Limit Order Book Simulation

Andrew Braun - V00955955
Evan Strasdin - V00907185

December 10, 2025

1 Problem Description and Goal of the Simulation Study

Modern electronic financial markets match buy and sell orders through a *limit order book* (LOB), which aggregates outstanding limit orders at discrete price levels and executes trades whenever compatible buy and sell orders meet. The microscopic dynamics of the LOB (i.e. arrival of new limit and market orders, cancellations of resting orders, and the quoting behaviour of liquidity providers) jointly determine key measures of market quality such as bid–ask spreads, execution latency, depth, and price volatility. Analytical models for these dynamics are typically intractable once realistic features such as price–time priority, stochastic order sizes, and strategic market making are introduced, which motivates the use of discrete-event simulation.

In this project we construct a continuous-time discrete-event simulation of a single-asset LOB with a single quoting market maker. Order flow on both sides of the book is driven by independent Poisson processes for limit orders, market orders, and cancellations, and the LOB is represented as two priority queues that enforce price–time priority. On top of this background order flow, a market maker posts bid and ask quotes at regular intervals according to a simple inventory-based strategy: quotes are placed symmetrically around the midprice with a configurable base spread and are skewed as a linear function of the current inventory.

The simulation study focuses on three structural components of the system:
(i) an *order strategy* parameter controlling the mean size of incoming orders,
(ii) a *market structure* parameter controlling how tightly limit order prices are distributed around the prevailing midprice, and
(iii) a *market maker strategy* parameter controlling the aggressiveness of the market maker's quotes.
By varying these components over a factorial grid of scenarios and running multiple independent replications in each scenario, the goal is to quantify how they affect market quality and market maker performance. Concretely, the study aims to estimate and compare average spreads, execution-time distributions, trading activity, LOB depth profiles, price volatility, and market maker profit

and inventory risk across scenarios, and to draw data-driven conclusions about which combinations of structural parameters lead to more efficient and stable markets.

2 Simulation Model

The simulated market is a continuous-time, double-auction system in which all state changes arise from discrete events: arrivals of limit orders, arrivals of market orders, cancellations of resting orders, and periodic submissions of market-maker quotes. The core of the model is the limit order book (LOB), which we represent explicitly as a pair of priority queues. This representation provides a clean mapping from market microstructure rules-specifically, price–time priority-to well-understood queueing constructs.

2.1 Mapping to a Priority Queueing Model

At any time t, the LOB maintains two ordered sets of outstanding limit orders:

$$\mathcal{B}(t) = \{\text{resting buy orders}\}, \qquad \mathcal{A}(t) = \{\text{resting sell orders}\}.$$

Each order is defined by its side (buy or sell), limit price, creation time, remaining quantity, and an identifier labeling the trader who submitted it. Because execution in electronic markets follows price–time priority, orders at more competitive prices must be considered first, and ties at the same price are broken by arrival time.

To encode these priorities efficiently, we treat $\mathcal{B}(t)$ as a *max-heap* keyed lexicographically by

$$(-\text{price},\ \text{time},\ \text{order\_id}),$$

so that the order with the highest price and earliest creation time always sits at the top. Analogously, $\mathcal{A}(t)$ is a *min-heap* keyed by

$$(\text{price},\ \text{time},\ \text{order\_id}),$$

ensuring that the lowest-priced, earliest-arriving sell order has highest priority. This mapping converts the LOB into a two-server priority queueing system in which the "service" operation corresponds to trade execution: whenever the book becomes crossed, meaning

$$P_{\text{best bid}}(t) \ \geq \ P_{\text{best ask}}(t),$$

the simulation executes one unit of quantity by removing one unit from the highest-priority buy and sell orders. Because the events are processed in continuous time, the queues evolve dynamically as orders arrive, are partially or fully executed, or are cancelled.

2.2 Order Dynamics and Event Types

The system evolves through five independent Poisson arrival processes:

- limit buy arrivals,
- limit sell arrivals,
- market buy arrivals,
- market sell arrivals,
- cancellations of resting orders.

If the next event at time t is a limit-order arrival, the simulation draws a limit price by sampling an exponentially distributed distance from the prevailing midprice and rounding it to the nearest tick. Market-order arrivals are modeled as sequences of one-unit executions against the opposite queue, again following priority rules.

Cancellations are modeled as abandoning the current best order on a randomly selected side of the book. Although simple, this abstraction corresponds to impatient traders withdrawing orders and has the practical effect of preventing the book from accumulating stale liquidity that might distort spread or execution-time statistics.

2.3 Market Maker Behaviour

A single market maker (MM) participates alongside the stochastic background order flow. At fixed intervals, the MM observes the current midprice,

$$M(t) = \tfrac{1}{2}\big(P_{\text{bid}}(t) + P_{\text{ask}}(t)\big),$$

and submits one-unit buy and sell limit orders at prices

$$M(t) - \delta + S(t) \qquad \text{and} \qquad M(t) + \delta + S(t),$$

where δ is a configurable base spread and $S(t)$ is an inventory-skew term proportional to the MM's current inventory. This state-dependent rule embeds a stabilizing mechanism: excessive inventory induces the MM to place quotes further away from the midprice on the side that reduces exposure.

When MM orders are executed, inventory and cash are updated immediately, and mark-to-market P&L is computed using the most recent trade price. This mechanism allows the simulation to evaluate the profitability and risk of the MM under different market conditions.

2.4 Discrete-Event Representation

All components described above are integrated into a discrete-event simulation (DES). Each event updates the system state and may trigger additional "service completions" (trades) if the book becomes crossed. The priority-queue formulation ensures that each such trade corresponds precisely to serving the highest-priority items in the two queues. Thus, the DES framework provides a faithful and computationally efficient representation of LOB microstructure while allowing controlled experimentation over structural parameters.

3 Simulation Parameters

The simulation operates under a set of fixed environmental parameters and a controlled set of experimental factors. These parameters govern the arrival patterns of orders, the behaviour of the market maker, the pricing rules for limit orders, and the overall duration and granularity of the simulated market. All simulations share the same baseline configuration, ensuring that differences in performance metrics arise only from the structural factors varied in the experimental design.

3.1 Fixed Parameters

Several parameters are held constant across all scenarios in order to establish a well-defined and realistic market environment:

- **Trading horizon.**
  Each simulation run spans 23,400 seconds (6.5 hours), corresponding to a stylized full trading session. This interval is long enough to generate substantial trading volume and to stabilize estimates of spreads, execution times, and market-maker performance.
- **Price discretization.**
  All prices evolve on a discrete grid with a tick size of 0.01. Limit prices generated by the model's stochastic pricing rule are rounded to the nearest tick, ensuring consistent granularity across scenarios.
- **Initial conditions.**
  At time $t = 0$, the order book is seeded with a minimal buy order at 99.99 and a sell order at 100.01. This guarantees a valid initial midprice and a strictly positive spread, avoiding undefined behaviour before the first stochastic event.
- **Order-flow intensities.**
  The market is driven by five independent Poisson processes with fixed rates:
  $$\lambda_{\text{limit buy}} = 0.6, \quad \lambda_{\text{limit sell}} = 0.6,$$
  $$\lambda_{\text{market buy}} = 0.2, \quad \lambda_{\text{market sell}} = 0.2, \quad \lambda_{\text{cancel}} = 0.1.$$

These values produce an active market in which limit order submissions dominate market orders, and cancellations occur frequently enough to prevent unrealistic liquidity accumulation.

- **Market-maker quoting rules.** The market maker posts new buy and sell quotations every 10 seconds. Quotes are placed symmetrically around the prevailing midprice and include an inventory-based adjustment term governed by a fixed skew coefficient of 0.05. This induces stabilizing behaviour by encouraging the market maker to reduce excess inventory.

Collectively, these fixed parameters define a neutral market environment that allows direct comparison of structural changes introduced by the experimental factors.

3.2 Experimental Factors

The study examines three structural components of the market microstructure, each varied over five levels. Together, these form a full $5 \times 5 \times 5$ factorial design containing 125 distinct scenarios. The factors are defined as follows:

1. **Mean Order Size:** This parameter controls the average volume (quantity of shares) for individual Limit and Market orders submitted by the simulated background traders. It dictates the typical "chunk size" of liquidity entering and exiting the market.

2. **Limit Price Decay Rate:** This parameter governs the probability distribution of prices for incoming limit orders relative to the best quotes. A higher decay rate concentrates new orders closer to the inside spread (simulating a competitive, tight market), while a lower rate disperses orders deeper into the book (simulating a "flatter" liquidity profile).

3. **Market Maker (MM) Base Spread:** This defines the initial distance at which the Market Maker posts their bid and ask quotes relative to the mid-price. It represents the Market Maker's baseline risk premium for providing liquidity before any adjustments for inventory or volatility are applied.

Component	Parameter Description	Levels
Order Strategy	Mean submitted order size	{1, 5, 10, 20, 30}
Market Structure	Decay rate of limit-price distribution	{0.05, 0.10, 0.20, 0.30, 0.50}
MM Strategy	Base half-spread of market-maker quotes	{0.10, 0.25, 0.50, 1.00, 2.00}

These factors were selected to span regimes of practical interest:

- Small order-size means represent highly granular, retail-like trading, while larger means create environments with deeper but more volatile liquidity.

- Small decay rates produce wide books with orders dispersed far from the midprice, whereas large decay rates produce very tight books with intense competition for queue priority.
- The market maker's base spread determines the aggressiveness of its liquidity provision strategy, directly affecting spreads, fill rates, and profitability.

The full factorial grid permits the isolation of main effects and interactions, enabling a systematic study of how structural market features interact with strategic quoting behaviour.

3.3 Replication Design

To obtain reliable statistical estimates, each scenario is simulated with five independent replications. This ensures that scenario-level performance metrics (means and confidence intervals) are supported by independent samples rather than single-run outcomes.

3.4 Common Random Numbers (CRN)

A common random numbers design is employed across the experiment: the same sequence of background randomness is used across all scenarios within a given replication. That is, replication r uses a shared random-number seed for all 125 scenarios. This induces strong pairing between scenarios, sharply reducing variance in scenario comparisons and yielding more sensitive detection of factor effects.

4 Methodology

The simulation is implemented as a continuous-time, discrete-event model in which all market activity is generated endogenously by stochastic event processes and by the quoting behaviour of the market maker. This section describes the structure of the simulation engine, the flow of events, and the procedures used to generate limit prices, track executions, and collect performance statistics. The description corresponds directly to the logic executed in the underlying code, but is presented in conceptual rather than implementation-specific terms.

4.1 Discrete-Event Flow

The simulation maintains a global event clock and advances time by processing whichever event is scheduled to occur next. At the start of each replication, the clock is set to zero, the order book is initialized with a minimal buy and sell order to establish an initial midprice, and candidate arrival times are generated for all stochastic event streams. The simulation then proceeds according to the following loop:

1. **Identify the next event.** Each stochastic event type (limit buys, limit sells, market buys, market sells, cancellations) has an associated next-arrival time generated from an exponential distribution. A separate deterministic timer tracks the next scheduled quoting action of the market maker. The simulation selects the smallest of these times and advances the global clock to that moment.

2. **Execute the event.** Depending on the event type, one of the following actions takes place:

   - *Limit order arrival.* A new limit buy or limit sell order is created. Its limit price is generated by sampling an exponentially distributed distance from the current midprice. The sampled distance is rounded to the nearest tick and applied on the appropriate side of the book (below the midprice for buys, above for sells). The order is inserted into the priority queue corresponding to its side.
   - *Market order arrival.* A market order triggers immediate execution against the opposite side of the book. The simulation repeatedly removes one unit from the best available quote until either the market order's required quantity is fully executed or no resting liquidity remains. Each one-unit execution is recorded as an individual trade, and the state of the limit order book is updated accordingly.
   - *Cancellation.* A random side (buy or sell) is chosen, and the current best order on that side is cancelled. Cancelling only the top-of-book order simplifies the abandonment mechanism while still capturing realistic turnover of stale liquidity.
   - *Market-maker quoting.* The market maker observes the best available buy and sell prices and computes the midprice. Based on this midprice, the configured base spread, and an inventory-dependent skew term, the market maker posts new one-unit buy and sell orders. These orders enter the book and are thereafter subject to the same execution and cancellation rules as all other resting orders.

   After executing any event, a new next-arrival time is drawn for that event type.

3. **Clear crossed markets.** After processing an event, the simulation checks whether the best buy price is greater than or equal to the best sell price. If so, the book is crossed, and the simulation executes one unit of a trade between the best bid and best ask. This procedure continues in a loop until the book becomes uncrossed. Each trade updates order quantities, removes inactive orders, adjusts market-maker inventory and cash (if the MM participated), and records data needed for execution-time and P&L calculations.

4. **Repeat until horizon is reached.** The event loop continues until the simulation clock reaches the fixed horizon of 23,400 seconds.

This event-based structure ensures that the simulation captures the precise timing, sequencing, and interdependence of order-flow events, executions, and market-maker actions, all while maintaining computational efficiency.

4.2 Generation of Limit Prices

Whenever a new limit order arrives, its price is determined by an exponential distance model reflecting how traders submit orders relative to the current midprice. Specifically:

1. Compute the current midprice as the midpoint of the best standing buy and sell prices.
2. Sample an exponential random variable with rate equal to the scenario's *limit-price decay parameter*. Smaller rates generate larger distances and more dispersed books, while larger rates concentrate orders near the midprice.
3. Convert this distance into integer ticks (based on a tick size of 0.01) and apply it on the appropriate side of the midprice.

This mechanism ensures that shifts in the decay parameter directly affect order placement behaviour and, consequently, spreads, depth, and volatility.

4.3 Tracking Executions

The simulation records detailed information about executions:

- Each one-unit trade is logged with its price, time, and whether the market maker was involved.
- For limit orders, execution time is recorded as the difference between creation time and completion time.
- Aggregate statistics such as total trades, executions per hour, and depth at the end of the run are updated incrementally throughout the simulation.

Market-maker inventory and cash are updated in real time during every execution in which the MM participates, allowing accurate computation of mark-to-market P&L at the end of the run.

4.4 Collection of Performance Metrics

At the end of each simulation run, a comprehensive set of performance metrics is computed and stored:

- average bid–ask spread over the entire run,
- mean, median, and 99th percentile execution times,

- trade counts and execution rates,
- midprice volatility based on sampled returns,
- depth within fixed tick bands around the midprice,
- market-maker inventory statistics and final mark-to-market P&L,
- total wall-clock runtime of the simulation.

Each individual run outputs a complete row containing these statistics. With five replications per scenario, these data support the estimation of means and confidence intervals for all performance measures at the scenario level.

4.5 Rationale

The methodology is designed to isolate how the structural parameters; order-size distribution, limit-price dispersion, and market-maker quoting aggressiveness, affect market quality and liquidity provision. The discrete-event structure ensures that all dynamics are captured at high temporal resolution, while the factorial design and CRN replication strategy provide the statistical power and variance reduction necessary for reliable comparison of scenarios.

5 Analysis

This section summarizes the empirical behaviour of the simulated limit order book across the $5 \times 5 \times 5 = 125$ scenarios defined by the three structural factors:
(i) the mean submitted order size `order_size_mean`,
(ii) the decay rate of the limit-price distribution `limit_price_decay_rate`, and
(iii) the market maker's base half-spread `mm_base_spread`.
For each scenario, we ran 5 independent replications using distinct random seeds and recorded a rich set of performance metrics per run. We then computed scenario-level point estimates and 95% confidence intervals based on the cross-replication data.

5.1 Metrics and Aggregation Procedure

At the end of each replication the simulator records a set of scalar summary statistics. The scenario-level dataset used in the analysis consists of the following variables:

- the structural factor levels (`order_size_mean`, `limit_price_decay_rate`, `mm_base_spread`);
- the number of replications n_{rep} (equal to 5 in all scenarios);
- *market-quality metrics*: the average bid–ask spread `avg_spread` and median execution time `median_exec_time`;

- *market-maker metrics*: final mark-to-market profit and loss per 1,000 trades `mm_final_pnl_per_1k_trades`;

For each scalar metric X and each fixed scenario we denote the replication-level values by $X_1, \ldots, X_R$. We form the usual sample mean

$$\hat{\mu} \;=\; \frac{1}{R}\sum_{r=1}^{R} X_r, \qquad \hat{\sigma}^2 \;=\; \frac{1}{R-1}\sum_{r=1}^{R}(X_r - \hat{\mu})^2,$$

and report a 95% confidence interval of the form

$$\hat{\mu} \,\pm\, t_{0.975,\,R-1}\,\frac{\hat{\sigma}}{\sqrt{R}},$$

where $t_{0.975,\,R-1}$ is the 97.5th percentile of the t distribution with $R-1=4$ degrees of freedom. In the scenario-level dataset, these appear as triplets of columns with suffixes `_mean`, `_CI95_lower`, and `_CI95_upper` (e.g., `avg_spread_mean`, `avg_spread_CI95_lower`, `avg_spread_CI95_upper`).

The resulting ranges give a sense of scale. Across the 125 scenarios, the average spread (`avg_spread_mean`) varies from roughly 0.35 to 4.7 price units, and median execution time (`median_exec_time`) spans about roughly three orders of magnitude. Market-maker profit per 1,000 trades (`mm_final_pnl_per_1k_trades_mean`) ranges from single-digit values up to nearly 700. At the same time, the 95% confidence intervals are generally tight relative to these between-scenario differences: for `avg_spread`, typical half-widths are only a few percent of the corresponding mean, and even for the noisier profit metric the half-widths are small compared with the spread of values across the design. Thus the qualitative patterns reported in the following subsections are not driven by Monte Carlo noise.

As an additional diagnostic on the simulation and the use of random seeds, we examined replication-level boxplots for several key metrics across seeds, pooling over all scenarios. In particular, we plotted the distributions of `avg_spread`, `median_exec_time`, and the market maker's raw final profit and loss `mm_final_pnl` by seed. The resulting boxplots (Figures 1–3) show comparable dispersion across seeds and no outlying seeds with systematically different behaviour. This provides a sanity check that the random-number streams are being used consistently and that no single seed is unduly influencing the results.

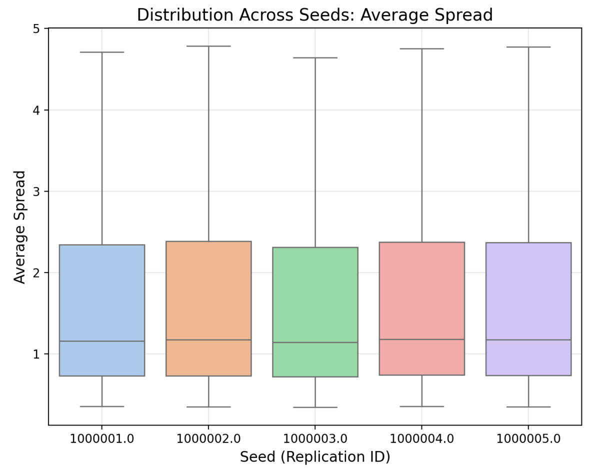

Figure 1: Distribution of replication-level average spread `avg_spread` across seeds.

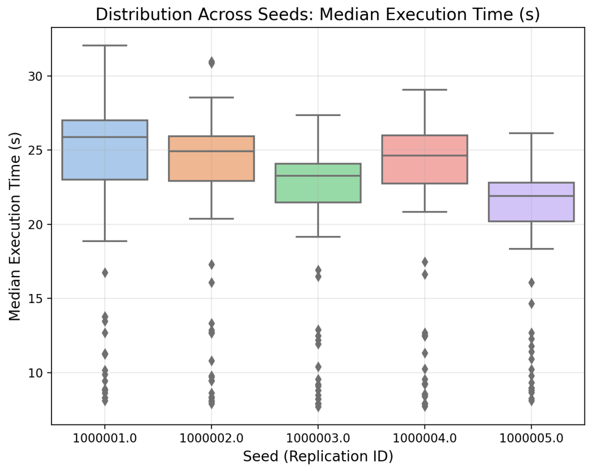

Figure 2: Distribution of replication-level mean execution time `mean_exec_time` across seeds.

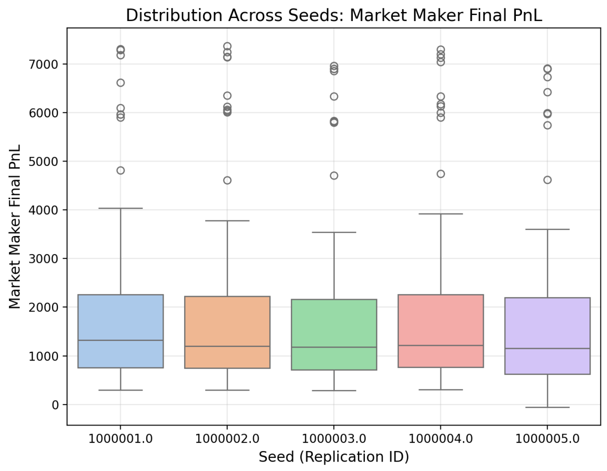

Figure 3: Distribution of replication-level market-maker final P&L `mm_final_pnl` across seeds.

5.2 Effects on Market Quality

This section analyzes how the three experimental factors `order_size_mean`, `median_exec_time`, and `mm_base_spread` shape the core market-quality outcomes:

5.2.1 Average Spread

The average bid–ask spread varies substantially across the 125 scenarios, with scenario-level means ranging from well under one price unit in the tightest configurations to nearly four units in the most dispersed books. The factor-level effects show a clear hierarchy of influence, driven primarily by the limit-price decay rate, followed by the market-maker's quoting width, and finally the mean submitted order size.

The decay parameter is the dominant driver of market tightness. Averaging over the remaining factors, the mean spread decreases monotonically from 3.94 at decay rate 0.05 to 2.15 at 0.10, 1.20 at 0.20, 0.86 at 0.30, and 0.55 at 0.50. This more-than-sevenfold reduction highlights the decisive role of order-book geometry: stronger decay concentrates liquidity near the midprice and compresses spreads accordingly. The corresponding main-effect curve is shown in Figure 4.

Wider market-maker quotes shift the spread upward across all decay regimes. The factor-averaged means rise from 1.51 at `mm_base_spread` = 0.10 to 1.56 at 0.25, 1.67 at 0.50, 1.85 at 1.00, and 2.12 at 2.00. These differences are modest compared with the decay-rate effect but remain quantitatively meaningful, indicating that quote width contributes an additive component to the observed spread.

Changes in `order_size_mean` have the smallest influence. The factor-averaged spread increases only from 1.51 at mean 1 to 1.81 at mean 30, with nearly identical values at means 10, 20, and 30. Most of the effect is realized when moving from very small to moderately sized orders, after which the response saturates. This behaviour is visible in the left panel of Figure 4.

Interaction surfaces reveal that the decay rate governs the global structure of the response, while the other two factors act primarily as vertical shifts. The strongest interaction occurs between the decay rate and the market-maker base spread (Figure 5): even for narrow market-maker quotes, scenarios with low decay rates exhibit spreads above 3, whereas at high decay rates all combinations fall below 1. Interactions involving the order-size mean are comparatively weak, consistent with its small main effect.

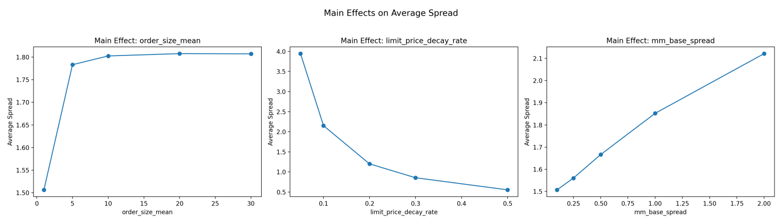

Figure 4: Main effects of the three structural factors on the average spread.

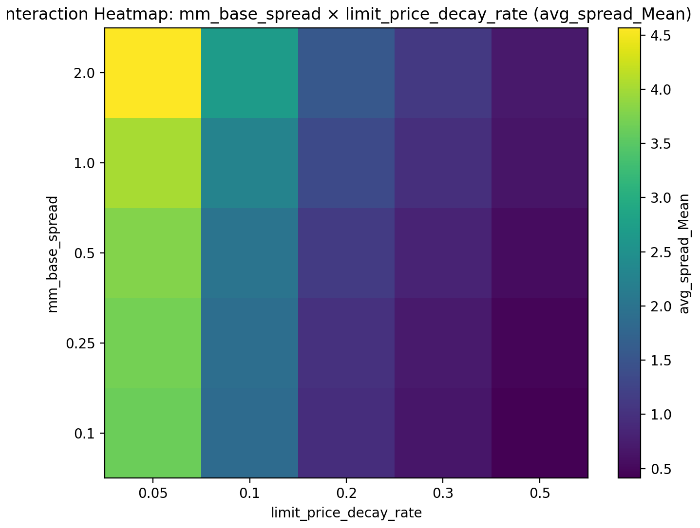

Figure 5: Interaction between limit-price decay rate and MM base spread for the average spread.

5.2.2 Median Execution Time

Median execution time reflects how long an order typically remains in the book before completion. Across all 125 scenarios, the scenario-level means range from approximately 15 to 26 time units, with most variation driven by the mean submitted order size and the limit-price decay rate. The overall range is modest compared with that of the spread, but the factor effects are systematic and quantitatively meaningful.

The mean submitted order size is the strongest driver of execution times. Averaging over all decay-rate and market-maker settings, median execution time rises from 15.37 at `order_size_mean` = 1 to 22.31 at 5, 23.96 at 10, 24.87 at 20, and 25.31 at 30. The response is monotone and displays clear diminishing returns: most of the increase occurs between order-size means 1 and 10, after which execution times grow slowly. This main-effect pattern is shown in Figure 6.

Execution times also increase with stronger decay, although the effect is milder than that of order size. The factor-averaged mean rises from 20.67 at decay rate 0.05 to 21.52 at 0.10, 22.71 at 0.20, 23.19 at 0.30, and 23.74 at 0.50. Higher decay compresses the book toward the midprice, leading to more competition for queue priority and slightly longer waiting times for execution.

The market-maker's quoting width exerts the weakest influence among the three factors, but its effect is still monotonic. The factor-averaged medians increase from 20.86 at `mm_base_spread` = 0.10 to 21.57 at 0.25, 22.38 at 0.50, 23.41 at 1.00, and 23.60 at 2.00. The effect size is small relative to that of order size, reflecting that execution times are driven primarily by the depth and congestion of liquidity, not by the width of the market-maker's quotes.

Interaction surfaces show that median execution times are largely additive in this design, with no large nonlinear effects. The most visible interaction occurs between order size and decay rate, where larger orders amplify the upward trend induced by stronger decay. However, these interaction terms remain modest compared with the pronounced main effect of order size. Interactions involving the market-maker base spread are weak.

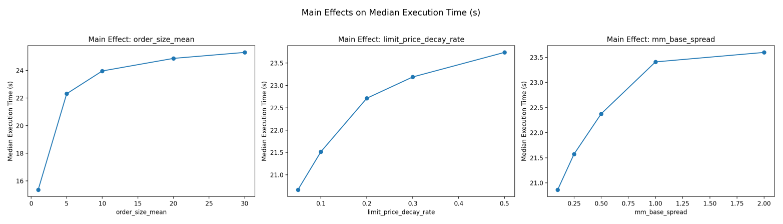

Figure 6: Main effects of the three structural factors on the median execution time.

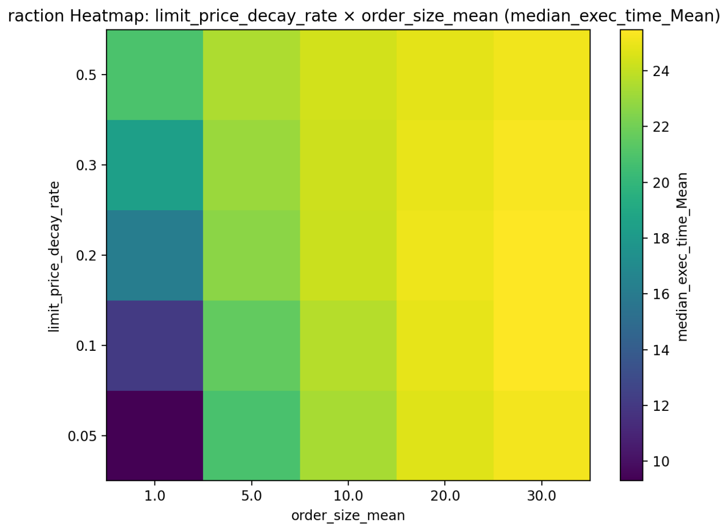

Figure 7: Interaction between limit-price decay rate and mean order size for the median execution time.

5.2.3 Market-Maker Profit per 1,000 Trades

The market maker's realized profit per 1,000 trades exhibits substantial variation across the design, with scenario-level means spanning more than two orders of magnitude. The factor effects reflect a combination of spread capture opportunities, inventory dynamics, and execution frequency. Unlike the average spread and execution-time metrics, all three experimental factors exert strong and nontrivial influence on the profit response surface.

The mean submitted order size is the single most influential determinant of market-maker profitability. Averaged over all decay-rate and quoting-width settings, the mean profit per 1,000 trades declines sharply from 167.85 at `order_size_mean` = 1 to 39.96 at 5, 20.25 at 10, 10.08 at 20, and 6.75 at 30. The response is strongly nonlinear: the transition from very small to moderately large orders produces a steep reduction in profitability, after which the curve flattens. This pattern is consistent with larger orders increasing exposure to adverse price movements and amplifying inventory risk.

Profitability also decreases systematically with stronger decay. The factor-averaged mean falls from 79.03 at decay rate 0.05 to 66.90 at 0.10, 46.58 at 0.20, 33.48 at 0.30, and 18.91 at 0.50. Tighter books restrict spread-capture opportunities and lead to more frequent inventory cycling, reducing overall gains. The decay-rate response is smoother and more linear than the order-size effect, reflecting a gradual erosion of profitable quoting conditions as order flow concentrates near the midprice.

The market-maker's quoting width exerts a large and monotonic effect on profits, reflecting the direct tradeoff between execution frequency and per-trade revenue. Averaging over the other factors, mean profit increases from 10.16 at `mm_base_spread` = 0.10 to 24.15 at 0.25, 43.66 at 0.50, 70.53 at 1.00, and 96.40 at 2.00. These substantial gains indicate that wider quotes compensate for lower fill rates by capturing more revenue per execution, though the effect interacts strongly with order size and decay conditions.

The interaction surfaces reveal a complex, highly nonlinear response shaped by all three factors. The most pronounced interaction occurs between order size and market-maker base spread: increasing the quoting width substantially boosts profits only when mean order sizes are small, with the benefit diminishing sharply as orders grow larger. A secondary interaction appears between decay rate and base spread, where wide quotes are profitable under low-decay conditions but yield much smaller gains when the book is tightly concentrated. These effects are illustrated in Figures 8 and 9, which highlight that market-maker profitability is jointly governed by order-flow granularity, liquidity distribution, and quoting aggressiveness.

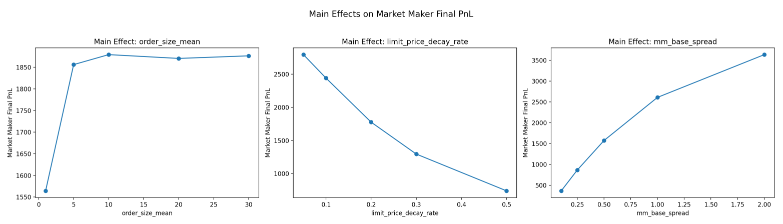

Figure 8: Main effects of the three structural factors on market-maker profit per 1,000 trades.

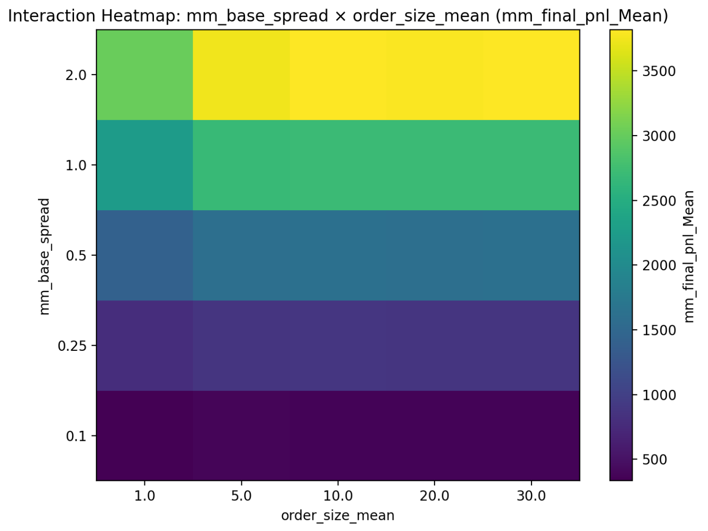

Figure 9: Interaction between market-maker base spread and mean order size for the market-maker profit per 1,000 trades.

6 Conclusion

This study developed and analyzed a continuous-time, discrete-event simulation of a single-asset limit order book with a quoting market maker. The model represents the bid and ask sides as priority queues implementing price–time priority, generates order flow through independent Poisson processes, and applies an inventory-sensitive quoting rule for the market maker. We conducted a full $5 \times 5 \times 5$ factorial experiment over three structural components of the simulated market: the mean submitted order size, the decay rate governing the distribution of limit-order prices, and the market maker's base half-spread. For each of the resulting 125 scenarios, five independent replications were run under a common-random-numbers scheme, and scenario-level means and 95% confidence intervals were computed for the key metrics.

The results show that the *limit-price decay rate* is the primary driver of overall market tightness. Across the design, average spreads fall from nearly four price units in low-decay environments to well below one price unit at high decay. This reflects the direct link between decay strength and the concentration of limit orders around the midprice.

For *median execution time*, the dominant factor is the *mean submitted order size.* Execution times increase monotonically with order size, with most of the rise occurring between mean sizes of 1 and 10. Higher decay rates also lengthen execution times modestly, while the market-maker base spread has only a small influence.

The *market maker's profit per* 1,000 *trades* depends strongly on all three factors. Profits decrease sharply with larger mean order sizes and also decline as the decay rate increases. In contrast, widening the market-maker base spread increases profits on average. The associated interaction surfaces reveal that wide quotes are most effective when order sizes are small and limit prices are more dispersed.

Confidence intervals for all metrics are narrow relative to the differences across scenarios, and replication-level boxplots show no anomalous seed behaviour. This indicates that the factorial comparisons are reliable and not driven by Monte Carlo variability.

Overall, the experiment highlights clear structural trade-offs in this stylized market. High decay rates produce tight spreads but reduce market-maker profits; large order sizes lengthen execution times and increase market-maker risk; and wide quoting spreads improve profitability at the cost of wider markets. Within this design, balanced regimes typically involve moderate to high decay rates, small to moderate mean order sizes, and intermediate quoting widths. These conclusions demonstrate how discrete-event simulations with explicit LOB structure can be used to explore how order-flow parameters and quoting behaviour jointly shape market quality and liquidity provision.

Appendix A: Code Repository

The full simulation codebase used in this study is available at the following GitHub repository:

`github.com/n4m3name/CSC446-Simulation`

This repository contains all scripts for:

- the limit order book simulation engine,
- scenario generation and replication control,
- data aggregation and confidence interval computation,
- figure generation,
- and reproducibility artifacts (e.g., random seed configuration).

Appendix B: 95% Confidence Interval Tables for Primary Metrics

Table 1: Average Spread

Scenario	Lower 95% CI	Upper 95% CI
S1	2.93678	3.02487
S2	3.00622	3.08047
S3	3.09528	3.14064
S4	3.38121	3.42679
S5	3.96404	4.0306
...	...	...
S121	0.42531	0.433911
S122	0.481095	0.491989
S123	0.563861	0.572033
S124	0.641248	0.653139
S125	0.681768	0.699942

Table 2: Market Maker Profit & Loss

Scenario	Lower 95% CI	Upper 95% CI
S1	32.4387	47.1176
S2	81.6958	105.87
S3	180.272	202.278
S4	360.822	387.76
S5	676.499	701.842
...	...	...
S121	1.04909	1.20533
S122	2.59229	2.74615
S123	4.04825	4.2999
S124	4.25351	4.52094
S125	1.3753	2.10697

Table 3: Median Execution Time

Scenario	Lower 95% CI	Upper 95% CI
S1	7.751	8.11981
S2	7.91883	8.4577
S3	8.27294	8.69959
S4	9.165	9.44409
S5	11.9344	13.279
...	...	...
S121	24.9831	28.1931
S122	24.5035	27.8622
S123	24.2356	26.7609
S124	23.0394	26.0164
S125	21.3192	24.3704

To see the full list of every confidence interval, see the github folder "data."