Stonk Bench: Unified Benchmark for Synthetic Data Generation for Financial Time Series

Uyen Lam Ho*
Eddison Pham*
uyenlam.ho@mail.utoronto.ca
eddison.pham@mail.utoronto.ca
University of Toronto
Toronto, Ontario, Canda

Figure 1: Toronto Stock Exchange, Toronto, 1981

Abstract

The ever emerging field of Synthetic Data Generation (SDG) has gain traction within many financial domains, including Finacial Time Series (FTS) data. However, there has been discorse and lack in universal agreement in what determines a better SDGFTS (Synthetic Data Generation for Financial Time Series), which maybe hindering progress in the field. In this paper, we propose and contribute a comprehensive benchmark for SDGFTS, covering various aspects such as data quality, privacy, and utility. We evaluate several state-of-the-art SDG methods using our benchmark and provide insights into their strengths and weaknesses. Our first-of-its-kind benchmark aims to facilitate the development of more effective SDG methods for FTS data, incorperating both classical statistical measures and utility benefits, and test over (insert number) to decicively determine what is the best SDGFTS model right now and for the future.

*Both authors contributed equally to this research.

Conference'17, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Keywords

Synthetic Data Generation, Financial Time Series, Benchmarking

ACM Reference Format:
Uyen Lam Ho and Eddison Pham. 2025. Stonk Bench: Unified Benchmark for Synthetic Data Generation for Financial Time Series. In . ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Synthetic Data Generation (SDG) has emerged as a crucial tool in financial technology, particularly for Financial Time Series (FTS) data [3]. The ability to generate high-quality synthetic financial data addresses several critical challenges in the field, including data privacy concerns, limited data availability, and the need for diverse training datasets in machine learning applications [4].

Despite the growing importance of SDGFTS (Synthetic Data Generation for Financial Time Series), there is a notable lack of standardization in evaluating the quality and effectiveness of these generative models [4]. Current evaluation methods vary widely across studies, making it difficult to compare different approaches objectively and determine their relative strengths and weaknesses.

Our research addresses this gap by introducing a comprehensive benchmark framework that encompasses:

- Statistical fidelity measures for comparing synthetic and real FTS
- Privacy preservation metrics to ensure sensitive financial information remains protected
- Utility metrics that assess the practical value of synthetic data in downstream tasks

- Performance evaluation across multiple SDGFTS models and datasets

By analyzing the results from our MLflow experiments and applying our unified benchmark, we aim to provide clear guidelines for evaluating SDGFTS models and establish a standard for future research in this domain.

1.1 Limitations in the SDGFTS Literature

Among other things, we observed that there is currently no universal, generally accepted approach to evaluating synthetic time series[4]; this issue extends beyond FTS to generative frameworks like GANs as a whole [5]. Many evaluation measures are insufficiently defined and lack public implementations, making reuse and reproduction troublesome and error-prone [4]. This presents a challenge unique to the generation task compared to areas like time series forecasting or classification [4]. Hence, future research would immensely benefit from a widely accepted, reasonably sized set of qualified measures for the central evaluation criteria.

1.2 Our Contributions

To put it in simple terms, we are adopting the time series evaluation taxonomy outlined from Stenger et al. [4], develop a comprehensive and unified SDG benchmark expanded from the works of Ang et al. [1] in specifics to FTS by incorporating a utility evaluation framework inspired by Boursin et al. [2] through portfolio evaluations generated by a deep hedger. Our key contributions include:

[C1] **Consolidated Evaluation Framework:** We systematically review and consolidate evaluation methods from leading papers (models and surveys) in SDG and financial time series [1, 2, 4], creating a comprehensive assessment toolkit based on established practices.

[C2] **Unified Statistical and Utility Measures:** For the first time, we integrate both statistical fidelity metrics and practical utility measures in a single SDGFTS benchmark, providing a more complete evaluation of synthetic data quality.

[C3] **Open Benchmark Platform:** We deliver an open-source benchmark framework for the research community, aiming to establish a gold standard for SDGFTS model evaluation and comparison.

2 Preliminaries

2.1 Problem Definition

Let us formally define the Synthetic Data Generation (SDG) problem for Financial Time Series (FTS). Given a financial time series X with N individual series of length T, we represent it as a matrix:

$X = (x_1, ..., x_n)^T$

where each individual series x_i is a T-dimensional vector:

$x_i = (x_{i,1}, ..., x_{i,t})$

and each $x_{i,t}$ corresponds to a single time point t of x_i.

Let $P(x_1, ..., x_n)$ denote the real distribution of the given time series X. The objective of SDG for FTS is to generate a synthetic time series:

$\hat{X} = (\hat{x}_1, ..., \hat{x}_n)$

such that its distribution $P(\hat{x}_1, ..., \hat{x}_n)$ approximates $P(x_1, ..., x_n)$, while preserving key statistical properties and financial characteristics of the original data. These properties include:

2.2 Scope of Project

2.2.1 Scope of Methods. Our benchmark encompasses a diverse range of SDGFTS methods, from traditional parametric approaches to modern deep learning architectures. We selected models based on three main criteria: (1) proven industry adoption, (2) research community acceptance, and (3) recent methodological innovations.

Our evaluation includes classical parametric models, which rely on predefined statistical distributions and assumptions about the underlying data generation process. These models have been extensively used in financial institutions for their interpretability and theoretical foundations.

We also evaluate modern non-parametric approaches, particularly deep learning-based models that learn the data distribution directly from observations without assuming a specific form. These include generative adversarial networks, variational autoencoders, and diffusion models, representing the cutting edge in synthetic data generation.

2.2.2 Scope of Datasets. We evaluate models on diverse financial datasets spanning:

- Stock market indices (S&P 500, NASDAQ, TSX)
- Individual stock prices from major exchanges
- Forex trading pairs

2.2.3 Scope of Evaluation Taxonomical Criteria. Our evaluation framework follows the taxonomy structure proposed by Stenger et al. [4], incorporating various evaluation measures from different papers in the field.

We systematically integrate evaluation metrics from multiple sources while maintaining this structured taxonomical approach, ensuring comprehensive coverage of all critical aspects of SDGFTS evaluation and setting a standard for future research in time series SDG as a whole.

3 Overview of Financial Time Series (FTS) Methods

Financial time series (FTS) methods refer to statistical and machine learning approaches used to model, forecast, or simulate financial data such as prices, returns, and volatility. These methods aim to reproduce key empirical properties including non-stationarity, heavy tails, volatility clustering, leverage effects, and long-memory in absolute returns.

We group them into two main families: classical parametric (stochastic) methods with explicit data-generating assumptions, and modern non-parametric (deep learning) methods that learn complex dependencies directly from data. A detailed overview of the methods considered in our benchmark is provided in the following to sub-sections.

3.1 Classical Statistical (Parametric) Methods

Parametric models assume a specific functional form for the data-generating process, characterised by a finite set of parameters.

These models are interpretable and analytically tractable, allowing for closed-form solutions in many cases. However, they may struggle to capture complex stylised facts or high-dimensional dependencies beyond their structural assumptions. Below we outline several widely-used parametric models in FTS.

Common notation and variables. S_t – asset price, $X_t = \ln S_t$ – log-price, $r_t = \ln(S_t/S_{t-1})$ – log-return, μ – drift, σ – volatility, W_t – standard Brownian motion, Δt – discrete time step.

[A1] Geometric Brownian Motion (GBM). The price process follows a continuous-time stochastic process with constant drift and volatility. Model-specific parameters are μ and σ.

$$dS_t = \mu S_t\, dt + \sigma S_t\, dW_t. \tag{1}$$

[A2] Ornstein–Uhlenbeck (OU). A mean-reverting Gaussian process for log-prices or spreads. Model-specific parameters are $\theta > 0$ (reversion rate), μ (long-term mean), and σ (volatility).

$$dX_t = \theta(\mu - X_t)\, dt + \sigma\, dW_t. \tag{2}$$

[A3] Merton Jump Diffusion (MJD). Extends GBM by incorporating normally-distributed jumps. Model-specific parameters are λ (jump intensity), μ_j and σ_j (mean and standard deviation of jump sizes).

$$\frac{dS_t}{S_t} = \mu\, dt + \sigma\, dW_t + J\, dN_t, \tag{3}$$

where $J \sim \mathcal{N}(\mu_j, \sigma_j^2)$, $\quad N_t \sim \text{Poisson}(\lambda t)$.

[A4] Double-Exponential Jump Diffusion (DEJD/Kou). A jump-diffusion model with asymmetric double-exponential jump sizes. Model-specific parameters are p (probability of upward jump) and η_1, η_2 (exponential parameters for upward/downward jumps).

$$\frac{dS_t}{S_t} = \mu\, dt + \sigma\, dW_t + Y\, dN_t, \tag{4}$$

$$Y = \begin{cases} \text{Exp}(\eta_1), & \text{with probability } p, \\ -\text{Exp}(\eta_2), & \text{with probability } 1-p. \end{cases}$$

[A5] GARCH(1,1). A discrete-time conditional heteroskedastic model for returns. Model-specific parameters are $\omega > 0,\ \alpha \geq 0,\ \beta \geq 0$ (GARCH coefficients) and $\mathcal{D}$ (innovation distribution, usually standard Normal or Student-t). Captures volatility clustering via time-varying conditional variance.

$$r_t = \sigma_t z_t, \quad z_t \sim \mathcal{D}, \tag{5}$$
$$\sigma_t^2 = \omega + \alpha r_{t-1}^2 + \beta \sigma_{t-1}^2$$

[A6] Block Bootstrap (non-parametric). A resampling technique that preserves short-range dependence by sampling contiguous blocks of returns. Although not parametric, it is included here for completeness and baseline comparison.

3.2 Deep Learning–based (Non-parametric) Methods

Non-parametric models (in this context) refer to generative (and often implicit) models that do *not* assume a fixed parametric form for the data-generating process; rather they learn, from data, latent structures via flexible architectures (e.g., neural networks) and can model highly nonlinear, high-dimensional, and multi-modal dependencies (for example multivariate FTS with interactions across assets, time horizons, features). While these models sacrifice direct parametrisation / interpretability, they offer greater freedom and expressive power — at the cost of larger data/training requirements, more hyper-parameter tuning, and less analytic tractability. Below we subdivide into three major deep learning generative model families: GANs, VAEs, Diffusion Models.

3.2.1 GANs (Generative Adversarial Networks). Generative Adversarial Networks (GANs) pit a generator network $G(\cdot)$ (which attempts to mimic the data distribution) against a discriminator (or critic) network $D(\cdot)$ (which attempts to distinguish real from synthetic). The generator is trained to fool the discriminator, and the discriminator is trained to detect the fakes. In the time-series context, one must additionally capture temporal dynamics and cross-feature dependencies.

* Example: the TimeGAN model [6] combines adversarial training with supervised latent-space losses to ensure realistic temporal transitions and feature-level realism. * In such models the loss typically is

$$\min_G \max_D \ \mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] \ + \ \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))] \tag{6}$$

with additional terms for supervised embedding, feature-matching, or temporal-consistency. * These models are very flexible but can suffer from mode collapse, unstable training, and require careful architecture/hyper-parameter tuning.

3.2.2 VAEs (Variational Autoencoders). Variational Autoencoders (VAEs) are generative latent-variable models that learn to map data x into a latent distribution (encoder) $q_\phi(z|x)$, and from latent z reconstruct x

$$\mathcal{L}(\theta, \phi; x) = \mathbb{E}_{z\sim q_\phi(z|x)}\big[\ln p_\theta(x|z)\big] \ - \ D_{KL}\big(q_\phi(z|x) \parallel p(z)\big). \tag{7}$$

In the time-series generation context, e.g., TimeVAE (Desai et al., 2021-22) uses a VAE architecture with temporal encoder/decoder and latent structure capturing sequence-level variability. :contentReference[oaicite:3]index=3 * Pros: more stable to train than GANs, latent space admits interpolation, can incorporate domain knowledge (e.g., trend/seasonality components) as in TimeVAE. * Cons: may under-represent heavy tails or complex multi-modal behaviour (latent distribution is often Gaussian, decoder may produce over-smoothed samples).

3.2.3 Diffusion Models. Diffusion models are a newer class of generative models that define a forward "noising" process (often a Gaussian Markov chain or SDE) gradually mapping data into noise, and then train a neural network to reverse this process (denoising) to sample from the data distribution. :contentReference[oaicite:4]index=4 In time-series generation (or forecasting/imputation) contexts, diffusion models have begun to show strong performance. :contentReference[oaicite:5]index=5 A canonical forward discrete-time chain (DDPM style) might be

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} \mathcal{N}\big(x_t; \sqrt{1-\beta_t}\, x_{t-1},\ \beta_t I\big), \quad x_0 \sim p_{\text{data}} \tag{8}$$

and the learned reverse process

$$p_\theta(x_{t-1} \mid x_t) = \mathcal{N}(x_{t-1};\, \mu_\theta(x_t, t),\, \Sigma_\theta(x_t, t)) \tag{9}$$

In the FTS-generation setting, recent work shows that one can incorporate financial SDE structure (e.g., heteroskedastic noise proportional to price) into the forward process. * Pros: very high fidelity and diversity, less mode collapse than GANs, stable training. * Cons: high computational cost (many denoising steps), less interpretability, often large memory/training requirements.

In summary, parametric models offer interpretability, analytic tractability and fast simulation (once calibrated), but may fail to reproduce complex stylised facts, interactions, or high-dimensional dynamics beyond their structural assumptions. Deep generative models (GANs, VAEs, Diffusions) increase expressive capacity and can generate diverse realistic synthetic sequences, but they bring heavier data/training demand, more hyper-parameter tuning, less direct interpretability, and often slower simulation or sampling times.

4 Stonk Bench Architechture

4.1 Datasets and Preprocessing

4.1.1 Data type. We use equity price data (e.g., AAPL) at daily frequency with four channels (Open, High, Low, Close). Let $X \in \mathbb{R}^{T\times N}$ denote a univariate stream with $N = 4$ channels and length T.

4.1.2 Transformations. We standardize to *log-returns* per channel via

$$r_t = \log P_t - \log P_{t-1}, \quad r_t \in \mathbb{R}^N.$$

For parametric models, we use contiguous T_{train} historical segments. For deep models, we construct 3D windows $\mathcal{W} \in \mathbb{R}^{A\times L\times N}$ by sliding window extraction of length L and sample A windows for train/validation/test, preserving temporal order in splits.

4.1.3 Data loaders. For deep models, we create mini-batches from $\mathcal{W}$ with fixed seeds for reproducibility (train/valid/test), enabling epoch-wise training and stable evaluation. For fair comparison across model families, parametric models generate sequences of the same length L and number of samples A as the deep models.

4.1.4 Visual diagnostics. We compute per-channel histograms, Q–Q plots, ACF/PACF on returns and absolute returns, rolling-volatility traces, and 2D embeddings (t-SNE/UMAP) of window-level features to validate preprocessing and detect leakage or anomalies prior to training.

4.2 Statistical Evaluation Measures

4.2.1 Feature-based Distance Measures. These metrics quantify how well synthetic data captures key marginal and second-order properties of real data.

[M1] Marginal Distribution Difference (MDD): Distance between real and synthetic marginals (e.g., average 1-Wasserstein across channels) to assess range and frequency alignment. Let $\widehat{F}_r^j$ and $\widehat{F}_s^j$ be empirical CDFs of returns for channel j. We compute

$$\mathrm{MDD} = \frac{1}{N}\sum_{j=1}^{N} W_1\left(\widehat{F}_r^j,\, \widehat{F}_s^j\right), \quad W_1(F,G) = \int_0^1 \left|F^{-1}(u) - G^{-1}(u)\right| du.$$

[M2] Mean Difference (MD): Absolute difference of per-channel means to test central tendency preservation. Let μ_r^j and μ_s^j be real and synthetic means for channel j. Then

$$\mathrm{MD} = \frac{1}{N}\sum_{j=1}^{N} \left|\mu_r^j - \mu_s^j\right|.$$

[M3] Standard Deviation Difference (SDD): Absolute difference of per-channel standard deviations to test volatility level fidelity. With σ_r^j and σ_s^j:

$$\mathrm{SDD} = \frac{1}{N}\sum_{j=1}^{N} \left|\sigma_r^j - \sigma_s^j\right|.$$

[M4] Kurtosis Difference (KD): Difference in excess kurtosis per channel to evaluate heavy-tailedness. Denoting excess kurtosis by $\kappa^{\mathrm{ex}}(x) = \frac{\mathbb{E}[(x-\mu)^4]}{\sigma^4} - 3$, we compute

$$\mathrm{KD} = \frac{1}{N}\sum_{j=1}^{N} \left|\kappa^{\mathrm{ex}}(r^j) - \kappa^{\mathrm{ex}}(s^j)\right|.$$

[M5] AutoCorrelation Difference (ACD): Absolute difference in lag-1 autocorrelation of returns per channel to gauge short-memory dynamics. For channel j:

$$\rho^j(1) = \frac{\sum_{t=2}^{T}(x_t^j - \bar{x}^j)(x_{t-1}^j - \bar{x}^j)}{\sum_{t=1}^{T}(x_t^j - \bar{x}^j)^2}, \quad \mathrm{ACD} = \frac{1}{N}\sum_{j=1}^{N} \left|\rho_r^j(1) - \rho_s^j(1)\right|.$$

4.2.2 Visualization Methods. Visual tools complement numeric metrics for face-validity and communication [1].

[M6] t-SNE/UMAP of window features: 2D embeddings of window-level features (e.g., pooled activations or hand-crafted statistics) for real vs. synthetic overlap and cluster structure.

[M7] Distribution and Q–Q comparisons: Channel-wise histograms and Q–Q plots for returns (and absolute returns) to visualize tails and skew.

4.2.3 Diversity Metrics. Diversity prevents mode collapse and encourages broad coverage.

[M8] Intra-Class Euclidean Distance (ICD-ED): Average pairwise Euclidean distance between generated samples in feature space. Given feature embeddings $\phi(X^{(a)}) \in \mathbb{R}^d$ for samples $a = 1, \ldots, A$:

$$\text{ICD-ED} = \frac{2}{A(A-1)} \sum_{1\le a<b\le A} \left\|\phi\left(X^{(a)}\right) - \phi\left(X^{(b)}\right)\right\|_2.$$

[M9] Intra-Class Dynamic Time Warping (ICD-DTW): Average pairwise DTW distance to capture temporal similarity under local warping. Let $d_{\mathrm{DTW}}(\cdot,\cdot)$ denote DTW distance between two multivariate sequences:

$$\text{ICD-DTW} = \frac{2}{A(A-1)} \sum_{1\le a<b\le A} d_{\mathrm{DTW}}\left(X^{(a)}, X^{(b)}\right).$$

4.2.4 *Efficiency Assessment.*

[M10] Generation Time: Wall-clock time to generate a fixed number of samples (e.g., 500), measured with identical hardware and sequence length. If t_0 and t_1 are start/end timestamps and S is the number of samples, then

$$T_{\text{gen}} = t_1 - t_0 \quad \text{(seconds for } S \text{ samples)}.$$

4.2.5 *Stylized Facts Verification.*

We verify canonical stylized facts of financial returns.

[M11] Heavy Tails: Excess kurtosis of returns ($\kappa - 3$) per channel and the absolute difference to real.

$$\kappa^{\text{ex}}(x) = \frac{\mathbb{E}[(x-\mu)^4]}{\sigma^4} - 3, \quad \Delta\kappa^{\text{ex}} = \frac{1}{N}\sum_{j=1}^{N} \left|\kappa^{\text{ex}}(r^j) - \kappa^{\text{ex}}(s^j)\right|.$$

[M12] Return Autocorrelation: Lag-1 autocorrelation of raw returns; should be close to zero for liquid assets. Using $\rho^j(1)$ as above, report per-channel values and/or their absolute differences to real.

[M13] Volatility Clustering: Autocorrelation of squared or absolute returns (lag-1) to measure volatility persistence.

$$\rho^j_{x^2}(1) = \text{Corr}\big((x^j_t)^2, (x^j_{t-1})^2\big), \quad \text{or} \quad \rho^j_{|x|}(1) = \text{Corr}\big(|x^j_t|, |x^j_{t-1}|\big).$$

[M14] Long Memory in Volatility: Decay profile of ACF in absolute returns; summarized as an average over lags $\mathcal{L} = \{\ell_1, \ldots, \ell_L\}$:

$$\text{LM} = \frac{1}{N}\sum_{j=1}^{N}\left(\frac{1}{L}\sum_{\ell\in\mathcal{L}} \rho^j_{|x|}(\ell)\right).$$

[M15] Non-Stationarity: Window-wise drift/variance drift statistics (e.g., KPSS/ADF summaries or rolling-moment drift) to ensure realistic non-stationary behavior. Using a partition of the series into windows $w = 1, \ldots, W$ with window means m_w and standard deviations s_w:

$$\text{NS} = \text{Var}_w(m_w) + \text{Var}_w(s_w), \quad \Delta\text{NS} = \left|\text{NS}_{\text{real}} - \text{NS}_{\text{synth}}\right|.$$

4.3 Utility Evaluation Measures: Deep Hedging

Utility measures are critical for evaluating synthetic data beyond statistical metrics, as they assess the practical value of generated data in real-world financial applications [2]. Our benchmark incorporates deep hedging as a utility measure for several key reasons:

[U1] Real-world Application Testing: Deep hedging provides a concrete way to evaluate how synthetic data performs in actual financial tasks, particularly in derivatives pricing and risk management.

[U2] Industry-relevant Metrics: By comparing hedging strategies trained on synthetic versus real data, we can assess the practical utility of synthetic data through metrics that matter to financial practitioners, such as replication errors and hedging performance.

[U3] Model Robustness Validation: Deep hedging helps verify if synthetic data maintains the complex relationships and market dynamics necessary for developing reliable trading strategies.

4.3.1 *Deep Hedger Problem Defintion.*

4.3.2 *Deep Hedger Architecture.*

4.3.3 *Deep Hedger Portfolio Evaluation.*

5 Results and Analysis

5.1 Statistical Evaluation Results

We report per-model, per-channel metrics. Copy values from the generated JSON (e.g., `complete_evaluation.json`) into the following templates.

5.2 Utility Evaluation Results

Results of the deep hedging evaluation (replication error, P&L distribution, risk-adjusted metrics) can be summarized in tables analogous to the above once computed.

5.3 Comprehensive Model Comparison

Provide radar plots or scorecards that combine normalized metrics (fidelity, diversity, stylized facts, efficiency) into composite ranks per task.

5.4 Ranking Analysis

Discuss trade-offs: fidelity vs. diversity, training time vs. quality, and sensitivity to window length and sample count.

6 Conclusion and Future Work

In our research, we recognize the existing limitations in the evaluation of synthetic time series, particularly the absence of a universally accepted framework. To address this, we adopt the evaluation taxonomy proposed by Stenger et al. and expand upon it to create a comprehensive benchmark specifically tailored for Synthetic Data Generation for Financial Time Series (SDGFTS). Our contributions include the development of a consolidated evaluation framework that systematically reviews and integrates various assessment methods from leading studies in the field. This approach not only enhances the rigor of evaluations but also facilitates the comparison of different SDGFTS models, ultimately providing a more standardized and reliable means of assessing synthetic data quality.

6.1 Summary of Findings

6.2 Implications for SDGFTS Research

6.3 Current Limitations and Shortcomings

6.4 Future Research Directions

Acknowledgments

To professor Irene Huang, University of Toronto, for her supervision and guidance throughout the project.

References

[1] Yihao Ang, Qiang Huang, Yifan Bao, Anthony K. H. Tung, and Zhiyong Huang. 2023. TSGBench: Time Series Generation Benchmark. *Proc. VLDB Endow.* 17, 3 (2023), 305–318.

[2] Nicolas Boursin, Carl Remlinger, Joseph Mikael, and Carol Anne Hargreaves. 2022. Deep Generators on Commodity Markets; application to Deep Hedging. arXiv:2205.13942 [q-fin.RM] https://arxiv.org/abs/2205.13942

Table 1: Fidelity metrics by model and channel (lower is better for differences). Channels C1–C4 correspond to dataset channels (e.g., Open, High, Low, Close).

	MDD				MD				SDD				KD			
Model	C1	C2	C3	C4	C1	C2	C3	C4	C1	C2	C3	C4	C1	C2	C3	C4
GBM																
OU_Process																
MJD																
GARCH11																
DEJD																
BlockBootstrap																
TimeGAN																
QuantGAN																
TimeVAE																

Table 2: Temporal and diversity metrics by model and channel (lower is better for differences; higher is better for diversity distances).

	ACD				ICD-ED				ICD-DTW			
Model	C1	C2	C3	C4	C1	C2	C3	C4	C1	C2	C3	C4
GBM												
OU_Process												
MJD												
GARCH11												
DEJD												
BlockBootstrap												
TimeGAN												
QuantGAN												
TimeVAE												

Table 3: Stylized facts by model and channel (differences reported real minus synthetic unless noted).

	Heavy Tails (diff)				Autocorr Raw (diff)				Volatility Clustering (diff)				Long Memory (diff)			
Model	C1	C2	C3	C4	C1	C2	C3	C4	C1	C2	C3	C4	C1	C2	C3	C4
GBM																
OU_Process																
MJD																
GARCH11																
DEJD																
BlockBootstrap																
TimeGAN																
QuantGAN																
TimeVAE																

[3] Vamsi K. Potluru, Daniel Borrajo, Andrea Coletta, Niccolo Dalmasso, Yousef El-Laham, Elizabeth Fons, Mohsen Ghassemi, Sriram Gopalakrishnan, Vikesh Gosai, Eleonora Kreacic, Ganapathy Mani, Saheed Obitayo, Deepak Paramanand, Natraj Raman, Mikhail Solonin, Srijan Sood, Svitlana Vyetrenko, Haibei Zhu, Manuela Veloso, and Tucker Balch. 2024. Synthetic Data Applications in Finance. arXiv:2401.00081 [cs.LG] https://arxiv.org/abs/2401.00081

[4] Michael Stenger, Robert Leppich, Ian Foster, Samuel Kounev, and André Bauer. 2024. Evaluation is key: a survey on evaluation measures for synthetic time series. *J Big Data* 11, 66 (May 2024). doi:10.1186/s40537-024-00924-7

[5] Zhengwei Wang, Qi She, and Tomas E. Ward. 2020. Generative Adversarial Networks in Computer Vision: A Survey and Taxonomy. arXiv:1906.01529 [cs.LG] https://arxiv.org/abs/1906.01529

[6] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. 2019. *Time-series generative adversarial networks.* Curran Associates Inc., Red Hook, NY, USA, 5508–5518.

Figure 2: Histograms and Q–Q plots of real vs. synthetic returns per channel. Replace with generated figures from `VisualAssessmentEvaluator`.

Figure 3: ACF/PACF of returns and absolute returns for real vs. synthetic data.

Figure 4: t-SNE/UMAP embeddings of window-level features; color-coded real vs. synthetic for overlap assessment.

Table 4: Non-stationarity (diff) and generation time per model.

Model	Non-stationarity (diff) C1	C2	C3	C4	Gen. Time (500) seconds
GBM					
OU_Process					
MJD					
GARCH11					
DEJD					
BlockBootstrap					
TimeGAN					
QuantGAN					
TimeVAE					

A Preprocessing Diagnostics

A.1 Channel-wise Distributions

A.2 Autocorrelation Analyses

B Embedding Visualizations

C Per-Model Visual Assessments

For each model (GBM, OU_Process, MJD, GARCH11, DEJD, Block-Bootstrap, TimeGAN, QuantGAN, TimeVAE), include side-by-side real/synthetic plots and distribution overlays.

Figure 5: Representative visual assessment panels for a selected model.