Blockhouse Work Trial Task

Name : Achyuth Kumar Miryala

Question 1: How I Chose to Model the Temporary Impact Function

1. My Goal and Approach

For this task, my main goal was to figure out the best way to model the temporary price impact, which the task calls $g_t(x)$. This is basically the "slippage" or extra cost you pay when you place a market order of size **x**. To do this, I used the order book data for the three tickers provided: **FROG**, **SOUN**, and **CRWV**.

My process was pretty straightforward:

- First, I loaded all the data you provided. Since there was a lot of it, I decided to sample the data to get a good mix of market conditions without being too slow to analyze.
- Next, for each of these snapshots, I simulated what would happen if I placed market orders of different sizes, from 1 share all the way up to 200 shares.
- For each simulated trade, I calculated the slippage. As the PDF explained, this is the difference between my average execution price and the mid-price of the stock right before the trade.
- Finally, with all this data, I had a clear picture of how order size affects slippage. I then tried to fit a few different mathematical models to this data to see which one described it best.

2. The Models I Tested

I decided to test three common models for market impact:

- **Linear Model ($g(x) = \beta x + \alpha$)**: This is the simplest model. It basically says that for every additional share you trade, the cost goes up by a constant amount (**β**). The **α** part is like a fixed fee for trading, related to the bid-ask spread.
- **Square-Root Model ($g(x) = \beta\sqrt{x} + c$)**: This model suggests that the impact grows more slowly as your order size gets bigger.
- **Power-Law Model ($g(x) = \beta x^{\alpha}$)**: This is a more flexible model that can describe different behaviors.

3. My Findings and Why I Chose the Linear Model

After fitting the models to the data, the results were very clear. The **Linear model was the best fit in almost every single case**. The plots and data files I generated show this clearly.

Here is a summary of my final results, with the correct formulas derived from the data you provided:

Ticker	Side	Best Model	R^2 Value	Final Formula
FROG	Buy	Sqrt	**0.990**	$g(x) = 0.0003\sqrt{x} + 0.0519$
FROG	Sell	Linear	**0.996**	$g(x) = 0.0001x + 0.0558$
SOUN	Buy	Linear	**0.899**	$g(x) = 0.00001x + 0.0057$
SOUN	Sell	Linear	**0.985**	$g(x) = 0.000002x + 0.0056$
CRWV	Buy	Linear	**0.997**	$g(x) = 0.00015x + 0.0543$
CRWV	Sell	Linear	**0.995**	$g(x) = 0.00014x + 0.0487$

Even though the task asked if linear models are an oversimplification, my analysis of this specific data shows that they are not. For order sizes up to 200 shares, the relationship between order size and slippage is almost perfectly linear. The R^2 values are consistently high, which means the linear model does a great job of predicting the cost.

This tells me that for these stocks, the market is deep enough to handle these order sizes without the cost spiraling out of control.

Question 2: My Mathematical Framework for Optimal Execution

1. Setting Up the Problem

The second task was to create a mathematical framework to figure out the best way to execute a large order of **S** shares over **N** different time periods. The goal is to minimize the total slippage cost.

Here's how I set up the problem:

- **S** = Total shares to execute.
- **N** = The number of times we can trade (e.g., 390 one-minute periods in a day).
- $\mathbf{x_i}$ = The number of shares I choose to trade in each period i.

My goal is to find the best list of trades $[x_1, x_2, ..., x_n]$ that minimizes the total cost. The problem is:
Minimize: Total Cost = $\Sigma\, g_i(x_i)$ (The sum of the costs of each individual trade).
With the rule that: $\Sigma\, x_i = S$ (I have to trade exactly S shares in total)

2. The Solution, Based on My Findings

Since my analysis in the first part showed that the impact is linear ($g(x) = \beta x + \alpha$), we can plug that into the cost formula. Assuming the market is stable and β and α don't change much during the day, the total cost is:

Total Cost = $\Sigma\, (\beta x_i + \alpha) = \beta * (\Sigma x_i) + (N * \alpha)$ – Because we know that Σx_i has to equal S, this simplifies to:

Total Cost = $\boldsymbol{\beta S + N\alpha}$

The Main Takeaway:

This is a really important result. It shows that for a market with linear impact, the **total cost doesn't depend on *how* I schedule my trades**. Whether I trade everything at the beginning, the end, or spread it out evenly, the total cost will be exactly the same.

Conclusion:

The "optimal" strategy from a pure cost perspective is any strategy that executes all S shares. The simplest and most practical of these is a **Time-Weighted Average Price (TWAP)** strategy, where you trade the same amount in each period ($x_i = S / N$). This means that for these stocks, the focus shouldn't be on finding a complex schedule to reduce costs, but rather on managing other things, like market risk.

Tools for More Complex Scenarios:

If the impact wasn't linear, I would need to use numerical optimization to solve this. The best tool for this in Python is the **minimize** function from the **SciPy** library, which can solve these kinds of complex problems.