On Market Making for a new exchange

Arthur Bagourd

January 2026

1 Context

You work as a quant trader at a reputable algorithmic FX market-making firm. A new exchange A has approached your firm as they are looking for a Designated Market Maker to bootstrap the liquidity on the exchange.

Your task is to develop a market making model to quote a bid/ask spread and hedge risk appropriately during the 3 phases of the deal.

1.1 Phase 1: Bootstrapping the liquidity without any trading history

Assume you will only be quoting EUR/USD Spot and there exist 2 other exchanges: B and C where B has on average 75% of the total volume and C 25% of the volume; these are the primary markets where price discovery occurs. You are not market making on exchanges B and C. You are the sole market maker on exchange A during phases 1 and 2 and your goal is to bootstrap liquidity as well as you can, not to quote the tighter spreads possible (though they still need to be sensible). You have committed to always quote both sides of the orderbook. You have to build a model for the day the exchange A goes live, you have 0 historical data about the traders you will be facing on that exchange or their behavior. You therefore cannot build a model that relies on empirical data and have to rely on heuristics. You are given a starting capital K which is half in EUR and half in USD. You are not the fastest market maker on the planet, it takes you 200ms to get data from exchange B and 170ms to get data from exchange C but HFTs can get it in 50ms and can use that information to trade on exchange A. Assume the same tick size of 1bp is used across all exchanges. You are asked to quote prices for 10 levels on each side. You have a delta risk limit of 90% after which you need to externalise your risk and hedge with market orders on other venues.
The fee structure is:

- market A: taker: 0.04%, maker: 0.01%
- market B: taker: 0.02%, maker: 0.009%
- market C: taker: 0.03%, maker: 0.009%

Simulate a backtest over 1 more month of data.

1.2 Phase 2: Bootstrapping with some history

A month has passed since you started phase 1, everything from phase 1 still holds true except that now have gathered some trading history about your fills, other traders' trades and orders on the exchange. You can use that data to build calibrate models. Your capital is reset to the initial K (with the same split). Simulate a backtest over 1 more month of data.

1.3 Phase 3: Fallback liquidity system

Another month has passed, HFT market makers are now aggressively quoting (meaning tight spreads) in the orderbook, these are HFTs that have access to the market data before you do. Assuming normal market circumstances, most orders should be filled by these aggressive market makers, while big orders get to be filled partly by them and partly by you. Your goal is not to compete with them on making money, but rather to act as a fallback liquidity provider should there be high volatility, big events and other market makers going one-sided or off (eg: their servers crash, the US actually invades Greenland, or the start of WW3). Your capital is reset to the initial K (with the same split). Simulate a backtest over 1 more month of data.

Step-by-step tasks

The following steps provide a structured guide for building and evaluating the market-making system:

1. Design a class representing an order book with n price levels on each side. Each level must contain a price and quantity for both bid and ask. Justify the choice of data structure.

2. Implement a simulator for markets B and C over time. Choose an appropriate approach (e.g. simulating order arrivals on an explicit order book, or generating a mid-price process with a synthetic order book around it, etc).

3. Implement the market-making logic for exchange A. Given the current states of the three markets and any additional relevant state variables, the function should output:

   - a list of orders to submit,
   - a list of existing orders to cancel.

4. Apply order submissions and cancellations to update the order book state.

5. Advance the simulation to the next time step and repeat the process.

6. Track inventory and P&L over time. Implement functions to compute:
   - MtM P&L for trades,
   - total realized P&L at the end of the period,
   - final inventory value.

   Clearly distinguish realized P&L from mark-to-market valuation, and inception spread from mid-to-mid revaluation effects.

7. Generate a backtesting report (in addition to the written paper). At a minimum, for each phase, include:
   - best bid, best ask, and mid-price over time for the three markets,
   - bid, ask, and mid-price evolution for each market, along with the top 10 trades by size,
   - EUR/USD price evolution alongside inventory over time,
   - average, median, 5th and 95th percentile mark-to-market P&L as a function of time since trade inception,
   - fill-rate analysis.

   This report should help you check that your trading logic behaves as you expect. Selected figures from this report may be reused and interpreted in the written paper.

8. **Phase 2**: Use the data generated during the Phase 1 backtest to calibrate and refine the models.

9. **Phase 3**: Analyze the impact of HFT market makers on exchange A. Compare outcomes for small versus large trades, and study scenarios involving volatility spikes or HFTs going one-sided or offline. Discuss the implications for both your P&L and exchange A customers.

2 Submission Instructions

Students are required to submit the following by **21st April 2026**:

- A written report explaining their thought process, research, formulas, and findings (interpretation) at every step. Support your work with the existing academic literature. This report should detail the models implemented, design decisions made, and any challenge faced.
- A complete python codebase hosted in a private GitHub repository. No jupyter notebook. It should include docs about how to run the code and generate a trading report. Invite (`@IlluvatarEru`) to the repository for evaluation.

- Should you have any question, require clarification or guidance, contact me by email at: `arthur@bagourd.com`.

All submissions should be made on Moodle.
NB: Ensure the report and code are well-documented and organized.