FIN 566
Problem Set #4

Adam D. Clark-Joseph

University of Illinois Urbana-Champaign

DUE October 4, 2017

Introduction

This problem set continues the ideas from PS#3, building on the foundations established in that problem set to develop a market-making algorithm that estimates a measure of adverse selection from past market data, and bases its decision about whether to participate in the market on that estimate. The third question of this problem set introduces in a very rudimentary form a subject to which we will return with increasing frequency throughout the term.

The template codes from PS#3 (together with the code addressing those questions) are the only pre-written code necessary for this assignment. Since I am allowing you more flexibility in how you solve some of the problems on this assignment, I am asking you to turn in several indicated portions of your code.

In the solutions that you turn in, please leave "max_quantity=1" "max_price=1000," "min_price=1," "num_bgt=10," "price_flex=1," "prob_last_order_price_sets_to_price_robot_1=0," and "burn_in_period=1322." Use the "`bgt_behavior_rw_price.m`" script for background-trader behavior.

As before:

- "`robot1_algo_mm_at_best.m`" denotes the market-making algorithm from PS#2 that enters an order at the best price in a randomly chosen direction.

1 Estimating The Degree of Adverse Selection

1.1 Question

Write a script to estimate the value of prob_last_order_price_resets from the sequence of prices at which background traders submit orders. Do not include the prices of any orders placed by robot1 in the data on which you base your estimates. Please ensure that your code can accommodate values of *price_flex* greater than 1. Although we will restrict attention to the case of $price\_flex = 1$ in this problem set, you are likely to need to consider the more general cases of $price\_flex > 1$ sometime soon.

TURN THIS CODE IN WITH YOUR PROBLEM SET SOLUTIONS. Please try to make your code easy to understand for a reader.

1.2 Question

Set prob_last_order_price_resets=0.01 and t_max=2322.

Generate a random sequence of background-trader order-prices and estimate prob_last_order_price_resets; repeat this 400 times and store each of your estimates. Report the 90% confidence interval for *an individual estimate.* Note that this is different from constructing a bootstrap confidence interval for the mean of all the individual estimates. Plot the empirical quantiles of the individual estimates. (Sort in ascending fashion, then plot.)

Hint for Questions 1.1 and 1.2

A sequence of background trader order-prices can be generated without doing any matching, and without switching among different background traders. Consider writing a short piece of code to generate these price sequences without all of the extraneous operations.

2 A New Market-Making Algorithm

Let the parameter "watch_interval" be an integer satisfying $burn\_in\_period \leq watch\_interval < t\_max$. Let the parameter "mm_trigger_value" be a real number taking values on the open interval $(0, 1)$.

2.1 Question

Modify the `robot1_algo_mm_at_best.m` algorithm so that robot1 estimates prob_last_order_price_resets using the sequence of entered order prices from times $t = 1$ through $t = watch\_interval$ (denote this estimate by "est_prob_last_order_price_resets"), and then incorporates that information into a trading strategy as follows:

- If est_prob_last_order_price_resets<mm_trigger_value, then for $t > watch\_interval$, robot1 behaves in exactly the manner prescribed by "`robot1_algo_mm_at_best.m`."
- However, if est_prob_last_order_price_resets≥mm_trigger_value, then robot1 does not trade while $t > watch\_interval$.
- Assume that robot1 never trades while $t < watch\_interval$.

TURN THIS CODE IN WITH YOUR PROBLEM SET SOLUTIONS.

2.2 Question

Suppose that prob_last_order_price_resets $\in \{0.01, 0.06\}$, and prob_last_order_price_resets assumes each of these values (0.01 and 0.06) with equal probability. Write a few lines of code so that you can run simulations in which prob_last_order_price_resets is determined in this manner. TURN THIS CODE IN WITH YOUR PROBLEM SET SOLUTIONS.

Set $watch\_interval = 2322$, and $t\_max = watch\_interval + 1000$.

Find an approximately optimal choice of mm_trigger_value (which is one of the MM's *ex-ante* choice parameters). Report this (near-) optimal parameter choice, the associated average profits, and a 95% confidence interval for the associated average profits.

Hint: on the previous problem set, you developed tools/code that should allow you to estimate with high precision the MM's average trading profits in the case of prob_last_order_price_resets=0.01, and also in the other case of prob_last_order_price_resets=0.06. You can estimate the conditional probabilities of prob_last_order_price_resets=0.01 and prob_last_order_price_resets=0.06, conditioning upon {est_prob_last_order_price_resets<trigger_value_in_question}. Combining such probabilities with the already-computed average trading profits in each of the cases, you can rapidly narrow your search space without particularly intensive computation.

3 Speed vs. Intelligence

3.1 Question

(a) Set prob_last_order_price_resets=0.01.

Repeat question (1.2) for t_max=2322, 3322, 4322. Report the associated confidence intervals in a table, and plot all of the empirical quantile-plots on the same graph.

(b) Repeat part (a) of this question, but set prob_last_order_price_resets=0.06.

3.2 Question

Repeat question (2.2) for $watch\_interval = 3322$ and $watch\_interval = 4322$. As before, set $t\_max = watch\_interval + 1000$.

3.3 Question

Instead of setting $t\_max = watch\_interval + 1000$, as in questions (2.2) and (3.2), set $t\_max = 5000$. Using the (near-)optimal values of mm_trigger_value obtained in questions (2.2) and (3.2), compute 95% confidence intervals for the new MM algorithm's average profits for $watch\_interval = 2322, 3322, 4322$.

4 *Open-ended, "no-wrong-answer" question

Please attempt this question to the best of your ability. You'll get full points just for trying, but I may ask you to talk about your approach to the question in class.

4.1 *Question

Set $t\_max = 5000$. Adapt your MM algorithm code from Questions 2 and 3 so that your algorithm keeps estimating prob_last_order_price_resets continuously throughout the simulation, and participates in the market when— and only when— est_prob_last_order_price_resets≥mm_trigger_value. Notice that est_prob_last_order_price_resets will now be an estimate that changes over time, so Robot_1 no longer faces just a single "participate today, or don't participate today" decision.

Please briefly describe what you did. You do NOT have to turn in any code.

4.2 *Question

Continuing Question 4.1, use the methods from other questions (or any other methods you like) to try to find a roughly optimal mm_trigger_value to use in your MM algorithm from Question 4.1.

Please briefly describe what you did. You do NOT have to turn in any code.