On Market Making an Illiquid Currency

Arthur Bagourd

November 2024

1 Context

You work as a quant trader at a reputable algorithmic market-making firm. The central bank of a small emerging market country, whose currency is not tradable on international markets, has approached your firm. They request that you create an OTC market for their currency against USD. Since no external market exists for this currency, the central bank has provided you with a significant percentage of the currency's total supply.

Your task is to develop a model to quote a bid/ask spread and hedge risk appropriately. The firm's research analysts have provided a fair price, denoted as p_t^f, and the central bank expects you to start quoting around this price.

2 Base Scenario

Given the above scenario, your objective is to design a pricing model based on the fair price time series p_t^f. The pricing model should resemble an order book, quoting prices up to $10 million in notional value with the following step sizes: $100k,$100k,$100k,$100k,$100k, $500k, $1M,$1M,$1M,$1M,$1M,$1M,$1M,$1M,$1M.

When a client trades with you, you will take on a risk position. You must build a model that skews your prices dynamically to manage this risk.

Step-by-step tasks:

1. Create a function to generate the fair price time series:

```
def simulate_fair_price(start_timestamp: datetime,
                        end_timestamp: datetime,
                        step_size_in_seconds: int,
                        <other parameters as you
                            see fit>):
```

   Research appropriate models for this simulation.

2. Design a class to represent an order book with `n_levels` levels. Each level should have a price and size on both bid and ask sides.

3. Implement a function to create the initial order book:

```
def get_base_pricing(<parameters>):
```

   This function should use the fair price and other parameters to define bid and ask prices and sizes that you are quoting.

4. Create a `Trade` class to handle trade logic:

```
class Trade:
    size: float
    side: str  # "Buy" or "Sell"

    def update_orderbook_with_trade(orderbook,
        trade):
        # Modify orderbook levels based on trade
```

5. Define two order types:

   (a) `MarketOrder` class:

```
class MarketOrder:
    size: float
    side: str

def post_market_order(orderbook, client_order):
    # Updates orderbook with new order
```

   (b) `LimitOrder` class (optional for this project):

```
class LimitOrder(MarketOrder):
    limit_price: float

def post_limit_order(orderbook, client_order):
    # Updates orderbook with new order
```

6. Simulate random client trades and update the order book accordingly.

7. Track your inventory and P&L over time. Implement a function to calculate:

- Total realized P&L at the end of the time period.
- Final inventory value.

Consider the distinction between realized P&L and mark-to-market valuation.

8. Develop a price-skewing model to adjust your base pricing dynamically in response to inventory and risk.
9. Resimulate the P&L over the time period using the adjusted pricing model.

Project Goals

1. Design a sound pricing model for market making
2. Maximize profitability while adhering to market-making principles

NB: The Volcker Rule prohibiting proprietary trading does not apply to FX spot trading.

3 Scenario with a Call Option

Building on the base scenario, assume the central bank is incentivized to increase the value of their currency. To align interests, they provide you with a free out-of-the-money (OTM) call option with a strike price $K = 1.2\ p^f_{t_0}$ (20% OTM).

Question: Modify your pricing and risk models to account for the call option.

4 Scenario with External Markets

Consider the base scenario again, but assume no call option exists. Instead, m external markets now provide observable order books (OBs), which you can use to infer a fair price (don't use the price provided by the research analysts anymore).

Tasks:

1. Simulate m external markets and their order books.
2. Use these external markets to derive the fair price, instead of relying on the research analyst's estimates.
3. Adjust your order book and trading strategies based on this external information.

5 Submission Instructions

Students are required to submit the following by **22rd April 2025**:

- A written report explaining their thought process, research, and findings. This report should detail the models implemented, design decisions made, and any challenges faced.
- A complete python codebase hosted in a private GitHub repository. Invite (`@IlluvatarEru`) to the repository for evaluation.

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