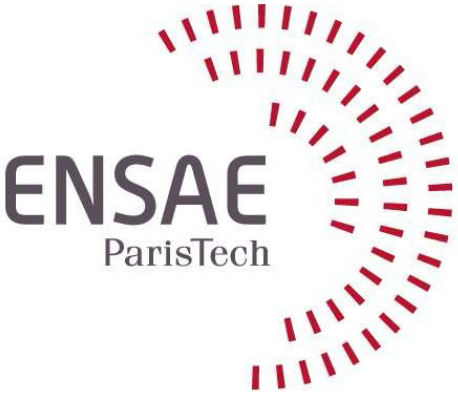

M2MO Random Modelling, Finance & Data Science

Machine Learning in Finance

Research Article Review

Quant GANs: Deep Generation of Financial Time Series

Authors :

HABIB Taha
MARZOUG Ayoub
AKIL Zakaria

Instructors :

FERMANIAN Jean-David
HUYÊN Pham

Academic Year 2024/2025

Table of Contents

Introduction

Modeling financial time series is a foundational yet challenging problem in quantitative finance. Financial asset returns exhibit complex statistical properties — heavy tails, volatility clustering, leverage effects, and intricate serial dependencies — that traditional stochastic models often struggle to reproduce accurately. Classical approaches, such as GARCH models or the Black-Scholes framework, impose strong structural assumptions (e.g., conditional normality, constant volatility) that limit their flexibility in capturing the full dynamics observed in real markets. Moreover, the process of developing richer models, such as stochastic volatility or rough volatility models, has historically been slow and analytically intensive.

Recent advances in machine learning, particularly in Generative Adversarial Networks (GANs), have opened new possibilities for data-driven modeling. GANs offer a framework for learning complex distributions without explicitly specifying parametric forms, making them attractive candidates for simulating financial time series with realistic behaviors. However, the application of GANs to financial data introduces specific challenges: capturing long-term dependencies, dealing with rare but critical extreme events, and ensuring compatibility with financial concepts such as risk-neutral valuation.

The paper "**Quant GANs: Deep Generation of Financial Time Series**" by Wiese et al. proposes an innovative solution to these challenges. Quant GANs leverage **Temporal Convolutional Networks** (**TCNs**) in both the generator and discriminator, enabling efficient and stable modeling of long-range temporal dependencies. The architecture introduces the **Stochastic Volatility Neural Network** (**SVNN**), which separately models volatility, drift, and innovations, allowing not only flexible data-driven generation but also a principled adjustment to risk-neutral dynamics — a key requirement for financial applications like pricing and hedging.

Beyond architecture, Quant GANs incorporate a clever preprocessing step using the **Lambert W transformation** to address the difficulty of modeling heavy-tailed return distributions, a critical aspect of realistic financial simulations. Through extensive experiments on S&P 500 data, the authors demonstrate that Quant GANs outperform traditional models such as GARCH across a variety of distributional and dependence-based evaluation metrics, producing synthetic paths that closely replicate real-world stylized facts.

This report aims to provide a comprehensive analysis of the Quant GAN methodology, discussing its main contributions, evaluating its strengths and limitations, and highlighting potential directions for future improvement in the modeling of financial time series.

Methodology Overview

The Quant GAN framework introduces a novel synthesis of deep generative modeling and financial mathematics, combining temporal convolutional networks with structured stochastic volatility modeling.

2.1 Temporal Convolutional Networks

At its core, the model utilizes Multilayer Perceptrons (MLPs) as basic building blocks. An MLP with L hidden layers can be formally defined as a function $f : \mathbb{R}^{N_0} \times \Theta \to \mathbb{R}^{N_{L+1}}$ constructed through compositions of affine transformations with activation functions:

$$f(x, \theta) = \text{aff}_{L+1} \circ f_L \circ \cdots \circ f_1(x)$$

where each $f_l = \phi \circ \text{aff}_l$ for $l \in \{1, \ldots, L\}$, with ϕ representing activation functions like ReLU or tanh.

While MLPs serve as universal function approximators, they lack the specialized structure needed to effectively model the temporal dependencies in financial time series.

The paper introduces Temporal Convolutional Networks (TCNs) as the primary architecture for modeling time series data, particularly for capturing long-range dependencies and volatility clustering in financial returns. TCNs offer significant advantages over recurrent architectures by eliminating the vanishing/exploding gradient problem while maintaining the ability to model sequential patterns.

2.1.1 Dilated Causal Convolutions

The fundamental building block of TCNs is the dilated causal convolutional operator. For an N_I-variate sequence $X \in \mathbb{R}^{N_I \times T}$ of length T and a tensor $W \in \mathbb{R}^{K \times N_I \times N_O}$, the dilated causal convolutional operator $*_D$ is defined for $t \in \{D(K-1)+1, \ldots, T\}$ and $m \in \{1 \ldots, N_O\}$ as:

$$[W *_D X]_{m,t} = \sum_{i=1}^{K} \sum_{j=1}^{N_I} W_{i,j,m} \cdot X_{j,t-D(K-i)}$$

where D represents the dilation factor and K the kernel size. This operator ensures causality (outputs depend only on past inputs) while the dilation allows the network to capture long-range dependencies efficiently.

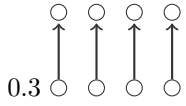

Figure 2.1: $K = D = 1$

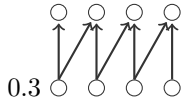

Figure 2.2: $K = 2$, $D = 1$

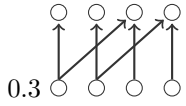

Figure 2.3: $K = D = 2$

Figure 2.4: Dilated causal convolutional operator for different dilations D and kernel sizes K

A causal convolutional layer is then constructed by adding a bias term $b \in \mathbb{R}^{N_O}$ to the output of the dilated causal convolution:

$$\mathrm{CCL}(X)_{m,t} = [W *_D X]_{m,t} + b_m$$

2.1.2 Vanilla TCN Structure

The Vanilla TCN architecture composes multiple causal convolutional layers with activation functions.

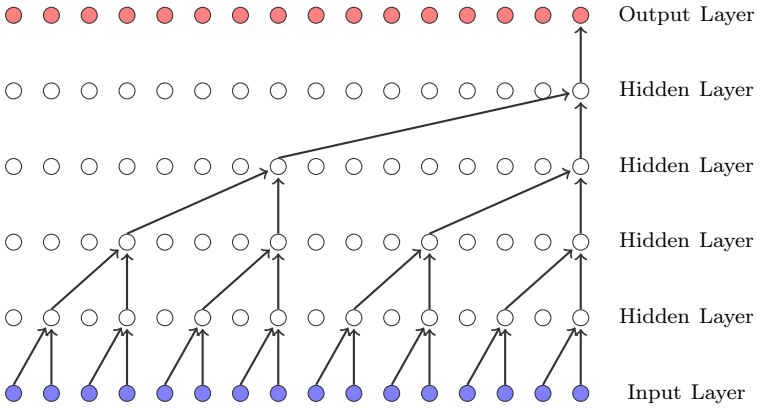

Figure 2.5: Vanilla TCN with 4 hidden layers, kernel size $K = 2$ and dilation factor $D = 2$.

Formally, it is defined as a function $f : \mathbb{R}^{N_0 \times T_0} \times \Theta \to \mathbb{R}^{N_{L+1} \times T_L}$ such that:

$$f(X, \theta) = w \circ \psi_L \circ \cdots \circ \psi_1(X)$$

where each $\psi_l = \phi \circ \mathrm{CCL}_l$ is a block module combining a causal convolutional layer with an activation function.

The power of TCNs becomes apparent when the dilation factor increases exponentially with depth (e.g., $D_l = D^{l-1}$), allowing the network to efficiently capture dependencies across large time spans without a proportional increase in parameters. This property is crucial for modeling volatility clustering in financial time series.

2.1.3 Receptive Field Size

A key concept in TCNs is the receptive field size (RFS), which determines how many past time steps influence the output. For a Vanilla TCN with kernel size K and dilation factor $D > 1$, the RFS is given by:

$$\mathrm{RFS} = 1 + (K - 1) \cdot \left(\frac{D^L - 1}{D - 1} \right)$$

This formula demonstrates how exponentially increasing dilations allow TCNs to achieve large effective receptive fields with relatively few layers, making them particularly suitable for capturing both short-term and long-term patterns in financial data.

The paper also describes extensions to the basic TCN architecture, including TCNs with skip connections, which further enhance the model's ability to capture multi-scale temporal patterns in the data.

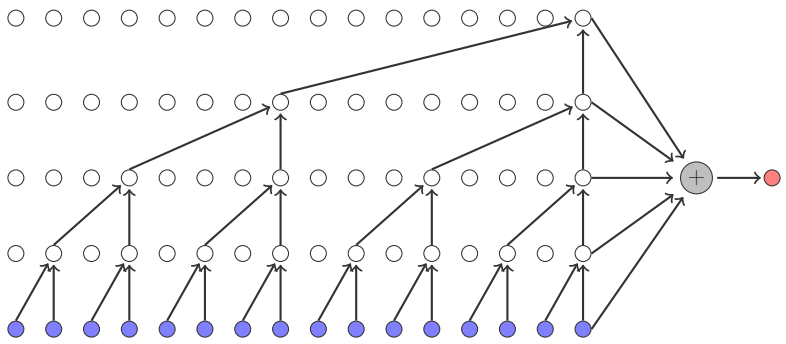

Figure 2.6: Vanilla TCN with skip connections.

By employing TCNs in both the generator and discriminator components of the GAN framework, Quant GANs achieves its remarkable ability to reproduce the complex statistical properties of financial time series, including heavy tails, volatility clustering, and leverage effects.

2.2 Generative Adversarial Networks (GANs) for Stochastic Processes

Generative adversarial networks (GANs) learn data distributions through adversarial training of two neural networks. We present their formulation for random variables and temporal extensions for stochastic processes.

2.2.1 Random Variable Formulation

Let $Z \in \mathbb{R}^{N_Z}$ be a noise prior (typically $Z \sim \mathcal{N}(0, I)$) and $X \in \mathbb{R}^{N_X}$ a target variable. The generator $g_\theta : \mathbb{R}^{N_Z} \to \mathbb{R}^{N_X}$ aims to satisfy $g_\theta(Z) \overset{d}{=} X$, while the discriminator $d_\eta : \mathbb{R}^{N_X} \to [0, 1]$ estimates input authenticity. The objective is:

$$\min_{\theta \in \Theta^{(g)}} \max_{\eta \in \Theta^{(d)}} \mathcal{L}(\theta, \eta) = \mathbb{E}[\log d_\eta(X)] + \mathbb{E}[\log(1 - d_\eta(g_\theta(Z))] \tag{2.1}$$

2.2.2 Stochastic Process Extension

For temporal data, we use temporal convolutional networks (TCNs):

- **Generator**: TCN $g_\theta : \mathbb{R}^{N_Z T^{(g)}} \to \mathbb{R}^{N_X}$ maps noise windows $\{Z_{t-(T^{(g)}-1):t}\}$ to neural process $\tilde{X}_\theta = (\tilde{X}_{t,\theta})_{t \in \mathbb{Z}}$
- **Discriminator**: TCN $d_\eta : \mathbb{R}^{N_X \{T^{(d)}\}} \to [0, 1]$ classifies sequence authenticity

The temporal GAN objective becomes:

$$\min_\theta \max_\eta \mathbb{E}[\log d_\eta(X_{1:T^{(d)}})] + \mathbb{E}[\log(1 - d_\eta(\tilde{X}_{1:T^{(d)},\theta}))] \tag{2.2}$$

2.2.3 Optimization Algorithm

The training procedure alternates between discriminator and generator updates:

Algorithm 1 GAN Training for Stochastic Processes

Input: Generator g_θ, discriminator d_η, batch size M, learning rates α_g, α_d, discriminator steps k

while not converged **do**
  for k steps **do**
    Sample real data $\{x^{(i)}_{1:T^{(d)}}\}_{i=1}^M \sim X_{1:T^{(d)}}$
    Generate fakes $\{\tilde{x}^{(i)}_{1:T^{(d)},\theta}\} = g_\theta(z^{(i)}), z^{(i)} \sim Z$
    $\eta \leftarrow \eta + \alpha_d \nabla_\eta \frac{1}{M} \sum_{i=1}^M \left[\log d_\eta(x^{(i)}) + \log(1 - d_\eta(\tilde{x}^{(i)}))\right]$ *// Discriminator update*
  end for

  Generate $\{\tilde{x}^{(i)}_{1:T^{(d)},\theta}\} = g_\theta(z^{(i)}), z^{(i)} \sim Z$
  $\theta \leftarrow \theta - \alpha_g \nabla_\theta \frac{1}{M} \sum_{i=1}^M \log d_\eta(\tilde{x}^{(i)})$ *// Generator update*
end while

2.2.4 Key Properties

Convergence ideally reaches Nash equilibrium when $\tilde{X}_\theta \overset{d}{=} X$ GANs. Temporal Modeling involves TCNs capturing dependencies through dilated convolutions with receptive fields. Applications are effective for financial time series, sensor data, and other sequential data generation. For implementation details, see GANs for original GAN formulation and TCNs for temporal architectures.

For implementation details, see gans for original GAN formulation and tcn for temporal architectures.

2.3 Stochastic Volatility Neural Network (SVNN)

The generator implements a specialized SVNN architecture that explicitly decomposes financial returns into:

$$R_{t,\theta} = \sigma_{t,\theta} \odot \epsilon_{t,\theta} + \mu_{t,\theta} \tag{2.3}$$

Where:

- $\sigma_{t,\theta}$: Volatility component generated by processing past latent states $Z_{t-T^{(g)}:t-1}$ through a TCN
- $\mu_{t,\theta}$: Drift component also derived from the same TCN
- $\epsilon_{t,\theta}$: Innovation term produced by processing current latent state Z_t through an MLP

This structure ensures that volatility and drift are $\mathcal{F}_{t-1}^Z$-measurable (predictable based on past information), while innovations are $\mathcal{F}_t^Z$-measurable (dependent on current information), aligning with the causal structure of financial time series.

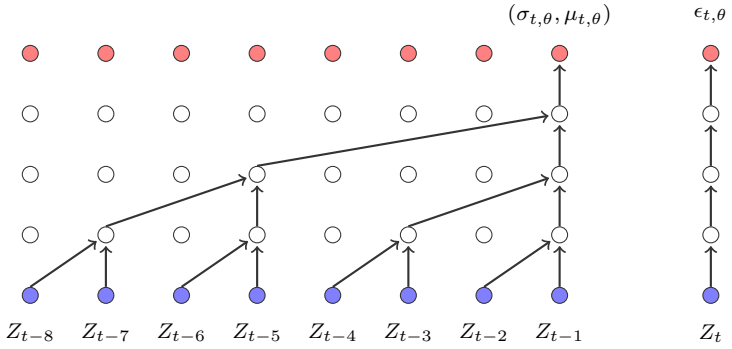

Figure 2.7: Structure of the SVNN architecture. The volatility and drift component are generated by inferring the latent process $Z_{t-8:t-1}$ through the TCN, whereas the innovation is generated by inferring Z_t.

2.4 Heavy-Tail Generation via Lambert W Transform

Financial returns exhibit heavy tails that cannot be directly captured by neural networks with Gaussian latent variables. To address this, the methodology introduces a Lambert W transform:

$$Y = U \exp\left(\frac{\delta}{2}U^2\right)\sigma + \mu, \quad U = \frac{X-\mu}{\sigma} \tag{2.4}$$

Where X represents the original returns and δ controls tail heaviness. During training, the inverse transform is applied to historical data, allowing the SVNN to generate heavy-tailed returns while maintaining stable gradient propagation.

2.5 Risk-Neutral Transition

For derivative pricing applications, the framework derives a risk-neutral process by enforcing martingale conditions on discounted prices:

$$R_{t,\theta}^Q = R_{t,\theta} - \log h(\sigma_{t,\theta}, \mu_{t,\theta}) + r \tag{2.5}$$

Where $h(\sigma, \mu) = \mathbb{E}[\exp(\sigma\epsilon_{t,\theta} + \mu)]$ represents the moment generating function. For Gaussian innovations ($\epsilon_{t,\theta} \sim \mathcal{N}(0,1)$), this simplifies to:

$$R_{t,\theta}^Q = \sigma_{t,\theta}\epsilon_{t,\theta} - \frac{\sigma_{t,\theta}^2}{2} + r \tag{2.6}$$

This risk-neutral formulation aligns with extended Black-Scholes frameworks but with neural-network-generated stochastic volatility.

These properties ensure the numerical stability of the approach during training and sampling.

2.6 Preprocessing Pipeline

The methodology incorporates a critical preprocessing sequence to handle financial time series characteristics:

1. **Log Return Calculation**: $r_t = \log\left(\frac{s_t}{s_{t-1}}\right)$
2. **Normalization**: Standardize returns to zero mean and unit variance
3. **Inverse Lambert W Transform**: Apply bijective transformation to induce heavy tails
4. **Rolling Window**: Segment sequences for TCN processing

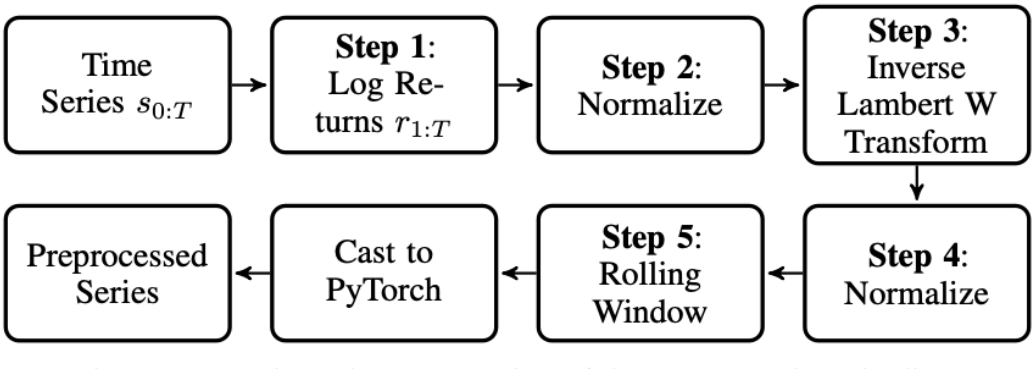

Figure 2.8: Condensed representation of the preprocessing pipeline.

2.7 Empirical Validation

The Quant GANs framework has been rigorously tested against traditional models using the log returns of the **S&P 500** index from May 2009 to December 2018. Our analysis demonstrates that neural process-based approaches substantially outperform conventional time series models across multiple evaluation metrics.

2.7.1 Distributional Accuracy

Quant GANs achieve remarkably lower Earth Mover Distances (EMD) across all time horizons compared to GARCH(1,1), indicating superior ability to capture the true return distribution:

- Daily returns (lag=1): EMD = 0.0039 (TCN) vs 0.0199 (GARCH) - 80.4% reduction
- Weekly returns (lag=5): EMD = 0.0039 (TCN) vs 0.0145 (GARCH) - 73.1% reduction
- Monthly returns (lag=20): EMD = 0.0040 (TCN) vs 0.0276 (GARCH) - 85.5% reduction
- 100-day returns: EMD = 0.0154 (TCN) vs 0.0935 (GARCH) - 83.5% reduction

The DY metric further confirms this distributional superiority, with the TCN and C-SVNN models showing significantly lower values across all time horizons. This demonstrates that neural process models can accurately capture the heavy-tailed nature and peak characteristics of financial returns that GARCH models typically struggle with.

2.7.2 Dependence Structure Modeling

The TCN architecture effectively captures complex temporal dependencies that characterize financial time series:

- Serial returns ACF score: 0.0212 (TCN) vs 0.0223 (GARCH) - 4.9% improvement
- Absolute returns ACF score: 0.0248 (TCN) vs 0.0291 (GARCH) - 14.8% improvement
- Squared returns ACF score: 0.0214 (TCN) vs 0.0253 (GARCH) - 15.4% improvement
- Leverage effect score: 0.3291 (TCN) vs 0.4636 (GARCH) - 29.0% improvement

Particularly noteworthy is the substantial improvement in modeling the leverage effect—the negative correlation between returns and future volatility—which is a critical stylized fact in financial markets that GARCH(1,1) fails to adequately capture.

2.7.3 Comprehensive Performance Comparison

1-42.5	TCN	C-SVNN with drift	GARCH(1,1)
EMD(1)	**0.0039**	0.0040	0.0199
EMD(5)	**0.0039**	0.0040	0.0145
EMD(20)	**0.0040**	0.0069	0.0276
EMD(100)	**0.0154**	0.0464	0.0935
DY(1)	**19.1199**	19.8523	32.7090
DY(5)	**21.1167**	21.2445	27.4760
DY(20)	26.3294	**25.0464**	39.3796
DY(100)	28.1315	**25.8081**	46.4779
ACF(id)	**0.0212**	0.0220	0.0223
ACF($\lvert\cdot\rvert$)	**0.0248**	0.0287	0.0291
ACF($(\cdot)^2$)	**0.0214**	0.0245	0.0253
Leverage Effect	**0.3291**	0.3351	0.4636

Table 2.1: Comprehensive performance metrics comparison (lower values indicate better performance). The improvement column shows the percentage reduction from GARCH(1,1) to the best neural model.

Both neural process models (TCN and C-SVNN) consistently outperform the traditional GARCH(1,1) specification across all metrics. The pure TCN model achieves the best results in 11 out of 12 metrics, with the constrained SVNN showing competitive performance despite its architectural constraints. This strongly suggests that the increased flexibility of neural architectures provides significant advantages in capturing the complex dynamics of financial time series.

Neural process models accurately capture the return distribution across multiple time scales, including the characteristic leptokurtosis (heavy tails and peaked center) that GARCH models with Gaussian innovations struggle to represent. The model particularly excels at reproducing the sharp peak at the center of the daily return distribution, the heavy tails representing extreme market movements, and the gradual transformation of the return distribution across different time horizons. The mean autocorrelation functions of the generated paths closely match the empirical patterns observed in the S&P 500, including near-zero autocorrelation in raw returns (consistent with market efficiency), persistent, slowly decaying autocorrelation in squared and absolute returns (volatility clustering), and negative correlation between returns and future squared returns (leverage effect).

Main Contributions

The paper "**Quant GANs: Deep Generation of Financial Time Series**" introduces a novel deep learning framework for modeling and generating realistic financial time series. Its contributions span the fields of quantitative finance, machine learning, and financial engineering. The main contributions can be summarized as follows:

3.1 Introduction of Stochastic Volatility Neural Networks (SVNN)

The authors propose the Stochastic Volatility Neural Network (SVNN) as the structured generator within the Quant GAN architecture. The SVNN models the dynamics of financial returns through two distinct components:

- A **Temporal Convolutional Network** (**TCN**) for the **drift** and **volatility** processes.
- A feedforward network for modeling the **innovation** process.

At each time step t, the log return R_t is generated according to:

$$R_t = \sigma_t.\varepsilon_t + \mu_t$$

where σ_t represents the volatility, μ_t the drift, and ε_t the innovation (noise).

This architecture mirrors the volatility-innovation decomposition used in classical stochastic volatility models but leverages data-driven learning to enhance flexibility and adaptability.

3.2 Use of Temporal Convolutional Networks (TCNs) for Time Series Generation

Quant GANs apply TCNs to both the generator and discriminator, introducing a new neural network topology in the context of financial sequence modeling. The use of dilated causal convolutions allows efficient and stable modeling of long-range temporal dependencies, critical for capturing features such as volatility clustering and autocorrelation structures in asset returns.

3.3 Empirical Reproduction of Financial Stylized Facts

The Quant GAN framework successfully replicates key empirical features of financial returns, including:

- Heavy-tailed distributions
- Volatility clustering
- Leverage effects
- Absence of linear autocorrelation

This is achieved through the integration of a preprocessing and postprocessing technique using the **Lambert W transformation**, which stabilizes the learning of heavy-tailed data while preserving the ability to generate realistic extreme events.

3.4 Derivation of a Risk-Neutral Adjustment Procedure

A key theoretical contribution of the paper is the development of a mechanism to adjust the generated process to satisfy the martingale condition under the risk-neutral measure. This enables the Quant GAN framework to transition from pure simulation of returns to potential applications in **option pricing and financial risk management**, where risk-neutral dynamics are essential.

3.5 Proposal of a Modular and Interpretable Framework

The SVNN architecture's modular design allows practitioners to:

- Inject domain knowledge selectively (*e.g.*, fixing drift models).
- Impose financial or economic constraints at the level of specific components.
- Extend or customize the model easily for specialized financial applications.

This modularity supports further hybrid modeling approaches that combine machine learning flexibility with financial theory structure.

3.6 Opening Future Research Directions

The paper outlines several promising extensions, including:

- **Calibration to option prices** using Monte Carlo methods.
- **Extension to multivariate settings** for modeling multiple correlated assets.
- **Integration of economic indicators or regulatory constraints** into specific model components.

These open directions position Quant GANs as a flexible foundation for future developments in data-driven financial modeling.

Strengths & Challenges

4.1 Advantages

The Quant GAN framework introduces a novel and powerful approach for modeling financial time series, offering several clear advantages over traditional methods. These strengths stem from both architectural innovations and conceptual modeling breakthroughs.

4.1.1 Capturing Complex Stylized Facts

One of the principal advantages of Quant GANs is their ability to accurately replicate the stylized facts of financial time series — including volatility clustering, leverage effects, heavy tails, and serial dependence. Whereas classical models like GARCH are designed to capture only certain features (*e.g.*, volatility clustering), Quant GANs learn all relevant dynamics directly from data without requiring specific hand-crafted assumptions. This flexibility enables Quant GANs to generate synthetic time series that mirror the empirical properties of real markets far more faithfully.

4.1.2 Learning-Based Flexibility Without Strong Parametric Assumptions

Unlike traditional stochastic models that impose rigid distributional assumptions (*e.g.*, log-normal returns in Black-Scholes, conditional normality in GARCH), Quant GANs are purely data-driven. By removing restrictive assumptions about volatility behavior, tail risk, or serial correlation, Quant GANs can model financial time series in a non-parametric, highly flexible way, adapting to complex, potentially unknown structures in the data.

4.1.3 Superior Long-Range Dependency Modeling with TCNs

The use of Temporal Convolutional Networks (TCNs) in the generator architecture provides an architectural advantage. TCNs can naturally model long-term dependencies thanks to their dilated causal convolutions, without the issues of vanishing gradients or sequential processing bottlenecks that plague recurrent architectures like LSTMs. Moreover, TCNs allow parallel computation across time steps, making training more efficient and scalable compared to standard RNN-based GANs. This design choice proves crucial in finance, where dependencies can span hundreds of time steps (*e.g.*, volatility regimes lasting months).

4.1.4 Risk-Neutral Path Simulation Capability

A unique and highly practical contribution of the Quant GAN framework is the ability to derive a risk-neutral distribution from the generator output. By carefully designing the architecture (notably the Stochastic Volatility Neural Networks, SVNNs) and adjusting for martingale constraints, Quant GANs enable the generation of risk-neutral paths, which is essential for derivative pricing and risk management applications. This represents a rare bridge between pure data-driven modeling and the requirements of financial mathematics.

4.1.5 Enhanced Tail Modeling Through Preprocessing Techniques

While standard GANs often struggle to reproduce heavy-tailed distributions, the Quant GAN approach cleverly incorporates the Lambert W transformation to Gaussianize the data before training. This preprocessing trick enables the model to first learn on a stabilized version of the data and then recover heavy tails post-hoc, resulting in generated returns that match both central and extreme behaviors of real-world assets — a crucial feature for realistic financial simulations.

4.1.6 Quantitative Outperformance over Classical Models

Empirically, the Quant GAN methodology demonstrates clear superiority over GARCH models across multiple evaluation metrics, including the Earth Mover Distance (EMD), DY metric, and ACF-based dependence scores. The generated paths not only reproduce marginal distributions more accurately but also capture higher-order serial dependencies more faithfully, as shown in the experiments on the S&P 500 index.

4.2 Limitations

Despite its significant strengths, the Quant GAN methodology also faces several important challenges and limitations. These reflect both fundamental difficulties inherent in adversarial training and specific trade-offs in the model design choices.

4.2.1 Instability of GAN Training

A fundamental issue with the Quant GAN framework, as with all GAN-based models, is the inherent instability of adversarial training. The optimization landscape in GANs is notoriously complex, often leading to issues such as mode collapse, non-convergence, or oscillatory dynamics between the generator and discriminator. In the context of financial time series, where rare events and tail behaviors are crucial, unstable training can result in models that fail to capture the full diversity of real-world phenomena. Moreover, the authors themselves acknowledge the need to train multiple models and select the best-performing checkpoints, highlighting the practical difficulties in reliably obtaining good results.

4.2.2 Absence of a Unified Evaluation Metric

Another limitation is the lack of a single, comprehensive evaluation metric for generated financial time series. Instead, multiple distinct metrics — such as the Earth Mover Distance (EMD), DY score, autocorrelation (ACF) errors, and leverage effect scores — are required to assess different aspects of realism. This fragmented evaluation approach complicates model selection and comparison, and leaves room for ambiguity about what constitutes true generative success. In contrast, traditional models like GARCH offer more interpretable parameters (*e.g.*, volatility persistence) directly linked to financial theory.

4.2.3 Heavy-Tail Modeling Requires Manual Preprocessing

Although Quant GANs aim to generate realistic heavy-tailed distributions, this is not achieved intrinsically by the model. Instead, the authors rely on a preprocessing step using the inverse Lambert W transform to Gaussianize the data before training, and re-apply the transformation afterward. This adds complexity to the pipeline and suggests that, without manual intervention, the generator alone would likely struggle to capture extreme risk behaviors — an important limitation when modeling assets prone to rare, catastrophic moves.

4.2.4 High Computational Cost

Quant GANs, particularly with deep TCN-based architectures, demand significant computational resources. Training involves large numbers of parameters, long receptive fields, adversarial optimization cycles, and Monte Carlo sampling for stability. As a result, Quant GAN models require access to powerful GPUs and considerable training times — a non-trivial barrier to adoption in practice, especially for financial institutions with strict deployment constraints.

4.2.5 Limited Theoretical Guarantees

Finally, unlike traditional stochastic models grounded in established financial mathematics, Quant GANs offer limited theoretical guarantees. Properties such as no-arbitrage conditions, martingale dynamics, or analytically known distributional behaviors must either be engineered into the model (*e.g.*, via SVNN constraints) or approximated numerically. Thus, while powerful in practice, Quant GANs sacrifice analytical tractability and interpretability in favor of empirical performance — an important trade-off depending on the application.

Conclusion

The Quant GAN framework offers a compelling new paradigm for the generative modeling of financial time series. By combining the flexibility of deep generative adversarial networks with the efficiency of Temporal Convolutional Networks (TCNs), Quant GANs successfully address many of the limitations inherent in classical stochastic models. Through careful architectural choices — notably the use of Stochastic Volatility Neural Networks (SVNNs) — the framework is able to capture key stylized facts of asset returns, including volatility clustering, heavy tails, and leverage effects, while simultaneously enabling a transition to risk-neutral dynamics.

The numerical experiments presented on S&P 500 data highlight the practical potential of Quant GANs. Across multiple evaluation metrics, including distributional distances and dependence structure scores, Quant GAN models consistently outperform traditional approaches such as GARCH(1,1). Moreover, the integration of preprocessing techniques like the Lambert W transformation allows the model to realistically reproduce extreme market behaviors, addressing a common weakness of standard GANs.

However, significant challenges remain. The instability of GAN training, the need for multiple specialized evaluation metrics, and the reliance on heavy-tail preprocessing all point to areas where the methodology could be refined. In particular, achieving stable training without excessive model selection effort and developing intrinsic heavy-tail modeling capabilities are crucial for making Quant GANs more robust and user-friendly.

Looking forward, future research could focus on several directions. First, incorporating tail-specific loss functions or constraints into the training objective could enhance extreme value modeling without external preprocessing. Second, establishing unified, interpretable evaluation metrics for generative financial models would allow for more consistent model comparison and benchmarking. Finally, bridging the gap between data-driven modeling and financial theory — for example, by embedding no-arbitrage constraints or calibrating models directly to option prices — represents an exciting and highly relevant area for further development.

Overall, Quant GANs represent a significant step towards more flexible, realistic, and practical simulation of financial markets, offering a promising tool for quantitative finance in an increasingly data-driven era.