Generative Adversarial Neural Networks for Realistic Stock Market Simulations

Badre Labiad[1], Abdelaziz Berrado[2], Loubna Benabbou[3]
AMIPS Research Team, Ecole Mohammadia d'Ingénieurs, Mohammed V University in Rabat, Morocco[1, 2]
Département Sciences de la Gestion, Universit du Québec Rimouski (UQAR), Campus de Lévis, Québec Canada[3]

***Abstract*—Stock market simulations are widely used to create synthetic environments for testing trading strategies before deploying them to real-time markets. However, the weak realism often found in these simulations presents a significant challenge. Improving the quality of stock market simulations could be facilitated by the availability of rich and granular real Limit Order Books (LOB) data. Unfortunately, access to LOB data is typically very limited. To address this issue, a framework based on Generative Adversarial Networks (GAN) is proposed to generate synthetic realistic LOB data. This generated data can then be utilized for simulating downstream decision-making tasks, such as testing trading strategies, conducting stress tests, and performing prediction tasks. To effectively tackle challenges related to the temporal and local dependencies inherent in LOB structures and to generate highly realistic data, the framework relies on a specific data representation and preprocessing scheme, transformers, and conditional Wasserstein GAN with gradient penalty. The framework is trained using the FI-2010 benchmark dataset and an ablation study is conducted to demonstrate the importance of each component of the proposed framework. Moreover, qualitative and quantitative metrics are proposed to assess the quality of the generated data. Experimental results indicate that the framework outperforms existing benchmarks in simulating realistic market conditions, thus demonstrating its effectiveness in generating synthetic LOB data for diverse downstream tasks.**

Keywords—Limit order book simulations; transformers; wasserstein GAN with gradient penalty; FI-2010 benchmark dataset

I. Introduction

Stock markets are complex systems, with underlying dynamics that remain largely unknown. Modeling such environments using traditional approaches poses unique challenges, including selecting appropriate hand-crafted features and verifying market assumptions. Leveraging the advancements in machine learning and deep learning techniques, numerous studies have sought to model market behaviors utilizing these innovative tools [1]-[4].

The dynamics of markets are influenced by interactions among multiple agents. These interactions are documented in the Limit Order Book (LOB) through buy and sell orders, providing rich insights into the market microstructure [5]. Such data is invaluable to traders, investors, regulators, and researchers. Unfortunately, the granular details within the LOB are not publicly accessible, with only aggregated daily summaries of price changes being made available. One potential solution involves adopting an agnostic approach toward the unknown underlying dynamics and embracing solutions capable of extracting crucial characteristics from the actual data.

Generative Adversarial Networks (GANs) [6] offer an intriguing solution for modeling the LOB data due to their exceptional ability to generate data from complex distributions. Recent breakthroughs in GANs have accelerated their adoption across various domains, including image generation [7], [8], text generation [9], and audio generation [10].

In finance, numerous studies have leveraged GANs to model financial data, demonstrating their competitiveness compared to other deep learning techniques [11–16]. While GANs exhibit promising attributes for producing realistic simulations, their application as a data generation technique for stock market data remains relatively nascent [17]. Furthermore, only a handful of studies have explored the potential of GANs for stock market simulation [18, 19]. This work aims to bridge this gap.

In this work, the aim is to develop a solution capable of simulating the LOB by generating synthetic data that closely resemble real data, capturing key statistical properties and mimicking the behavior of stock order dynamics realistically. The output of the framework is synthetic yet realistic LOB data. The practical implications of this endeavor are manifold. The generated data can be utilized for training forecasting models, calibrating trading strategies, conducting stress tests, performing backtests, and detecting anomalies. Additionally, the proposed procedure helps address data access limitations by creating synthetic data.

The main contribution of this study is a new framework based on GANs models for generating synthetic Limit Order Book data to simulate stock market behaviors. The proposed solution relies on a specific data representation and preprocessing scheme, along with conditional Wasserstein GAN with a gradient penalty for model training. An assessment methodology is proposed to evaluate the quality and realism of the generated LOB data, and a comparison with state-of-the-art models is provided.

The paper is organized as follows: a review of related works is presented in Section II. The developed framework is detailed in Section III. Subsequently, experimental settings and results are presented and discussed in Sections IV and V, respectively. Finally, Section VI provides conclusions and outlines possible future avenues of research.

II. Related Works

In this section, an overview of the LOB and GANs basics is provided, along with a review of works that have simulated LOB data using various techniques.

A. *Limit Order Book Background and Simulation*

The LOB serves as a comprehensive record of all orders submitted to an exchange system, providing a detailed snapshot of market activities at a microstructure level. At any given moment, the LOB includes active orders organized by price levels. An active order is an order that is still unmatched or uncancelled. Orders are categorized as either asks or bids, and they can be modified or cancelled until they are executed. An execution, or trade, occurs when there is a match between ask and bid prices. Orders may be market orders, executed immediately at the best available price, or limit orders, executed only when there is a matching sell (buy) order at the desired price. Fig. 1 presents a simplified visualization of an LOB. The LOB undergoes continuous updates due to the arrival of new orders, cancellations, and executions, altering the current state of the market.

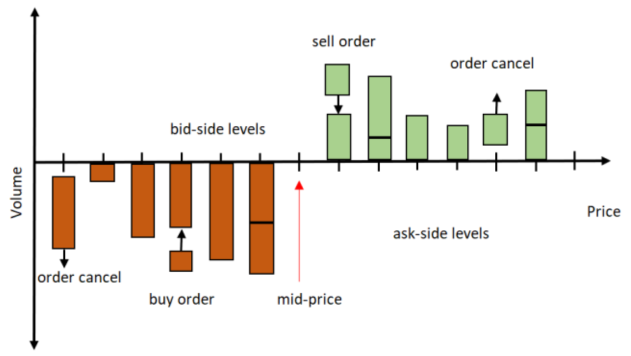

Fig. 1. Simplified graphical representation of the LOB showing the bid and ask sides prices structure and the impact of different order types on the LOB's price levels.

The deployment of new algorithms or trading strategies in real environments necessitates extensive testing under various market scenarios. These tests are typically conducted within a simulation framework that mimics the states of the LOB. The LOB, being a dynamic and complex system, poses a challenge for both market practitioners and researchers. Explicitly expressing the LOB as a function is often infeasible due to the hidden complexity of its underlying dynamics.

Furthermore, despite the increasing use of GANs in various stock market applications, their application for LOB simulation remains relatively understudied. Only a few studies have explored certain aspects of GANs for stock market simulation, with other works focusing solely on generating individual stock price time series. This section reviews related works pertaining to the aforementioned aspects of LOB simulation (see Table I).

The multi-agent approach is widely adopted as a technique for market generation, simulating interactions among agents in the market. It involves emulating multiple types of traders with diverse trading strategies and testing the performance of new experimental trading strategies by simulating market responses to modifications of agent archetypes within the simulation.

The authors of [21] proposed an Agent-Based Interactive Discrete Event Simulation (ABIDES) environment, which provides the capability to simulate interactions among various types of trading agents. This simulation occurs within a continuous double-auction mechanism, with an exchange agent, utilizing a Limit Order Book (LOB) featuring price-then-FIFO matching rules. Additionally, ABIDES incorporates a simulation kernel responsible for managing the flow of time and facilitating all inter-agent communication. The objective of ABIDES is to replicate a realistic financial market environment by simulating the characteristics observed in real financial markets.

TABLE I. Market Simulation Techniques

Ref.	Objective	Technique
[17]	Generation of financial time series	Generative adversarial networks (GANs)
[19]	Simulating market orders	Conditional GAN
[18]	Simulating market orders	Conditional Wasserstein GAN with Gradient Penalty.
[20]	Simulating market orders	Conditional GAN
[21]	Simulating agents interactions	ABIDES
[22]	Simulating orders execution	Reinforcement learning (RL)
[23]	Calibration of multi-agent simulation	GAN with self-attention
[24]	Simulating multi-agent systems	Reinforcement learning
[25]	Simulating multi-agent systems	Model generating transactions
[26]	Simulation of the order flow	Sequence Generative Adversarial Network (SeqGAN)
[27]	Simulating order optimal execution	ABIDES and Re-inforcement learning
[28]	Generation of financial time series	Wasserstein GAN

Furthermore, in study [27] the authors proposed a multi-agent LOB simulation environment for the training of RL execution agents within ABIDES. By comparing the LOB stylized facts on simulations using their method with the ones of a market-replay simulation using real LOB data, they showed the realism of their simulations.

In study [22], an approach is deployed to model order execution decisions based on signals derived from LOB knowledge by a Markov Decision Process; and train an execution agent in a LOB simulator, which simulates multi-agent interaction.

The reference in [23] introduced a method, for the calibration of multiagent simulators, that can distinguish between "real" and "fake" price and volume time series as a part of GAN with self-attention, and then utilize it within an optimization framework to tune the parameters of a simulator model with known agent archetypes to represent market scenarios.

In study [24], the authors designed a multi-agent stock market simulator, in which each agent learns to trade autonomously via reinforcement learning. The authors showed that the proposed simulator can reproduce key market microstructure metrics, such as various price autocorrelation scalars over multiple time intervals.

Lastly, in study [25] a simulative model of a financial market, based on the LOB data is presented. The traders' heterogeneity is characterized by their trading rules, and by introducing a behavioral individual risk aversion and a learning ability.

The aforementioned works depend on explicit hand-crafted rules and intricate assumptions to tailor simulations to desired specifications. These simulations are influenced by numerous hyperparameters, including the selection of agent types, the variety and quantity of permissible orders, and other factors. While this approach provides considerable flexibility in generating diverse simulated scenarios, it may lead to shortcomings in simulating market dynamics realistically, particularly when compared to real market conditions.

B. Generative Adversarial Networks Background

GANs are frameworks for training generative models [6]. A GAN has two components: a Generator G which learns to produce synthetic examples looking like the real ones and, a Discriminator D which tries to discriminate between real and synthetic examples. To train a GAN, a vector of noise Z~P_Z is fed to Generator G which tries to map this vector to real data. The adversarial learning process corresponds to the following Minmax function:

$$min_G max_D \; E_{x\sim p_{data(x)}}[logD(x)] + E_{z\sim p_{z(z)}}\big[\log(1 - D\big(G(Z)\big))\big] \quad (1)$$

In practice, GANs with the Minmax function are hard to train due to the mode collapse challenge [29]. Ref. [30] proposed a solution to fix this issue, namely the Wasserstein GAN (WSGAN).

Even if, the WSGAN shows smooth training, [31] explained that vanishing or exploding gradients can always occur. They proposed a solution, the WGAN-GP, which adds a gradient penalty to the objective WSGAN function:

$$min_G max_D \; E_{x\sim p_{data(x)}}[D(x)] - E_{z\sim p_{z(z)}}\big[D\big(G(Z)\big)\big] - \lambda E_{\hat{x}\sim p_{\hat{x}}(\hat{x})}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1\,)^2\,] \quad (2)$$

λ is the gradient penalty coefficient. Gradients are calculated in linear interpolation $\hat{x}\sim p_{\hat{x}}(\hat{x})$ between real and synthetic examples, $p_{\hat{x}}$ is the sampling distribution of those linear interpolations. WGAN-GP is the actual state of the art in many domains: image field [32], airfoil shapes simulation [33], and text processing [34].

1) GAN for Time-series generation: In study [17] the authors proposed a generative adversarial network for financial time-series modeling. The model relies on a generator with a multilayer perceptron (MLP), convolutional neural networks (CNNs), and the combination of these two neural networks (MLP-CNNs), the same architecture is used for the discriminator. The proposed approach is assessed regarding the ability to reproduce some major stylized facts of the studied data. They showed that the proposed model produces a time series that recovers the statistical properties of financial time series.

In study [26] the authors used the Sequence Generative Adversarial Network (SeqGAN) for modeling the order flow to simulate the intraday price variation. To assess the performance of the proposed framework a comparison is made between the generated data and the real ones regarding the returns distribution tails and the volatility of the mid-price time series. The experimental results showed that their method reproduces the statistics of real data better than the benchmark.

Authors in study [28] used Wasserstein GAN for data augmentation to generate stock market order time series. Using data from Tokyo Stock Exchange, they showed that the probability distribution of synthetic order events generated by the GAN was close to reality.

The aforementioned approaches aim to enhance stock market prediction accuracy by augmenting training datasets with synthetic examples. While the application of GANs in this context shows promise in improving stock market modeling, it primarily addresses aspects related to price variation. However, market simulation presents a more complex challenge, as it seeks to model the microstructure dynamics of the stock market at the order level.

2) GAN for stock market simulation: In study [19] the authors proposed an approach to generate stock market orders based on conditional GANs. The adopted architecture relies on LSTM and convolutional layers for the generator and discriminator. The generator considers, in addition to historical data, handcrafted features that approximate market mechanisms. This work includes an ablation study to assess the importance of each component of the proposed architecture and provides a set of assessment metrics to evaluate the quality of generated data. For comparison purposes, two baseline models are used: Variational auto-encoder and Deep Convolutional Generative Adversarial Network (DCGAN). The proposed method showed better results than the benchmarks.

In study [20], a conditional GAN was used to build a framework called LOB-GAN to simulate the market ordering behavior. They used the LOB-GAN to help a reinforcement learning-based trading portfolio agent to make better generalizations. Their experimental results suggest that the framework improves out-of-sample portfolio performance by 4%.

Authors in study [18] proposed a market generator to simulate synthetic market orders. The quality of generated data is assessed in terms of stylized facts. The adopted architecture relies on a Conditional Wasserstein Generative Adversarial Network with Gradient Penalty. The generator and discriminator use long short-term memory (LSTM) and convolutional layers. The output of the proposed approach is a single order to feed to the exchange given the current state of the market. This work does not provide an ablation study to assess the contribution of each component in the proposed framework.

Authors in study [35] surveyed the metrics to assess the robustness and realism of the market simulation. This work proposes a catalog of known stylized facts regarding LOB microstructure behavior: return distributions, volumes, and order flow, non-stationary patterns, order market impact, and, cross-asset correlations.

Although the studies mentioned earlier show promise, they are not without significant challenges and limitations. These include reliance on manually engineered features, limitations in simulating individual orders rather than entire market dynamics, and the complexity inherent in the models utilized. These factors may hinder the scalability and accuracy of the simulations, thus impacting their effectiveness in capturing real-world market behavior.

III. The Proposed Framework

The objective is to simulate the dynamics of the Limit Order Book (LOB) by generating synthetic data. To achieve this goal, a generative framework has been developed. This section provides a detailed exposition of the principal components constituting the proposed framework.

A. *Overview of the Frameworks*

Fig. 2 offers an overview of the main steps of the proposed framework. Initially, raw Limit Order Book (LOB) data undergo preprocessing, adopting a spatial-temporal representation conducive to machine learning tasks. To capture the spatial-temporal dependencies inherent in market data, a transformer-based temporal model is selected for the generator, while a convolutional neural network serves as the discriminator. Additionally, both a condition vector and a noise vector are inputted into the generator, enhancing its capability to effectively capture local and temporal correlations.

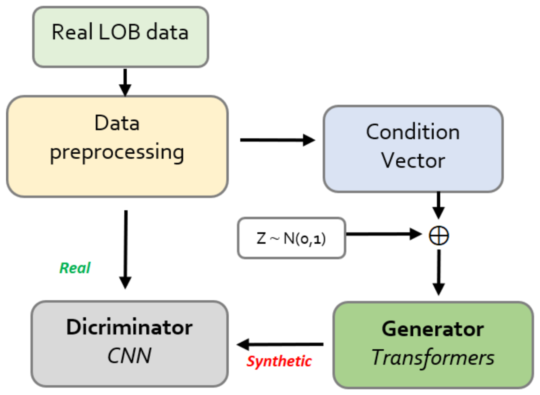

Fig. 2. The principal components of the proposed framework are depicted as a data flow pipeline. Real data undergo preprocessing before being passed to the Generator and the Discriminator, operating within an adversarial scheme.

B. *Data Representation and Preprocessing*

The commonly-used representation of the LOB consists of a time series of multiple levels of orders. It's a series of timely indexed snapshots with a local structure of the ask and bid orders organized by price levels as illustrated in Fig. 3.

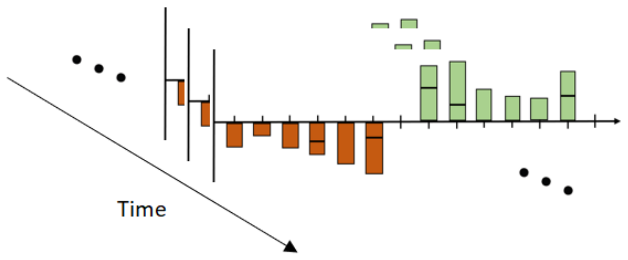

Fig. 3. Representation of the LOB as time-indexed consecutive snapshots. A snapshot represents the state of the price level structure of the LOB at a given moment.

Each input data point in the LOB can be expressed as $\vec{x} \in \mathbb{R}^{T\times 4L}$. Where T is the history of the stock snapshots reflecting the evolution of the LOB after each event such as execution, modification, or cancellation. L is the number of price levels on each side of the LOB. The snapshot is a spatial representation of the LOB in terms of the price level. Let's i in $[1, L]$, the snapshot is a vector $x_t = \{p_a^i(t), v_a^i(t), p_b^i(t), v_b^i(t)\}_{i=1}^{L}$ the $p_a^i(t)$ and $p_b^i(t)$ are the ask and bid prices and $v_a^i(t)$ and $v_b^i(t)$ are the ask and bid volumes. Hence, the 4L representation expresses the length of each snapshot in the LOB at time t.

The spatial-temporal representation of the LOB implies many challenges from a machine-learning point of view. The prices-levels representation does not yield local smoothness of the LOB features. As an illustration, let's consider a LOB with only three levels of prices on each side. Fig. 4 shows the initial state of the LOB at t=i. Each side contains active orders at the price levels 95, 97, and 99 for the bid side and 101, 103, and 106 for the ask side. This state will be perturbed by a bid at the price of 98 and ask at price of 102.

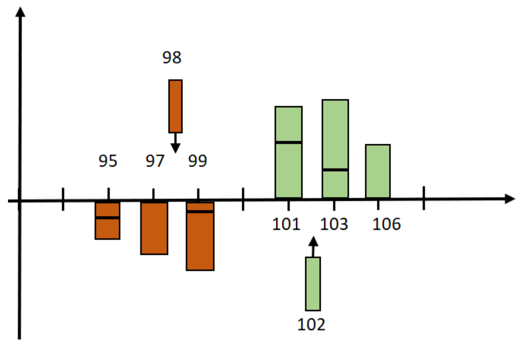

Fig. 4. LOB's snapshot at time t=i with price levels ranging from 95 to 106 and an upcoming new ask and bid orders at prices 102 and 98.

The new state of the LOB resulting from these two orders is depicted in Fig. 5. It is evident that the new price levels on each side have undergone a complete transformation, as price levels 95 and 106 are no longer visible in this LOB with 3 price levels. Instead, the bid side now consists of price levels 97, 98, and 99, while the ask side comprises levels 101, 102, and 103. This illustration highlights how even a minor perturbation can induce a substantial shift in the data structure. Put differently, any slight perturbation of the LOB due to changes in price level values causes a significant alteration in the LOB snapshots. Such variability poses a challenge to model robustness, as consecutive LOB snapshots can exhibit entirely different data structures. To address this challenge, a modification in data representation is adopted by employing "mid-price-centered moving windows" as introduced by [36].

In LOB simulation, the mid-price data is a region of particular interest since it is generally where the stock price is formed by the matching of the bid and ask order. The "mid-price-centered moving windows" representation [36] consists of a two-dimensional window around the mid-price for a time point, it contains N LOB history and 2W + 1 continuous price levels stepped by the tick size Δp. This representation gives a view of the LOB within a history of N time point and a price range [p(t) − WΔp,p(t) + WΔp].

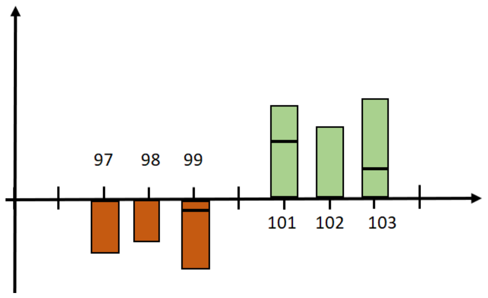

Fig. 5. LOB's snapshot structure at time t=i+1 following the processing of the new ask and bid orders with price levels ranging from 97 to 103 instead of 95 to 106.

In this new two-dimensional LOB's representation x ∈ RN×2W+1, each element xn,i, n = 1,...,N, i = 0,...,2W of the moving window representation indicates the volume of limit orders at price level p(t)−WΔp+i and at LOB snapshot t − N + n The ask side is marked by xn,i > 0 and the bid side by xn,i < 0 and |xn,i| for volume size. Fig. 6 illustrates this 2-dimensional representation of the first 200 LOB snapshots from the FI-2010 dataset [37].

This data representation provides an efficient data structuration around the mid-price region and data are summarised in a spatial-temporal presentation of the ask and bid volume. The extent of this region [p(t)−WΔp,p(t)+WΔp] is a hyperparameter to be determined during the training process.

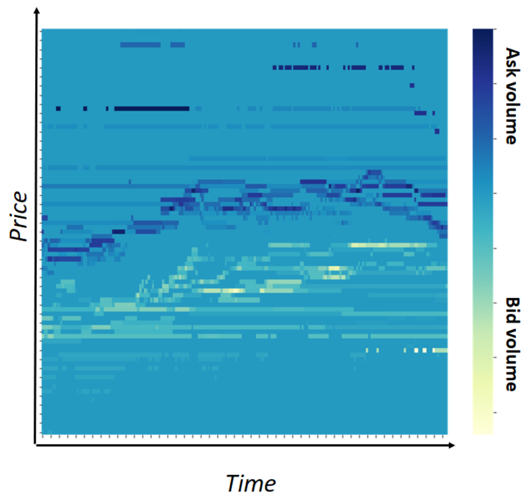

Fig. 6. Mid-price-centered moving windows of the LOB snapshots.

While providing a harmonious data view, the mid-price-centered moving windows are a sparse presentation of the LOB data. To overcome this limitation, [36] proposes an interesting variation namely the accumulated moving window representation. Each element xn,i becomes the sum of total volumes up to the corresponding price level on each side in the n-th snapshot. Fig. 7 illustrates the accumulated moving window representation of the first 200 LOB snapshots from the FI-2010 dataset.

After the aforementioned preprocessing steps, the input data are in the form of x ∈ RN×2W+1. For the FI-2010 dataset used for this study, we consider N = 50 and W = 20.

C. The Generator

The choice of a Transformer [38] to model LOB data is motivated by its ability to efficiently capture interactions and long-range dependencies in sequential data. In addition, Transformer offers a great computational performance due to its parallel matrix multiplication operations since it has no recurrence.

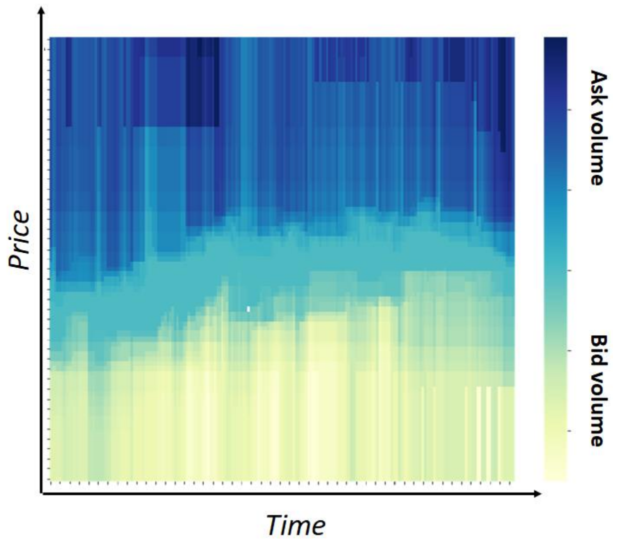

Fig. 7. The accumulated moving windows of the LOB snapshots.

The Transformer model relies on a self-attention mechanism that learns regions of interest by considering all past snapshots in the LOB history. In particular, it is expected from the Transformer Generator to efficiently capture the dynamic around the mid-price region.

The Transformer is composed of an Encoder-Decoder structure. In the Encoder, we find two sublayers: self-attention followed by a position-wise feed-forward layer. The Decoder has three sublayers: self-attention, cross attention, and position-wise feed-forward layer. To prevent information loss, the Transformer uses residual connections between sublayers. Self-attention sublayers employ multiple attention heads that learn different sets of attention projections.

The attention used by the Transformer is given by:

$$Attention(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = softmax(\frac{QK^T}{\sqrt{D_k}}) \quad (3)$$

where, $\boldsymbol{Q} \in \mathbb{R}^{N \times D_k}$ are queries, $\boldsymbol{K} \in \mathbb{R}^{M \times D_k}$ are keys, $\boldsymbol{V} \in \mathbb{R}^{M \times D_v}$ and N, M are the lengths for queries and keys (or values). D_k, D_v for keys (or queries) and values dimensions.

In the LOB context, the next snapshot is highly influenced by the past ones. Hence, for the positional encoding, a relative positional encoding [39] is opted for, as it is believed that the relative positions or distances between snapshots are a key element in LOB simulation.

D. The Transformer

The task of the discriminator (or the Critic when using WGAN-GP) is to accurately distinguish between real and synthetic data. In our context, the discriminator task is a classification between real and synthetic snapshots. Convolutional Neural Networks (CNN) are known for their good performance in images and natural language processing. Since snapshots have spatial-temporal structures, it's believed that the use of a Convolutional Neural Network (CNN) as a discriminator is a good choice. Within the GAN settings, to get a well trained discriminator the generator is expected to produce a wide variety of diverse examples.

The CNN is composed mainly of two components: a convolution layer and a pooling layer. The convolution layer applies several filters to the input data which extract the key features of the input data. The number of convolution filters and layers is a hyperparameter to be determined. The pooling layer downsamples the feature map while preserving most information to ensure a more robust representation regarding the location change of the feature map produced by the convolution filters. To perform classification, the extracted features are connected to a linear layer followed by a sigmoid function.

E. The Condition Vector

To further enhance the capacity of the generator to efficiently capture the past snapshots dynamic and produce a "contextualized" output, a conditional GAN architecture [40] is adopted. To do so, the generator relies on random vectors as well as a condition vector to produce synthetic examples. As a condition, a vector of the last 20 snapshots is used. Under these settings, the produced samples ($\hat{x}$) by the Generator can be interpreted as a conditional distribution of x given Y : $\hat{x} \sim \mathbb{P}_G(x \mid Y)$. The random vector size is set to 100.

F. Training and Hyperparameters

The proposed framework includes several essential hyperparameters that require precise tuning to achieve consistent outcomes. Extensive experimentation with the FI2010 dataset revealed optimal parameter values, as detailed in Tables II and III, which yielded the most favorable results.

TABLE II. GENERATOR HYPERPARAMETERS

Hyperparameter	Value
Batch size	64
Number of heads	5
Number of blocks	2

TABLE III. DISCRIMINATOR HYPERPARAMETERS

Hyperparameter	*Value*
Convolution layers	2
Filter size	5
Convolution activation function	tanh
Pooling size	2
Pooling activation function	ReLu

A dropout layer with a rate of 0.2 is applied before the final linear layer of the generator.

The choice of the aforementioned hyperparameters is adjusted regarding a trade-off between the computational complexity and the output quality.

The model is trained with the WGAN-GP loss using the Adam Optimiser. The learning rate for the generator and critic is 2 × 10−4 and the maximum epoch number is 100.

G. Baselines and Evaluation Procedure

The evaluation of GANs models poses a significant challenge due to their inherent complexity and the multitude of factors influencing their performance. In order to determine the effectiveness of the framework in generating realistic LOB data, a comprehensive approach is imperative. Hence, the following strategy is meticulously devised to assess the fidelity and quality of the generated data:

1) *Baselines:* For comparison purposes, three state-of-the-art benchmarks are used:

a) A framework for market simulation based on a Conditional GAN (CGAN) as in [18, 19].

b) A Recurrent Variational Autoencoder (VAE) [39].

c) And a Deep Convolutional Generative Adversarial Network (DCGAN) [41].

2) *Qualitative assessment:* To evaluate the framework, qualitative and quantitative approaches are employed. The qualitative approach involves visual comparisons between the distributions of real and synthetic data. While it is acknowledged that this evaluation is not definitive, it is considered to provide valuable intuition regarding the output quality.

3) *Quantitative assessment:* To quantitatively assess the performance of the framework, the two-sample Kolmogorov-Smirnov test is employed. In this context, the null hypothesis (H0) posits that "the synthetic and the real data are drawn from the same distribution.

4) *Ablation study:* To explore the individual contributions of each component within the framework towards generating realistic data, an ablation study is conducted. This rigorous analysis aims to systematically examine the effects of each component, providing insights into their respective influences on the quality of the generated data.

IV. The Experimental Settings

The framework is trained and tested on the FI-2010 dataset [37] which is the new benchmark dataset for LOB modeling [36]. The FI-210 records the LOB, for ten days, for five instruments from the Nasdaq Nordic stock market (Helsinki Stock Exchange). The FI-210 consists of 10 orders on each side of the LOB. The F-210 can be downloaded from: « https://etsin.fairdata.fi ».

The proposed methods of this study were implemented using Python programming language. Keras library was used to implement the GAN-based framework. Experiments were conducted on a computer running the Windows 10 operating system with the configuration of Intel(R) Core (TM) i5-8250U CPU @ 1.60GHz (8 CPUs), 1.8GHz, 8 GB RAM, and 500 Gigabytes hard disk drive.

V. Results and Discussion

As previously outlined, with the adoption of the 2-dimensional LOB representation, the input data are predominantly distributed around the mid-price. Therefore, the primary assessment objective is to evaluate the framework's ability to generate mid-price data that closely approximates real-world observations.

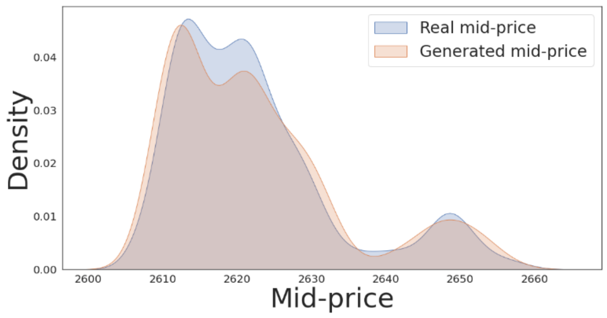

Fig. 8. Mid-price distributions.

Fig. 8 depicts a notable similarity between the generated mid-price data and the real counterparts. Both distributions exhibit similar characteristics, notably in terms of their multi-mode attributes, suggesting a strong correspondence between the simulated and actual data.

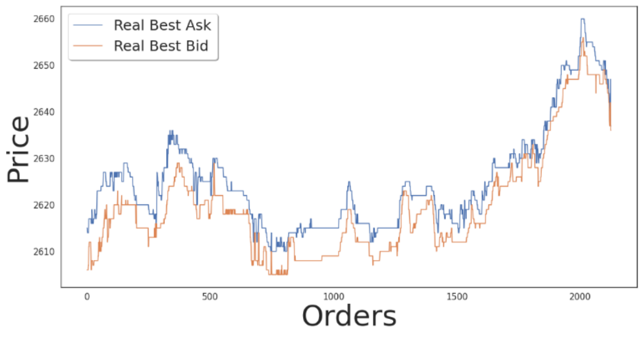

Fig. 9. Real ask and bid of consecutive orders.

One critical aspect that cannot be overlooked is the temporal correlation inherent in the distributions of the best ask and bid data (see Fig. 9). The framework must generate these data with attributes similar to those observed in real market conditions.

Fig. 10 provides insight into the coherence of temporal correlations exhibited in the generated distributions. This aspect is crucial for capturing the dynamic nature of market data over time. The effectiveness of maintaining such coherence largely hinges on the self-attention mechanism employed by the generator, highlighting its pivotal role in ensuring the fidelity and accuracy of the generated data.

Another crucial aspect to consider is the distributions of the best ask and bid data. It is essential for the framework to generate best ask and bid data that exhibit similar attributes to those observed in real market conditions. Thus, ensuring consistency in these distributions is imperative for the fidelity of the generated data.

Fig. 11 and Fig. 12 visually confirm the close resemblance between the generated best ask and bid data and their real counterparts, showcasing the framework's ability to accurately capture essential market dynamics.

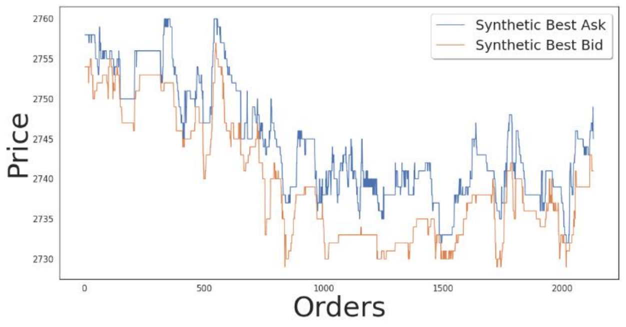

Fig. 10. Synthetic ask and bid distribution.

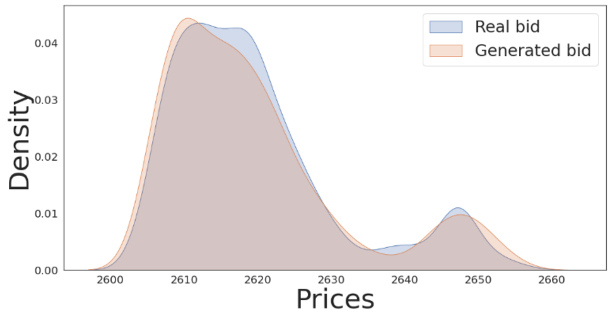

Fig. 11. The bid distributions.

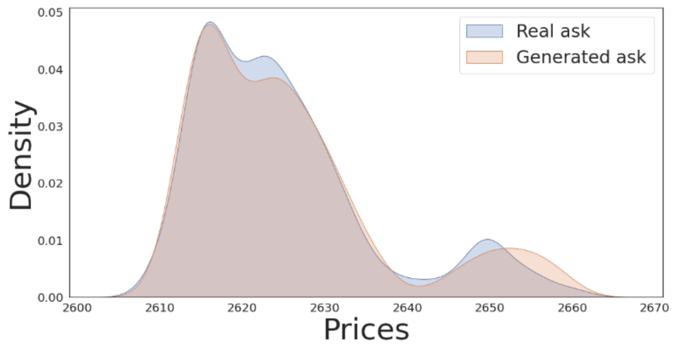

Fig. 12. The ask distributions.

To quantitatively evaluate the proximity and similarity of the generated data to the real ones, a Kolmogorov-Smirnov (K-S) test is conducted. This statistical analysis allows for a comprehensive assessment of the degree of correspondence between the distributions of the generated and real data sets, providing valuable insights into the fidelity and accuracy of the simulated data.

TABLE IV. K-S DISTANCES

	Mid-price	*Best ask*	*Best bid*
Real vs Our framework	0.23	0.32	0.11
Real vs CGAN	0.22	0.39	0.25
Real vs VAE	0.61	0.67	0.72
Real vs DCGAN	0.42	0.45	0.51

These results, presented in Table IV, underscore the framework's performance in generating synthetic data relative to other models, as indicated by the Kolmogorov-Smirnov (K-S) distances. The framework demonstrates superior performance compared to CGAN, particularly in replicating the distributions of best ask and best bid data. While CGAN shows slightly lower K-S distances for mid-price, the framework consistently outperforms across all metrics. Furthermore, when compared to the VAE and DCGAN models, the framework exhibits superior performance. The lower K-S distances achieved by the framework underscore its capability to generate synthetic data that closely resembles real market observations.

To thoroughly assess the contributions of each framework component, an ablation studyis conducted. This analysis enables us to systematically evaluate the impact of individual elements on overall performance, aiding in the optimization and refinement of the framework's design for generating synthetic data.

Table V presents the results of the ablation study, showcasing the impact on the framework's performance, measured by the K-S distances, when key components are removed. When the data representation component is omitted from the framework (using the original data structure instead), there is a noticeable increase in the K-S distances across all metrics. Similarly, removing the condition from the framework leads to a moderate increase in the K-S distances. Furthermore, omitting the Wasserstein Gradient Penalty (WGAN-GP) results in a discernible rise in the K-S distances, indicating a significant contribution of this component to the framework's performance. Similarly, when the Transformer component is replaced with LSTM, there is a notable increase in the K-S distances, highlighting the importance of Transformers in capturing essential temporal dependencies. Moreover, removing the CNN component and replacing it with LSTM also leads to an increase in the K-S distances, indicating the importance of CNNs in capturing spatial dependencies. Overall, these results underscore the critical role played by each component in enhancing the framework's performance in generating synthetic data that closely resembles real market observations.

TABLE V. ABLATION STUDY RESULTS (K-S DISTANCES)

	Mid-price	*Best ask*	*Best bid*
Real vs Our framework	0.23	0.32	0.11
Real vs framework w/o data representation (using original data structure)	0.51	0.63	0.65
Real vs framework w/o condition	0.36	0.47	0.35
Real vs framework w/o WGAN-GP (vanilla GAN)	0.44	0.57	0.49
Real vs framework w/o Transformer (LSTM instead)	0.56	0.59	0.61
Real vs framework w/o CNN (LSTM instead)	0.42	0.51	0.43

VI. CONCLUSION AND FUTURE WORK

In this study, a generative adversarial framework is introduced to produce real-looking synthetic Limit Order Book (LOB) data for simulating stock market dynamics. The proposed framework applies a specific data preprocessing scheme and utilizes conditional Wasserstein GAN with a gradient penalty function to effectively capture the underlying structure of the data. The framework is assessed using both quantitative and qualitative criteria, demonstrating through experimental results that it outperforms existing benchmarks in simulating realistic market conditions. The synthetic data generated by the framework holds promise for various downstream tasks such as forecasting and calibrating trading strategies. The contributions of this research are two fold: aiding in the development of more realistic simulation tools and providing traders with the ability to simulate diverse market scenarios, thereby enhancing financial risk management practices. For future research directions, exploring alternative architectures for the generative framework and incorporating realistic trading strategies to further assess the practical applicability of the proposed solutions is recommended. Additionally, investigating advanced techniques for hyperparameter tuning to ensure the attainment of globally optimal solutions can enhance the overall quality of the evaluated frameworks.

REFERENCES

[1] R. Singh and S. Srivastava, “Stock prediction using deep learning,” Multimed Tools Appl, vol. 76, no. 18, pp. 18569–18584, Sep. 2017, doi: 10.1007/s11042-016-4159-7.

[2] M. Nabipour, P. Nayyeri, H. Jabani, A. Mosavi, E. Salwana, and S. S., “Deep Learning for Stock Market Prediction,” Entropy, vol. 22, no. 8, Art. no. 8, Aug. 2020, doi: 10.3390/e22080840.

[3] H. M, G. E.a., V. K. Menon, and S. K.p., “NSE Stock Market Prediction Using Deep-Learning Models,” Procedia Computer Science, vol. 132, pp. 1351–1362, Jan. 2018, doi: 10.1016/j.procs.2018.05.050.

[4] W. Long, Z. Lu, and L. Cui, “Deep learning-based feature engineering for stock price movement prediction,” Knowledge-Based Systems, vol. 164, pp. 163–173, Jan. 2019, doi: 10.1016/j.knosys.2018.10.034.

[5] A. Madhavan, “Market microstructure: A survey,” Journal of Financial Markets, vol. 3, no. 3, pp. 205–258, Aug. 2000, doi: 10.1016/S1386-4181(00)00007-0.

[6] I. Goodfellow et al., “Generative adversarial networks,” Commun. ACM, vol. 63, no. 11, pp. 139–144, Oct. 2020, doi: 10.1145/3422622.

[7] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive Growing of GANs for Improved Quality, Stability, and Variation.” arXiv, Feb. 26, 2018. doi: 10.48550/arXiv.1710.10196.

[8] A. Brock, J. Donahue, and K. Simonyan, "Large Scale GAN Training for High Fidelity Natural Image Synthesis." arXiv, Feb. 25, 2019. doi: 10.48550/arXiv.1809.11096.

[9] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and Tell: A Neural Image Caption Generator," presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3156–3164.

[10] A. van den Oord et al., "WaveNet: A Generative Model for Raw Audio." arXiv, Sep. 19, 2016. doi: 10.48550/arXiv.1609.03499.

[11] K. Zhang, G. Zhong, J. Dong, S. Wang, and Y. Wang, "Stock Market Prediction Based on Generative Adversarial Network," Procedia Computer Science, vol. 147, pp. 400–406, 2019, doi: 10.1016/j.procs.2019.01.256.

[12] A. Koshiyama, N. Firoozye, and P. Treleaven, "Generative Adversarial Networks for Financial Trading Strategies Fine-Tuning and Combination," arXiv:1901.01751 [cs, q-fin, stat], Jan. 2019.

[13] X. Zhou, Z. Pan, G. Hu, S. Tang, and C. Zhao, "Stock Market Prediction on High-Frequency Data Using Generative Adversarial Nets," Mathematical Problems in Engineering, vol. 2018, pp. 1–11, 2018, doi: 10.1155/2018/4907423.

[14] G. Mariani et al., "PAGAN: Portfolio Analysis with Generative Adversarial Networks." arXiv, Sep. 19, 2019. doi: 10.48550/arXiv.1909.10578.

[15] J. Engelmann and S. Lessmann, "Conditional Wasserstein GAN-based oversampling of tabular data for imbalanced learning," Expert Systems with Applications, vol. 174, p. 114582, Jul. 2021, doi: 10.1016/j.eswa.2021.114582.

[16] M. Diqi, M. E. Hiswati, and A. S. Nur, "StockGAN: robust stock price prediction using GAN algorithm," Int. j. inf. tecnol., vol. 14, no. 5, pp. 2309–2315, Aug. 2022, doi: 10.1007/s41870-022-00929-6.

[17] S. Takahashi, Y. Chen, and K. Tanaka-Ishii, "Modeling financial time-series with generative adversarial networks," Physica A: Statistical Mechanics and its Applications, vol. 527, p. 121261, Aug. 2019, doi: 10.1016/j.physa.2019.121261.

[18] A. Coletta et al., "Towards realistic market simulations: a generative adversarial networks approach," in Proceedings of the Second ACM International Conference on AI in Finance, in ICAIF '21. New York, NY, USA: Association for Computing Machinery, Nov. 2021, pp. 1–9. doi: 10.1145/3490354.3494411.

[19] J. Li, X. Wang, Y. Lin, A. Sinha, and M. Wellman, "Generating Realistic Stock Market Order Streams," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 01, Art. no. 01, Apr. 2020, doi: 10.1609/aaai.v34i01.5415.

[20] C.-H. Kuo, C.-T. Chen, S.-J. Lin, and S.-H. Huang, "Improving Generalization in Reinforcement Learning–Based Trading by Using a Generative Adversarial Market Model," IEEE Access, vol. 9, pp. 50738–50754, 2021, doi: 10.1109/ACCESS.2021.3068269.

[21] D. Byrd, M. Hybinette, and T. H. Balch, "ABIDES: Towards High-Fidelity Market Simulation for AI Research." arXiv, Apr. 26, 2019. doi: 10.48550/arXiv.1904.12066.

[22] S. Vyetrenko and S. Xu, "Risk-Sensitive Compact Decision Trees for Autonomous Execution in Presence of Simulated Market Response." arXiv, Jan. 06, 2021. doi: 10.48550/arXiv.1906.02312.

[23] V. Storchan, S. Vyetrenko, and T. Balch, "Learning who is in the market from time series: market participant discovery through adversarial calibration of multi-agent simulators." arXiv, Aug. 02, 2021. doi: 10.48550/arXiv.2108.00664.

[24] J. Lussange, I. Lazarevich, S. Bourgeois-Gironde, S. Palminteri, and B. Gutkin, "Modelling Stock Markets by Multi-agent Reinforcement Learning," Comput Econ, vol. 57, no. 1, pp. 113–147, Jan. 2021, doi: 10.1007/s10614-020-10038-w.

[25] A. E. Biondo, "Learning to forecast, risk aversion, and microstructural aspects of financial stability," Economics, vol. 12, no. 1, Dec. 2018, doi: 10.5018/economics-ejournal.ja.2018-20.

[26] Y.-S. Lim and D. Gorse, "Intra-Day Price Simulation with Generative Adversarial Modelling of the Order Flow," in 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), Dec. 2021, pp. 397–402. doi: 10.1109/ICMLA52953.2021.00068.

[27] M. Karpe, J. Fang, Z. Ma, and C. Wang, "Multi-agent reinforcement learning in a realistic limit order book market simulation," in Proceedings of the First ACM International Conference on AI in Finance, in ICAIF '20. New York, NY, USA: Association for Computing Machinery, Oct. 2021, pp. 1–7. doi: 10.1145/3383455.3422570.

[28] Y. Naritomi and T. Adachi, "Data Augmentation of High Frequency Financial Data Using Generative Adversarial Network," in 2020 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), Dec. 2020, pp. 641–648. doi: 10.1109/WIIAT50758.2020.00097.

[29] T. Salimans et al., "Improved Techniques for Training GANs," in Advances in Neural Information Processing Systems, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., Curran Associates, Inc., 2016.

[30] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein Generative Adversarial Networks," in Proceedings of the 34th International Conference on Machine Learning, PMLR, Jul. 2017, pp. 214–223.

[31] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved Training of Wasserstein GANs," in Advances in Neural Information Processing Systems, Curran Associates, Inc., 2017.

[32] Q. Jin, R. Lin, and F. Yang, "E-WACGAN: Enhanced Generative Model of Signaling Data Based on WGAN-GP and ACGAN," IEEE Systems Journal, vol. 14, no. 3, pp. 3289–3300, Sep. 2020, doi: 10.1109/JSYST.2019.2935457.

[33] K. Yonekura, N. Miyamoto, and K. Suzuki, "Inverse airfoil design method for generating varieties of smooth airfoils using conditional WGAN-gp," Struct Multidisc Optim, vol. 65, no. 6, p. 173, Jun. 2022, doi: 10.1007/s00158-022-03253-6.

[34] M. Hu, M. He, W. Su, and A. Chehri, "A TextCNN and WGAN-gp based deep learning frame for unpaired text style transfer in multimedia services," Multimedia Systems, vol. 27, no. 4, pp. 723–732, Aug. 2021, doi: 10.1007/s00530-020-00714-0.

[35] S. Vyetrenko et al., "Get real: realism metrics for robust limit order book market simulations," in Proceedings of the First ACM International Conference on AI in Finance, in ICAIF '20. New York, NY, USA: Association for Computing Machinery, Oct. 2020, pp. 1–8. doi: 10.1145/3383455.3422561.

[36] Y. Wu, M. Mahfouz, D. Magazzeni, and M. Veloso, "Towards Robust Representation of Limit Orders Books for Deep Learning Models." arXiv, Oct. 10, 2021. doi: 10.48550/arXiv.2110.05479.

[37] A. Ntakaris, M. Magris, J. Kanniainen, M. Gabbouj, and A. Iosifidis, "Benchmark dataset for mid-price forecasting of limit order book data with machine learning methods," Journal of Forecasting, vol. 37, no. 8, pp. 852–866, 2018, doi: 10.1002/for.2543.

[38] A. Vaswani et al., "Attention is All you Need," in Advances in Neural Information Processing Systems, Curran Associates, Inc., 2017.

[39] J. Chung, K. Kastner, L. Dinh, K. Goel, A. Courville, and Y. Bengio, "A Recurrent Latent Variable Model for Sequential Data." arXiv, Apr. 06, 2016. doi: 10.48550/arXiv.1506.02216.

[40] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets." arXiv, Nov. 06, 2014. doi: 10.48550/arXiv.1411.1784.

[41] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks." arXiv, Jan. 07, 2016. doi: 10.48550/arXiv.1511.06434.