Code Summary

Willie Xia and Thomas Su

https://github.com/thomaswsu/Trading_Assignment_2

Code Summary of Graphs

The underlying methodology to creating our 3-d plots of the order size, bid-ask spread, and bid size can be broken down into 5 parts:

1. Isolate the targeted columns
2. Remove unnecessary trailing time values from time column
    a. Milliseconds for quotes
    b. Seconds and milliseconds for trades
3. Create weights in order to calculate weighted average
4. Calculate weighted averages
    a. Sum for values like size
5. Plot required data

The biggest challenge is determining the weights, which was achieved by taking the time second difference from the previous time second and pair that as the weight for the previous row. For example, if a had second 45 and b had second 48, a would have weight 3. In the matter when a minute shifts, we subtract the last second value before the minute changed from 60. Ie. b would have weight 12 if it had no further rows within the same minute. We can apply the same logic every time an hour and date changes. After calculating the weights, we can use a lambda function to calculate the weighted averages of all the required values for each column.

For the graph, because we already cleaned the data and calculated the required values, we can simply format and plot the necessary columns. The graph plotting methodology of choice is trisurf, which colorizes the peaks of our graph to give a beautiful visualization that helps to distinguish peaks from other peaks.

Selling Algorithm

At a high level the following is our selling algorithm:
1. Generate the intervals when we can sell. In this case it is every 15 minutes of the trading day.
2. Random select to sell or not to sell with equal weight.
3. If we decide to sell, sell a random amount of shares, with normal distribution. (Look at code for how mu and sigma are configured.)
4. Repeat until we have no more shares left to sell, or we are out of time. If we are out of time, sell all remaining shares.

In order to implement the selling algorithm our group implemented a discrete stochastic algorithm of the following form:

$$dV_t = -\left|N(\mu_t, \sigma_t)\right| \cdot Bern(1, 0.5)$$

Where $dV_{\Delta t}$ is the change in the number of stocks, N is a normal distribution, μ is the expected number of shares to be sold, σ is the desired volatility number of shares to be sold, and $Bern()$ is a Bernouli distribution which is sampled once with an equal probability of returning 0 or 1.
μ is value that is recalculated at every time step is determined by the following equation:

$$\mu_t = 2 \cdot \text{remaining share/remaining time steps}$$

σ is calculated every time step by:

$$\sigma_t = \frac{\mu_t}{2}$$