Simulation-based Analysis on Optimal High-Frequency Market-making Using a Deep Neural Network Method

Bekhzodbek Najmiddinov[b], Hyun Jin Jang[b,], So Eun Choi[a]

[a]*Samsung Advanced Institute of Technology (SAIT), Suwon 16678, Republic of Korea*
[b]*School of Business Administration, Ulsan National Institute of Science and Technology (UNIST), Ulsan 44919, Republic of Korea*

Abstract

This study investigates the relationships for high frequency (HF) market-making strategies and the expected terminal wealth to diverse market scenarios through simulation analysis. We employ the optimal trading strategy for HF market-making constructed on the Hawkes arrival model, which incorporates self/mutually-exciting factors reflecting synchronizing tendency in buy and sell order dynamics. We employ the deep Galerkin method (DGM) to estimate a solution of the high-dimensional partial differential equation that solves the market-maker's optimization problem. With the validation scheme by verifying the distribution of the value function, we assess the DGM-approximated solutions under training of various batch size and learning rate for the selection of the most efficient and robust DGM training model. Using the optimal training model, we conduct sensitivity analysis on the market-maker's profitability with respect to changes in HF market stability caused by several noise components and potential market manipulation by spoofers. Finally, we discuss the practical implications of our findings for enhancing efficacy in HF market-making based on our simulation results.

Keywords: Optimal market-making strategy, Deep neural network, Deep Galerkin method, Validation, Market stability, Spoofing, Sensitivity analysis

1. Introduction

High-frequency (HF) trading has experienced significant growth over the past two decades largely propelled by regulatory changes during the 2000s. Decimalization in 2001[1] and Reg NMS in 2007 (SEC, 2005) are two key changes that played pivotal roles in the expansion of HF trading. Decimalization mandated that securities be priced in increments of one cent rather than fractions of a dollar. This change made it economically feasible for market participants to engage in HF trading by reducing bid-ask spreads and transaction costs,

Email addresses: behzod1996@unist.ac.kr (Bekhzodbek Najmiddinov), cksoeun@kaist.ac.kr (So Eun Choi)

[1] "SEC adopts rules requiring stock markets to quote prices in decimals by early 2001" Press Release No. 2000-84 by US SEC (2000). https://www.sec.gov/news/testimony/ts092000.htm

yielding effectively lowering the barriers to entry for HF traders. Moreover, Reg NMS aimed to enhance competition among exchanges by promoting the best execution of trades and ensuring that investors receive the best possible prices. This regulation further encouraged the adoption of HF trading strategies, as emphasizing the importance of speed and efficiency in executing trades to obtain favorable prices. As a result of these regulatory changes along with development of IT technologies, HF trading has become a dominant force in modern financial markets. Estimates suggest that HF trading volume accounts for a substantial portion of trading activity, ranging from 50 to 70% in US equity and futures markets, while it is estimated to be 40% in the European equity market (Biais and Woolley, 2011; Miller and Shorter, 2016).

Among the strategies employed by HF traders, market-making accounts for 80% in incumbent exchanges and new entrant markets (Menkveld, 2013), which is the most prevailing strategy. Market-making focuses on enhancing liquidity for market participants while generating profits from the bid-ask spread, which involves managing risks associated with inventory, execution, and adverse selection. The distinguishing factor of HF market-making lies in its reliance on order execution speed and automated systems, setting it apart from traditional slower traders. HF market-makers leverage the minimal latency in systems to exploit profit opportunities ahead of slower traders.

Due to the huge involvement of HF market-making trading, the relevant literature primarily focuses on the impact and implications of HF traders' market participation in terms of market stability and efficiency. Ait-Sahalia and Brunetti (2020) provide empirical evidence suggesting that increased noise in price processes leads to a greater number of trading opportunities for HF traders. However, they argue that this heightened activity does not necessarily translate into increased market volatility. In contrast, Zhang (2010) presents a differing perspective, demonstrating that HF trading activity exhibits a positive correlation with stock price volatility and negatively linked to market ability to incorporate firm's fundamental news into asset prices. This suggests that the presence and actions of HF traders may indeed contribute to fluctuations in market volatility, rather than market efficiency. These contrasting findings highlight the complexity of the relationship between HF trading, market noise, and volatility, underscoring the need for further in-depth study to better understand the dynamics at play.

However, there remains a lack of literature on the HF market-maker's expected profit associated with the HF market environment. This is primarily due to the innate complexity of analyzing the relationship in the market microstructure framework. This is also because the optimal values for HF market-making strategies and profits involve numerous market factors as a non-linear manner. The distinguishing factors of HF trading from conventional asset trading include correlations between buy and sell orders, autocorrelation in order arrivals, short-term volatility, and order book imbalance. This study tackles the technical challenges involved in analyzing HF market-making activities through the use of the deep neural network (DNN) approach. We generate the required dataset using the DNN method to investigate our research questions: (i) how changes in HF market stability and potential

market manipulation affect the market-maker's profitability, and (ii) what primary factors influence the relationship between the market-maker's optimal wealth and the HF environment. To the best of our knowledge, this aspect has not been explored in previous literature.

As a market microstructure model, we employ the limit order book model (LOB) proposed by Choi et al. (2021), which is developed based on the Hawkes process including self and mutual-exciting factors and synchronizing tendency between buy and sell order dynamics. Given that optimal market-making trading strategies involve numerous variables in the model of Choi et al. (2021), the challenge of high-dimensionality is addressed by adopting the deep Galerkin method (DGM). This method enables us to estimate solutions for the high-dimensional partial differential equation (PDE) by representing the unknown function of interest through a DNN structure. The conceptual groundwork and its proof for the convergence are laid out by Sirignano and Spiliopoulos (2018) and Grohs et al. (2023). The DGM trains the network function by minimizing losses associated with the differential operators acting on the function, as well as any initial, terminal, or boundary conditions that the solution must satisfy. The input data in DGM consists of samples *randomly generated* in the domain of the PDE, referred to as batches, which are trained to conform to the PDE conditions through sufficient iterations. A notable feature of this approach is its mesh-free nature, distinguishing it from other conventional numerical methods like finite difference methods. Many simulation studies suggest that the DGM may overcome the curse of dimensionality more effectively than other numerical techniques (Beck et al., 2019, 2021).

This study draws three main contributions in aspects of efficiency improvement in DGM and HF market-making management. First, one notable issue to run DGM is the high computational expense required to reach a certain level of accuracy. As stated in Choi et al. (2021), a single training under suitable hyper-parameters demends 16.9 hours, presenting a considerable barrier to the further application. Another issue in this regard is the impossibility of validation for the trained model, because the true value, namely the solution of the PDE, is generally unknown including our problem. This renders testing efficiency for the DGM models across various hyper-parameters infeasible. To address these issues, we propose an indirect scheme for validation in DGM that simulates the distribution of the value function employing the DGM-approximated optimal controls under a newly generated data set. This approach enables for assessing the trained models from a different perspective of accuracy along with the training loss profile. It also allows a comprehensive evaluation of potential overfitting issues in the DGM estimation.

Next, we establish the practical relationship between batch size and the number of iterations applied in DGM, which are fundamental components to run the training loss minimization procedure. Our findings in this regard are as follows: (i) the number of iteration decreases as batch size increases, and (ii) the increase in batch size demands more necessary computing resource, despite of the trade-off between batch size and the number of iterations. In other words, smaller batch necessitates further iterations to achieve a comparable performance to that of a larger batch model. This is because it estimates less accurate gradients, requiring more steps to reach the ultimate optimal point. However, the total

minimum training cost rises as batch size grows, and the corresponding computing time also does accordingly. Our findings are aligned with the prevailing results in general neural network training including Shallue et al. (2019) who empirically derive the relationship that increasing the batch size decreases the number of required training steps proportionally. Such relationships allow us to have an optimal DGM model selection given the conditions available computational cost and desired accuracy level.

Lastly, we verify the relationship between the market-maker's maximal expected wealth and the degrees of stability and potential manipulation in the HF market using the optimally generated data from DGM. HF market stability is assessed through the bounded condition of the Hawkes model, which is further decomposed into multiple influencing components, including long-term persistent trader relationships and transient noises. Additionally, HF market manipulation focuses on spoofing behaviors. For the market stability test, we observe that higher market instability leads to higher profits, and vice versa, but this relationship holds true only when buy and sell orders are more synchronized or when market orders exhibit greater self-excitement. In essence, the synchronization and self-exciting tendencies in HF market order arrivals significantly contribute to enhancing the market-maker's profitability. However, in cases where market stability is influenced by other parameters, such as mutual-excitement and mean-reversion speed factors, no significant relationship is observed.

For the HF market manipulated by spoofers, we examine how spoofing acting on limit orders can escalate adverse impacts on the market-maker's profit, considering both the frequency and intensity of spoofing patterns. Our finding in this regard is that frequency does indeed significantly worsen the market-maker's profitability, whereas intensity has a limited impact on reducing profit. This implies that frequent and coordinated spoofing behaviors, even with smaller sizes of unintended limit orders to execute, can greatly disrupt market-makers from performing effectively and diminish their profitability. These observations offer valuable insights for both HF market-making traders and policymakers to monitor HF trading practices effectively.

The remainder of this paper is organised as follows. Section 2 reviews the relevant literature. Section 3 explains the LOB model by Choi et al. (2021) incorporating with the synchronising factor and the optimal market-making strategies as a solution of the associated PDE, and Section 4 presents the DGM procedure for estimate the PDE. Section 5 discusses a novel scheme of validation for DGM and overfitting issues through training two representative cases. Section 6 demonstrates the relationship between batch size and the number of iterations to seek the optimal DGM model. Section 7 presents the simulation results for a market-maker's wealth with respect to HF market stability using the optimal strategies. Section 8 concludes, and supplementary figures are displayed in Appendix A.

2. Literature Review

The optimal market-maker's problem under order dynamics has been widely studied in the market microstructure field. Ho and Stoll (1981) discuss the optimal market-making

policy by specifying a true price for assets on the supply and demand curves of a public market. They derive the optimal bid and ask quotes around the true price by accounting for the inventory effect. In this spirit, Avellaneda and Stoikov (2008) propose a market-making model in an order book by employing a diffusion process for a mid-price and a Poisson process for executed limit orders. For the exponential utility function, this provides an asymptotic solution for quoting spreads and reservation prices. Rosu (2009) derives an equilibrium transaction price between market and limit orders in terms of utility by considering the trade-off between execution prices and waiting costs at the bounded discrete price levels. Gueant et al. (2013) study the same problem as Avellaneda and Stoikov (2008) by adding inventory volume constraints and then approximate the optimal control with asymptotic limits over an infinite time horizon. Cartea et al. (2014) model the dynamics of the arrival of market orders and resulting changes in the LOB's shape with self-exciting and mutually exciting Hawkes processes. Guilbaud and Pham (2013) investigate the optimal market-making policy when an agent uses both limit, and market orders by employing a numerical method that estimates the optimal problem. Cartea and Jaimungal (2013) study modelling price revision and duration for HF trading activities using the hidden Markov model with regime switching. Veraart (2010) models two types of market-making actions in foreign exchange markets, removing, and adding liquidity with two-dimensional Brownian motions. Guo et al. (2017) employ a correlated random walk for the best bid/ask prices and solve the optimal placement problem with a reflection principle.

Since the drastic emergence of HF traders, there have been debates and contraversies of the roles played by HF trading, but large empirical studies support the views that HF traders enhance market quality and efficiency. Hendershott et al. (2011) find that HF traders improve liquidity and enhance the informativeness of quotes, using the automation of quote dissemination as an exogenous change in market structure. They find that for large stocks in particular, HF traders narrow spreads, reduce adverse selection, and reduce trade-related price discovery. Hasbrouck and Saar (2013) show that increased HF traders' activity is associated with lower posted and effective spreads, increased depth, and lower short-term volatility. Brogaard et al. (2014) investigate benevolent roles for HF traders, as they provide liquidity, yielding to increase price discovery. Chaboud et al. (2014) estimate that HF trading is responsible for 60 to 80% of price discovery procedure mainly through limit orders. Biais et al. (2016) find that HF traders provide liquidity by leaving limit orders in the LOB, thus helping the market absorb shocks. Brogaard et al. (2019) find that HF traders are major contributors of limit orders and thus influence price discovery significantly. However, their behavior and contribution decrease with higher market volatility. This suggests that HF traders' informational advantage is partly due to their faster reactions to public information.

Empirical studies on market-maker profit opportunities have been largely focused on the analysis in terms of trading and regulatory systems and operations. Transaction fees and trading duration are a substantial part of HF trader's profit. Menkveld (2013) estimate that positions lasting less than five seconds make a profit, whereas ones lasting longer than that generally lose money. Baron et al. (2019) find that differences in relative latency account for

large differences in HF trading firms' performance. They also find the stronger performance for HF market-makers is substantially linked with relatively higher speed. Menkveld and Zoican (2017) investigate that faster exchanges may actually harm liquidity by intensifying competition among HF market-makers and speculators, whereas they allows the HF market-maker to update one's quotes more quickly and reduce one's payoff risk.

The DGM, an innovative numerical method for solving PDEs, is theoretically grounded from the universal approximation theorem (Hornik, 1991) which states that DNNs can estimate a wide variety of arbitrarily continuous functions on compact sets with appropriate weights. Since being introduced by Sirignano and Spiliopoulos (2018) for DGM, a substantial body of literature has provided rigorous mathematical proofs regarding its approximation capabilities and their extensions. For the convergence, Grohs et al. (2023), Jiao et al. (2023), Hutzenthaler et al. (2020) provide convergence and tractability results with dimension independent convergence rates and error constants depending polynomially on the input dimension. For the extension, Chen et al. (2021) propose the DGM to solve hyperbolic PDEs with discontinuous solutions and random uncertainties. Al-Aradi et al. (2022) extend the DGM to solve for the value function and the optimal control simultaneously by characterizing both as DNN, for the case where the PDE has a constraint. Beck et al. (2022) recently overview all the literature of DGM.

3. Limit Order Book Model and Optimization Problem

We define a filtered probability space $(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, \mathbb{P})$ satisfying the usual conditions. Assume that all stochastic processes in this paper are defined on $(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, \mathbb{P})$. Let S_t be the mid-price of the asset at time t with the dynamics

$$\mathrm{d}S_t = \sigma S_t \mathrm{d}W_t,$$

where σ is a positive constant and W is a standard Brownian motion.

Consider a market-maker who continuously posts a limit buy order and sell order of the asset with depth $\delta_t^-, \delta_t^+ \geq 0$, respectively. In other words, the market-maker posts a buy limit order at a price of $S_t - \delta_t^-$, and a sell limit order at a price of $S_t + \delta_t^+$. The market-maker provides liquidity to the market and earns profits from the bid-ask spread.

We assume that transactions only occur when market orders arrive and match with pending limit orders posted by the market-maker. Let the counting processes M_t^+ and M_t^- with intensities λ_t^+ and λ_t^- denote the arrival of other participants' buy and sell market orders, respectively. We denote the market-maker's filled buy and sell limit orders by the counting processes N_t^- and N_t^+, respectively. As a measure of the chance with which the market-maker's limit buy and sell orders are executed, we consider the fill probabilities $h(\delta_t^\pm, c_t^\pm)$ at time t for limit orders placed $\delta_t^\pm$ away from S_t. The process c_t can be interpreted as parameters directly determining the shape of the limit orders.

From this setup, the processes $N_t^{\pm}$ can be regarded as the pathwise stochastic integral with respect to $M_t^{\pm}$:

$$N_t^- = \int_0^t I_s^- \mathrm{d}M_s^- \text{ and } N_t^+ = \int_0^t I_s^+ \mathrm{d}M_s^+. \tag{1}$$

Here, $I_t^{\pm}$ are defined by

$$I_t^- = \begin{cases} 0 & \text{if } M_t^- - M_{t-}^- = 0 \\ \epsilon_t^- & \text{otherwise} \end{cases} \text{ and } I_t^+ = \begin{cases} 0 & \text{if } M_t^+ - M_{t-}^+ = 0 \\ \epsilon_t^+ & \text{otherwise} \end{cases}, \tag{2}$$

where $\epsilon_t^{\pm}$ are Bernoulli random variables with probability $h(\delta_{t-}^{\pm}, c_{t-}^{\pm})$, respectively. We assume that market order volumes are independent and identically distributed and exponentially distributed, and the shape of the piled limit orders is flat. Given that market orders arrive, the probability that a limit order at price level $S_t \pm \delta_t^{\pm}$ is executed is equal to $h(\delta_{t-}^{\pm}, c_{t-}^{\pm}) = \mathrm{e}^{-\delta_t^{\pm} c_t^{\pm}}$.

The market-maker's execution processes have two driving factors, how often market orders are walking into the LOB $\boldsymbol{\lambda}_t = (\lambda_t^-, \lambda_t^+)_{t \geq 0}$ and how deep is the volume depth for the current limit order $\boldsymbol{c}_t = (c_t^-, c_t^+)$. We employ a synchronising order book model (Choi et al., 2021). In this model, the market orders arrive at the intensity $\boldsymbol{\lambda}_t$ given as

$$\begin{aligned} \mathrm{d}\lambda_t^- &= \beta(\theta^- - \lambda_t^- + \kappa\lambda_t^+)\mathrm{d}t + \eta \mathrm{d}M_t^- + \nu \mathrm{d}M_t^+, \\ \mathrm{d}\lambda_t^+ &= \beta(\theta^+ - \lambda_t^+ + \kappa\lambda_t^-)\mathrm{d}t + \eta \mathrm{d}M_t^+ + \nu \mathrm{d}M_t^- \end{aligned} \tag{3}$$

with constants $\beta, \theta^{\pm} > 0$ and $\eta, \nu, \kappa \geq 0$, and the limit orders have the depth $\boldsymbol{c}_t$ given as

$$\begin{aligned} \mathrm{d}c_t^- &= \beta_c(\theta_c^- - c_t^- + \kappa_c c_t^+)\mathrm{d}t + \eta_c \mathrm{d}M_t^- + \nu_c \mathrm{d}M_t^+, \\ \mathrm{d}c_t^+ &= \beta_c(\theta_c^+ - c_t^+ + \kappa_c c_t^-)\mathrm{d}t + \eta_c \mathrm{d}M_t^+ + \nu_c \mathrm{d}M_t^- \end{aligned} \tag{4}$$

with constants $\beta_c, \theta_c^{\pm} > 0$ and $\eta_c, \nu_c, \kappa_c \geq 0$. This model features that the market order arrival intensity jumps up immediately after any market order arrival, where the parameters η and ν govern how responsive the self-exciting and mutually-exciting components of the intensity are due to additional market orders, respectively. Their states revert to the mean-reversion level $\theta^{\mp} + \kappa\lambda_t^{\pm}$ with speed β since the exciting impacts from market order arrivals are temporary. The depth process of the limit order jumps up with size η_c, ν_c when market orders arrive, which has a one-way effect, unlike the market order intensities. The other components play the same role with the ones in $\boldsymbol{\lambda}_t$, where $\theta_c^{\pm}$ are the long-run mean level, κ_c is the synchronising factor, and β_c is the mean-reverting speed.

Remark 1. *(i) To examine the conditions that guarantee the intensity processes for market orders be stable, we define the mean future rate, $m_t^{\pm}(u) = \mathbb{E}[\lambda_u^{\pm} | \mathcal{F}_t]$, for $u \geq t$. For the processes $\lambda_t^{\pm}$ to be stable, $m_t^{\pm}$ must remain bounded as a function of u for each t. The mean future rates $m_t^{\pm}(u)$*

$$\begin{bmatrix} m_t^-(u) \\ m_t^+(u) \end{bmatrix} = e^{-D(u-t)} \left(\begin{bmatrix} \lambda_t^- \\ \lambda_t^+ \end{bmatrix} - D^{-1}\pi \right) + D^{-1}\pi \tag{5}$$

are bounded for all $u \geq t$ if and only if $(1-\kappa)\beta > \eta + \nu$. Furthermore,

$$\lim_{u\to\infty} m_t^{\pm}(u) = D^{-1}\pi, \quad \textit{where} \quad D = \begin{bmatrix} \beta-\eta & -\kappa\beta-\nu \\ -\kappa\beta-\nu & \beta-\eta \end{bmatrix} \quad \textit{and} \quad \pi = \begin{bmatrix} \beta\theta^- \\ \beta\theta^+ \end{bmatrix}.$$

(ii) We obtain $n_t^{\pm}(u) = \mathbb{E}[c_u^{\pm}|\mathcal{F}_t]$ by solving a system of ODEs derived by taking the integral, conditional expectation, and derivative in Eq.(4). The mean future level $n_t^{\pm}(u)$ is given by, for all $u \geq t$,

$$\begin{bmatrix} n_t^-(u) \\ n_t^+(u) \end{bmatrix} = D_c^{-1}\pi_c + e^{-D_c(u-t)}\left(\int_t^u e^{D_c(s-t)}G(s)ds + \begin{bmatrix} c_t^- \\ c_t^+ \end{bmatrix} - D_c^{-1}\pi_c\right), \tag{6}$$

where

$$D_c = \begin{bmatrix} \beta_c & -\kappa_c\beta_c \\ -\kappa_c\beta_c & \beta_c \end{bmatrix}, \quad \pi_c = \begin{bmatrix} \beta_c\theta_c^+ \\ \beta_c\theta_c^- \end{bmatrix}, \quad \textit{and} \quad G(s) = \eta_c \begin{bmatrix} m_t^-(s) \\ m_t^+(s) \end{bmatrix} + \nu_c \begin{bmatrix} m_t^+(s) \\ m_t^-(s) \end{bmatrix} \tag{7}$$

with $m_t^{\pm}(s)$ in Eq.(5). Eq.(6) shows the mean long future rate yields

$$\lim_{u\to\infty} n_t^{\pm}(u) = D_c^{-1}\pi_c,$$

which are only affected by the synchronizing factor κ_c and the mean-reversion levels $\theta_c^{\pm}$. The impact from the exciting factors μ, ν by market order changes ultimately disappears as time goes, meaning that it is only temporary and not affecting its fundamental level of the limit orders's depths.

From their setup, the market-maker's cash process X_t satisfies

$$\mathrm{d}X_t = (S_t + \delta_{t-}^+)\mathrm{d}N_t^+ - (S_t - \delta_{t-}^-)\mathrm{d}N_t^-,$$

which accounts for the cash increase when a sell limit order is lifted by a buy market order, and the cash decrease when a buy limit order is hit by a sell market order. Accordingly, the market-maker's inventory process q_t is given as

$$\mathrm{d}q_t = \mathrm{d}N_t^- - \mathrm{d}N_t^+.$$

A market-maker seeks the strategy $(\delta_t^-, \delta_t^+)_{0\leq t\leq T}$ that maximises the cash value at the terminal date T. At time T, the market-maker liquidates the terminal inventory q_T using market orders at a price lower than the mid-price to account for liquidity costs as well as the market orders walking the LOB. The performance of the market-maker achieved during $[t, T]$ is given by

$$\Phi_T = X_T + q_T(S_T - \phi q_T) - \psi\int_t^T q_u^2 du, \tag{8}$$

where $\phi \geq 0$ is a cost attributed to liquidity as well as the impact of the market order walking the LOB, and $\psi \geq 0$ is the running inventory penalty parameter.

The value function of the market-maker is given by

$$V(t,x,s,q,\boldsymbol{\lambda},\boldsymbol{c}) = \max_{(\delta_u^+,\delta_u^-)_{t\le u\le T}} \mathbb{E}\left(\Phi_T \middle| X_t = x, S_t = s, q_t = q, \boldsymbol{\lambda}_t = \boldsymbol{\lambda}, \boldsymbol{c}_t = \boldsymbol{c}\right), \tag{9}$$

where ϕ, $\psi \ge 0$, and the initial states of the cash amount x, the stock price s, the inventory amount q, and the intensity levels $\boldsymbol{\lambda} = (\lambda^-, \lambda^+)$, $\boldsymbol{c} = (c^-, c^+)$ are given. From the HJB derived from Eq.(8), the ansatz solution can be obtained

$$V(t,x,s,q,\boldsymbol{\lambda},\boldsymbol{c}) = x + qs + g(t,q,\boldsymbol{\lambda},\boldsymbol{c}),$$

and the optimal controls can be derived in Remark 2.

Remark 2. *(Optimal market-making strategy) The optimal controls $\delta_t^\pm$ are derived as*

$$\begin{aligned}
(\delta_t^-)^* &= \frac{1 - c^-(\Delta^-_{q,\lambda,c} g - \Delta^-_{\lambda,c} g)}{c^-} \mathbb{1}_{\{c^-(\Delta^-_{q,\lambda,c} g - \Delta^-_{\lambda,c} g) < 1\}}, \\
(\delta_t^+)^* &= \frac{1 - c^+(\Delta^+_{q,\lambda,c} g - \Delta^+_{\lambda,c} g)}{c^+} \mathbb{1}_{\{c^+(\Delta^+_{q,\lambda,c} g - \Delta^+_{\lambda,c} g) < 1\}},
\end{aligned} \tag{10}$$

where g satisfies the following PDE

$$\begin{aligned}
&\frac{\partial g}{\partial t} + \beta(\theta^- - \lambda^- + \kappa\lambda^+)\frac{\partial g}{\partial \lambda^-} + \beta(\theta^+ - \lambda^+ + \kappa\lambda^-)\frac{\partial g}{\partial \lambda^+} \\
&\quad + \beta_c(\theta_c^- - c^- + \kappa_c c^+)\frac{\partial g}{\partial c^-} + \beta_c(\theta_c^+ - c^+ + \kappa_c c^-)\frac{\partial g}{\partial c^+} \\
&\quad + \lambda^-(\Delta^-_{q,\lambda,c} g - g)\mathbb{1}_{\{c^-(\Delta^-_{q,\lambda,c} g - \Delta^-_{\lambda,c} g) \ge 1\}} + \lambda^+(\Delta^+_{q,\lambda,c} g - g)\mathbb{1}_{\{c^+(\Delta^+_{q,\lambda,c} g - \Delta^+_{\lambda,c} g) \ge 1\}} \\
&\quad + \lambda^- \left(\frac{\mathrm{e}^{c^-(\Delta^-_{q,\lambda,c} g - \Delta^-_{\lambda,c} g)}}{\mathrm{e} c^-} + \Delta^-_{\lambda,c} g - g\right) \mathbb{1}_{\{c^-(\Delta^-_{q,\lambda,c} g - \Delta^-_{\lambda,c} g) < 1\}} \\
&\quad + \lambda^+ \left(\frac{\mathrm{e}^{c^+(\Delta^+_{q,\lambda,c} g - \Delta^+_{\lambda,c} g)}}{\mathrm{e} c^+} + \Delta^+_{\lambda,c} g - g\right) \mathbb{1}_{\{c^+(\Delta^+_{q,\lambda,c} g - \Delta^+_{\lambda,c} g) < 1\}} - \psi q^2 = 0
\end{aligned} \tag{11}$$

with the terminal condition $g(T,q,\boldsymbol{\lambda},\boldsymbol{c}) = -\phi q^2$. and the shift operators $\Delta^\pm_{q,\lambda,c}$ and $\Delta^\pm_{\lambda,c}$ are defined as follows:

$$\begin{aligned}
\Delta^-_{q,\lambda,c} g(t,q,\boldsymbol{\lambda},\boldsymbol{c}) &= g(t,q+1,\boldsymbol{\lambda}+(\eta,\nu),\boldsymbol{c}+(\eta_c,\nu_c)), \\
\Delta^+_{q,\lambda,c} g(t,q,\boldsymbol{\lambda},\boldsymbol{c}) &= g(t,q-1,\boldsymbol{\lambda}+(\nu,\eta),\boldsymbol{c}+(\nu_c,\eta_c)), \\
\Delta^-_{\lambda,c} g(t,q,\boldsymbol{\lambda},\boldsymbol{c}) &= g(t,q,\boldsymbol{\lambda}+(\eta,\nu),\boldsymbol{c}+(\eta_c,\nu_c)), \\
\Delta^+_{\lambda,c} g(t,q,\boldsymbol{\lambda},\boldsymbol{c}) &= g(t,q,\boldsymbol{\lambda}+(\nu,\eta),\boldsymbol{c}+(\nu_c,\eta_c)).
\end{aligned}$$

4. Deep Neural Network Methods

This section discusses the DGM to approximate for a solution of the HJB equation derived in Section 3, and also shows the optimizor to run DGM. The DGM trains batches of

randomly sampled time and space points from the domain of the PDE as input values and produce a candidate solution of the PDE through a smooth neural networks with activation function repeatedly. The training procedure aims to find the best DNN parameters that minimize a loss function, which indicates how close the DNN architecture is to satisfying the PDE's given conditions.

We choose a domain of the PDE such that $\mathcal{D} = (-\infty, \infty) \times [0, \infty)^4 \subset \mathbb{R}^5$ and $\mathcal{D}_T = [0, T) \times \mathcal{D}$. We assume that there exists a unique solution $u(t, q, \boldsymbol{\lambda}, \boldsymbol{c}) \in C^1(\mathcal{D}_T)$ to the following PDE:

$$\begin{aligned} &\frac{\partial u}{\partial t}(t, q, \boldsymbol{\lambda}, \boldsymbol{c}) + \mathcal{L}u(t, q, \boldsymbol{\lambda}, \boldsymbol{c}) = 0, \text{ for } (t, q, \boldsymbol{\lambda}, \boldsymbol{c}) \in \mathcal{D}_T \\ &u(T, q, \boldsymbol{\lambda}, \boldsymbol{c}) = -\phi q^2, \text{ for } (q, \boldsymbol{\lambda}, \boldsymbol{c}) \in \mathcal{D}, \end{aligned} \tag{12}$$

where $\mathcal{L}$ is a nonlinear PDE operator with a one time variable and five state variables defined in Eq.(11). Suppose that $g(t, q, \boldsymbol{\lambda}, \boldsymbol{c})$ is a solution of the PDE (11). We can set

$$u(t, q, \boldsymbol{\lambda}, \boldsymbol{c}) = g(t, q, \boldsymbol{\lambda}, \boldsymbol{c}), \quad \text{for } (t, q, \boldsymbol{\lambda}, \boldsymbol{c}) \in \mathcal{D}_T,$$

which can be approximated by a DNN function $f(\cdot, \Theta)$, where Θ is a set of the DNN parameters. As an architectures of f, we employ a fully-connected feedforward network

$$\begin{aligned} h_1 &= \tanh(W_0 \boldsymbol{x} + b_0) \\ h_{l+1} &= \tanh(W_l h_l + b_l), \qquad l = 1, \cdots, L \\ f(\boldsymbol{x}; \Theta) &= W_{L+1} h_{L+1} + b_{L+1}, \end{aligned} \tag{13}$$

with $h_l \in \mathbb{R}^n$ and L hidden layers having n hidden units in each hidden layer, where hyperbolic tangent is chosen as the activation function[2], and Θ consists of weight matrices $W_0 \in \mathbb{R}^{n \times 6}$, $W_l \in \mathbb{R}^{n \times n}, W_{L+1} \in \mathbb{R}^{1 \times n}$ and bias vectors $b_0 \in \mathbb{R}^n$, $b_l \in \mathbb{R}^n$, $b_{L+1} \in \mathbb{R}^1$.

To proceed DNN training, we set a compact domain $\tilde{\mathcal{D}} = [-N_1, N_1] \times [n_2, N_2]^2 \times [n_3, N_3]^2 \subset \mathcal{D}$ for any large positive numbers N_1, N_2, N_3 and small positive numbers n_2, n_3, and let $\tilde{\mathcal{D}}_T = [0, T) \times \tilde{\mathcal{D}}$. In the reduced domain, the following loss function is employed:

$$L(f; \Theta) = \left\| \frac{\partial f}{\partial t}(t, q, \boldsymbol{\lambda}, \boldsymbol{c}; \Theta) + \mathcal{L}f(t, q, \boldsymbol{\lambda}, \boldsymbol{c}; \Theta) \right\|^2_{\tilde{\mathcal{D}}_T, \mu_1} + \left\| f(T, q, \boldsymbol{\lambda}, \boldsymbol{c}; \Theta) + \phi q^2 \right\|^2_{\tilde{\mathcal{D}}, \mu_2}, \tag{14}$$

where μ_1 and μ_2 are probability measures of $\tilde{\mathcal{D}}_T$ and $\tilde{\mathcal{D}}$, respectively, which are absolutely continuous with respect to the Lebesgue measure.

[2]From the tests of various types of architectures including the classic/modified LSTM, monotonic nonlinear transformation, drop-out layers, skip connections models and different activation functions, we chose the best practice.

Training is conducted by the stochastic gradient descent (SGD) method based on the mean squared error $\tilde{L}(f(\boldsymbol{x});\Theta)$

$$\tilde{L}(f(\boldsymbol{x});\Theta) = \frac{1}{m}\sum_{i=1}^{m}\Bigg\{ \left(\frac{\partial f}{\partial t}(t_i, q_i, \boldsymbol{\lambda}_i, \boldsymbol{c}_i;\Theta) + \mathcal{L}f(t_i, q_i, \boldsymbol{\lambda}_i, \boldsymbol{c}_i;\Theta)\right)^2 + \left(f(T, \tilde{q}_i, \tilde{\boldsymbol{\lambda}}_i, \tilde{\boldsymbol{c}}_i;\Theta) + \phi\tilde{q}_i^2\right)^2\Bigg\}, \tag{15}$$

where $\boldsymbol{x}$ is the input variables randomly sampled with a batch size m. The SGD employs a random estimator $\tilde{L}(f(\boldsymbol{x});\Theta)$ that is unbiased of the original loss function, i.e., $L(f;\Theta) = \mathbb{E}_{\boldsymbol{x}}[\tilde{L}(f(\boldsymbol{x});\Theta)]$, instead of obtaining the true loss value on the whole input domains. The mechanism of running the SGD method is as follows: The DNN parameters Θ are updated in the opposite direction of the gradient of $\tilde{L}(f(\boldsymbol{x});\Theta)$ such that

$$\Theta \leftarrow \Theta - \ell\,\nabla_\Theta \tilde{L}(f(\boldsymbol{x});\Theta)$$

with a given learning rate ℓ and the mean squared error (15) evaluated by averaging out the values computed on m sets of input samples randomly generated from the domains $\tilde{\mathcal{D}}_T$ and $\tilde{\mathcal{D}}$ under the probability measures μ_1 and μ_2, respectively. Whenever every step is iterated, different m sets of randomly generated samples are used to compute the gradient of $\tilde{L}(f(\boldsymbol{x});\Theta)$, and as iteration goes on, ℓ can be chosen differently (commonly reducing).

5. Validation and Overfitting

The DNN training in the DGM framework has two features distinct with conventional DNN training. First, DGM does not have a fixed and finite sample size, unlike the conventional DNN training scenarios, because a distinct random sample is freshly drawn from the input space for each iteration. Instead, the sample size in DGM is determined by the product of the total number of iterations used in training and the batch size. Therefore, the overall sample size varies based on the selection of hyper-parameters, particularly batch size, learning rate, and the resultant total number of training iterations. Second, DGM does not employ the concept of an epoch, which is traditionally defined as one complete pass through the entire training dataset. Instead, DGM expands the training dataset with each iteration. This continuous expansion alters the typical training dynamics and necessitates a different approach to model validation and assessment.

In this context, this section outlines these unique characteristics and their implications for the training process and proposes the methods to address this distinction.

5.1. Indirect Validation with Simulation

Direct computation of validation of the trained model is infeasible for the DGM due to the unique nature of our dataset, as described above. Additionally, the true solution of the PDE is unknown except at the terminal trading time, which is provided as boundary and

terminal conditions. This renders it impossible to validate the trained model across nearly the entire domain. In typical scenarios with a fixed training dataset, it is standard practice to divide the dataset into proportions such as 80/20 or 70/30 for training and validation, respectively. However, the DGM dataset comprises randomly generated batches of samples produced by the same random number generator, which prevents this approach.

To address this challenge, we propose an indirect validation method. Post-training, we validate our models by conducting trading simulations and comparing the profit and loss (PnL) performance of different models, by generating samples of the terminal wealth

$$P_T = X_T + q_T(S_T - \phi q_T). \tag{16}$$

During these simulations, the data set is not randomly generated but rather "simulated" according to the modeled dynamics of each variable. This proposed indirect measure involves applying the trained DNN structure to a new dataset and simulating the distribution of the final PnL using the estimated optimal strategies δ_t^*.

Although the true values, that is the solution of the PDE, are unknown except at the terminal trading time, this method provides a practical sense to verify the model. If the trained DNN is close to the true solution, the model should be able to produce the highest terminal wealth during trading on average across various market scenarios. Thus, this simulation-based validation, as proposed, allows us to assess the model's performance effectively, ensuring that it performs well under different market conditions.

5.2. Overfitting

The unique nature of our training problem implies that overfitting, a major concern in neural network training, may not apply to DGM problems in the same manner as it does in typical scenarios. In conventional settings, one would expect the probability of overfitting to increase with the number of epochs because the model tends to learn the training data too well capturing noise and outliers, which can lead to poor generalization to unseen data. However, this may not be entirely applicable to the DGM case since each iteration explores a new subset of points within the input domain.

In the SGD optimization, utilizing a larger batch size and a smaller learning rate, coupled with a significant number of iterations, can result in a model that is closer to the true minimum. Theoretically, $\tilde{L}(f(x);\Theta)$, which serves as a sample mean estimator for the true value $L(f;\Theta)$, becomes more accurate as the batch size increases. Additionally, as the convergence of DGM shown in the literature (Grohs et al., 2023; Jiao et al., 2023; Hutzenthaler et al., 2020) and for our market-making problem (Theorem 3, Appendix A.3 in Choi et al., 2021), an infinite number of iterations should theoretically lead the model to the true minimum. A small learning rate ensures that the DNN parameters, including weights and biases, converge to their optimal values slowly but surely.

However, when considering the computational cost and its effectiveness, it is crucial to find a balance. While a large batch size and small learning rate can improve the model's accuracy, the practical constraints of computational resources and time must be taken into

account. Therefore, we need to determine a practically feasible configuration for the batch size m, learning rate ℓ, and the number of iterations N to ensure computational efficiency and effectiveness.

5.3. Case Study for Overfitting

We investigate overfitting risk that can apply to the DGM problem by focusing on the role of the learning rate and batch size through two case studies. For the test, we select the DNN model of $L = 3$ hidden layers and $n = 900$ hidden nodes, which is tested as the best performed model than other choices in Choi et al. (2021). The required market model parameters are chosen as $\beta = 100$, $\theta^- = 0.5$, $\theta^+ = 0.4$, $\kappa = 0.2$, $\eta = 20$, $\nu = 15$ (for the intensity processes $\boldsymbol{\lambda}$ in Eq.(3)), $\beta_c = 40$, $\theta_c^{\pm} = 0.3$, $\kappa_c = 0.1$, $\eta_c = 8$, $\nu_c = 5$ (for the depth processes $\boldsymbol{c}$ in Eq.(4)), $\phi = 0.1$ and $\psi = 0.01$ (penalty on the inventory in Eq.(8)). Our computing is performed on a Windows 10 PC with an AMD Ryzen Threadripper 1950X CPU, a 3.40 GHz 16-core processor, 64 GB RAM, and an NVIDIA Titan V GPU. Testing is conducted using TensorFlow in Python, and the DNN parameters are updated using the Adam optimisation algorithm, which is a momentum-based stochastic optimisation method (Kingma and Ba, 2015). The DNN parameters are initialised using the Xavier initialisation.

For PnL validation, we employ a customized version of the thinning algorithm (Ogata, 1978) to generate the required sample paths including market order intensity $\lambda^{\pm}$ and the corresponding market order process $M_t^{\pm}$, as well as the LOB depth $c_t^{\pm}$. Samples of $N_t^{\pm}$ are then filtered from $M_t^{\pm}$ using a Bernoulli variable generator based on the fill probability $h(\delta_t^{\pm}, c_t^{\pm}) = e^{-\delta_t^{\pm} c_t^{\pm}}$ with $\delta_t^{\pm}$ estimated under different models. The stock mid-price path S_t, the market-maker's cash process X_t, and the inventory process q_t are generated accordingly. Starting with an initial wealth of zero, we track the market-maker's cash process until the accumulated inter-arrival time of either N_t^+ or N_t^- reaches the terminal trading time $T = 300$ by executing market-making trades with limit orders, guided by the fill probabilities of posted limit buy and sell orders $S_t + \delta_{t-}^+$ and $S_t - \delta_{t-}^-$, respectively. This procedure is repeated K times to obtain the terminal PnL values.

Given under the computing configuration and the parameter conditions, we consider the DGM model trained with two cases of learning rate, 1×10^{-5} and 5×10^{-5} with batch size 25,000.

5.3.1. Case 1: learning rate 1×10^{-5}

We train the model of a batch size 25,000 and the initial learning rate 1×10^{-5} and progressively reduce it over training iterations[3]. We train this model for a total of 600,000 iterations while saving checkpoints at various stages throughout the training process[4]. We

[3]The Adam optimizer adjusts the optimal learning rate inside the algorithm yet, the initial value needs to be input.

[4]The learning rate is set to 1×10^{-5} for $0 \leq N \leq 200,000$; 5×10^{-6} for $200,000 \leq N \leq 300,000$; 1×10^{-6} for $300,000 \leq N \leq 400,000$; and 5×10^{-7} for $400,000 \leq N \leq 600,000$.

then simulate the market-maker's wealth process using the DGM-approximated optimal strategies derived with different number of iterations.

Figure 1 presents the training loss over various training steps, highlighting the model's convergence behavior. The training loss exhibits smooth and consistent convergence throughout the training process, continuing to decrease even after 600,000 iterations. The terminal training loss was 90.83 for the model trained over 600,000 iterations, which took 33.3 hours, and the loss was still on a decreasing trend when training was stopped. Additionally, the model appears to be progressively approximating the terminal condition with greater accuracy. This can be seen in Figure 2 which illustrates the terminal condition of g alongside its DNN-approximated solution f, as well as their absolute difference at different stages of training[5]. The absolute error for the model trained over 600,000 iterations was minimal, remaining below 2 across most of the inventory range.

Table 1 presents the number of paths with positive (profit) and negative (loss) terminal PnL out of 1,000 simulations for models at various stages of training, along with the corresponding final training loss. Additionally, it includes the mean and standard deviation of the training loss over the last 1,000 iterations of each specific training stage.

Despite most models appearing well-trained in terms of training loss and terminal condition approximation, they still yield some negative terminal losses in some of the 1000 PnL simulations. Notably, these negative terminal losses are significantly larger in magnitude than the profits. Even in the case of a well-trained DNN model with generally low error levels, both overall and in the terminal condition – for instance, at 480,000 iterations, achieving a final loss of 131.34 – we observe 13 extreme negative losses (biggest of which are -598, -509, and -352). These outcomes are difficult to regard as instances of optimal wealth. This issue can be better visualized in Figure A.9, where terminal PnL is plotted over simulations for models at different stages of training. Moreover, as the number of training steps increases, the number of negative PnL cases gradually diminishes, indicating an improvement in model performance. Given sufficient iterations, the model is capable of converging to the global minimum, even with a large batch size, a small learning rate, and an extensive number of iterations.

5.3.2. Case 2: learning rate 5×10^{-5}

We train the model with learning rate 5×10^{-5} without reseting the magnitude during training, under that the other conditions are the same with Case 1. Training was stopped at 200,000 iterations, since the training loss no longer decreased, while saving checkpoint at the stage of 120,000th iteration. The terminal loss is 185.04 and the corresponding computing time is 4.44 hours.

Surprisingly, we observe a significant improvement in the model's PnL performance even though computing cost is largely reduced. Table 2 displays the results of this case, presented

[5]100,000 terminal f values (for each q) are computed from 100,000 random samples of $\lambda^{\pm}$ and $c^{\pm}$, and the average of the computed terminal f values is taken.

Table 1: Results of the model with learning rate 1×10^{-5} progressively decreasing: The final training loss, and the mean (Mean) and standard deviation (Std) of the loss over the last 1000 iterations for the employed DNN model at various training steps, and the corresponding PnL validation resulting in the number of positive and negative terminal PnL for 1000 samples.

Iterations	Training					PnL Validation	
	Batch Size	Learning Rate	Training Loss	Mean	Std	# Profits	# Losses
120,000	25,000	1×10^{-5}	275.31	307.93	23.44	976	23
160,000	25,000	1×10^{-5}	292.97	265.12	17.95	975	24
200,000	25,000	1×10^{-5}	243.54	238.32	14.11	981	18
300,000	25,000	5×10^{-6}	213.85	194.63	10.00	981	18
400,000	25,000	1×10^{-6}	157.32	166.94	9.9	979	20
480,000	25,000	5×10^{-7}	131.34	141.88	13.4	986	13
520,000	25,000	5×10^{-7}	174.52	128.88	14.84	989	10
560,000	25,000	5×10^{-7}	103.85	121.05	17.24	998	1
600,000	25,000	5×10^{-7}	90.83	113.00	24.45	999	0

in a similar manner to Table 1. We can see that even after 100,000 iterations, no negative PnL cases were observed, highlighting the robustness and efficiency of the model under the revised learning rate.

Table 2: Results of the model with learning rate 5×10^{-5}: The final training loss, and the mean (Mean) and standard deviation (Std) of the loss over the last 1000 iterations for the employed DNN model at various training steps, and the corresponding PnL validation resulting in the number of positive and negative terminal PnL for 1000 samples.

Iterations	Training					PnL Validation	
	Batch Size	Learning Rate	Training Loss	Mean	Std	# Profits	# Losses
100,000	25,000	5×10^{-5}	246.49	213.65	24.96	1000	0
150,000	25,000	5×10^{-5}	236.85	232.00	84.93	1000	0

5.4. Overfitting in DGM

The observations in Section 5.3 carry significant implications for training in DGM. It is characterized that some models trained in Case 1 exhibiting *overfitting* to the new dataset, which it had not encountered during training. Despite most models in Case 1 demonstrating favorable training loss and terminal condition error metrics over ones in Case 2, it produced highly anomalous results that would lead a significant operational risk (poorly executed algorithms) in real trading places. Overfitting may stem from employing excessively large input data and a unnecessarily small learning rate.

Increasing the batch size tends to decrease the variability in computed gradients by averaging out losses. While this can reduce noise in the optimization process, it also limits

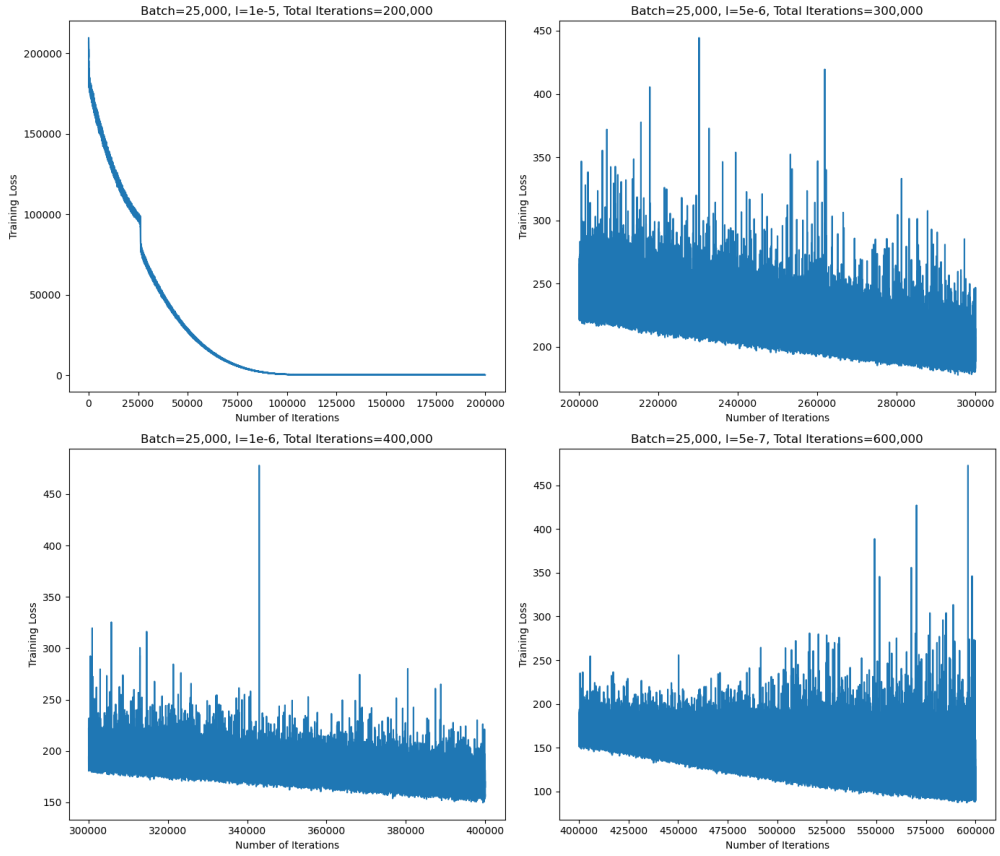

Figure 1: Training loss over iterations for different configurations. Each subplot shows the training loss as a function of the number of iterations for a fixed batch size of 25,000 with varying learning rates: 1×10^{-5} (top left), 5×10^{-6} (top right), 1×10^{-6} (bottom left), and 5×10^{-7} (bottom right).

exploration of the parameter space, potentially trapping the loss in sharp minima[6], thus increasing the risk of overfitting. Additionally, a small learning rate exacerbates this situation, as larger samples smooth out noise in gradient, making it more probable for optimization to remain in sharp minima regions of the loss landscape.

Even if the training loss continues to decrease, but if it is exploring in sharp mimima area, *generalization performance* of the model may be degraded on unseen data as it becomes increasingly specialized to the training set. To escape a sharp minima region, a large batch and small step size model needs more iterations, meaning that largely expensive computational costs are required to obtain a reliable model, where the cost was at least 600,000 iterations in Case 1.

The issue of model generalization is a crucial aspect in training DNNs, first introduced by Hochreiter and Schmidhuber (1997). They proposed the concept of a flat minimum in the loss

[6]Sharp minima are regions where the loss function has steep gradients. Models trained at sharp minima may have lower training error, but they are more susceptible to overfitting because small changes in the parameters can lead to large changes in the loss.

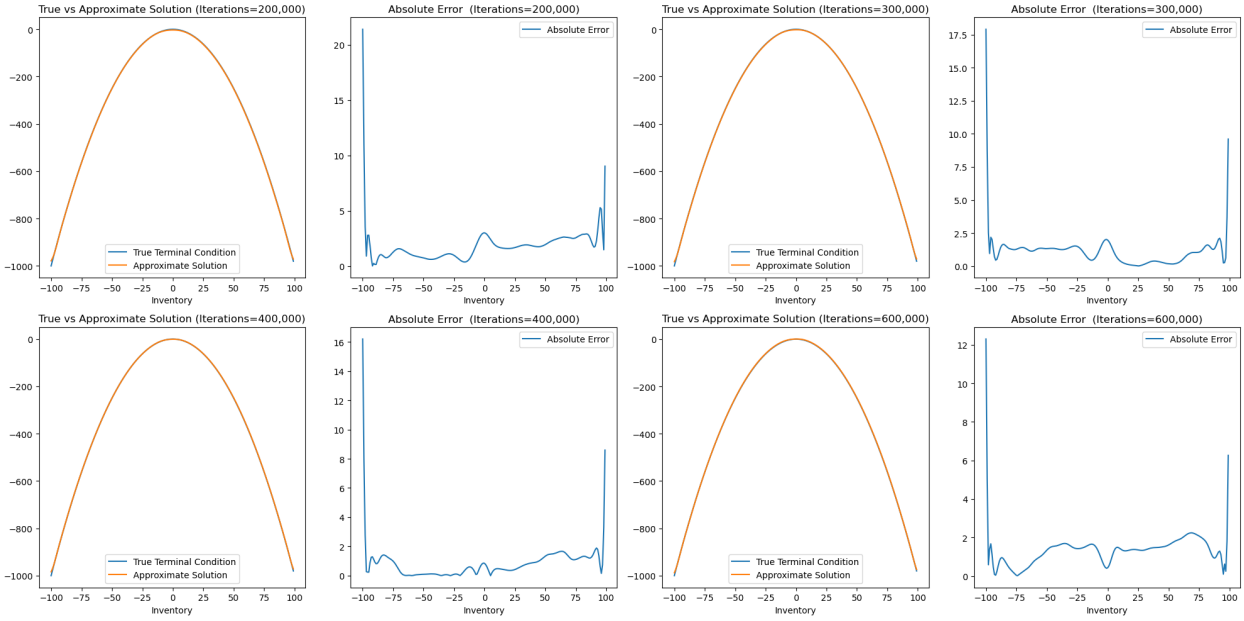

Figure 2: Comparison of the true terminal condition g and its approximate solution f at various training iterations across the inventory level with the number of iterations 200,000 (top left), 300,000 (top right), 400,000 (bottom left), and 600,000 (bottom right). Each subplot has the true terminal condition and the approximate solution in the left column, and their absolute error (blue) in the right column.

function and developed an algorithm to discover neural networks with low complexity and high generalization capacity. Numerous recent studies focus on enhancing a neural network's generalization performance by aiming for a flat minimum in optimizers (Cha et al., 2021; Kaddour et al., 2022; Zhang et al., 2023). Conversely, small batch sizes introduce more noise into the optimization process, which can help the model escape sharp minima and find flatter regions of the loss landscape[7], which has lower sensitivity of the loss function to a little change in weights. Small batch sizes enable the optimization algorithm to explore a wider range of parameter configurations, potentially leading to better generalization performance.

The overfitting in a large batch and small learning rate model may not be a concern in DGM context because it has the potential to overcome sharp minima by increasing the number of iterations. However, this strategy needs a larger training set for the convergence, leading to diminishing benefits as batch size increases[8]. Taking such a long route is computationally intensive; thus, it may be practically infeasible. Moreover, early stopping training

[7]Flat minima are regions in the optimization landscape where the loss function is relatively flat. Models trained to a flat minimum often generalize better because they are less sensitive to small perturbations in the input data. Flat minima typically correspond to regions where the model's parameters have similar performance, indicating robustness to variations in the training data

[8]This observation is particularly interesting because, in typical machine learning settings, the likelihood of overfitting tends to increase with additional training steps, often resulting in poor generalization and suboptimal validation results.

may lead to overfitting, as evidenced by validation through PnL simulation. Thus, careful calibration of batch size and learning rate is essential to balance computational efficiency and model performance. This issue is discussed in detail in the next section.

6. Optimal Model Selection

This section focuses on identifying the optimal DGM model by analyzing training loss, computational efficiency, and validation performance. Through a series of experiments, we assess the effects of different batch sizes and training step configurations on model performance. By conducting PnL simulations, we validate our findings and highlight the implications of batch size and training duration on overfitting and model robustness. The insights derived from these experiments offer valuable guidance for selecting hyper-parameters that balance computational cost and predictive accuracy in high-dimensional optimization problems.

In general neural network training, Shallue et al. (2019) derive empirically the relationship between batch size and the number of iterations required to reach a goal of out-of-sample error, finding that increasing the batch size decreases the number of required training steps proportionally. However, there are no further benefit for large increasing the batch size does not reduce the number of training steps. Smith (2018); McCandlish et al. (2018) empirically illustrate the trade-off between time and compute resources spent to train a model to a given level of performance. In the tests, our focus lies in investigating a particular relationship between MESS and batch size. This correlation can show the degree of a trade-off between improved gradient accuracy and the increased likelihood of encountering sharp minima with respect to the stochasticity in loss.

6.1. Measurement of Compute Cost for DGM

We measure a minumum computing cost required for a model to achieve a reasonablly small loss but to be robust for PnL performance. For a given batch size, we define minimum effective iterations (MEI) as the number of iterations required to reduce the training loss below a threshold level and maintain it under this threshold for 1000 consecutive iterations for a given batch size. Furthermore, we define minimum effective sample size (MESS) as the product of MEI and the given batch size,

$$\text{MESS} = \text{Batch Size} \times \text{MEI}. \tag{17}$$

This represents the minimum size of computing costs that is required that the estimated solution of the PDE has reliable and general performance (not to overfitting with good generalization).

We conduct the test of estimating MEI and MESS with various size of batch and the number of iterations setting a loss threshold level of 500. The batch size is chosen as 200, 400, 600, 800, 1000, 3000, 5000, 10000, 15000, 20000, and 25000. For learning rate, 5×10^{-5} is chosen for all batch sizes except for the batch size of 400 which uses 1×10^{-5}. The training

cases are examined with several performance metrics including terminal loss, the mean and the standard deviation of the loss over the last 1,000 iterations, the absolute error at terminal time, and the PnL performance.

After estimating MEI for each batch case with the base learning rate, we continue training for each model beyond the MEI to ensure that the identified MEI results in robust PnL performance. This is to assess whether models reaching the MES perform reasonably well and if further training beyond the MES can lead to additional enhancements in the aforementioned criteria. We also adjust the learning rate to investigate the potential benefits of refining the model with smaller learning rates after reaching MEI. Each training configurations are defined as follows:

- MEI: Base estimate for each batch size.
- Continued = MEI + 50,000: Given that MEI varies for each model depending on the batch size, each case is trained for additional 50 thousand iterations beyond the respective MEI.
- Extended = Total 200,000 iterations or more: Batch sizes 400 and 600 are trained for 500 thousand iterations; batch sizes 800 and 1000 are for 300 thousand; and the other cases are for 200 thousand iterations in total.
- Refined = MEI + 50,000 with learning rate reducing: Additional 50,000 iterations from MEI are conducted with progressively decreasing learning rates. The learning rate is reduced five times at every 10,000 iterations post-MEI.

Each model is saved at the point of reaching the MEI, continued, extended, and refined cases. This comprehensive analysis allows us to ascertain whether extending training beyond MEI at the same learning rate or employing a strategy of progressively decreasing learning rates contributes to improved model performance and robustness.

6.2. Test Results: Training Loss

Figure 3 depicts the training loss over MEI, continued, and extended numbers of iterations, while Figure 4 illustrates the training loss under the refined case. Table 3 summarizes the statistics of the training loss, mainly the mean and standard deviation of the last 1000 losses under MEI, continued, extended, and refined number of iterations. One clear observation is that any of the continued, extended, or refined cases do not add significant improvements over MEI in terms of the mean and variance of training loss.

It is worth noting a feature regarding sudden spikes in the loss value within these curves, especially before reaching MEI. These spikes result from the Adam optimizer, which adjusts the effective learning rate based on the first and second moments of the gradients. During the initial training phase, gradients are typically very large, especially with large batch sizes, making it easier to reduce the loss. Consequently, Adam makes substantial updates, leading to sudden spikes when encountering outlier samples (Kingma and Ba, 2015).

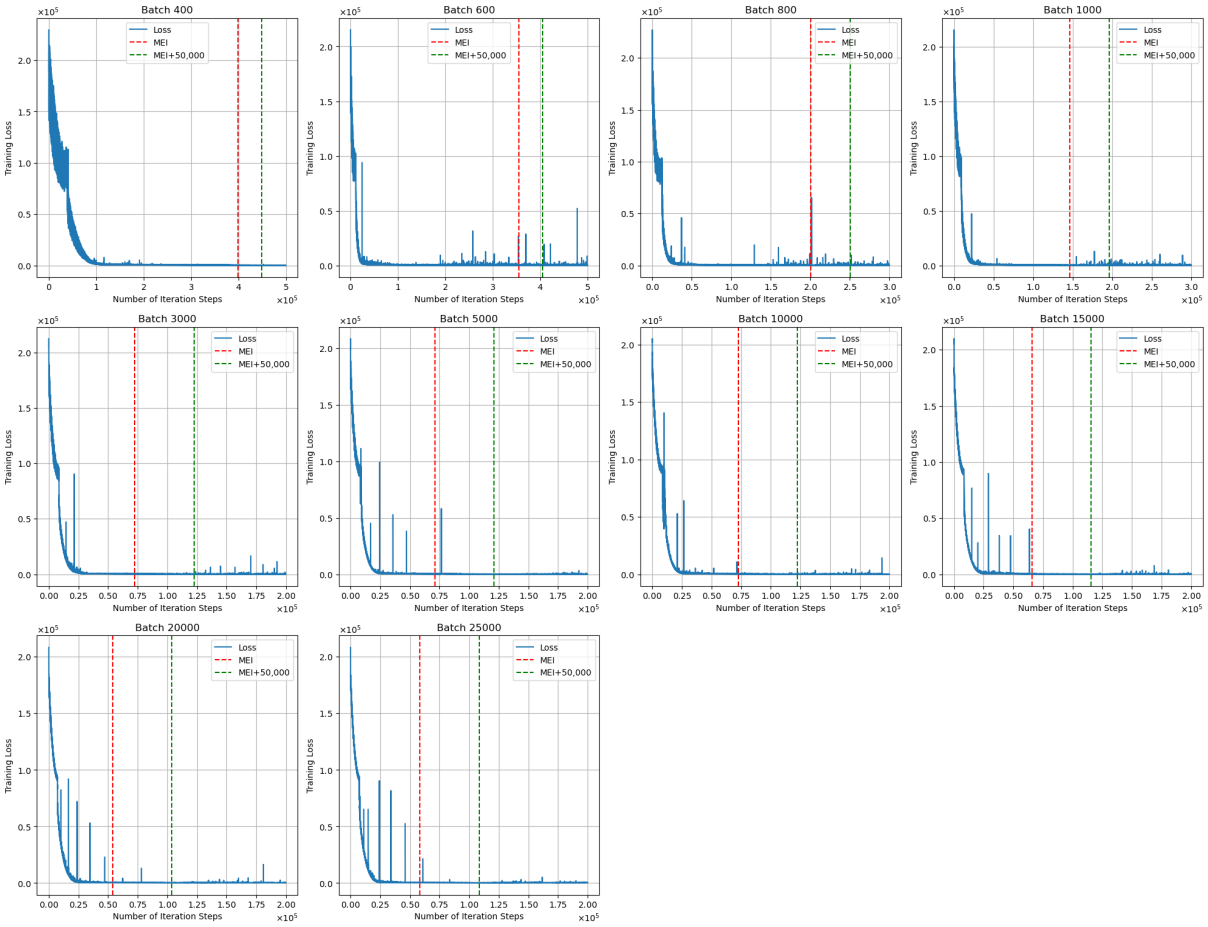

Figure 3: Training loss over MEI and continued upto extended iterations. The vertical red line denotes the MEI, and the blue line marks the continued iterations.

Additionally, the frequency of these spikes increases as batch sizes grow larger. Bigger batches result in more stable gradient estimates (i.e., smaller variance), prompting Adam to take larger, more confident steps. Conversely, smaller batches result in higher gradient variance relative to the mean, leading Adam to take smaller steps and causing fewer spikes. As training progresses, spikes become less frequent. This occurs because, in the regions that gradients are small of the loss landscape, the gradients' magnitude (both mean and variance) is smaller, causing Adam to take very small, careful steps. Figures A.10, A.11, A.12, and A.13 in Appendix A provide plots of the last 1000 training losses under MEI, continued, extended, and refined configurations, respectively, for further reference.

6.3. Test Results: MEI, MESS, and Computing Cost

Table 4 presents the MEI, MESS, and corresponding computing time required to train the models up to MEI. We note that batch size 200 results in that an appropriate number of training for the loss thereshold 500 is not attained even for 600 thousand iterations and

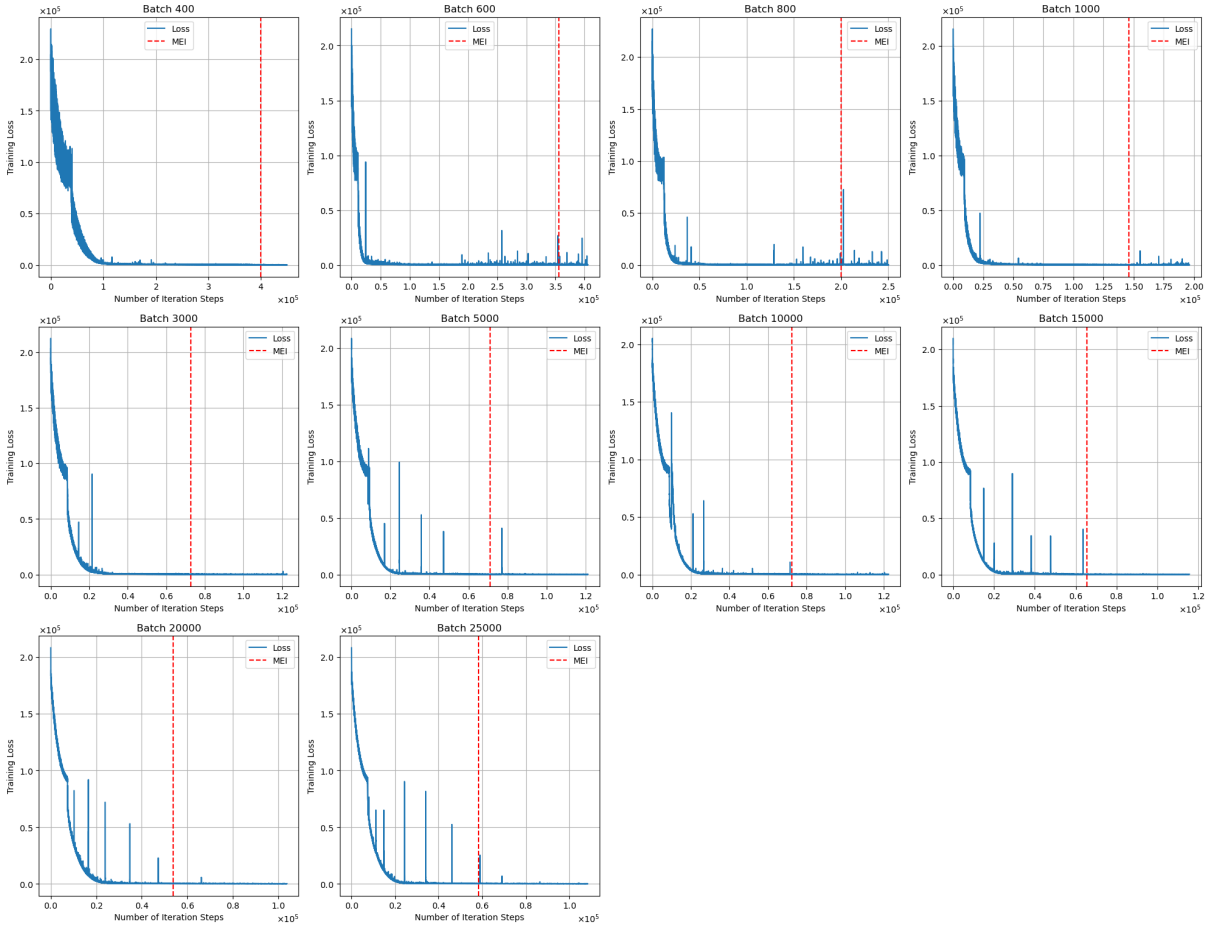

Figure 4: Training loss over 50,000 more iterations after MEI (Refined case). The vertical red line denotes MEI for each configuration. Training loss after MEI corresponds to refined loss at progressively smaller learning rates.

proper learning rate adjustment. The results from batch size 400 are visually illustrated in Figures 5 and 6.

We observe that as batch size increases, MEI tends to decrease, indicating a trade-off between MEI and batch size in DGM training. Regression analysis further reveals this negative relationship, estimated to follow a power function of $kx^{-0.41}$ with a coefficient of determination $R^2 = 0.83$. This trend occurs because larger batch sizes yield more accurate gradient estimates, facilitating faster convergence in terms of training steps. In other words, fewer steps are needed to reach the loss threshold level. It is known that the pattern of diminishing MEI with increasing batch size is a common observation in machine learning settings (Shallue et al., 2019). However, after a batch size of 3000, the rate of decline in MEI notably levels is reduced, indicating that computational costs do not diminish from further increment in batch size.

On the other hand, MESS and computing time display a rising trend, except for batch sizes less than 1000, where there is a slight decline, resulting in a J-shaped curve. The

Table 3: Summary statistics of the last 1000 training losses for different batch sizes with four iteration conditions – the MEI, continued, extended, and refined cases.

Batch	**MEI**			**Continued**			**Extended**			**Refined**		
	Mean	Std	Iterations	Mean	Std	Iterations	Mean	Std	Iterations	Mean	Std	Iterations
400	358.7	36.7	399,356	321.6	31.9	449,356	306.1	33.2	500,000	321.6	31.3	449,356
600	310.4	37.00	355,270	341.0	73.0	405,270	321.4	50.0	500,000	304.8	39.8	405,270
800	298.5	37.9	200,400	334.7	120.7	250,400	346.2	162.6	300,000	309.1	43.0	250,400
1,000	330.9	43.7	146,206	322.5	56.2	196,206	289.4	63.8	300,000	318.2	123.3	196,206
3,000	323.2	37.8	72,589	285.3	48.2	122,589	273.6	119.7	200,000	281.3	46.6	122,589
5,000	304.4	35.5	71,081	252.6	23.5	121,081	236.3	31.2	200,000	253.5	28.3	121,081
10,000	291.6	33.9	72,434	244.0	34.8	122,434	232.0	18.0	200,000	244.9	31.4	122,434
15,000	283.9	30.7	65,590	224.9	22.1	115,590	229.9	40.4	200,000	223.8	22.4	115,590
20,000	274.1	35.2	53,746	219.1	23.8	103,746	226.1	60.7	200,000	218.3	25.6	103,746
25,000	260.1	33.4	58,431	211.8	26.0	108,431	226.4	53.1	200,000	212.9	27.3	108,431

positive relationship between MESS and batch size is described by a power function of $kx^{0.59}$ with $R^2 = 0.91$, implying that increasing batch size necessitates more computing resources, particularly for batch sizes that are not too small.

Although the reduction in MEI as batch sizes grow, both computing time and MESS generally show an upward trend. Initially, the computing time and MESS decline with batch sizes up to 3000, likely due to the improved higher accuracy of estimated gradients. However, beyond a batch size of 3000, MESS and the computing time start to rise. This suggests that a batch size of 3000 may be optimal for DGM models trained up to MEI in terms of computational efficiency. The next section will discuss PnL validation, where this observation becomes even clearer.

Batch Size	MEI	MESS (thousands)	Computing Time (min)
200	Unattainable	-	-
400	399,356	159,742.4	83.199
600	355,270	213,162.0	82.896
800	200,400	160,320.0	48.430
1000	146,206	146,206.0	36.552
3000	72,589	217,767.0	36.295
5000	71,081	355,405.0	52.718
10000	72,434	724,340.0	96.579
15000	65,590	983,850.0	128.447
20000	53,746	1,074,920.0	138.844
25000	58,431	1,460,775.0	194.770

Table 4: MEI, MESS (in thousands), and the corresponding computing time (in minutes) for models trained with different batch sizes.

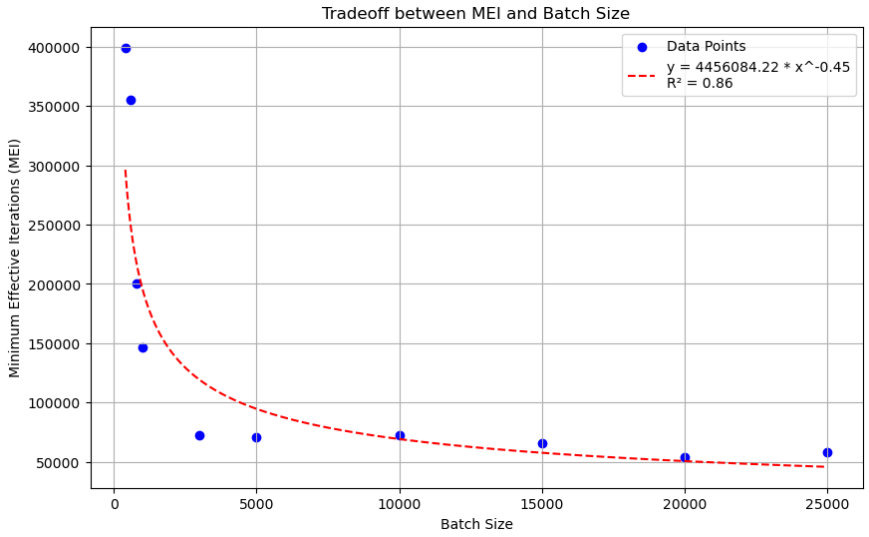

Figure 5: Trade-off relationship between MEI and batch size with regression by a power function $kx^{-0.41}$.

6.4. Test Results: Validation by PnL Simulations

To validate the trained DGM model's performance, we simulate 1,000 paths of the market-makers' wealth process and obtain 1,000 terminal PnL values. Table 5 presents the summary statistics of terminal PnL for models trained with different batch sizes under the MEI, continued, extended, and refined configurations, which is further illustrated in Figure 7. The error at the terminal condition are displayed in Figures A.14, A.15, A.16, and A.16 for the MEI, continued, extended, and refined cases, respectively.

Table 5: Summary statistics of PnL validation for different batch sizes with four iteration conditions – the MEI, continued, extended, and refined cases.

Batch	**MEI**				**Continued**				**Extended**				**Refined**			
	Mean	Std	Min	Max	Mean	Std	Min	Max	Mean	Std	Min	Max	Mean	Std	Min	Max
400	283.7	129.7	-1063.8	439.2	296.2	124.1	-851.2	433.0	304.6	107.5	-338.2	442.7	300.9	111.7	-660.6	452.5
600	306.7	31.0	199.9	409.7	296.7	31.1	215.7	395.3	310.9	31.87	209.0	420.1	291.7	31.7	196.9	425.9
800	341.4	30.2	232.0	440.0	333.2	31.2	231.7	455.3	326.9	30.90	229.3	443.6	333.7	30.9	235.6	435.9
1,000	341.9	31.3	238.2	438.1	330.4	30.6	231.0	437.7	310.0	32.31	216.7	418.6	332.7	31.1	247.7	439.2
3,000	345.7	30.9	259.1	445.2	342.0	29.7	238.2	444.0	301.4	31.66	198.7	397.9	340.2	30.3	237.6	448.8
5,000	333.6	37.0	-289.3	427.5	303.1	30.4	194.8	394.2	222.5	31.27	136.3	331.8	310.1	30.9	219.7	421.3
10,000	327.9	30.7	230.1	442.8	295.8	31.5	201.8	380.4	245.8	32.35	124.5	362.9	296.3	31.4	210.4	410.6
15,000	328.8	32.1	233.7	429.9	268.3	30.9	179.4	377.5	205.7	31.86	112.9	297.3	269.3	31.3	169.3	399.3
20,000	336.3	29.9	250.7	419.8	319.2	30.9	229.3	435.2	214.8	31.93	114.6	303.1	318.8	30.5	214.9	424.4
25,000	324.7	31.5	228.1	417.3	257.6	32.0	161.8	355.6	167.9	31.68	72.9	262.7	261.9	31.9	183.4	383.6

These results highlight two distinct features: First, the models trained up to the MEI configuration outperform those trained under other configurations across almost all batch sizes in terms of mean of terminal PnLs and the coefficient of variation (CV), which is the standard deviation scaled by the mean. This indicates that training the model beyond the MEI does not provide additional improvement. Instead, it appears to introduce overfitting,

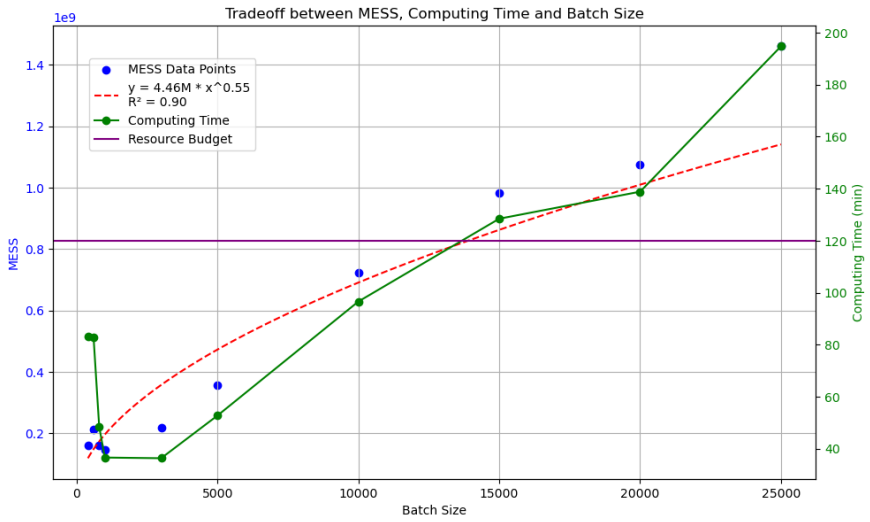

Figure 6: Relationship between MESS and batch size with the corresponding computing time. The left x-axis shows the MESS values fitted with a power function $kx^{0.59}$, and the right x-axis depicts the computing time for different batch sizes. The horizontal line indicates an example of resource budget for computing time at 120 minutes.

resulting in inferior performance in terminal PnL compared to the MEI case. Moreover, the models trained under configurations other than MEI demonstrate a downward trend in mean terminal PnL as batch size increases. This also implies that potential overfitting becomes more pronounced with larger batch sizes, particularly when the number of iterations is higher.

Second, the models trained under the MEI configuration exhibit consistent mean terminal PnL for batch sizes greater than 800. This consistency allows for flexibility in choosing a batch size that is computationally efficient. Notably, the batch size of 3000 yields the highest terminal PnL and the shortest computing time. The batch size of 3000 with MEI can be a computationally the optimal model in DGM.

These findings have substantial implications for practical applications in the DGM approach. Contrary to the common assumption that random sampling mitigates overfitting, our results demonstrate that overfitting can still occur if models are over trained, with the issue becoming more pronounced as batch sizes increase. This highlights that it is critical to select appropriate batch sizes to optimize computational resources and stable model outcomes. These insights are critical particularly for whom has a limited budget for computing resourse. For example, if a time budget is 2 hours per training, a model with a batch size less than 10,000 can be chosen.

7. Sensitivity Analysis

In Section 3, we have discussed we explored how the value function of the market-maker's problem and its optimal strategies are influenced not only by a multitude of underlying processes, which continuously self/mutual-interacting in short and lont term levels, but also

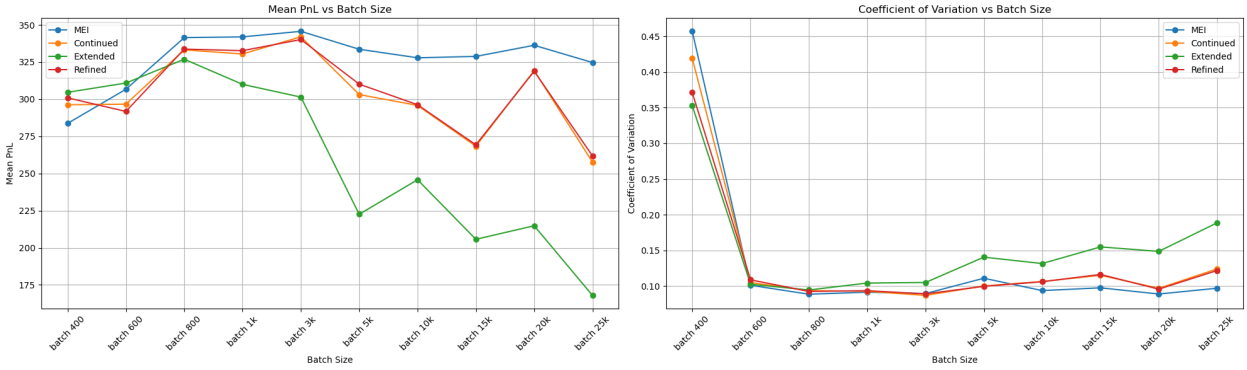

Figure 7: Mean PnL (left) and its CV (right) across different batch size models trained under four conditions of MEI, continued, extended, and refined cases.

by a set of model parameters necessary for constructing a LOB in a HF dynamic setting. For given penalty terms ϕ and ψ, the optimal strategy to a limit buy order is dependent on a time variable and five state variables of $q_t, \lambda_t^{\pm}, c_t^{\pm}$, meaning that

$$(\delta_t^+)^* = F_0(t, q_t, \lambda_t^+, \lambda_t^-, c_t^+, c_t^-), \tag{18}$$

where F_0 represents a positive function determined by solving the HJB (11) through optimal DGM training. The final wealth attained by the optimal strategies δ_t^* of the market-maker at time T depends on the mid-price process S_t, along with an initial wealth x, in addition to the LOB processes and model parameters mentioned earlier. This same scenario applies to the corresponding sell strategy $(\delta_t^-)^*$.

In this section, we conduct sensitivity tests on the optimal market-making strategies and the corresponding maximal expected wealth achievable at the end of trading, denoted as $\mathbb{E}[P_T]$, obtained through δ_t^*, where the final wealth at time T is defined in Eq.(16). We employ DGM-approximated functions to assess the inherent relationships concerning the variables of interest. This analytical investigation has not been previously explored in the literature due to the high complexity of the HF market structure, as shown in Eq.(18). The primary variables of interest in the sensitivity tests are the stability and manipulation of the HF market.

For the tests, we select a batch size 3,000, learning rate 1×10^{-5}, and MEI iterations for DGM approximation under the LOB parameters of $\theta^{\pm} = 0.5$, $\xi = 40$, $\alpha^{\pm} = 0.3$, $\kappa_c = 0.1$, $\eta_c = 8$, $\nu_c = 5$, and the market-maker's parameters setting to trading time $T = 300$ seconds, liquidation cost $\phi = 0.1$ and inventory penalty $\psi = 0.0$, implying the case that a market-maker is not penalized for variations in inventory during trading.

7.1. Market Stability

The Hawkes model requires stability conditionsto ensure that order arrival processes do not diverge over time, attributed to various influencing factors. For the market order arrivals governed by the intensity processes $\lambda_t^{\pm}$, the bounded condition is $(1 - \kappa)\beta/(\mu + \nu) > 0$, as

shown in Remark 1. We define an indicator of HF market stability as

$$I_{\text{st}} = \frac{(1-\kappa)\beta}{\mu+\nu}.$$

A larger value of I_{st} denotes heightened stability in the HF market, while a smaller value indicates the opposite.

The market stability I_{st} is influenced by four components, which are the self-exciting parameter μ and the mutual-exciting parameter ν governing the degree of transitory noises; the synchroning parameter κ reflecting long-term correlation between buy and sell order arrivals; and the mean-reverting speed β indicating the extent to diminish for temporary noise orders in the order arrival process. An increase in either μ or ν tends to reduce market stability, as does a higher value of κ. Consequently, I_{st} shows negative associations with μ, ν, or κ. However, a higher β implies greater stability in the HF market, leading to a positive correlation between I_{st} and β. Unlike market order processes, we can assume that limit order processes $c_t^{\pm}$ remain stable as long as μ_c and ν_c remain finite. This is because limit order dynamics only experience jumps when market orders are stimulated, remaining continuous otherwise.

We hypothesize that a HF market-maker earns higher expected profit as the market stability decreases with respect to μ, ν, κ, and β, and test it through linear regression. We choose

$$\eta = 20, \mu = 15, \beta = 100, \kappa = 0.2 \tag{19}$$

as the base parameter case yielding $I_{\text{st}} = 2.3$, and select five $I_{\text{st}} = 2.8, 2.5, 2.3, 2.0, 1.5$ levels for high, moderate high, base, moderate low, and low HF market stability cases, respectively, as displayed in Table 6. Then, we generate the data set for the market-maker's expected PnL by simulating 1000 optimal wealth values after estimating the optimal strategies training MEI iterations for 20 cases specified in Table 6. Table 7 includes descriptive sample statistics of the generated data for the final PnL with estimated MEI for each case, and Figure A.18 illustrates the mean of the terminal PnLs with respect to the five stability levels attributed to the driving factors κ, η, ν, β.

We test that the null hypothesis H_0 of no relationship with respect to individual factors η, ν, κ, β through linear regression, and the results are in Table 8. The test reveals that κ and η are significant at the 1% and 5% levels, respectively, whereas ν and β are not. It means when buy and sell orders are more synchronized or when market orders are more self-excited, higher market instability corresponds to higher profits, and lower market instability leads to lower profits. In the components in HF market stability, synchrony reflecting the long-term tendency in buy and sell orders emerges as the primary driving factor, and the self-excited transient noise orders serve as a secondary factor, in influencing to a maker-maker's expected profit. When market stability rises due to factors like mutual excitement and mean-reversion speed, In the cases when market stability rises due to the other factors such as mutual excitement and mean-reversion speed, no significant relationship is observed.

Table 6: Parameter selection for the sensitivity test to change in the synchronizing factor κ, the self-exciting η factor, the mutually-exciting factor ν, and the mean-reversion speed β for the five stability level cases I_{st}.

	Synchronizing κ				Self-exciting η				Mutual-exciting ν				Mean-reversion β			
I_{st}	η	ν	β	κ	η	ν	β	κ	η	ν	β	κ	η	ν	β	κ
2.8	20	15	100	0.02	14	15	100	0.2	20	9	100	0.2	20	15	123	0.2
2.5	20	15	100	0.13	17	15	100	0.2	20	12	100	0.2	20	15	109	0.2
2.3	20	15	100	0.20	20	15	100	0.2	20	15	100	0.2	20	15	100	0.2
2.0	20	15	100	0.30	25	15	100	0.2	20	20	100	0.2	20	15	88	0.2
1.5	20	15	100	0.48	38	15	100	0.2	20	33	100	0.2	20	15	66	0.2

Table 7: Summary statistics for simulated terminal PnL samples for different stability levels I_{st} attributed to various factors (κ, η, ν, β) under the optimal DGM model trained with MEI iterations. The statistics include sample mean, sample standard deviation (Std), minimum (Min), maximum (Max), and coefficient of variation (CV).

Factor	Value	MEI	Stability I_{st}	Mean	Std	Min	Max	CV
	0.02	50,912	2.80	286.7	30.0	206.8	397.2	0.1046
	0.13	55,682	2.50	330.0	31.1	240.6	431.3	0.0943
κ	0.20	41,280	2.29	346.2	32.2	226.3	453.3	0.0930
	0.30	62,259	2.00	376.8	33.0	273.5	477.2	0.0874
	0.48	113,706	1.00	505.9	38.4	378.0	636.3	0.0759
	14	41,887	2.80	344.9	33.2	251.0	448.9	0.0964
	17	41,863	2.50	343.8	32.4	246.8	437.3	0.0944
η	20	41,280	2.29	346.2	32.2	226.3	453.3	0.0930
	25	50,136	2.00	331.3	31.8	236.5	424.0	0.0959
	38	85,726	1.00	356.0	31.0	253.7	454.4	0.0872
	9	39,556	2.80	342.8	31.9	246.8	445.1	0.0930
	12	40,143	2.50	343.1	32.4	238.0	465.7	0.0945
ν	15	41,280	2.29	346.2	32.2	226.3	453.3	0.0930
	20	38,604	2.00	342.8	31.9	263.0	452.6	0.0931
	33	115,217	1.00	364.8	31.9	270.3	466.7	0.0874
	123	46,387	2.80	341.9	31.8	230.1	465.0	0.0931
	109	39,346	2.50	348.3	32.5	245.3	477.1	0.0932
β	100	41,280	2.29	346.2	32.2	226.3	453.3	0.0930
	88	56,538	2.00	335.5	31.8	242.7	442.4	0.0949
	66	159,038	1.00	368.7	32.6	238.3	468.2	0.0883

Next, we test multiple null hypothesis that the mean PnL under lower stability is less than or equal to the mean PnL under higher stability for the following four pairs: high $\geq$ moderate high, moderate high $\geq$ base, base $\geq$ moderate low, and moderate low $\geq$ low. To ensure that the results are is statistically significant in multimple comparison tests, we use

Table 8: Linear regression results of the market-maker's expected PnL with respect to each parameter κ, η, ν, β determining the HF market stability I_{st}. t-statistics are in parentheses, and standard errors are in square brackets. *** $p < 0.01$, ** $p < 0.05$, * $p < 0.1$.

	κ	η	ν	β
Intercept	244.97***	315.39***	337.31***	387.5442***
	(18.654)	(25.308)	(25.275)	(11.499)
	[13.132]	[12.462]	[13.346]	[33.703]
κ	5.1284***			
	(10.733)			
	[0.478]			
η		1.2054**		
		(2.351)		
		[0.513]		
ν			0.4559	
			(0.673)	
			[0.678]	
β				-0.5238
				(-1.540)
				[0.340]
R-squared	0.975	0.648	0.131	0.442

a 1% significance level by applying the Bonferroni correction (Dunn, 1961). This correction divides the desired significance level by the number of tests, leading to an adjusted threshold, that is, 0.01/4. Thus, the null hypothesis is rejected if the p-value is smaller than this adjusted threshold. This is to adjust the significance level of multiple comparisons, typically to mitigate the issue of inflated Type I error rates.

The linear regression examined overall relationship between a market-maker's PnL and the market stability environment, while the multiple comparison test can give the insight into how the market-maker's profitability changes when the market stability regime shift from one to the other level. Table 9 presents four t-tests comparing the mean PnL across different stability levels attributed to the four factors κ, η, ν, and β. P-values are provided in parentheses with the rejection rule based on a Bonferroni corrected alpha of 0.0025.

From these results, we observe the following: For κ, the null hypothesis is rejected in all four tests. This indicates that as market orders become more synchronized (i.e., as stability decreases), the market-maker's expected optimal wealth increases. As this trend is evident in Figure A.18, the plot for κ clearly demonstrates a positive linear relationship between stability and mean PnL. Notably, the mean terminal PnL under low stability is nearly twice as high as that under high stability, highlighting a significant and surprising finding.

Another significant finding is the rejection of the null hypothesis in the case (4) that the mean PnL under the moderate low stability is greater than or equal to one under the low

case for each factor. This means that when market stability shifts from moderate low to low, the market-maker's expected profit increases significantly, regardless of type of attributing factors κ, ν, η, or β. In contrast, we fail to reject the null hypothesis in all other tests except for high $\geq$ moderate high for β. This suggests that the expected optimal wealth of the market-maker is not significantly affected by changes in stability levels for η and ν under the conditions tested.

These findings carry implications of the relationship between the market-maker's optimal value function and HF stability environment. The market-maker's optimal value function demonstrates a linear relationship with κ, while it appears to have nonlinear (i.e. combination of monotonic and linear) structures with respect to η, ν, and β. As a whole, the market-maker's expected optimal wealth increases when market stability is lowered. This tendency appears consistently for change in κ across all market stability levels, while it meaningfully emerges for the changes in η, ν, and β only when the degree of market instability changes from a moderate low level to a low level.

Table 9: Multiple t-tests comparing the mean PnL across different stability levels for the four factors: κ, η, ν, and β. The hypothesis tests determine if the mean PnL under lower stability is higher than the mean PnL under higher stability. P-values are provided in parentheses with the rejection rule based on a Bonferroni corrected significance level of 0.0025.

H_0	κ	η	ν	β
(1) High $\geq$ Moderate high	Reject	Not Reject	Not Reject	Reject
	(0.000)	(0.770)	(0.395)	(0.000)
(2) Moderate high $\geq$ Base	Reject	Not Reject	Not Reject	Not Reject
	(0.000)	(0.042)	(0.014)	(0.912)
(3) Base $\geq$ Moderate low	Reject	Not Reject	Not Reject	Not Reject
	(0.000)	(1.000)	(0.991)	(1.000)
(4) Moderate low $\geq$ Low	Reject	Reject	Reject	Reject
	(0.000)	(0.000)	(0.000)	(0.000)

7.2. Market Manipulation

We are concerned about the relationship between the optimal market-making strategy and the extent to market manipulation by other HF traders. As one of market disrupting behaviors, we consider *spoofing*. The act of spoofing is known as a specific trading activity that aims at artificially modifying the supply (limit orders) on the market without an intent to trade. Spoofers post a relatively large number of limit buy or sell orders with the intent to cancel before the orders are executed, to make other market participants believe that there is pressure to sell or buy the asset, illustrated in Figure 8. Spoofers can profits from timed buying and selling based on this manipulation. Since it can cause a sharp rise and fall of asset prices in a sudden and extremely turbulent market with exaggerated volatility and correlation, spoofing has been regarded as an illegal activity since the Dodd–Frank

Act (Dodd-Frank, 2010). However, from a practical perspective, it is difficult to clearly understand the intention to cancel for manipulation purposes unless the volume feigned to buy or sell is sufficiently large. Thus, at the individual trader level, the surveillance and detection of spoofing may not be valid.

Cartea et al. (2018, 2020) introduce the volume imbalance index as a quantitative measure of spoofing. The index ρ_t is defined as the degree of imbalance in the buy and sell limit orders currently being stack in the market such that

$$\rho_t = \frac{c_t^- - c_t^+}{c_t^- + c_t^+},$$

where $c_t^\pm$ are the best prices of the limit buy (-) and sell (+) orders at time t. This index can indirectly measure the extent to how much spoofing behavior (spoofing-like) are getting involved in the current market. The value ρ_t lies between ± 1, where the extreme values of ± 1 represent the case when only buy (or sell) limit orders are piled in the book and $\rho_t = 0$ means the case of exactly buy-sell balanced. The value of ρ_t is divided into three buckets: $(-1, 1/3)$ for buy-heavy; $(-1/3, 1/3)$ for neutral; and $(1/3, 1)$ for sell-heavy cases.

We consider the model of the limit order dynamics that spoofers are involved

$$\begin{aligned} \mathrm{d}c_t^- &= \beta_c(\theta_c^- - c_t^- + \kappa_c c_t^+)\mathrm{d}t + \eta_c \mathrm{d}M_t^- + \nu_c \mathrm{d}M_t^+ + \gamma^- \mathrm{d}J_t^-, \\ \mathrm{d}c_t^+ &= \beta_c(\theta_c^+ - c_t^+ + \kappa_c c_t^-)\mathrm{d}t + \eta_c \mathrm{d}M_t^+ + \nu_c \mathrm{d}M_t^- + \gamma^+ \mathrm{d}J_t^+, \end{aligned} \tag{20}$$

where $\gamma^\pm$ are positive constants and $J_t^\pm$ are independent Poisson processes with a constant intensity $\mu^\pm$, respectively. The LOB in Eq.(20) contains Poisson jumps occurring with a frequency rate $\mu^\pm \mathrm{d}t$ on average and the jump size $\gamma^\pm$ in each buy and sell dynamic, besides of temporary upsurge influence by market order arrivals. μ represents how often spoofers are posting the unintended limit orders in a buy (or sell) side, and γ shows how large the posted limit orders by spoofers at once are. These unexpected jumps of the limit order depth dynamics with μ and γ signify that transitory imbalance between buy and sell limit orders emerge. The differences between μ^+ and μ^- and between γ^+ and γ^- can be larger as more spoofing activities are engaged in the market. Figure A.19 simulates the sample paths of the limit order depth processes when the buy-sell limit orders are balanced. Figure A.20 illustrates the buy-heavy LOB case samples with intensity $\gamma^- = 20$ and frequency $\mu^- = 0.1$, and the degree of imbalance ρ_t is accordingly exhibited in Figure A.21.

We test how spoofing behavior can worsen the optimal expected profit of a market-maker for diverse selection of μ and γ. Table 10 shows the parameter choice to illustrate five different levels of buy-heavy imbalance including the base case with no manipulation. To see the effect from frequency μ and intensity γ seperately, we divide into two cases – μ-varying with no change in γ and γ-varying with no change in μ. Note that sell-heavy imbalance can be illstrated and tested in the same manner.

We consider the situation that a market-maker employs the optimal trading strategy under assuming no spoofing activities, yet the market is being manipulated by spoofers.

We compare the market-maker's expected terminal PnL between under the scenarios that the LOB is not manipulated (which is the case the market-maker is employing the optimal strategies) and is manipulated across various spoofing frequencies and intensities in Eq.(20) (which is the case the market-maker is using suboptimal strategies but does not know). All the scenarios are estimated by DGM with the base case parameter in Eq.(19). Generating 1,000 samples of terminal PnL under each scenario, we compare the changes in the mean terminal PnL relative to the base case without spoofing.

Table 11 presents summary statistics of terminal PnL for different spoofing frequency μ^- on the buy side of the LOB, with the spoofing intensity γ^- fixed at 10. We observe a clear and consistent decrease in mean terminal PnL as spoofing frequency increases, which is not surprising. It is linear of the extent to worsen for the PnL to the increase in the frequency level, where the rate of decreasing in the PnL is approximately \$12.4 for rising 0.1 in the spoofing frequency.

Table 12 shows the mean terminal PnL under different spoofing intensity γ^- while keeping the spoofing frequency μ^- fixed at 0.3, which are visually illustrated in Figure A.22. Unlike the frequency case, the mean terminal profit displays almost no change to the different size of spoofing intensity after it largely drops when market imbalance has appeared, as seen in the PnL values of the base case versus the others. The effect of spoofing intensity on PnL might appear counterintuitive, but it can be explained as follows. Such indifference relationship to varying γ occurs because the optimal posting strategy has a reciprocal relation with the depth of the LOB $c^{\pm}$. LOB markets follows a price-time priority rule, where limit orders offering the best price get executed first. Among orders with the same price, those placed earlier have execution priority. When the buy side of the LOB is manipulated by spoofing orders, the buy-side queue will get longer. Then, the market-maker is required to post the limit orders much closer to the mid-price to increase the probability of execution or to reduce execution risk, but one cannot place buy limit orders higher than the mid-price, that is, $\delta^* \geq 0$. Consequently, the effect of worsening PnL for the market-maker due to the overstated limit orders by spoofers can be limited, regardless of spoofing intensity size.

Concludingly, frequent spoofing activity even with smaller sizes of fake orders can hurt the market-maker's profits as these small distortions accumulate over time. This finding gives important implications to market-makers about danger of a particular spoofing behavior pattern, as collective spoofing can have substantial effects. Trading risk of market-making may be limited incurred by one large-sized spoofing orders; but, it can be significantly enlarged by frequent and collective small-sized spoofing behavior. Because of the nature that smaller spoofing sizes make it increasingly difficult to differentiate, it is more complicate to detect spoofing activities. Authorities and regulatories should exercise great caution in defining and identifying what constitutes spoofing to protect genuine traders.

Table 10: Parameter selection for five buy-heavy imbalance levels across different spoofing frequency (left) and intensity (right) including the base case of no manimulation with the other LOB model parameters chosen with Eq.(19).

	Varying Frequency				Varying Intensity			
Case	γ^+	γ^-	μ^+	μ^-	γ^+	γ^-	μ^+	μ^-
0. Base	0	0	0	0	0	0	0	0
1. Weak	0	10	0	0.1	0	5	0	0.3
2. Moderate	0	10	0	0.2	0	10	0	0.3
3. Moderate Strong	0	10	0	0.3	0	15	0	0.3
4. Strong	0	10	0	0.4	0	20	0	0.3
5. Very Strong	0	10	0	0.5	0	25	0	0.3

Table 11: Summary statistics of terminal PnL samples estimated by DGM with the base case, when spoofers are engaged at different frequency μ^- but the same intensity $\gamma^- = 10$ in the buy side LOB.

Case	γ^-	μ^-	Mean	Std	Min	Max	CV
0. Base	0	0	329.0	33.0	203.8	430.8	0.100
1. Weak	10	0.1	315.2	31.5	214.7	429.2	0.010
2. Moderate	10	0.2	300.1	30.9	195.8	398.8	0.103
3. Moderate Strong	10	0.3	288.5	30.3	205.2	376.6	0.105
4. Strong	10	0.4	277.7	30.2	177.7	368.4	0.109
5. Very Strong	10	0.5	266.8	29.2	165.1	365.0	0.109

Table 12: Summary statistics of terminal PnL samples estimated by DGM with the base case, when spoofers are engaged at different intensity γ^- but the same frequency $\mu^- = 0.3$ in the buy side LOB.

Case	γ^-	μ^-	Mean	Std	Min	Max	CV
0. Base	0	0	329.0	33.0	203.8	430.8	0.100
1. Weak	5	0.3	289.4	30.0	198.8	394.8	0.104
2. Moderate	10	0.3	288.5	30.3	205.2	376.6	0.105
3. Moderate Strong	15	0.3	288.6	30.9	172.0	383.0	0.107
4. Strong	20	0.3	288.8	30.8	206.6	386.3	0.107
5. Very Strong	25	0.3	289.0	29.4	204.9	382.9	0.102

8. Concluding Remarks

This study has discussed the two aspects for the HF market-maker's problem constructed in the LOB setup by Choi et al. (2021).

First, we proposed an efficient DGM model to estimate the optimization problem characterized as a solution of the high-dimensional PDE. We introduced a novel scheme of validating the DGM-approximated solutions through simulation of the sample distribution of the value

function. With the validation method, we were able to observe overfitting issues in DGM, which has never been discussed in the previous literature, by exploring the cases studies. Large batch size and small learning rate can easily lead to be overfitting for DGM models, because less noise in estimating gradient in the SGD optimizer can degrade the DNN model generalization power, yielding the model to more likely get stuck in sharp minima areas than smaller batch and larger learning rate models.

This investigation motivated us to find the optimal size of batch and the learning rate to obtain a robust and efficient DGM model in terms of training loss and validation. By employing new measurements – MEI and MESS that gauge the minimum training iterations and the minimum total samples in DGM, respectively, for a given batch size, we derived empirical relationships between batch size and MEI, and batch and MESS of the DGM models that achieve the similar training loss and model performance levels. Our findings are as follows. Batch size and MEI had a trade-off relationship, whereas batch size and MESS exhibited a positive relationship, which implies larger batch requires more computing cost to achive the same level of training loss and PnL validation. These results provide us a reference for selecting the optimal DGM model given a limited computing budget to obtain a robust DGM model.

Second, we examined the interaction of the diverse environments to determine the degrees of HF market stability and market manipulation with the market-maker's maximized expected wealth, by generating proper data sets using the optimal training configurations for DGM. The sensitivity analysis brought us the following findings. Based on the HF market stability measured with the four parameters including the synchronizing factor, the self/mutual-exciting componenets, and the mean-reversion speed, as a whole, lower market stability enhances profitability of the market-maker, yet which is the case only when the synchrony level between buy and sell orders increase, or the self-exciting tendency in each market order arrival rises. For the case that market stability increased by mutual-exciting and mean-reversion speed factors, there exhibited a significant relationship only when the market stability regime shifted from a moderate low to a low level, not for the whole range.

HF market manipulation was discussed with spoofing behaviors. We examined how much spoofing behaviors can deteriorate the market-maker's profit in terms of the spoofing frequency and the intensity size. We observed that the frequency of the emergence of spoofers does matter to worsen for the optimal market-maker's wealth significantly, while its intensity has a limited effect to hurt the market-maker's profit. It means frequent and collective spoofing behaviors even with smaller sizes of unintended orders to execute can largely disrupt to play primary roles for a market-maker for provision of liquidity by hurting its profitability. As considering the nature that small-sized spoofing or spoofing-like trading is increasingly difficult to be detected due to largely unknown sophisticated schemes employed by HF traders, regulatories should take great cautions to set right policies for HF illegal trading including spoofing and sniffing, to assist genuine HF traders to play their own roles.

The proposed validation method available in DGM and the strategy to select the optimal DGM model are widely applicable, not limited to our market-maker's problem, offering bene-

fits to fields such as quantitative finance, reinforcement learning, and other high-dimensional optimization problems. Also, for the empirical findings for HF trading in terms of market stability and manipulation, we believe that market practitioners and relevant policy makers should recognize our empirical results that the market-makers intimately interact with delicate changes in HF market evironments for better management of HF trading.

Acknowledgements

The authors are grateful to anonymous reviewers for providing their constructive comments and suggestions.

References

Ait-Sahalia, Y., Brunetti, C., 2020. High frequency traders and the price process. Journal of Econometrics 217, 20–45.

Al-Aradi, A., Correia, A., Jardim, G., Naiff, D., Saporito, Y., 2022. Extensions of the deep galerkin method. arXiv Working Paper -, –.

Avellaneda, M., Stoikov, S., 2008. High-frequency trading in a limit order book. Quantitative Finance 8, 217–224.

Baron, M., Brogaard, J., Hagstromer, B., Kirilenko, A., 2019. Risk and return in high-frequency trading. Journal of Financial and Quantitative Analysis 53, 994–1024.

Beck, C., Becker, S., Cheridito, P., Jentzen, A., Neufeld, A., 2021. Deep splitting method for parabolic pdes. SIAM J. SCI. COMPUT 43, 3135–3153.

Beck, C., E, W., Jentzen, A., 2019. Machine learning approximation algorithms for high-dimensional fully nonlinear partial differential equations and second-order backward stochastic differential equations. Journal of Nonlinear Science 29, 1563–1619.

Beck, C., Hutzenthaler, M., Jentzen, A., Kuckuck, B., 2022. An overview on deep learning-based approximation methods for partial differential equations. arXiv Working Paper -, –.

Biais, B., Declerck, F., Moinas, S., 2016. Who supplies liquidity, how and when? BIS Working Papers -, –.

Biais, B., Woolley, P., 2011. High frequency trading. Toulouse School of Economics and London School of Economics -, –.

Brogaard, J., Hendershott, T., Riordan, R., 2014. High-frequency trading and price discovery. The Review of Financial Studies 27, 2267–2306.

Brogaard, J., Hendershott, T., Riordan, R., 2019. Price discovery without trading: Evidence from limit orders. The Journal of Finance 74, 1583–2143.

Cartea, A., Donnelly, R., Jaimungal, S., 2018. Enhancing trading strategies with order book signals. Applied Mathematical Finance 25, 1–35.

Cartea, A., Jaimungal, S., 2013. Modelling asset prices for algorithmic and high-frequency trading. Appl Math Fin 20, 512–547.

Cartea, A., Jaimungal, S., Ricci, J., 2014. Buy low, sell high: A high frequency trading perspective. SIAM Journal on Financial Mathematics 60, 415–444.

Cartea, A., Jaimungal, S., Wang, Y., 2020. Spoofing and price manipulation in order-driven markets. Applied Mathematical Finance -, 1–32.

Cha, J., Chun, S., Lee, K., Cho, H., Park, S., Lee, Y., Park, S., 2021. Swad: Domain generalization by seeking flat minima. arXiv Working Paper -, –.

Chaboud, A., Chiquoine, B., Hjalmarsson, E., , Vega, C., 2014. Rise of the machines: Algorithmic trading in the foreign exchange market. The Journal of Finance 69, 1851–2341.

Chen, J., Jin, S., Lyu, L., 2021. A deep learning based discontinuous galerkin method for hyperbolic equations with discontinuous solutions and random uncertainties. arXiv Working Paper -, –.

Choi, S., Jang, H., Lee, K., Zheng, H., 2021. Optimal market-making strategies under synchronised order arrivals with deep neural networks. Journal of Economic Dynamics and Control 125, 104098.

Dodd-Frank, 2010. Public law 111-203, reform, dodd-frank wall street and act, consumerprotec- tion. US Statutes at Large 124, –.

Dunn, O., 1961. Multiple comparisons among means. Journal of the American Statistical Association 56, 52–64.

Grohs, P., Hornung, F., Jentzen, A., Wurstemberger, P., 2023. Convergence analysis of the deep galerkin method for weak solutions. Memoirs of the American Mathematical Society -, –.

Gueant, O., Lehalle, C.A., Fernandez-Tapia, J., 2013. Dealing with the inventory risk: a solution to the market making problem. Math Finan Econ 8, 477–507.

Guilbaud, F., Pham, H., 2013. Optimal high-frequency trading with limit and market orders. Quantitative Finance 13, 79–94.

Guo, X., Larrard, A., Ruan, Z., 2017. Optimal placement in a limit order book: an analytical approach. Math Finan Econ 11, 189–213.

Hasbrouck, J., Saar, G., 2013. Low-latency trading. The Journal of Financial Markets 16, 646–679.

Hendershott, T., Jones, C., Menkveld, A., 2011. Does algorithmic trading improve liquidity? The Journal of Finance 66, 1–33.

Ho, T., Stoll, H.R., 1981. Optimal dealer pricing under transactions and return uncertainty. Journal of Financial Economics 9, 47–73.

Hochreiter, S., Schmidhuber, J., 1997. Flat minima. Neural Computation 9, 1–42.

Hornik, K., 1991. Approximation capabilities of multilayer feedforward networks. Neural Networks 4, 251–257. doi:10.1016/0893-6080(91)90009-T.

Hutzenthaler, M., Jentzen, A., Kruse, T., Nguyen, T., 2020. Deep splitting method for parabolic pdes. SN Partial Differ. Equ. Appl. 1, 1–10.

Jiao, Y., Lai, Y., Wang, Y., Yang, H., Yang, Y., 2023. Convergence analysis of the deep galerkin method for weak solutions. arXiv Working Paper -, –.

Kaddour, J., Liu, L., Silva, R., Kusner, M., 2022. When do flat minima optimizers work? 36th Conference on NIPS 36, –.

Kingma, D.P., Ba, J., 2015. Adam: A Method for Stochastic Optimization arXiv:1412.6980.

McCandlish, J., Kaplan, J., Amodei, D., 2018. An empirical model of large-batch training. arXiv Working Paper -, –.

Menkveld, A., 2013. High frequency trading and the new market makers. Journal of Financial Markets 16, 712–740.

Menkveld, A., Zoican, M., 2017. Need for speed? exchange latency and liquidity. The Review of Financial Studies 30, 712–740.

Miller, R.S., Shorter, G., 2016. High frequency trading: Overview of recent developments. Congressional Research Service .

Ogata, Y., 1978. The asymptotic behaviour of maximum likelihood estimators for stationary point processes. Annals of the Institute of Statistical Mathematics 30, 243–261. doi:10.1007/BF02480216.

Rosu, I., 2009. A dynamic model of the limit order book. The Review of Financial Studies 22, 4601–4641.

SEC, 2005. Sec adopts regulation national market system (reg nms). US Securities and Exchange Commission August, –. URL: https://www.sec.gov/files/rules/final/34-51808.pdf.

Shallue, C., Lee, J., Antognini, J., Sohl-Dickstein, J., Frostig, R., Dahl, G., 2019. Measuring the effects of data parallelism on neural network training. Journal of Machine Learning Research 20, 1–49.

Sirignano, J., Spiliopoulos, K., 2018. DGM: A deep learning algorithm for solving partial differential equations. Journal of Computational Physics 375, 1339–1364. URL: https://linkinghub.elsevier.com/retrieve/pii/S0021999118305527, doi:10.1016/j.jcp.2018.08.029.

Smith, L., 2018. A disciplined approach to neural network hyper-parameters. US Naval Research Laboratory Technical Report 5510, –.

Veraart, L.A.M., 2010. Optimal market making in the foreign exchange market. Appl Math Fin 17, 359–372.

Zhang, F., 2010. High-frequency trading, stock volatility, and price discovery. SSRN -, –.

Zhang, J., Shi, Y., Gao, Y., 2023. Exploring flat minima for domain generalization with large learning rates. arXiv Working Paper -, –.

Appendix A. Supplementary Figures

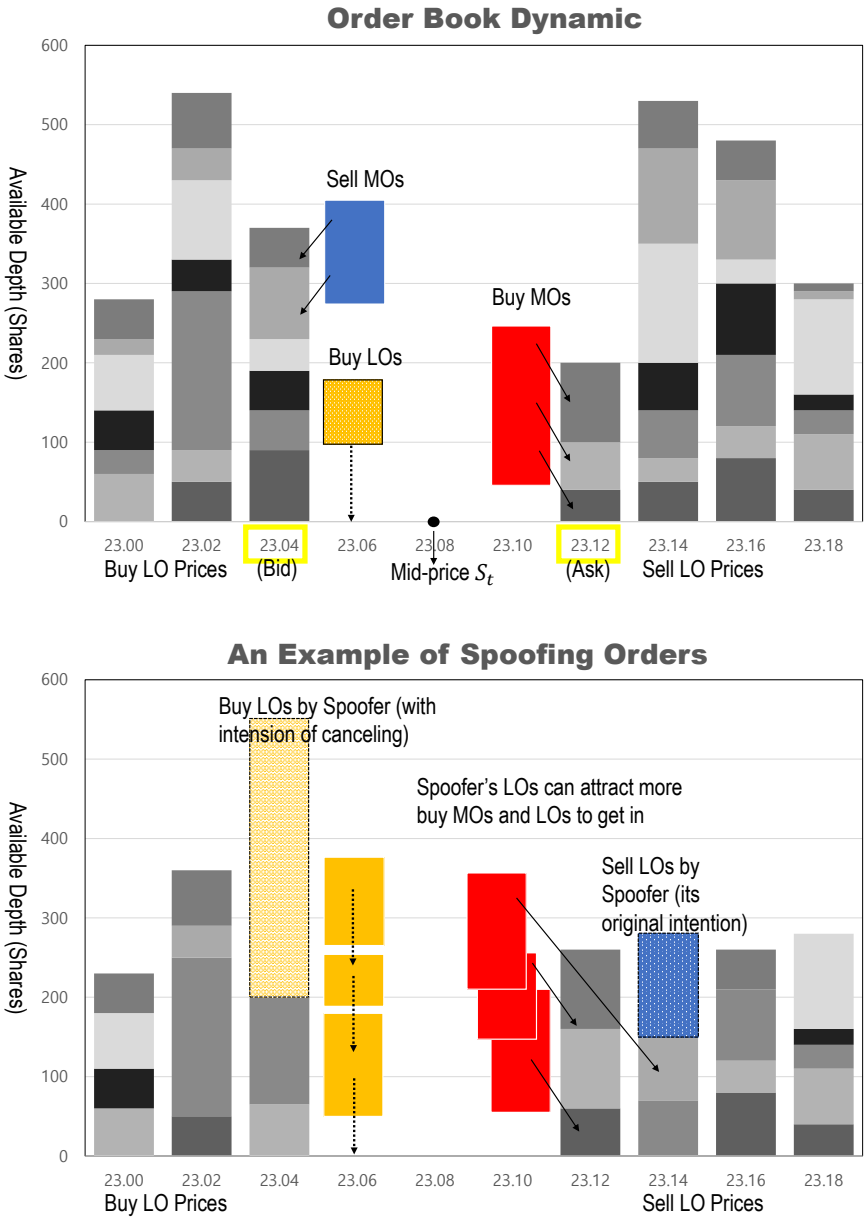

Figure 8: Illustration of the act of spoofing versus normal trading in the LOB.

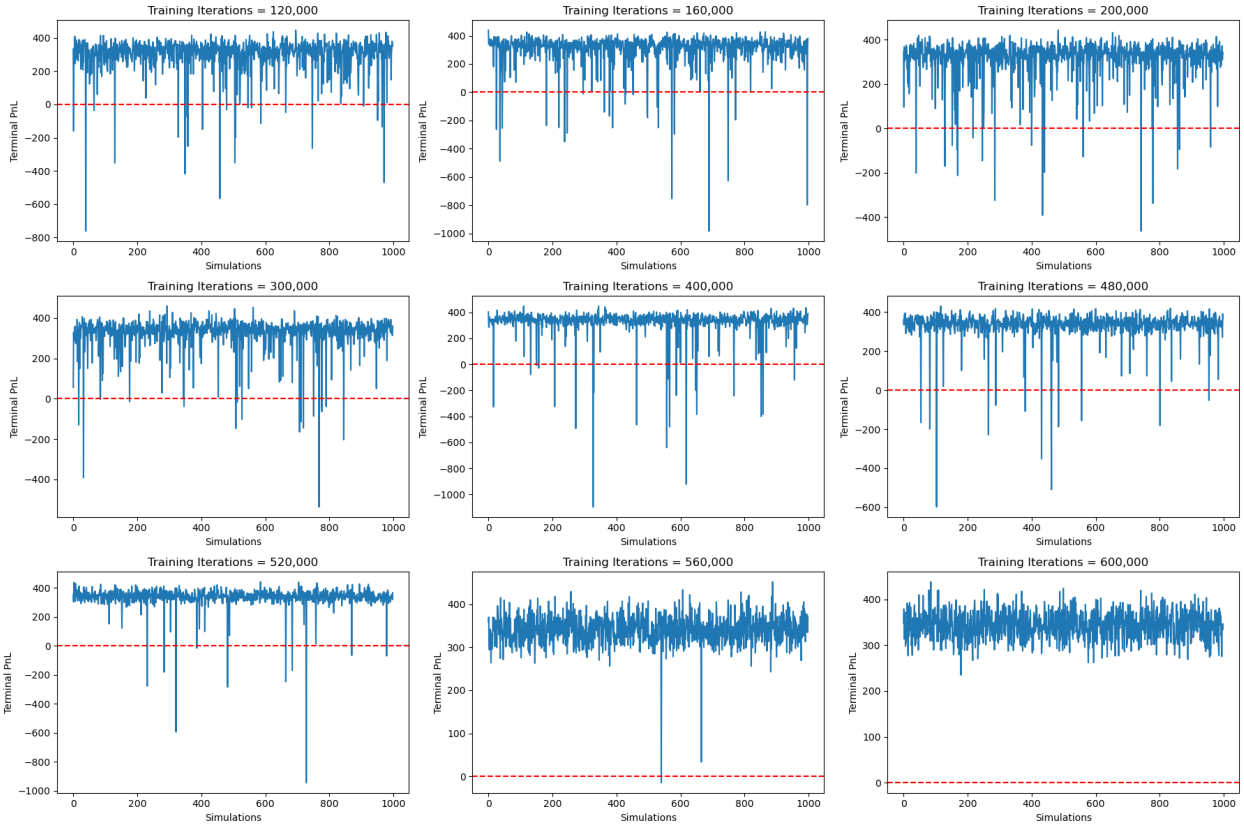

Figure A.9: Terminal PnL over simulations for different training iterations. Each subplot shows the Terminal PnL corresponding to the same model at different stages of training

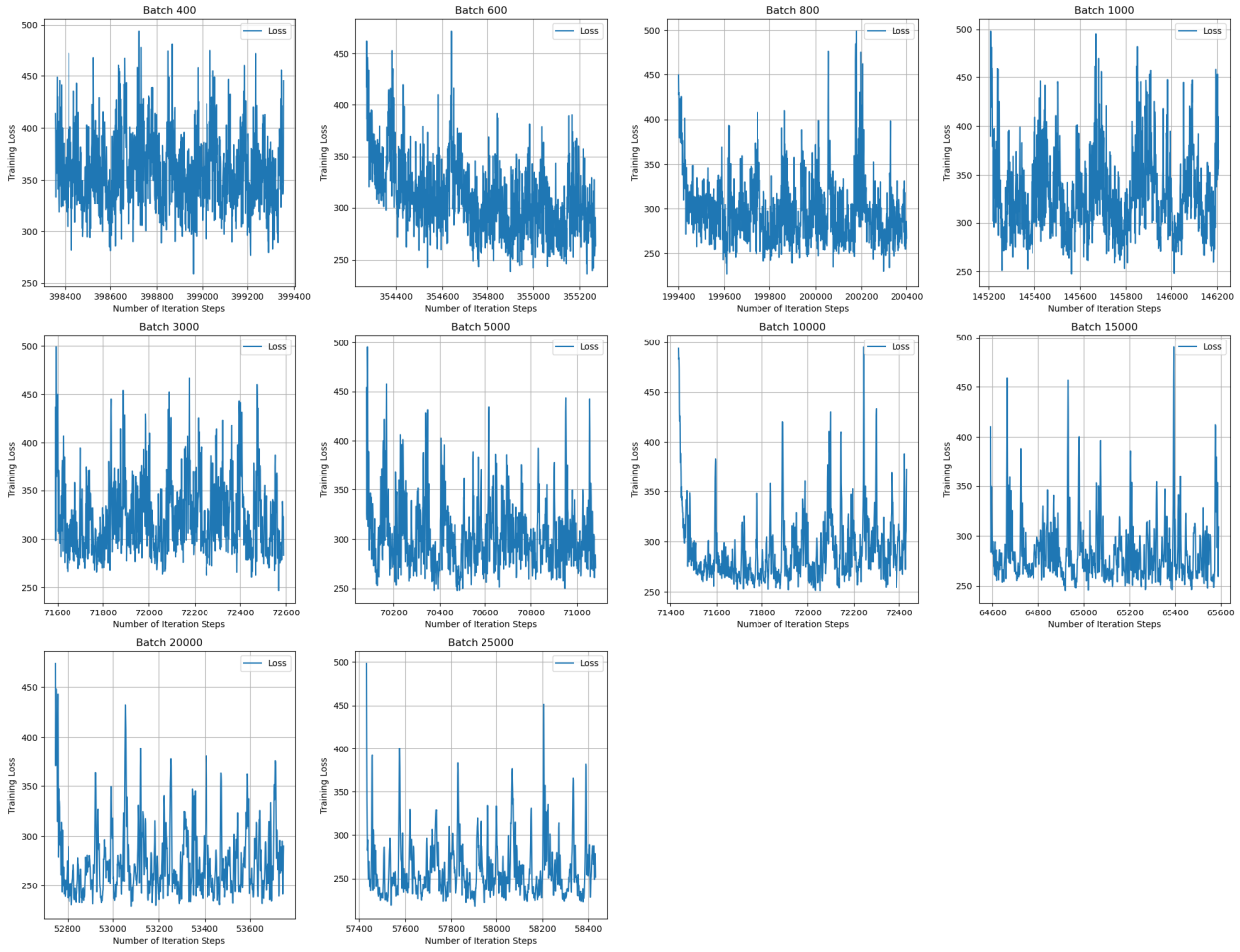

Figure A.10: MEI case: The last 1000 training losses for the MEI case, which shows the stability and convergence behavior of the training process near MEI.

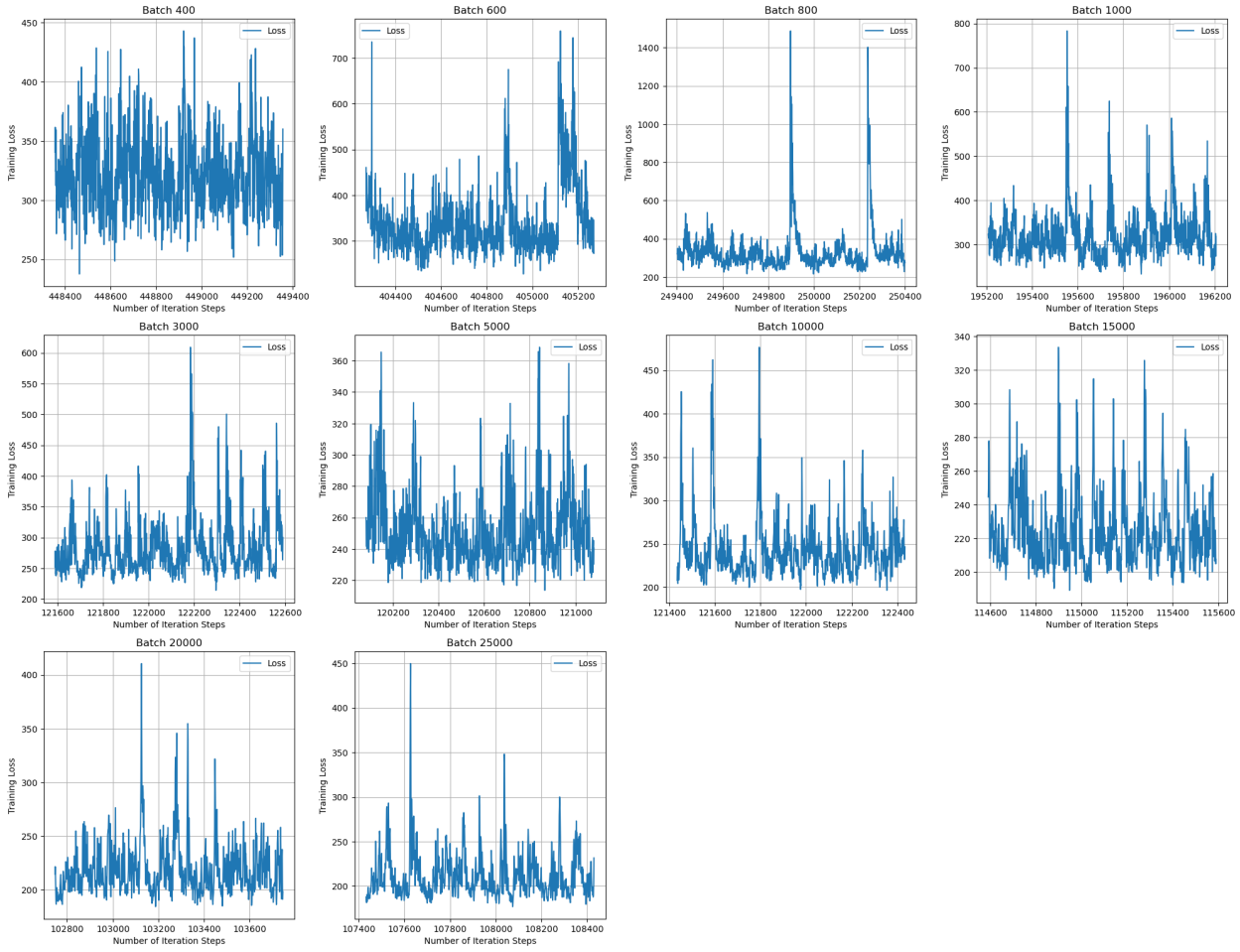

Figure A.11: Continued case: The last 1000 training losses for the continued case, which illustrates the training behavior during the additional 50,000 iterations beyond MEI.

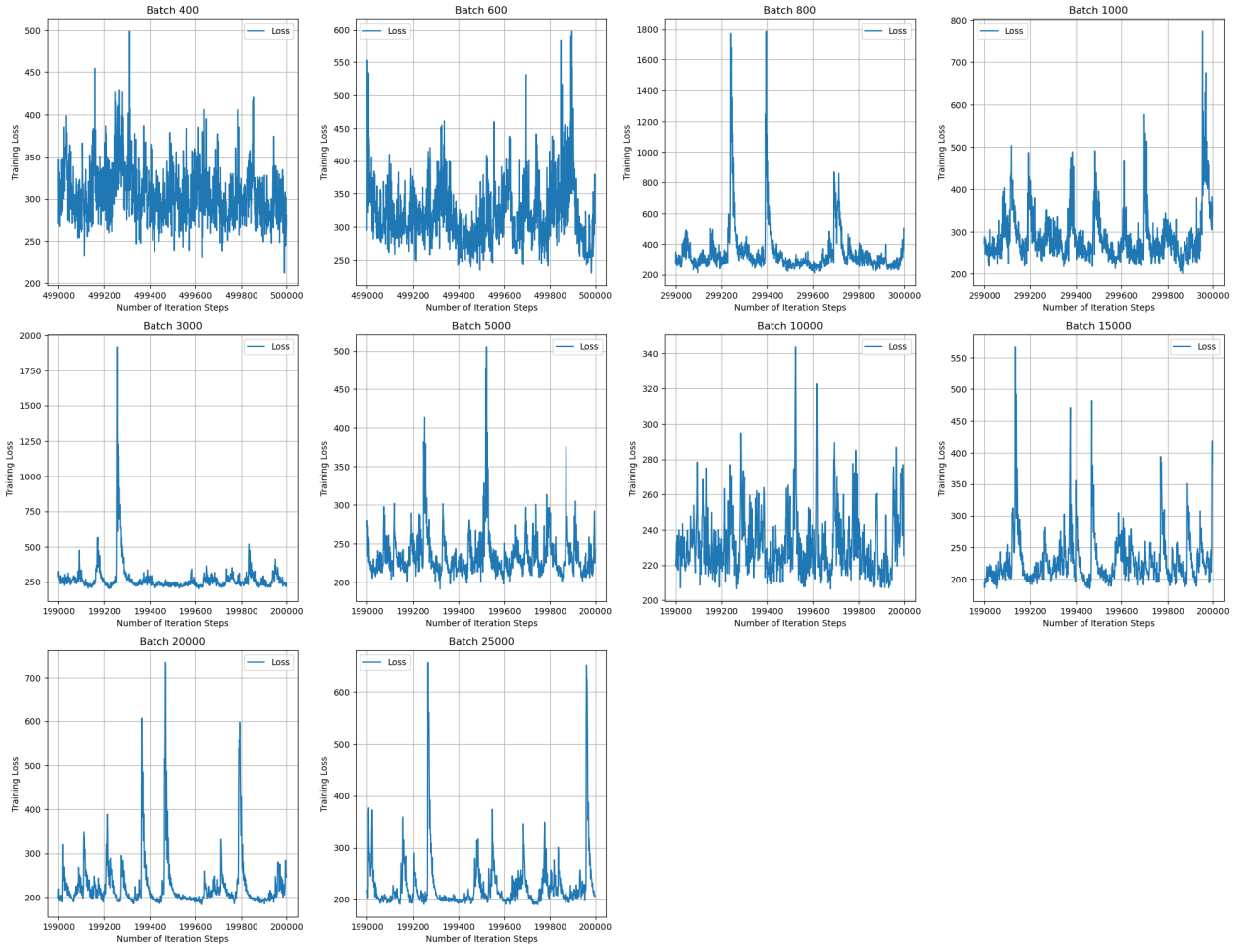

Figure A.12: Extended case: The last 1000 training losses for the extended case, which highlights the training dynamics during the prolonged training iterations beyond MEI.

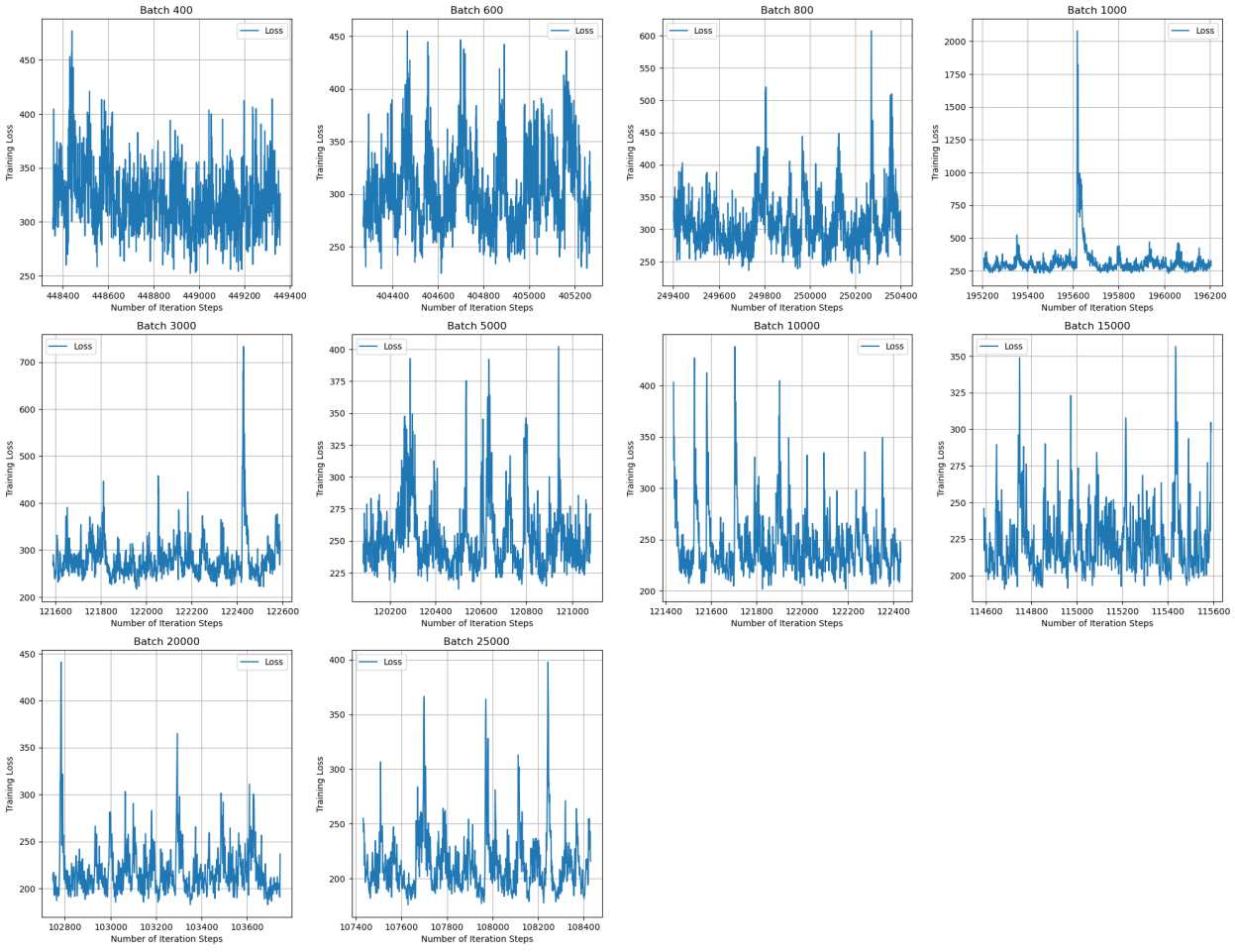

Figure A.13: Refined case: Last 1000 training losses for the refined case, which shows the training behavior as the learning rate is progressively reduced following MEI.

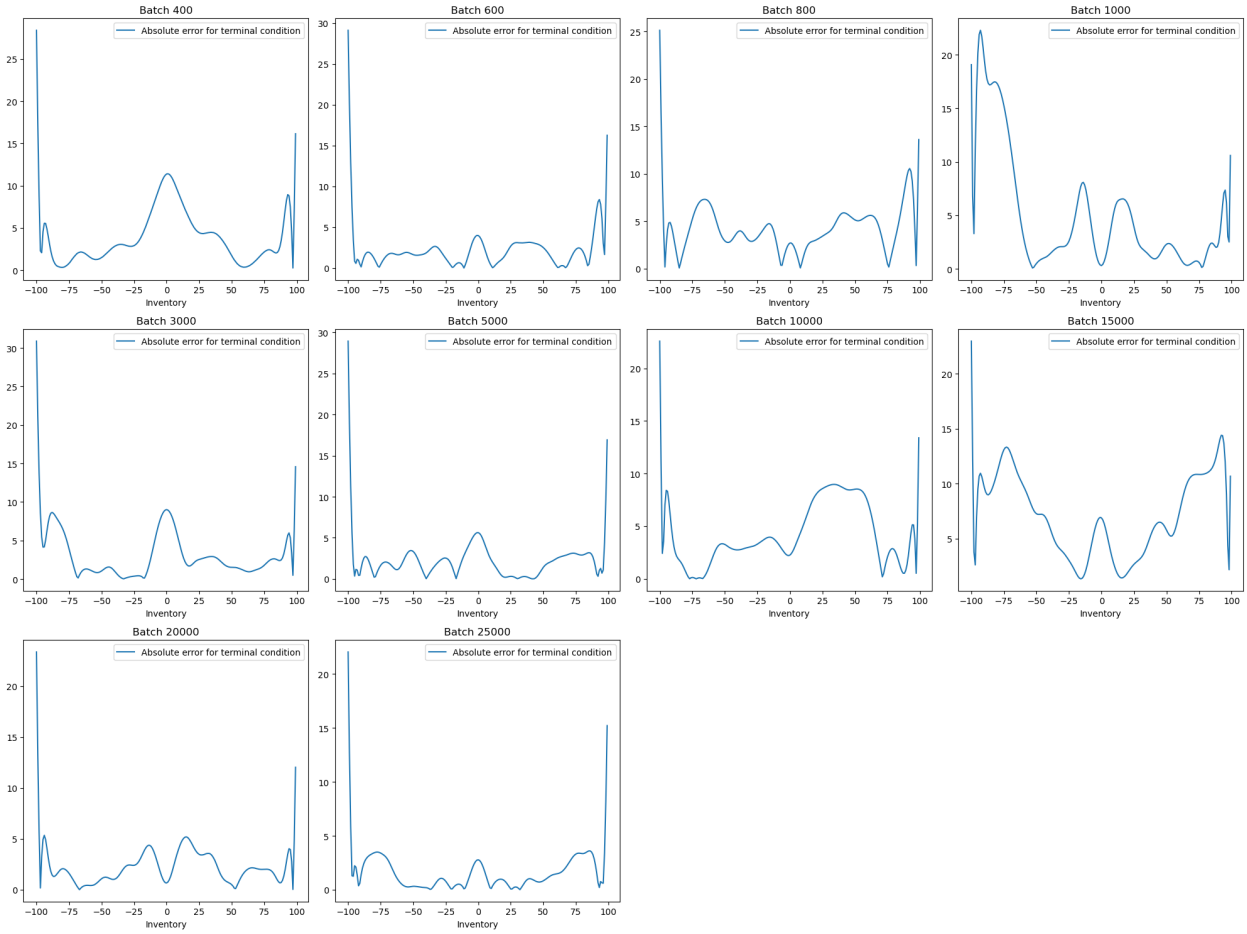

Figure A.14: MEI case: Absolute difference between the terminal condition of g and its DNN-approximated solution f for models trained with different batch sizes under the MEI case.

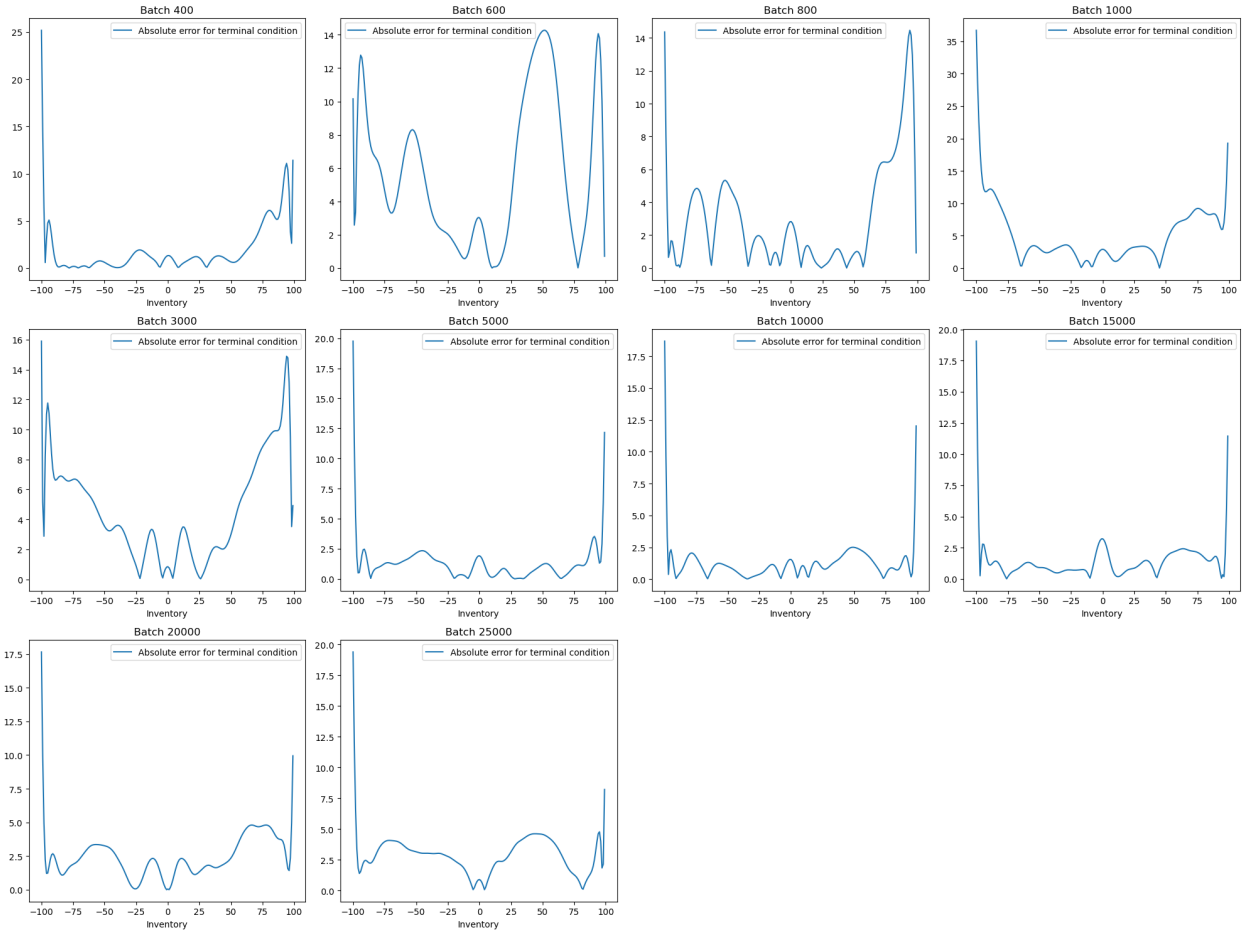

Figure A.15: Continued case: Absolute difference between the terminal condition of g and its DNN-approximated solution f for models trained with different batch sizes under the continued case.

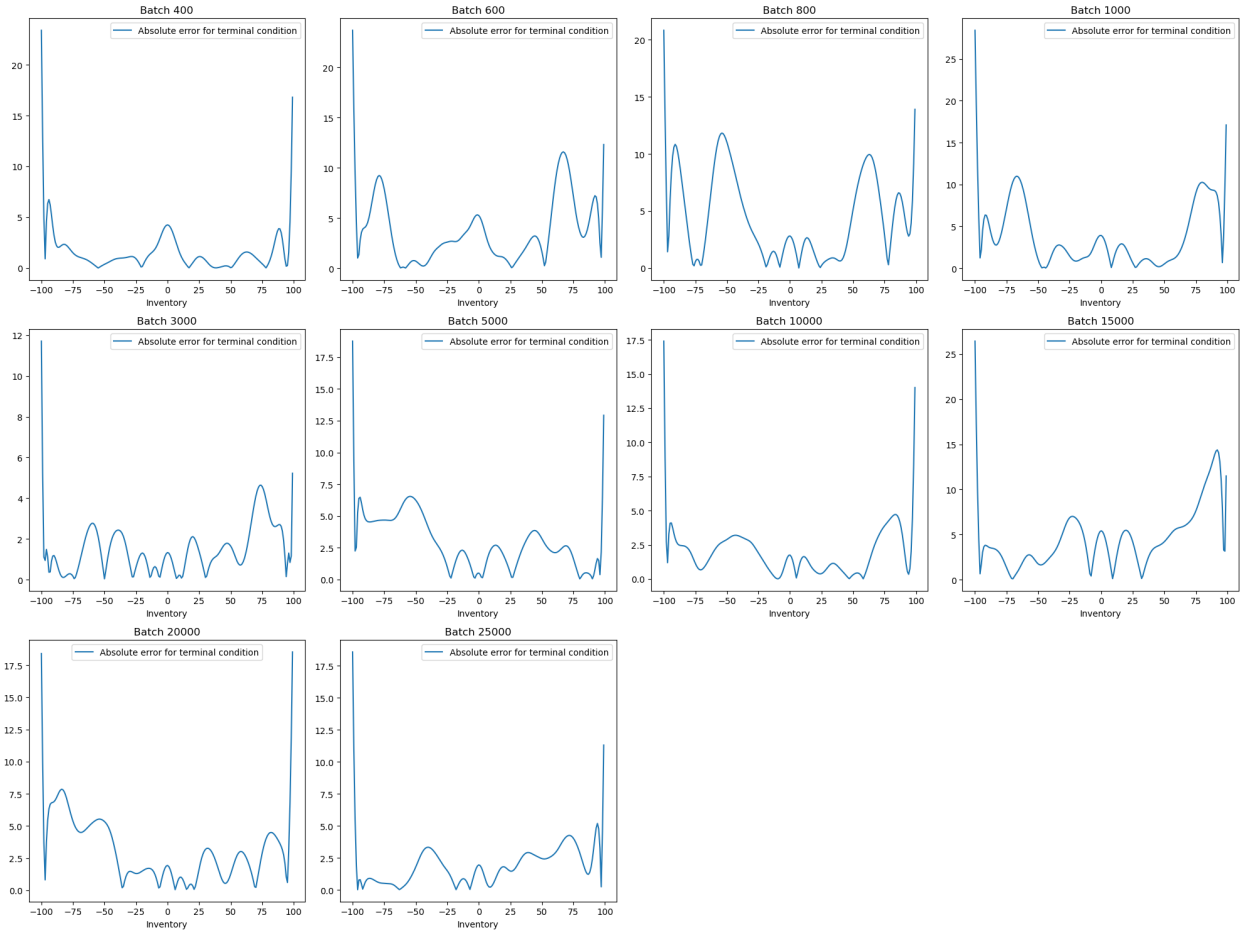

Figure A.16: Extended case: Absolute difference between the terminal condition of g and its DNN-approximated solution f for models trained with different batch sizes under the extended case.

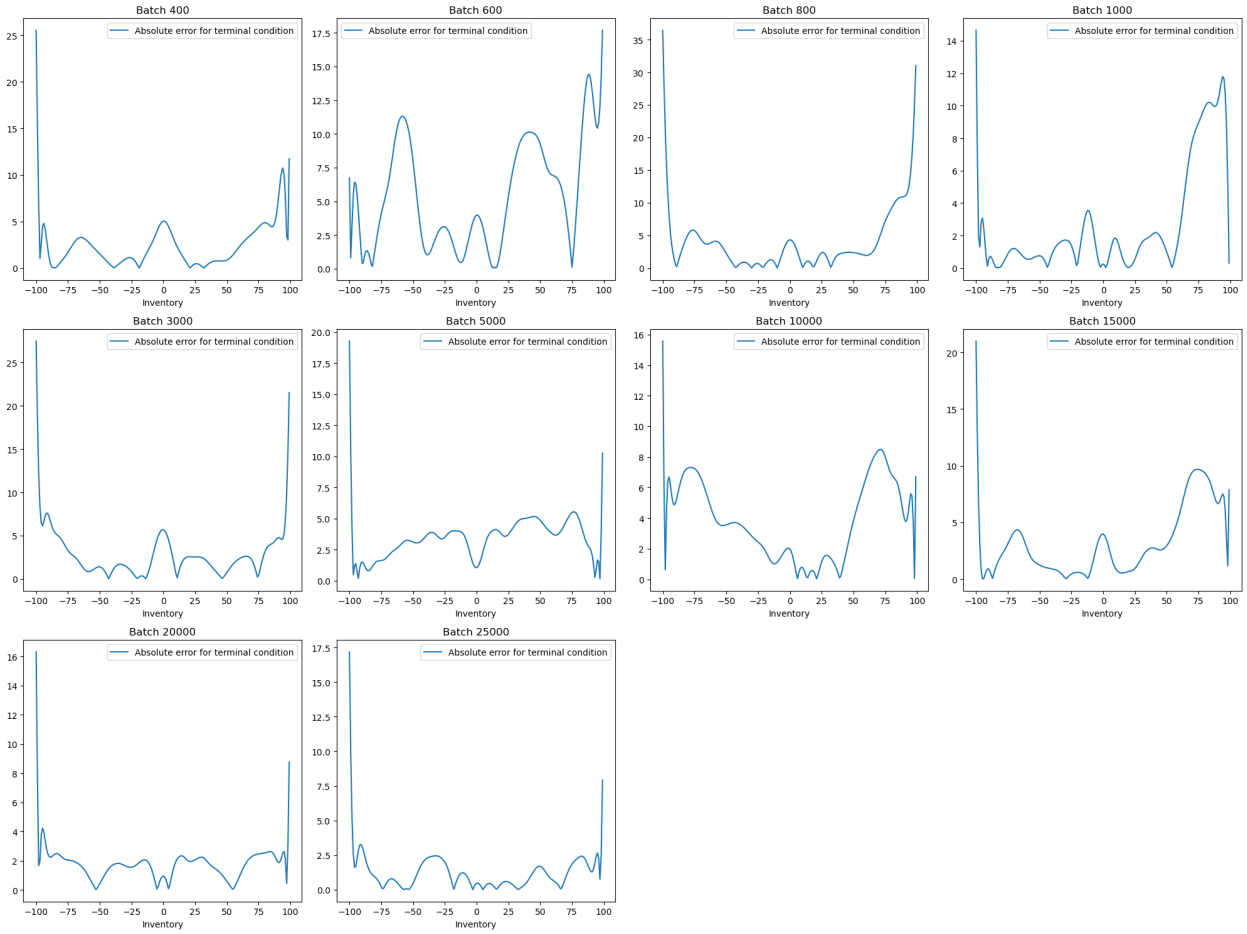

Figure A.17: Refined case: Absolute difference between the terminal condition of g and its DNN-approximated solution f for models trained with different batch sizes under the refined case.

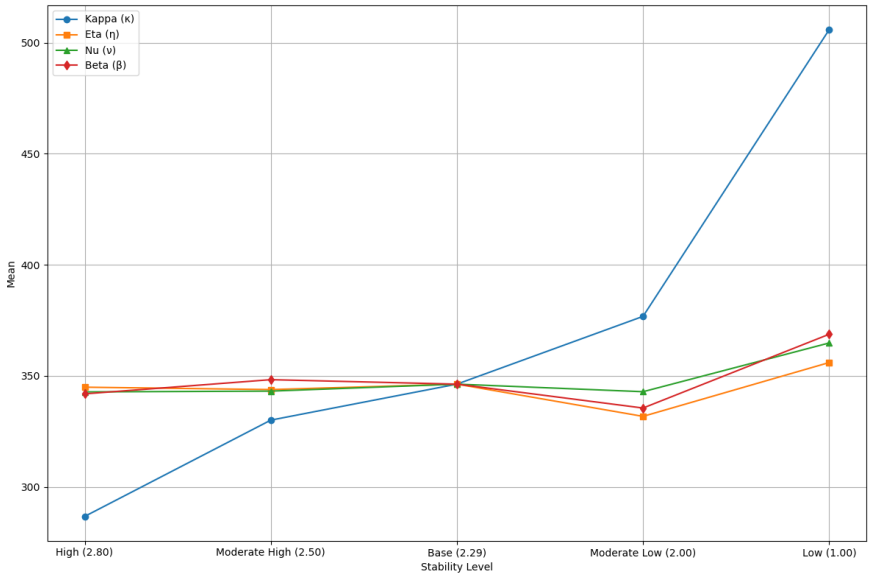

Figure A.18: Comparative sensitivity of the mean terminal PnL across levels of stability attributed to factors (κ, η, ν, and β).

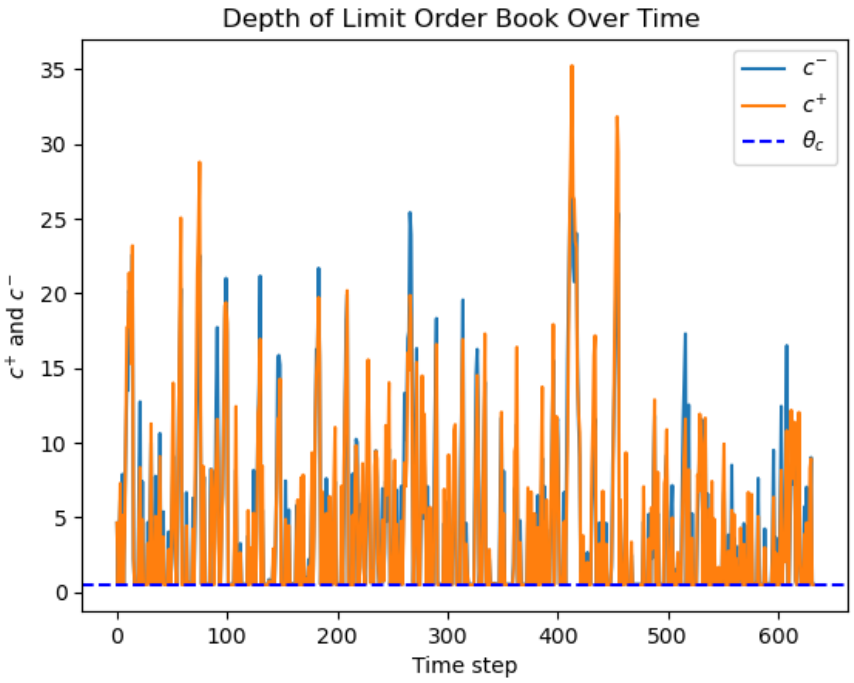

Figure A.19: LOB depth dynamics over time under no spoofing (buy/sell-balanced).

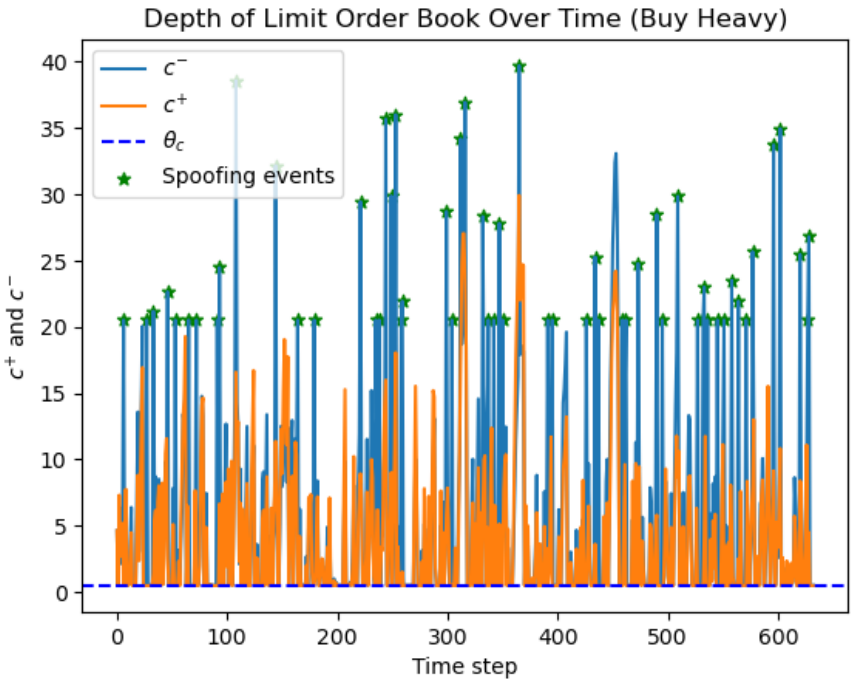

Figure A.20: LOB depth dynamics over time with spoofing activity engaged in the buy LOB with intensity $\gamma^- = 20$ and frequency $\mu^- = 0.1$ (buy-heavy).

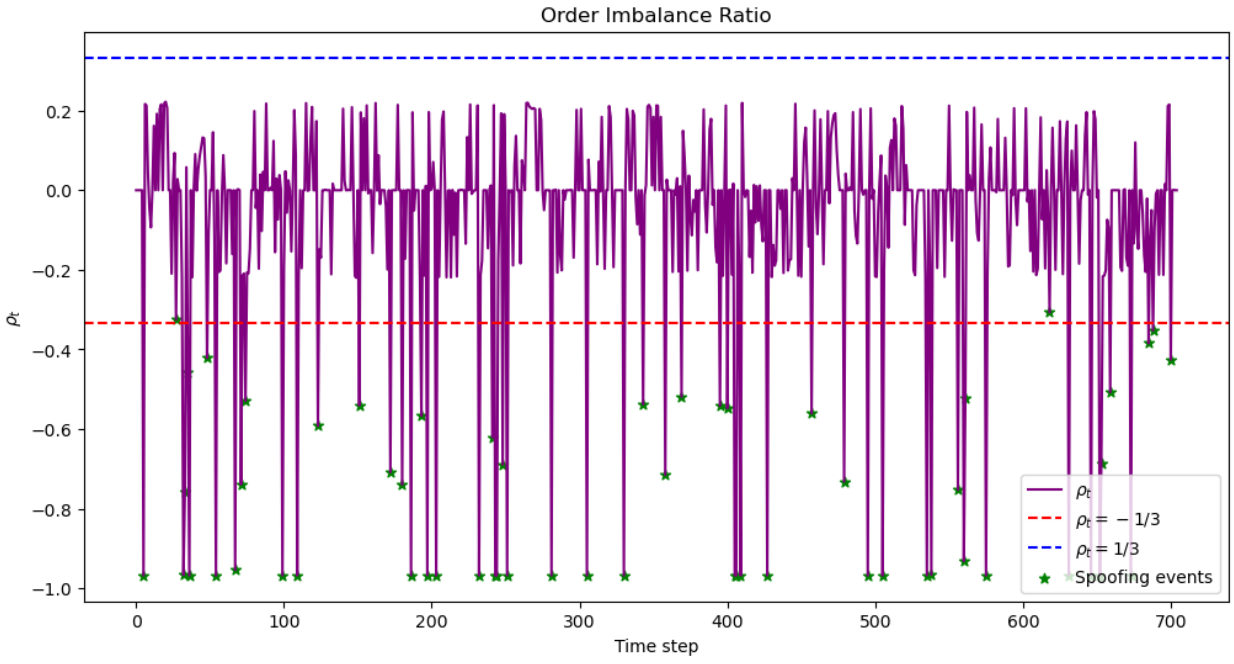

Figure A.21: Order imbalance ρ_t over time with the parameters in Figure A.20. Stars represent the timings of spoofed buy orders. The horizontal blue and red lines correspond to order imbalance values of 1/3 and -1/3, respectively. Order imbalance is considered neutral between these lines, sell-heavy above the blue line, and buy-heavy below the red line.

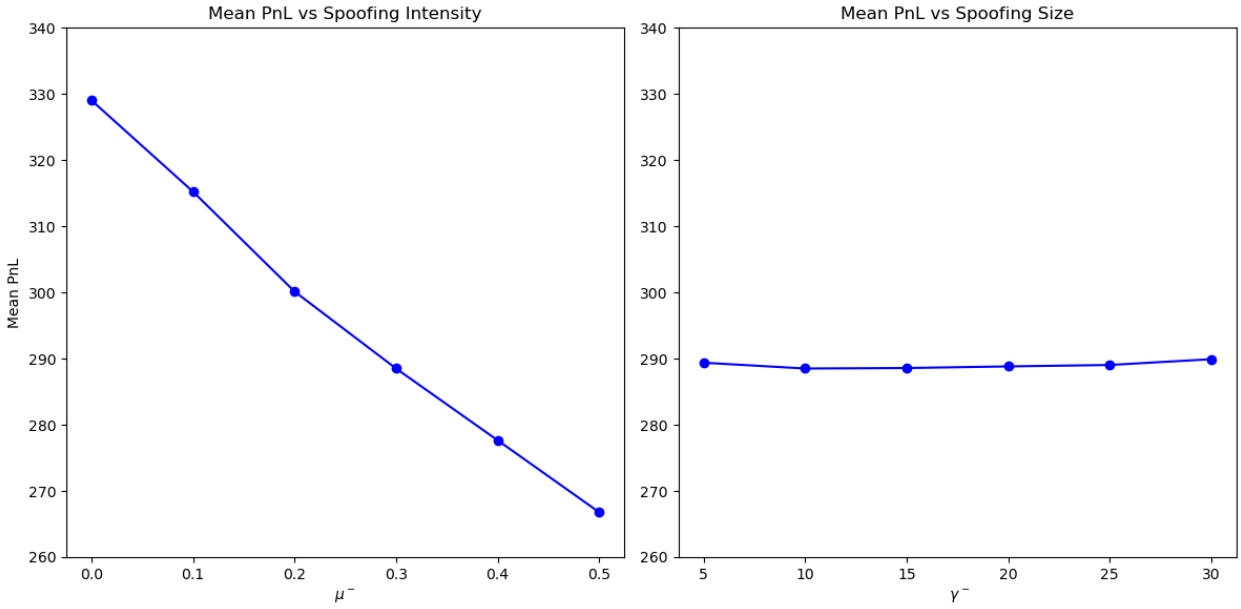

Figure A.22: Mean terminal PnL across different spoofing frequencies with the same spoofing intensity $\gamma^- = 10$ (left) and across different intensity sizes with the same spoofing frequency $\mu^- = 0.3$ (right) in the buy side of LOB.