results are omitted. Bold numbers in the table indicate the best performance.

Methods	100% Stock				10% Stock			
	CCD	Transformer	Timesnet	COSCIGAN	CCD	Transformer	Timesnet	COSCIGAN
MDD↓	0.315	0.341	0.38	0.334	0.511	0.692	1.036	0.606
ACD↓	0.015	0.027	0.029	0.037	0.049	0.132	0.135	0.083
SD↓	0.127	0.149	0.133	0.217	0.200	0.431	0.344	0.581
KD↓	0.446	0.449	0.485	1.647	0.861	1.392	1.492	1.422
ED↓	1.068	1.094	1.100	1.101	2.239	2.58	2.98	1.935
DTW↓	2.772	2.805	2.959	3.014	5.717	5.93	6.64	4.913
PS↓	0.073	0.097	0.091	0.086	-	-	-	-
DS↓	0.327	0.339	0.341	0.396	-	-	-	-

6.5 Validation of CCD

6.5.1 Ablation Study of CCD. This section examines the effect of the central discriminator within the channel-independent generation framework.

Firstly, We compared COSCI-GAN using common attention mechanisms and TimesNet, a state-of-the-art time series classification framework, as baselines. Experiments were conducted on both the full Stock dataset and a reduced version with 10% of the data, maintaining the same number of channels in COSCI-GAN. The central discriminator in COSCI-GAN was replaced by the Transformer Encoder, TimesNet, and CCD for evaluation. The Transformer Encoder structure followed [46] with $N = 6$. Detailed information on parameter metrics and datasets is provided in Appendix Section C and D.

As shown in Table 5, these methods struggled with small sample scenarios, exhibiting a notable performance drop with 10% of the Stock data. The heavy parameterization of attention mechanisms likely made them less effective for limited data. TimesNet, designed for single-channel time series, failed to capture inter-channel dependencies, a critical aspect for multi-channel time series generation. In contrast, CCD demonstrated superior performance, particularly with the 10% Stock data, where its advantage was most pronounced.

Moreover, we validated the effectiveness of the Period and the Conv2D modules on the Stock[53] and Energy[6] datasets. **No Period** indicates the removal of the Period module from CCD, avoiding manipulation of real and fake time series dimensions. **No Conv2D** denotes the replacement of the Conv2D block with three linear layers.

The results in Table 6 show that removing either the Period or Conv2D block decreases performance. This underscores the importance of both components in CCD, highlighting their role in enhancing and capturing various types of local information across different datasets.

6.5.2 Exploration of Period Blocks in CCD. We investigated the impact of using FFT versus Wavelet Transform in the Period Block. While FFT focuses solely on the frequency domain, the Wavelet Transform offers both frequency and time localization. Experiments

on the Stock dataset, with consistent parameters except for frequency extraction, revealed minimal performance differences between FFT and Wavelet Transform, as shown in Table 19.

6.5.3 Analysis of Kernel Size. We investigated the selection of Conv2D kernel sizes, testing two strategies: fixed kernel sizes (1, 3, and 5) and varied kernel sizes defined by $kernel\ size = 2 \times i + 1$ for $i = 0, 1, \dots, k$ with $k = 3$ as the number of Conv2D blocks. Results on the Stock and Energy datasets, shown in Table 7, demonstrate that the varied kernel size strategy outperforms the fixed kernel size approach. Stacking Conv2D modules with increasing kernel sizes allows the model to capture more diverse and richer feature representations.

The benefits of adopting such a strategy are as follows: 1) Compared to a fixed convolutional kernel, by stacking conv2d modules with increasing kernel sizes, the model can extract more diverse and richer feature representations. Different kernel sizes can capture different types of features. 2) By using smaller kernel sizes in the initial convolutions, the model can simultaneously focus on finer local features. As the conv2d blocks are stacked, the model needs to acquire broader global features.

6.5.4 Effect of no. of Channels on CCD. We explored the impact of the number of channels N , using the Energy dataset as an example. We randomly selected channel data with numbers 4, 8, and 12 in the Energy dataset. The results, as shown in Table 8, indicate that the number of channels has almost no impact on CCDGAN.

We investigated the impact of subsequence length l . The Stock dataset was divided into Stock ($l = 24$) and Stock Long ($l = 125$) lengths, and the Energy dataset was divided into Energy ($l = 24$) and Energy Long ($l = 125$). Experimental results are shown in Tables 9–12. As the subsequence length l increases, the performance metrics of CCDGAN degrade but still outperform other methods.

6.5.5 Effect of No. of convolution layers on CCD. For the number of convolution layers, we experimented with 2, 3, and 4 layers on the Stock dataset, while keeping other experimental settings and parameters constant. The results are shown in Table 9. The results indicate that setting the convolution layers to 3 yields relatively good experimental results on the Stock dataset. Therefore, this

Table 6: Ablation study of Period and Conv2D modules on the Stock and Energy datasets.

Dataset	Stock			Energy		
Metrics	No Period	No Conv2d	CCD	No Period	No Conv2d	CCD
MDD↓	0.433	0.431	0.305	0.226	0.395	0.219
ACD↓	0.022	0.039	0.015	0.037	0.044	0.023
SD↓	0.149	0.142	0.127	0.29	0.285	0.272
KD↓	0.829	0.848	0.556	0.781	0.801	0.748
ED↓	1.103	1.097	1.068	0.994	0.979	0.966
DTW↓	4.442	4.39	4.252	7.131	7.195	7.045
PS↓	0.09	0.127	0.073	0.305	0.28	0.247
DS↓	0.408	0.483	0.327	0.424	0.433	0.419

Table 7: Analysis of Kernel size on Energy and Stock dataset.

Dataset	Stock				Energy			
Kernel Size	1	3	5	varied	1	3	5	varied
MDD↓	0.485	0.433	0.572	0.305	0.306	0.289	0.251	0.219
ACD↓	0.171	0.211	0.193	0.015	0.049	0.051	0.052	0.023
SD↓	0.174	0.16	0.152	0.127	0.296	0.267	0.284	0.272
KD↓	0.807	0.719	0.692	0.556	0.912	0.959	0.761	0.700
ED↓	1.102	1.079	1.092	1.068	0.992	0.979	0.971	0.966
DTW↓	5.104	4.829	4.701	4.252	7.281	7.163	7.082	7.045
PS↓	0.166	0.092	0.096	0.073	0.466	0.301	0.288	0.247
DS↓	0.468	0.391	0.383	0.327	0.484	0.466	0.451	0.419

Table 8: Analysis of number of channels on Energy dataset.

No. of channels	4	8	12
MDD↓	0.314	0.315	0.315
ACD↓	0.017	0.015	0.013
SD↓	0.128	0.127	0.127
KD↓	0.442	0.446	0.445
ED↓	1.068	1.068	1.07
DTW↓	2.773	2.772	2.773
PS↓	0.071	0.073	0.081
DS↓	0.32	0.327	0.331

Table 9: Analysis of number of layers on Stock dataset.

No. of layers	2	3	4
MDD↓	0.307	0.315	0.318
ACD↓	0.017	0.015	0.014
SD↓	0.133	0.127	0.127
KD↓	0.442	0.446	0.445
ED↓	1.066	1.068	1.067
DTW↓	2.775	2.772	2.78
PS↓	0.08	0.073	0.079
DS↓	0.331	0.327	0.325

setting can be initially applied to other datasets, with adjustments made based on the actual situation.

7 Conclusion

In this work, we tackled the limitations in existing multivariate time series generation frameworks by proposing an updated recommendation guide, TSGGuide, aimed at improving the selection of generation methods. Our comprehensive analysis revealed gaps in prior work, particularly in the limited exploration of channel-independent frameworks and the absence of evaluations for diffusion-based models. We have demonstrated that by enhancing the central discriminator within the channel-independent framework—integrating methods like TimesNet and attention mechanisms—and ultimately introducing CCD, significant performance improvements can be achieved, particularly in small sample scenarios.

Our contributions offer a more balanced assessment of both channel-independent and diffusion-based methods, underscoring their value in time series generation tasks. This work provides a more nuanced framework for selecting TSG methods, addressing both the data-specific needs of users and the current state of the field. Future research should continue to investigate the evolving landscape of MTSG, with an emphasis on addressing periodicity and inter-channel correlation challenges. With the conditional time series generation problem posed, it makes sense to explore the advantages and disadvantages of these schemes.

References

- [1] Ahmed Alaa, Alex James Chan, and Mihaela van der Schaar. 2021. Generative time-series modeling with fourier flows. In *International Conference on Learning Representations*.

- [2] Yihao Ang, Qiang Huang, Yifan Bao, Anthony K. H. Tung, and Zhiyong Huang. 2023. Tsgbench: time series generation benchmark. *Proc. VLDB Endow.*, 17, 3, (Nov. 2023), 305–318.
- [3] Yihao Ang, Qiang Huang, Anthony KH Tung, and Zhiyong Huang. 2023. A stitch in time saves nine: enabling early anomaly detection with correlation analysis. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. IEEE, 1832–145.
- [4] Donald J Berndt and James Clifford. 1994. Using dynamic time warping to find patterns in time series. In *Proceedings of the 3rd international conference on knowledge discovery and data mining*, 359–370.
- [5] David Campos, Tung Kieu, Chenjuan Guo, Feiteng Huang, Kai Zheng, Bin Yang, and Christian S Jensen. 2021. Unsupervised time series outlier detection with diversity-driven convolutional ensembles—extended version. *arXiv preprint arXiv:2111.11108*.
- [6] Luis Candanedo. 2017. Appliances energy prediction. UCI Machine Learning Repository. DOI: <https://doi.org/10.24432/C5VC8G>. (2017).
- [7] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. 2018. Neural ordinary differential equations. *Advances in neural information processing systems*, 31.
- [8] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. (2014). arXiv: 1406.1078 [cs.LG].
- [9] Razvan-Gabriel Cirstea, Bin Yang, Chenjuan Guo, Tung Kieu, and Shirui Pan. 2022. Towards spatio-temporal aware traffic time series forecasting. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 2900–2913.
- [10] Andrea Coletta, Sriram Gopalakrishnan, Daniel Borrajo, and Svitlana Vyetenko. 2023. On the constrained time-series generation problem. In *Advances in Neural Information Processing Systems*. A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, (Eds.) Vol. 36. Curran Associates, Inc., 61048–61059.
- [11] Abhyuday Desai, Cynthia Freeman, Zuhui Wang, and Ian Beaver. 2021. Timevae: a variational auto-encoder for multivariate time series generation. *arXiv preprint arXiv:2111.08095*.
- [12] Chris Donahue, Julian McAuley, and Miller Puckette. 2018. Adversarial audio synthesis. *arXiv preprint arXiv:1802.04208*.
- [13] Luo donghao and wang xue. 2024. ModernTCN: a modern pure convolution structure for general time series analysis. In *The Twelfth International Conference on Learning Representations*. <https://openreview.net/forum?id=vpJMjXHU>.
- [14] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. 2017. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*.
- [15] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial networks. (2014). arXiv: 1406.2661 [stat.ML].
- [16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (June 2016).
- [17] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9, 8, 1735–1780.
- [18] Hongbin Huang, Minghua Chen, and Xiao Qiao. 2024. Generative learning for financial time series with irregular and scale-invariant patterns. In *The Twelfth International Conference on Learning Representations*. <https://openreview.net/forum?id=CdjnzWsQax>.
- [19] Jon Hutchins. 2006. Dodgers Loop Sensor. UCI Machine Learning Repository. DOI: <https://doi.org/10.24432/C51P50>. (2006).
- [20] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. 2019. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33, 4, 917–963.
- [21] Daniel Jarrett, Ioana Bica, and Mihaela van der Schaar. 2021. Time-series generation by contrastive imitation. *Advances in Neural Information Processing Systems*, 34, 28968–28982.
- [22] Paul Jeha, Michael Bohlke-Schneider, Pedro Mercado, Shubham Kapoor, Rajbir Singh Nirwan, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. 2021. Psa-gan: progressive self attention gans for synthetic time series. In *International Conference on Learning Representations*.
- [23] Jinsung Jeon, Jeonghak Kim, Haryong Song, Seunghyeon Cho, and Noseong Park. 2022. Gt-gan: general purpose time series synthesis with generative adversarial networks. *Advances in Neural Information Processing Systems*, 35, 36999–37010.
- [24] James Jordon, Jinsung Yoon, and Mihaela Van Der Schaar. 2018. Pate-gan: generating synthetic data with differential privacy guarantees. In *International conference on learning representations*.
- [25] Patrick Kidger, James Foster, Xuechen Li, and Terry J Lyons. 2021. Neural sdes as infinite-dimensional gans. In *International conference on machine learning*. PMLR, 5453–5463.
- [26] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.
- [27] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. 2021. Diffwave: a versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*. <https://openreview.net/forum?id=a-xFK8Ymz5J>.
- [28] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, 95–104.
- [29] Daesoo Lee, Sara Malacarne, and Erlend Aune. 2023. Vector quantized time series generation with a bidirectional prior model. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research)*. Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent, (Eds.) Vol. 206. PMLR, (25–27 Apr 2023), 7665–7693.
- [30] Hongming Li, Shujian Yu, and Jose Principe. 2023. Causal recurrent variational autoencoder for medical time series generation. *arXiv preprint arXiv:2301.06574*.
- [31] Shujian Liao, Hao Ni, Lukasz Szpruch, Magnus Wiese, Marc Sabate-Vidales, and Baoren Xiao. 2020. Conditional sig-wasserstein gans for time series generation. *arXiv preprint arXiv:2006.05421*.
- [32] Zinan Lin, Alankar Jain, Chen Wang, Giulia Fanti, and Vyas Sekar. 2020. Using gans for sharing networked time series data: challenges, initial promise, and open questions. In *Proceedings of the ACM Internet Measurement Conference*, 464–483.
- [33] Olof Mogren. 2016. C-rnn-gan: continuous recurrent neural networks with adversarial training. *arXiv preprint arXiv:1611.09904*.
- [34] Ilan Naiman, N. Benjamin Erichson, Pu Ren, Michael W. Mahoney, and Omri Azencot. 2024. Generative modeling of regular and irregular time series data via koopman VAEs. In *The Twelfth International Conference on Learning Representations*. <https://openreview.net/forum?id=eY7sLb0dVF>.
- [35] Hao Ni, Lukasz Szpruch, Marc Sabate-Vidales, Baoren Xiao, Magnus Wiese, and Shujian Liao. 2021. Sig-wasserstein gans for time series generation. In *Proceedings of the Second ACM International Conference on AI in Finance*, 1–8.
- [36] Hao Ni, Lukasz Szpruch, Magnus Wiese, Shujian Liao, and Baoren Xiao. 2020. Conditional sig-wasserstein gans for time series generation. *arXiv preprint arXiv:2006.05421*.
- [37] Emanuel Parzen. 1963. On spectral analysis with missing observations and amplitude modulation. *Sankhyā: The Indian Journal of Statistics, Series A*, 383–392.
- [38] Jeha Paul, Bohlke-Schneider Michael, Mercado Pedro, Kapoor Shubham, Singh Nirwan Rajbir, Flunkert Valentin, Gasthaus Jan, and Januschowski Tim. 2022. Psa-gan: progressive self attention gans for synthetic time series. (2022). arXiv: 2108.00981 [cs.LG].
- [39] Hengzhi Pei, Kan Ren, Yuqing Yang, Chang Liu, Tao Qin, and Dongsheng Li. 2021. Towards generating real-world time series data. In *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE, 469–478.
- [40] Gorgia Ramponi, Pavlos Protopapas, Marco Brambilla, and Ryan Janssen. 2018. T-cgan: conditional generative adversarial network for data augmentation in noisy time series with irregular sampling. *arXiv preprint arXiv:1811.08295*.
- [41] Oliver Roesler. 2013. EEG Eye State. UCI Machine Learning Repository. DOI: <https://doi.org/10.24432/C57G7J>. (2013).
- [42] Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. 2019. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32.
- [43] Ali Seyfi, Jean-Francois Rajotte, and Raymond Ng. 2022. Generating multivariate time series with common source coordinated gan (cosci-gan). *Advances in Neural Information Processing Systems*, 35, 32777–32788.
- [44] Mohammad Shokoohi-Yekta, Bing Hu, Hongxia Jin, Jun Wang, and Eamonn Keogh. 2017. Generalizing dtw to the multi-dimensional case requires an adaptive approach. *Data mining and knowledge discovery*, 31, 1–31.
- [45] Kaleb E Smith and Anthony O Smith. 2020. Conditional gan for timeseries generation. *arXiv preprint arXiv:2006.16477*.
- [46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.
- [47] Lei Wang, Liang Zeng, and Jian Li. 2023. Aec-gan: adversarial error correction gans for auto-regressive long time-series generation. In *Proceedings of the AAAI Conference on Artificial Intelligence* number 8. Vol. 37, 10140–10148.
- [48] Magnus Wiese, Robert Knobloch, Ralf Korn, and Peter Kretschmer. 2020. Quant gans: deep generation of financial time series. *Quantitative Finance*, 20, 9, 1419–1440.
- [49] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. 2022. Timesnet: temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*.
- [50] Xinle Wu, Dalin Zhang, Chenjuan Guo, Chaoyang He, Bin Yang, and Christian S Jensen. 2021. Autots: automated correlated time series forecasting. *Proceedings of the VLDB Endowment*, 15, 4, 971–983.

- [51] Saining Xie, Ross Girshick, Piotr Dollar, Zhuowen Tu, and Kaiming He. 2017. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (July 2017).
- [52] Tianlin Xu, Li Kevin Wenliang, Michael Munn, and Beatrice Acciaio. 2020. Cot-gan: generating sequential data via causal optimal transport. *Advances in neural information processing systems*, 33, 8798–8809.
- [53] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. 2019. Time-series generative adversarial networks. In *Advances in Neural Information Processing Systems*. H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, (Eds.) Vol. 32. Curran Associates, Inc.
- [54] Xinyu Yuan and Yan Qiao. 2024. Diffusion-TS: interpretable diffusion for general time series generation. In *The Twelfth International Conference on Learning Representations*. <https://openreview.net/forum?id=4h1apFjO99>.
- [55] Yu Zheng, Xiuwen Yi, Ming Li, Ruiyuan Li, Zhangqing Shan, Eric Chang, and Tianrui Li. 2015. Forecasting fine-grained air quality based on big data. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, 2267–2276.
- [56] Linqi Zhou, Michael Poli, Winnie Xu, Stefano Massaroli, and Stefano Ermon. 2023. Deep latent state space models for time-series generation. In *Proceedings of the 40th International Conference on Machine Learning* (Proceedings of Machine Learning Research). Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, (Eds.) Vol. 202. PMLR, (23–29 Jul 2023), 42625–42643.

A CCDGAN

Since time series generation tasks are motivated by small sample sizes, the central discriminator in the channel independent generation framework is essentially a small sample time series classifier. The main challenges in small-sample time series classification arise from the difficulty of adequately training neural networks with limited data, which reduces their classification performance. Possible solutions include: 1) performing data augmentation to increase the data size; 2) decomposing the problem to reduce the classification difficulty; 3) enhancing the information to capture key features that aid classification; and 4) using a small-parameter, high-performance mapping framework. The designed CCD is shown in Figure 2.

A.0.1 Period Block. In multivariate time series, changes in one channel often have short-term effects on other channels. COSCI-GAN [43] may tend to overlook the crucial mechanism of sharing information between adjacent time steps. Additionally, considering the periodic nature of time series, models should be able to capture time patterns across different cycles. Each time point involves two types of temporal changes: **intra-cycle changes** and **inter-cycle changes**, which correspond to adjacent regions and different phases of the same cycle.

For a multivariate time series T , we utilize Frequency to obtain its periodicity in the frequency domain.

$$\alpha = \text{Avg}(\text{Amp}(\text{Frequency}(T))) \quad (4)$$

where $\text{Frequency}(\cdot)$ represents the solution to obtain the frequency. Here, we use FFT or WT. α represents the calculated amplitude of each frequency, which is averaged from the dimensions N using $\text{Avg}(\cdot)$.

In consideration of the sparsity in the frequency domain and to avoid noise caused by irrelevant high frequencies, we conducted testing and found that selecting only the first amplitude value is sufficient. We denote the non-normalized amplitude as α and obtain the most significant frequency f from it. The true periodicity of T is computed using $l_p = \lceil \frac{1}{f} \rceil$. Depending on the selected frequency and the corresponding period length, we can transform the dimension of the multivariate time series T and \hat{T} into $(K, l_p, N \times f)$.

$$T_p \in \mathcal{R}^{K \times l_p \times f}, \hat{T}_p \in \mathcal{R}^{K \times l_p \times f} = \text{Reshape}(T, \hat{T}) \quad (5)$$

It is worth noting that this transformation enhances three types of local information to the transformed 3D tensor: 1) local information between adjacent time points in the same channel (within one cycle), 2) local information between adjacent periods in the same channel (across cycles), and 3) local information of adjacent time steps in different channels. Therefore, through the receptive field of the convolution structure, CCD can better preserve the correlations between channels.

A.0.2 Conv2d Block. After passing through the Period module, we utilize multiple conv2d blocks to capture the three types of local information mentioned earlier, distinguishing between real and synthesized multivariate time series. The formulation is as follows:

$$t_c, \hat{t}_c = \text{Conv2d Block}(T_p, \hat{T}_p) \quad (6)$$

where we transform 2D representations $T_p, \hat{T}_p \in \mathcal{R}^{K \times l_p \times f}$ into 1D space $t_c, \hat{t}_c \in \mathcal{R}^{K \times d}$.

After conducting tests, we recommend adopting a progressive kernel size expansion strategy for convolution operations. The specific sizes for the kernel are $\text{kernel size} = 2 \times i + 1$, where $i = 1, 2, \dots, k$ and k is the number of Conv2d blocks. In addition to the aforementioned approach, we can choose various convolution modules from computer vision, such as widely used models like ResNet [16] and ResNeXt [51], for feature extraction within the Conv2d Block. Generally, stronger 2D backbones for representation learning lead to better performance. Considering both performance and efficiency, we opt for experiments using the `nn.conv2d()` block based on PyTorch for our main experiments. Later we will talk about the influence on kernel size.

Finally, the features extracted through the Conv2d blocks are further processed by the feedforward module, which consists of a linear layer followed by a sigmoid activation function, yielding the ultimate classification results.

A.0.3 Loss Function. In designing the loss function, our approach propagates the CD loss to each single-channel time series generation module. By incorporating CD loss terms, we aim to enhance the modules' focus on inter-channel correlations.

If we select pure GAN-based methods to be the single-channel time series generation module, the objectives function can be summarized as follows:

$$\begin{aligned} \min_{\theta_i} \max_{\phi_i} \max_{\alpha} \mathcal{L} = & \mathbb{E}_{t_i \sim P_{\text{data}}} [\log(D_{i,\phi_i}(t_i))] \\ & + \gamma \cdot \log(\text{CCD}_{\alpha}(t_i)) \\ & + \mathbb{E}_{z \sim P_z} [\log(1 - D_{i,\phi_i}(G_{i,\theta_i}(z)))] \\ & + \gamma \cdot \log(1 - \text{CCD}_{\alpha}(G_{i,\theta_i}(z), G_{j \neq i})). \end{aligned} \quad (7)$$

where G_{i,θ_i} represents the generator of the i -th channel independent GAN with parameters θ_i , and D_{i,ϕ_i} represents the discriminator of the i -th channel independent generator with parameters ϕ_i . CCD_{α} represents the central discriminator with parameter α . P_{data} represents the distribution of real time series. $G_{j \neq i}$ represents all other generators with fixed parameters, except G_{i,θ_i} , in the optimization steps. γ is a hyper-parameter that controls the balance between three types of locality among channels and generates higher-quality signals within each channel. z is a shared noise vector sampled from the distribution P_z .

A.0.4 Model Training. Our training approach using the channel-independent framework is similar to that of COSCI-GANs. In each training iteration, N channel-independent generators are trained. These modules generate single-channel time series \hat{t}_i for their respective channels, which are then concatenated to form \hat{T} . This concatenated time series \hat{T} , along with the real MTS T , is then used to train the CCD and channel-independent generators.

B Additional Information for Simulated Datasets

B.0.1 Simulated Datasets. To effectively assess the performance and significance of CCD, we require a customizable time series dataset. Therefore, we utilized three synthetic datasets with dual channels as proposed by [43].

Simple sine The formula for a basic sine function is given as: $x = A \sin(2\pi ft) + \epsilon$, where A , f , and ϵ are hyperparameters. Across all time series data, channel 1 maintains a frequency f of 0.01, while channel 2's frequency is set to 0.005. For the first data type, the amplitude A is sampled from $N(0.4, 0.05)$, and for the second type, it is sampled from $N(0.6, 0.05)$.

Sine with frequency changes This variant is derived from a basic sine wave with a doubled frequency at the midpoint of the time series. By altering the frequency, we can observe how the model generates data with varying frequencies.

Anomalies Anomalies are generated from basic sine waves by replacing the original data at the midpoint of the time series with Gaussian noise. This allows us to evaluate the model's performance in generating data with outliers.

B.0.2 Parameters. The CCD module was configured with $k = 3$. Moreover, the training epochs were fixed at 500, simulating a resource and time-limited scenario, enabling a more effective evaluation of CCD.

B.0.3 Metrics. Following [43], we conducted a quantitative comparison of the correlation matrices between the two channels using several metrics: (1) Mean Squared Error (MSE), (2) Frobenius norm (FN), (3) Spearman's ρ , and (4) Kendall's τ . The MSE and Frobenius norm metrics indicate a higher similarity between the correlation matrices of the real dataset and the generated synthetic dataset when smaller values are obtained. On the other hand, Spearman's coefficient and Kendall's coefficient approach a value of 1 as the similarity increases.

C Dataset Selection

Dataset Selection. To ensure reproducibility and mitigate biases or oversimplification in our evaluations, we exclusively employ publicly available, real-world datasets. It is crucial to emphasize that our objective is not to accumulate an exhaustive collection of datasets, but rather to curate a diverse set encompassing multiple domains, showcasing varied data statistics and distributions. Table 1 summarizes their statistics. Below, we provide a brief description of each dataset.

- Stock [53]. It comprises daily historical Google stock data from 2004 to 2019, including volume and high, low, opening, closing, and adjusted closing prices.
- Stock Long [53]. It is identical to the Stock dataset but with a sequence length of 125.
- Energy [6]. It includes information on appliance's energy use in a low-energy building.
- Energy Long [6]. It is identical to the Energy dataset but with a sequence length of 125.
- EEG [41]. It is with the measurements derived from ElectroEncephaloGraphy (EEG) data captured by Emotiv EEG Neuroheadset. It helps to understand brainwave patterns, especially those under specific cognitive conditions or stimuli.
- Dodgers Loop Game (DLG) [19]. It consists of loop sensor data from the Glendale on-ramp for the 101 North freeway in Los Angeles.

- Air [55]. It has air quality, meteorological, and weather forecast data from 4 major Chinese cities: Beijing, Tianjin, Guangzhou, and Shenzhen from 2014/05/01 to 2015/04/30.

This study selected five datasets from different real-world domains: the Stock dataset from the financial domain, the Energy dataset from the energy sector, the EEG dataset from biological signals, the DLG dataset from traffic, and the Air dataset. To better investigate the impact of subsequence length on the model, we set different values of l for the Stock and Energy datasets. Therefore, our method is applicable to real cases or practical applications.

D Evaluation Measure Suite

Numerous metrics are available to assess the quality of TSG (Time Series Generation) methods, which commonly adhere to principles such as diversity, fidelity, and utility.

D.0.1 Model-based Measures. These measures primarily follow the TSTR scheme [14, 24], wherein synthetically generated series are utilized to train a post-hoc neural network, which is then evaluated on the original time series.

- Discriminative Score (DS) [53]. This study utilizes a post-hoc time-series classification model, leveraging 2-layer GRUs or LSTMs, to discern between original and generated series [53]. The original series are denoted as "real," whereas the generated series are labeled as "synthetic." Subsequently, an RNN classifier is trained using these labels. The fidelity of the generation model is assessed by measuring the classification error on a separate test set.
- Predictive Score (PS) [53]. It focuses on training a post-hoc time series prediction model using synthetic data [53]. The model leverages GRUs or LSTMs to predict either the temporal vectors of each input series for future steps [53, 21] or the entire vector [23]. To assess performance, the model is evaluated on the original dataset using the mean absolute error metric.

D.0.2 Feature-based Measures. These measures aim to capture inter-series correlations and temporal dependencies, evaluating the extent to which the generated time series preserves the original characteristics. Feature-based measures offer a distinct advantage by providing clear and deterministic results, ensuring an unambiguous assessment of the quality of the generated time series.

- Marginal Distribution Difference (MDD) [35]. This measure calculates empirical histograms for each dimension and time step in the generated series. The bin centers and widths from the original series are used for this purpose. By computing the average absolute difference between these histograms and those of the original series across bins, it assesses the alignment of the distributions between the original and generated series.
- AutoCorrelation Difference (ACD) [28]. This measure calculates the autocorrelation of both the original and generated time series and determines their difference [37, 28]. By comparing the autocorrelations, we can assess the preservation of dependencies in the generated time series.
- Skewness Difference (SD). In addition to ACF, this study incorporates statistical measures to assess the quality of the

generated time series [47]. One such measure is skewness, which quantifies the distribution asymmetry of a time series and is crucial for analyzing its marginal distribution. Given the mean (standard deviation) of the train time series T_s^{tr} as μ_s^{tr} (σ_s^{tr}) and the generated time series T_s^{gen} as μ_s^{gen} (σ_s^{gen}), we evaluate the fidelity of T_s^{gen} by computing the skewness difference between them as:

$$SD = \frac{\mathbb{E}[(T_s^{gen} - \mu_s^{gen})^3]}{\sigma_s^{gen3}} - \frac{\mathbb{E}[(T_s^{tr} - \mu_s^{tr})^3]}{\sigma_s^{tr3}}. \quad (8)$$

- **Kurtosis Difference (KD).** Similar to skewness, kurtosis is employed to evaluate the tail behavior of a distribution, uncovering extreme deviations from the mean. Using notations from Equation 8, the kurtosis difference between T_s^{tr} and T_s^{gen} is calculated as:

$$KD = \frac{\mathbb{E}[(T_s^{gen} - \mu_s^{gen})^4]}{\sigma_s^{gen4}} - \frac{\mathbb{E}[(T_s^{tr} - \mu_s^{tr})^4]}{\sigma_s^{tr4}}. \quad (9)$$

D.0.3 Training Efficiency. Training efficiency plays a critical role, especially in scenarios that require fast time series generation methods or when computational resources are limited. However, only a limited number of studies, such as [11, 23], have been utilized for evaluation in this particular context.

- **Training Time.** Training time, referring to the wall clock time required for training a time series generation (TSG) method, is a crucial metric for evaluating and deploying TSG methods. It holds significant importance due to economic considerations.

D.0.4 Distance-based Measures. To address the challenges associated with data synthesis (DS) and privacy preservation (PS), we propose the integration of two distance-based measures as a means of achieving an efficient and deterministic evaluation.

- **Euclidean Distance (ED).** The mean of the Euclidean distance (ED) is computed for all series and samples. As the input time series has been preprocessed to fall within the range of [0, 1], ED allows for a deterministic evaluation of the similarity between s^{gen} and s^{tr} . It facilitates a value-wise comparison of the time series.
- **Dynamic Time Warping (DTW)** [4]. In order to account for alignment, we incorporate DTW to capture the optimal alignment between series, regardless of their pace or timing. The alignment facilitated by DTW provides valuable insights into the predictive quality of the generated series. Additionally, studies such as [44] have demonstrated that multi-dimensional DTW can enhance downstream classification tasks, making it a discriminative measure.

By leveraging the metrics of ED and DTW, we can efficiently and effectively assess the quality of generated time series. These metrics provide streamlined alternatives to evaluate time series generation, with similar goals as those of DS and PS.

D.1 AED and AWD

Time series generation tasks consider two key factors: Fidelity and Correlation Preservation. Fidelity aims to generate results that match the distribution of the original time series while avoiding mode collapse. We use the Average Wasserstein Distance (AWD) to reflect the diversity of the generated results. A smaller AWD

indicates a closer resemblance to the true distribution and better fidelity. Correlation Preservation aims to maintain the same channel correlations as the original time series for different channels of the generated results. We map the generated time series onto a two-dimensional plane and calculate the Average Euclidean Distance (AED) between the generated series and the line with an amplitude and slope of 1. We use AED to measure the correlation preservation of the generated results.

E Results on Synthetic Dataset

E.0.1 Experimental Setup. To effectively assess the performance and significance of CCD, we require a customizable time series dataset. Therefore, we utilized three synthetic datasets (*Simple sine*, *Sine with frequency changes*, and *Anomalies*) with dual channels as proposed by [43]. Following [43], we conducted a quantitative comparison of the correlation matrices between the two channels using several metrics: (1) Frobenius norm (FN), (2) Spearman’s ρ , and (3) Kendall’s τ . The detailed information on datasets, parameters, and metrics can be found in Appendix Section B.

E.0.2 Comparison with COSCIGAN. We compare our method with the state-of-the-art (SoTA) channel-independent method to demonstrate the performance improvements achieved by our method. To ensure experimental fairness, CCDGAN and COSCIGAN utilize the same single-channel generator and single-channel discriminator. The single-channel generator consists of a 1-layer LSTM network and three linear layers, while the single-channel discriminator consists of four linear layers. COSCIGAN’s CD consists of four linear layers. Table 10 presents the results, highlighting significant advancements in three key metrics. For the Anomalies dataset, our performance in Kendall’s τ falls short of COSCIGAN. This discrepancy can be attributed to Kendall’s τ ’s focus on order correlation while remaining insensitive to outliers. Given the Anomalies dataset’s substantial number of anomalies, the resulting weaker order correlation adversely impacts the fairness of metric evaluation.

E.0.3 Analysis of Channel Independent Generator. We explore the impact of different single-channel TSG modules on the overall framework. Considering the channel independence, we selected MLP, GRU, TimeGAN, and VAE as our channel-independent TSG modules incorporating the CCD as the central discriminator.

If we select pure VAE-based methods to be single-channel time series generation module, the objectives of these three components can be summarized as follows:

$$\min_{\theta_i} \max_{\alpha} \mathcal{L}(\text{VAE}_{i,\theta_i}, \text{CCD}_{\alpha}) = \mathcal{L}_{\text{VAE}_{i,\theta_i}} + \gamma \cdot \log(1 - \text{CCD}_{\alpha}(\text{VAE}_{i,\theta_i}(z), \text{VAE}_{j \neq i}(z))) \quad (10)$$

$$\begin{aligned} \mathcal{L}_{\text{VAE}_{i,\theta_i}} = & -\mathbb{E}_{i,\theta_i,q(z|t_i)} [\log p(t_i|z)] \\ & + \beta \cdot \text{KL}_{i,\theta_i}(q(z|t_i) \| p(z)) \\ & + \gamma \cdot \log(\text{CCD}_{\alpha}(t_i)) \end{aligned} \quad (11)$$

where $-\mathbb{E}_{q(z|t_i)} [\log p(t_i|z)]$ represents the reconstruction loss in the original loss function of VAE. Here, $q(z|t_i)$ represents the approximate posterior distribution of latent variables generated by the encoder, and $p(t_i|z)$ represents the reconstruction data distribution produced by the decoder. $\text{KL}(q(z|t_i) \| p(z))$ represents the

Table 10: Results of CCDGAN and COSCIGAN on simulated datasets, where bold indicates methods that perform well in the respective metric. The performance of the central discriminator was evaluated by employing the same channel generator. \uparrow represents the larger the value, the better, while \downarrow represents the smaller the value, the better.

Dataset	Method	FN \downarrow	$\rho\uparrow$	$\tau\uparrow$
Simple Sine	CCDGAN	3.725	0.667	0.021
	COSCIGAN	5.002	0.208	-0.01
Freq changes	CCDGAN	1.868	0.659	-0.009
	COSCIGAN	2.285	-0.075	-0.05
Anomalies	CCDGAN	2.041	0.767	-0.005
	COSCIGAN	2.531	0.032	0.02

Table 11: The types of central discriminator (MLP or CCD) on the performance of channel independent framework. The smaller the values for AWD and AED, the better.

Dataset	CD type	AWD \downarrow	AED \downarrow
Simple Sine	None	0.047	0.133
	MLP	0.08	0.018
	CCD	0.055	0.014
Freq changes	None	0.04	0.077
	MLP	0.068	0.024
	CCD	0.061	0.017
Anomalies	None	0.054	0.077
	MLP	0.073	0.077
	CCD	0.066	0.071

KL divergence loss, where $p(z)$ represents the prior distribution, typically assumed to be a multivariate Gaussian distribution. β and γ are hyperparameters used to balance the KL divergence loss and the CCD loss.

Table 12 indicates that choosing alternative modules for the single-channel TSG modules is a viable approach. Regarding the module selection, it is advisable to opt for methods that exhibit superior performance in channel-mixing techniques while striking a balance between performance and computational resources. We also analyzed the performance of the entire framework when incorporating the non-GAN module VAE as the single-channel TSG module. Using VAE yields inferior results compared to GAN-based approaches. This discrepancy can be attributed to the possibility that the current training objectives of the channel-independent framework may not be suitable for VAE-based methods.

E.0.4 Ablation Study of CCD. We verified the effectiveness of the CCD module in capturing temporal dependencies within multivariate time series, both within and across periods. The Average Wasserstein Distance (AWD) was used to measure the diversity of generated results, while the Average Euclidean Distance (AED) assessed correlation preservation.

A channel-independent GAN method served as the baseline, with identical settings for all channel generators as described in Section E.0.2. The CD type used 'None' to denote the absence of a CD module and 'MLP' for the MLP-based central discriminator, which consists of consists of four linear layers. It concatenates the time

series ($l \times N$) horizontally, resulting in a tensor with dimensions $l \times N$.

Table 11 shows that achieving both high fidelity and strong correlation preservation is challenging. While the central discriminator improves correlation, it may slightly reduce fidelity. In contrast, CCD outperforms the MLP-based central discriminator in both fidelity and correlation preservation.

F Results on Real Dataset

The experimental results for Section 5.5 are included in the supplementary materials due to space limitations.

G Channel Independent TimeVAE adding CCD

We utilized TimeVAE, a VAE-based model, and incorporated CCD loss into the loss function to investigate whether non-GAN models, apart from GAN, can adopt the channel-independent approach. In the following, bold indicates methods that perform well in the respective metric.

H Additional Information for CCDGAN v.s. Diffusion-based Methods

H.0.1 Baselines. We compared CCDGAN with three diffusion model-based approaches: Diffusion-TS[54], DiffWave[27], and DiffTime[10]. These methods, including Diffusion-TS, DiffWave, and DiffTime, are all based on DDPM, with modifications tailored to the characteristics of time series data.

H.0.2 Parameters. For Diffusion-TS, DiffWave and DiffTime, to ensure the fairness of the experiments, we strived to maintain consistency in parameter settings. We chose 4 attention heads, each with a dimension of 16, and selected 2 encoder and decoder layers.

H.0.3 Datasets. For dataset selection, we chose the previously mentioned Stock and Energy datasets. For the Sine dataset, we opted for the sine wave dataset provided in [53], which is channel-independent and exhibits more diverse variations.

H.0.4 Evaluation Metrics. 1) **Context-Fréchet Inception Distance (Context-FID) score** [38] quantifies the quality of the synthetic time series samples by computing the difference between representations of time series that fit into the local context; 2) **Correlational score (CS)** [36] uses the absolute error between cross correlation matrices by real data and synthetic data to assess the temporal dependency.

2) **Correlational Score** To mitigate the challenges associated with DS and PS, we propose the incorporation of distance-based measures to provide an efficient, deterministic evaluation.

3) **Discriminative Score** For a quantitative measure of similarity, we train a post-hoc time-series classification model (by optimizing a 2-layer LSTM) to distinguish between sequences from the original and generated datasets. First, each original sequence is labeled real, and each generated sequence is labeled not real. Then, an off-the-shelf (RNN) classifier is trained to distinguish between the two classes as a standard supervised task.

Table 12: The results of different channel-independent generators with CCD. Among them, channel-independent GANs based on MLP, GRU, and TimeGAN were employed. We also utilized VAE to investigate whether non-GAN models, apart from GAN, can adopt the channel-independent approach.

Dataset	Model	FN↓	ρ ↑	τ ↑
Simple Sine	MLP	6.737	0.122	-0.108
	GRU	5.509	0.235	-0.01
	TimeGAN	5.268	0.288	-0.007
	VAE	7.338	0.170	-0.016
Freq changes	MLP	3.004	-0.144	-0.198
	GRU	2.291	-0.081	-0.059
	TimeGAN	2.334	-0.118	-0.124
	VAE	5.004	-0.104	-0.037
Anomalies	MLP	4.889	-0.1	-0.016
	GRU	2.531	0.039	0.032
	TimeGAN	3.829	0.027	0.107
	VAE	3.482	0.002	0.021

Table 13: Detail results about Figure 3 on Stock dataset

Methods Metrics	CCD	TimeVQVAE	TimeVAE	COSCI-GAN	LS4	RTSGAN
MDD↓	0.315	1.186	0.327	0.334	0.502	0.319
ACD↓	0.015	0.017	0.078	0.037	0.031	0.029
SD↓	0.127	0.059	0.4	0.217	0.215	0.138
KD↓	0.446	0.458	2.54	1.647	0.958	0.771
ED↓	1.068	1.051	1.088	1.101	1.098	1.075
DTW↓	2.772	2.791	2.787	3.014	2.781	2.784
PS↓	0.073	0.094	0.084	0.086	0.109	0.088
DS↓	0.327	0.163	0.177	0.396	0.482	0.339
Time	52min11s	1h43min	1min33s	55min35s	47min50s	1h03min

Table 14: Detail results about Figure 3 on Stock Long dataset

Methods Metrics	CCD	TimeVQVAE	TimeVAE	COSCI-GAN	LS4	RTSGAN
MDD↓	0.451	1.207	0.455	0.457	0.482	0.516
ACD↓	0.128	0.143	0.235	0.131	0.135	0.157
SD↓	0.206	0.303	0.528	0.215	0.069	0.238
KD↓	2.223	0.782	2.541	1.414	0.569	1.115
ED↓	2.388	2.697	2.486	2.539	2.722	2.532
DTW↓	6.229	6.269	6.319	6.625	6.698	6.626
PS↓	0.093	0.112	0.086	0.095	0.093	0.084
DS↓	0.314	0.244	0.124	0.399	0.471	0.382
Time↓	49min	4h41min	56s	57min	44min	1h40min

4) **Predictive Score** In order to be useful, the sampled data should inherit the predictive characteristics of the original. In particular, we expect TimeGAN to excel in capturing conditional distributions over time. Therefore, using the synthetic dataset, we train a post-hoc sequence-prediction model (by optimizing a 2-layer LSTM) to predict next-step temporal vectors over each input sequence. Then, we evaluate the trained model on the original dataset.

5) **Training Time** It refers to the wall clock time for training a TSG method. It is a vital measure for evaluating and deploying TSG methods due to economic considerations.

6) **2000 Data Sampling Time** It evaluates the time spent by the model in generating data, which was not previously emphasized, but significantly affects the user experience with the model. Additionally, methods based on diffusion models require longer generation times, leading to increased attention to sampling time.

Table 15: Detail results about Figure 3 on Energy dataset

Methods Metrics	CCD	TimeVQVAE	TimeVAE	COSCI-GAN	LS4	RTSGAN
MDD↓	0.289	0.328	0.386	0.307	0.294	0.306
ACD↓	0.023	0.074	0.118	0.028	0.075	0.099
SD↓	0.062	0.129	0.138	0.07	0.218	0.34
KD↓	0.572	0.421	0.674	0.588	0.824	0.723
ED↓	0.966	0.956	0.987	0.972	0.92	0.775
DTW↓	6.045	6.025	5.799	6.267	5.928	4.984
PS↓	0.247	0.252	0.288	0.256	0.492	0.311
DS↓	0.419	0.335	0.487	0.469	0.471	0.488
Time↓	2h32min	3h27min	15min19s	2h11min	1h38min	1h9min

Table 16: Detail results about Figure 3 on Energy Long dataset

Methods Metrics	CCD	TimeVQVAE	TimeVAE	COSCI-GAN	LS4	RTSGAN
MDD↓	0.319	0.413	0.431	0.328	0.783	0.433
ACD↓	0.103	0.123	0.27	0.109	0.29	0.403
SD↓	0.088	0.153	0.183	0.093	0.592	0.192
KD↓	0.289	0.185	0.621	0.489	6.735	0.823
ED↓	2.017	2.189	2.035	2.297	2.342	2.349
DTW↓	13.377	13.549	12.527	13.776	13.788	14.325
PS↓	0.311	0.253	0.289	0.254	0.488	0.253
DS↓	0.477	0.492	0.499	0.483	0.486	0.496
Time↓	1h57min	2h58min	43min	1h42min	3h05min	2h24min

Table 17: Detail results about Figure 3 on EEG dataset

Methods Metrics	CCD	TimeVQVAE	TimeVAE	COSCI-GAN	LS4	RTSGAN
MDD↓	0.208	0.214	0.347	0.316	0.249	0.233
ACD↓	0.043	0.068	0.148	0.047	0.143	0.076
SD↓	0.167	0.188	0.241	0.174	0.381	0.27
KD↓	0.509	0.361	0.958	0.662	1.51	0.663
ED↓	1.331	1.693	1.517	1.831	1.578	1.762
DTW↓	6.263	6.371	6.538	7.182	6.121	7.049
PS↓	0.041	0.033	0.039	0.045	0.218	0.041
DS↓	0.442	0.339	0.475	0.451	0.491	0.404
Time↓	2h02min	4h30min	42min	2h12min	3h51min	2h12min

Table 18: Channel Independent TimeVAE adding CCD.

Dataset	Model	MSE↓	FN↓	$\rho\uparrow$	$\tau\uparrow$
Simple Sine	Without CCD	0.167	9.071	-0.019	-0.129
	With CCD	0.121	7.338	0.17	-0.016
Freq changes	Without CCD	0.179	6.382	-0.191	-0.235
	With CCD	0.123	5.004	-0.104	-0.037
Anomalies	Without CCD	0.236	7.711	-0.136	-0.113
	With CCD	0.143	3.482	0.002	0.021

I Analysis of Channel-independent Framework

1.0.1 Exploration of Different Channel Independent Generators. Following [43], we conducted a quantitative comparison on three synthetic datasets (*Simple sine*, *Sine with frequency changes*, and *Anomalies*). We explore the impact of different channel independent generators on the overall framework. The detailed results can be found in Appendix Section E. TimeGAN is a better team for CCD than MLP, GRU, and VAE.

Table 19: Analysis of Period Block on Stock dataset. WT stands for Wavelet transform.

Period Block	FFT	WT
MDD↓	0.315	0.313
ACD↓	0.015	0.019
SD↓	0.127	0.132
KD↓	0.446	0.448
ED↓	1.068	1.062
DTW↓	2.772	2.779
PS↓	0.073	0.086
DS↓	0.327	0.319

J TSGGuide versus TSGBench

In contrast to TSGBench [2], the updated sections are marked in blue.

J.1 TSGBench

- (1) As a foundational step, we advocate for users to commence with VAE-based methods (e.g., TimeVAE and LS4). Their consistent leading performance and superior computational efficiency make them go-to choices for initial exploration.
- (2) In applications that emphasize autocorrelation or forecasting, such as predictive maintenance or stock market analysis, the ACD measure becomes crucial. Fourier Flow, which is recognized for maintaining temporal dependencies, is highly suitable for these scenarios. On the other hand, for capturing complex multi-variate relationships in datasets, COSCI-GAN is the recommended choice.
- (3) Subsequent considerations focus on dataset size and domain specificity. For small-sized datasets, RTSGAN and LS4, which excel in single DA, are strong choices. For heterogeneous datasets, or when the goal is to generate time series for a new target domain, TimeVAE and COSCI-GAN stand out for their effectiveness in cross DA.
- (4) Users can further fine-tune their method selection based on specific real-world application needs, which involves identifying the most relevant evaluation measures. In this case, Figure 1 serves as a valuable visual guide.

J.2 TSGGuide

- (1) In Figure 3, VAE-based methods demonstrate faster training times and rank above average on several metrics. This makes VAE-based methods suitable for initial attempts due to their lower time requirements. As a foundational step, we recommend that users start with the VAE-based method. Its

consistently excellent computational efficiency makes it the preferred choice for initial exploration and handling of small data sets.

- (2) If we emphasize autocorrelation or forecasting, such as predictive maintenance or stock market analysis, the ACD measure becomes crucial. CCDGAN is highly suitable for these scenarios.
- (3) In Figure 3, CCD shows a leading performance on the majority of datasets, proving that CCDGAN performs well on real-world datasets, even though they come from different domains. When the dataset originates from a novel domain or exhibits complex multivariate relationships, CCDGAN is a recommended choice. It shows excellent performance across data sets from various domains and outperforms other methods, particularly on datasets with a large number of variables.
- (4) In Table 4, diffusion model-based methods achieve certain advantages in metrics. However, these methods require significant training and sampling time, leading to higher computational costs. Therefore, if ample computational resources are available, diffusion model-based approaches should be considered. When computational resources and dataset are abundant and optimal results are desired, it is recommended to utilize Diffusion-TS. These methods typically exhibit stable training and yield superior performance.
- (5) Users can further adjust method selection based on specific application requirements, including identifying the appropriate channel-independent TSG module for CCDGAN (see Appendix Section E).

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009