# Blockhouse Report for Q1 and Q2

## Q1: Modeling Temporary Impact Function $g_t(x)$

Introduction

When executing large trades in electronic markets, a key concern is temporary market impact, or slippage. This refers to how the price received diverges from the reference mid-price due to the order size and current market liquidity. To minimize costs when executing a total number of shares `S`, we must understand the structure of the impact function `g_t(x)` — how much slippage we incur by trading `x` shares at time `t`. This section provides a data-driven model for `g_t(x)` using high-frequency limit order book (LOB) data across three tickers (SOUN, FROG, CRWV), with extensive exploratory analysis, simulation, and fitting techniques.

### 1. Data and Preprocessing

We work with millisecond-resolution LOB snapshots in the MBP-10 format, with top-10 levels for bids and asks. From each day of data, we extract 1-minute LOB snapshots (`N = 390` per day) using the last message per minute. Each snapshot gives us:

- Bid/Ask prices and sizes at levels 0 to 9
- Computed mid-price: $mid_t = (bid_0 + ask_0)/2$
- Total depth: $\sum size_i$ over levels
- Spread and imbalance indicators

### 2. Slippage Simulation via VWAP

We simulate market buy and sell orders of various sizes by walking through the LOB:

$VWAP(x) = (\sum_i price_i \,.\, fill_i)/ x$

Then, slippage is computed as:

- Buy-side: ( $g_t^{buy(x)} = VWAP_{buy(x)} - mid_t$)
- Sell-side: ($g_t^{sell(x)} = mid_t - VWAP_{sell(x)}$

This approach captures the true execution cost for `x` shares under perfect information.

### 3. Functional Forms for $g_t(x)$

We evaluated three families:

(a) Linear: $g_t(x) = \boldsymbol{\beta}_t\, x$

- Too simplistic, underestimates risk for large orders.

(b) Piecewise or spline-based fits

- Captures local non-linearity, but hard to use in optimization.

(c) Power Law: $g_t(x) = \alpha_t\, x^{\gamma_t}$
- Theoretically supported and fits our data well.
- Convex if $\gamma_t > 1$, which holds empirically.

We use `scipy.optimize.curve_fit` to estimate $\alpha_t$, $\gamma_t$ per snapshot using simulated `x ∈ [100, 2000]`.

**4. Empirical Findings**

- Buy vs Sell: Both sides are modeled separately — asymmetries observed.
- Convexity: $\gamma_t$ in [1.1, 1.5] typically.
- Stability: Impact functions are consistent within a day, but vary across days and instruments.
- Normalization: We model `x` in absolute shares, not relative depth, to keep optimization interpretable.

**5. Visual Analysis**

Overlaid plots of slippage curves for random timestamps show:

- Low slippage up to ~500 shares
- Steep convex rise beyond 1000 shares
- Good agreement between empirical and fitted power law curves

**6. Limitations and Extensions**

- Real execution involves latency and hidden liquidity — our model assumes perfect fill.
- For high-frequency extensions, need to include volatility and adverse selection.
- Future: add Kalman filtering on $\alpha_t$, $\gamma_t$ for smoothing.

**Conclusion**

We model the temporary impact function $g_t(x)$ using a power law form:

$g_t(x) = \alpha_t x^{\gamma_t}$

This captures convex slippage growth and aligns with both theory and empirical LOB behavior. These models are used in Q2 to build an optimal trading strategy.

## Q2: Optimization Framework to Minimize Slippage

### Objective

We are given a target of S shares to buy throughout the day, over N = 390 trading intervals. At each interval t, we can choose $x_t$ shares to buy.

The goal is to minimize total slippage across the day:

$\min_{x_1,\dots,x_N} \sum_{t=1}^{N} g_t(x_t)$ subject to $\sum x_t = S$

### Modeling g□(x)

From Q1, we model:

$g_t(x) = \alpha_t x^{\gamma_t}$

Where:

- $\alpha_t > 0$: scaling parameter (slippage per share)
- $\gamma_t > 1$: convexity exponent (typically ~1.2)

This is a convex function if $\gamma_t >= 1$, so the optimization is tractable.

### Final Optimization Problem

$\mathrm{Min}_{\{x_t >= 0\}} \sum_{\{t=1\}}^{\{390\}} \alpha_t x_t^{\gamma_t}$ such that $\sum_{\{t=1\}}^{\{390\}} x_t = S$

This is a convex program with one equality and non-negativity constraints.

### Optional Constraints

To reflect real execution limitations, we can add:

- Max participation: $x_t <= \eta \cdot D_t$, where $D_t$ is top-10 depth
- Volatility penalty: $\lambda \sum \sigma_t x_t^2$, where $\sigma_t$ is rolling volatility

### Solving Strategy

We use CVXPY to solve this problem numerically:

### Interpretation

The output is a vector $x_t$ that balances:

- Lower slippage times $\alpha_t$
- Lower convexity $\gamma_t$
- Global budget constraint $\sum x_t = S$

The allocation is more aggressive during liquid, low-volatility windows.

### Conclusion

This framework enables data-driven, convex optimization of execution schedules using fitted impact models. It can be extended for multi-asset execution, real-time adaptation, or inclusion in RL systems.