Engineering Leader Project

Brief Description

Design and implement a basic autohedger for a Spot Bitcoin and Ethereum trading business.

Requirements

1. Take in a time series of BTC-USD and ETH-USD trades.
2. Provide an API to return positions as-of any given time in the time series.
3. An API already exists that will execute hedging trades.
4. The process is standalone, with no human interaction or oversight necessary.
5. The goal of the process is risk reduction of client flow.

Assumptions

1. Assume that the overarching goal of the hedger is to return the book to as flat as possible as quickly as possible, subject to other reasonable considerations.
2. Assume that there can be an adverse market impact to submitting large hedging orders to exchanges, so break large hedges into smaller chunks in certain situations.
3. Assume that a goal is to minimize transaction costs by not hitting the market to trade after every client trade regardless of size. Set a maximum client volume that should be reached during a specified time window before hedging.

Specifications

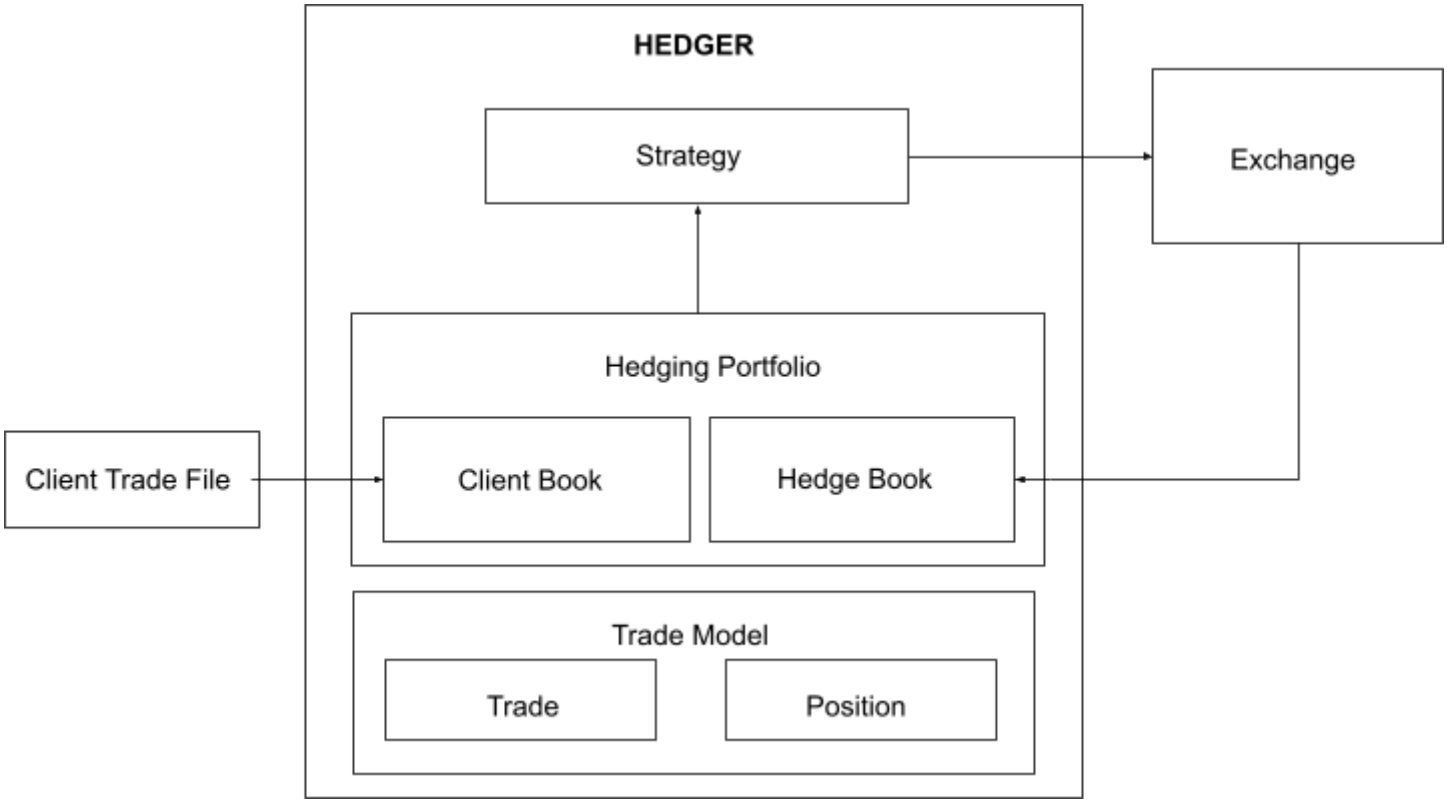
To keep the process simple, it will complete the following steps and then stop, rather than being a server process. This simple model could be easily converted.

1. Load list of client trades into memory.
2. Build a list of hedging trades according to the hedging strategies below.
3. Export a list of hedging trades executed.

Hedging Strategies

- Incrementally hedge in the market to reduce the impact of large orders moving the market adversely.
- Reduce transaction costs by avoiding executing a hedge for every client trade. Group hedging together subject to some reasonable constraints.
- Avoid warehousing a large amount of risk by changing order sizes according to changing evolving client activity.
- Use one of three different modes:
    a. SLOW: When there has not been limited client activity within a recent window, hold off on hedging to minimize transaction costs. Wait until total client volume within a given window reaches a certain size to hedge. The hedger will stay in SLOW mode and either switch to NORMAL or STEALTH, then the risk will be hedged.
    b. NORMAL: A simple strategy which will hedge the net total quantity for each asset. This reduces transaction costs versus hedging every individual trade.
    c. STEALTH: If we receive an risk from client trades above a certain size in the latest batch then we can do some more careful hedging in the market by chunking up the position over time.

Design Sketch



Possible Extensions

Given the limited time to implement, there are a number of improvements that could be made to this process. Here are some ideas on how the hedger could be extended.

- Adapt to work on a “push” basis where each client trade triggers a re-assessment of the hedger’s stance and next actions. Iterating through the client trade file in time could be replaced by an interrupt-driven mechanism that gets triggered when a client trade happens.
- Calculate a “hedge cost” for each client trade executed, in order to calculate an expected PNL by which to measure the effectiveness of the hedger itself.
- Greater sophistication determining which strategy to use at any given time, and how to execute. For example, moving averages of position size could be used to make the strategies more responsive.
- Support for multiple portfolios with different parameters for hedging. This could be easily added by building on top of the current implementation.
- Support beyond Bitcoin and Ethereum as assets. This would be an easy change following current design.
- Support Crypto / Crypto hedging such as BTC-ETH in case there is an advantage of moving risk between Crypto assets instead of hedging versus FIAT currencies. This would require some minor changes.
- Support non-USD FIAT currencies and Foreign Exchange hedging also. The Trade class supports denominated (e.g. USD/EUR) in addition to quantity unit (e.g. BTC or ETH), so trade booking would need to be adapted to enter the crypto and FIAT legs of the trade into different positions - e.g. one for BTC, one for USD.
- Creating an additional “Strategy” class along with a helper functions class would cut down the size of the Hedger class.