Creating a robust Transaction Cost Analysis (TCA) framework for American equity options

Vinh Nguyen

July 8, 2024

1 Introduction

Transaction Cost Analysis (TCA) is a process used to measure and evaluate the costs associated with trading securities. These costs can include explicit costs (like commissions and fees) and implicit costs (such as market impact and slippage). Lower liquidity generally increases transaction costs, as larger trades are more difficult to execute without moving the market. Hence incorporating liquidity factors into TCA provides a deeper understanding of the cost and efficiency of trades. We chose stock GME (GameStop) for TCA due to its historical volatility and variability in liquidity, which can provide rich data for machine learning and optimization applications. The associated implementation with this note includes the following content:

- Download and Preprocess Data
- Key Metrics in TCA
- Visualization
- Define objective function minimizing it

While other things are common are explained clearly in the code, we focus more on the part where we built the transaction cost as a cost function and minimize it.

2 Transaction Cost as an objective function

We define the transaction cost as a function of:

- variables that can be calculated from market data, change over time, and affect transaction costs include Trade Size (the number of shares we intend to buy), Liquidity (ease of buying or selling in the market), and Market Volatility (measure of price fluctuations) ,
- with parameters $\beta(t), \gamma(t), \delta(t), \epsilon(t)$ to be optimized, and α as a large enough constant to ensure that the transaction cost is positive, for a natural and logical discussion of cost

$$TransactionCost = \alpha + \beta(t).TradeSize + \gamma(t).TradeSize^2 + \delta(t).Liquidity + \epsilon(t).Votality \quad (1)$$

where the trade size is defined as above, Liquidity is chosen to be the Amihud Illiquidity Measure, a higher values indicate a lower liquidity.

We choose spline fitting to capture non-linear and smooth variations over time for the following parameters. Main steps involving in the code are:

- Initial Splines: Fit initial spline models using random values.
- Extract Control Points: Extract the control points from these splines for optimization.
- Optimize Control Points: Optimize the control points directly, ensuring the correct length of control points.
- Update Spline Models: Use the optimized control points to update the spline models.
- Predict Time-Varying Parameters: Use the updated spline models to predict the parameters over time.

And then we used Library Scipy to find optimized parameters.

3 Set up the experiment and interpret the obtained results.

The purpose of this experiment section and the provided code is to understand the transaction cost for buying a number of shares of GME (data['Trade Size'] = np.random.randint(100, 2000, size=len(data))), with all variables such as Liquidity (Amihud Illiquidity Measure) and Market Volatility (the rolling standard deviation of daily returns over a 20-day window) varying over time in 2023. The obtained results are visualized in the associated code, showing how the cost and parameters $\beta(t), \gamma(t), \delta(t), \epsilon(t)$ change over time.

4 Next steps to improve the current model

Consider incorporating more factors that affect transaction costs, such as:

- Incorporating more factors such as Bid-ask spread,Market impact models
- Allow parameters to dynamically change based on different market conditions, not just over time.
- Validate the model using a separate dataset and perform backtesting to ensure the model performs well on unseen data.
- Integrate the model into trading systems for real-time usage.
- Build better transaction cost function.