Online Markets Project 4

Online Revenue Maximization

May 2022

1 Introduction

The following study aims to analyze the performance of learning algorithms optimizing truthful auctions in which bidders arrive online. In particular, we will be looking at the exponential weights algorithm, and its performance in optimizing revenue for online auctions. In online auctions, bidders arrive simultaneously at the start of each round, and the learning algorithm has no knowledge of the distributions from which their values have been drawn. The performance will be analyzed in the following two parts:

In the first part of this study, we will analyze the performance of the exponential weights algorithm as it attempts to find the optimal reserve price in a second-price auction (SPA) with reserve. In the SPA with reserve online auction, the winner is the bidder with the highest bid, and they pay the $max(2^{\text{nd}}$ highest bid, reserve price). If no bidders bid above the reserve price, there is no winner, and the item is not sold. The bidders will always bid truthfully. Our exponential weights algorithm will attempt to find the optimal reserve price for the auction as rounds of the auction go on. We will then analyze the performance of the algorithm in finding the optimal reserve price and optimizing revenue. We will test the algorithm using different variations of distributions and number of bidders. Additionally, we will vary the approach by allowing the algorithm to wait a given k rounds before attempting to optimize reserve price, in a similar idea to the Secretary Algorithm. These variations will be described in greater detail in the preliminaries section. Across all variations, our algorithm quickly converged to nearly the optimal reserve price.

In the second part of this study, we will analyze the performance of the learning algorithm in a bilateral exchange scenario. The algorithm will broker an exchange between a buyer and seller with the following behavior. There is a single buyer with value v drawn from distribution F, and there is a single seller with a cost c drawn from distribution G. Both buyer and seller will bid truthfully, that is, they will bid their true value or cost. The expected revenue of the broker is equal to the expected virtual welfare which is the virtual value of the buyer minus the virtual cost of the seller if the good is exchanged, and

zero otherwise. Given the distributions F and G we will identify the optimal truthful mechanism for a bilateral exchange, and then provide an online learning algorithm that is able to learn this mechanism online. We came up with an algorithm that gets fairly close to the optimal truthful mechanism in the context we evaluated.

2 Preliminaries

In this study, data for each scenario will be generated at random from the given distributions using standard Python techniques. This includes values in part 1, and values and costs in part 2.

Before discussing the details of our tests and scenarios, we will describe our implementation of an online learning algorithm. The exponential weights algorithm is an algorithm which chooses an action from a set of actions based on certain probabilities, adjusting them as it runs. In this context, the exponential weights algorithm will be choosing the "action" of setting a reserve price for an SPA with reserve auction. It makes these decisions based on the following parameters. The first is a list of payoffs for each possible action, which is determined by the revenue generated from that reserve price. The second is a learning rate, which determines how fast/slow the algorithm learns, and for this study, learning rate will be fixed at the theoretical optimal rate of $\epsilon = \sqrt{\frac{\ln(k)}{n}}$. We determined that our results did not change significantly after $n = 100$, so we used this value. The action space, and thus the value we chose for k, will be explained soon. The algorithm and the auction run in the following way:

1. Choose a reserve price based on the probabilities for each
2. Set that reserve price for the upcoming round
3. Bidders arrive and place bids
4. Determine winner (if any), and obtain payoff
5. Update payoffs of reserve price options
6. Given learning rate, update probabilities of actions

As such, the algorithm and auction will run together as bidders arrive and the algorithm attempts to maximize revenue with its reserve price. With full information, one can calculate the optimal reserve price using virtual value (which is what we will use to analyze performance, as discussed below), but because the algorithm is unaware of the distributions from which the bidders' values have been drawn, it must find a way to estimate reserve price otherwise. To do this, the algorithm looks at the set of bids it has received, and finds the minimum and maximum. From there, we chose to define the range of our action space as

[min, max] and then we evenly distributed the actions within this range according to k actions. We wanted to choose k such that it is large enough that it is likely one of our discretized actions is close to the optimal reserve value. We decided to use $k = 30$ for this study.

For part 1 of this study, we will consider the following SPA with reserve online auction. Consider m bidders with values drawn i.i.d. from a distribution F in each round. These bidders will always bid truthfully. In order to obtain accurate results for the performance of our learning algorithm at optimizing reserve price, we will analyze the following auction scenarios:

1. Baseline: $m = 2$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$
2. Variation 1.2: $m = 2$ bidders arrive each round with $F(z) = z$ for $z \in [0, 3]$
3. Variation 1.3: $m = 2$ bidders arrive each round with $F(z) = z$ for $z \in [2, 3]$
4. Variation 1.4: $m = 5$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$
5. Variation 1.5: $m = 10$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$
6. Variation 1.6: $m = 25$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$

With these variations, we will obtain a greater understanding of the algorithm's performance. For each test, we use virtual value to calculate the optimal reserve and the optimal expected revenue, and then compare those values to that which our algorithm converges upon. The baseline provides a quick and easily digestible test of our algorithm. Variations 1.1-1.3 alter the distribution from which the bidders get their values. Variations 1.4-1.6 alter the number of bidders which arrive each round to participate in the auction.

Additionally, we aim to improve upon the basic strategy of simply applying the online learning algorithm to this problem. Specifically, note that our algorithm's action space can be very unrepresentative of the underlying distribution given we define it according to only the first round of bidding. To mitigate this problem, we will implement a similar strategy to the Secretary Algorithm, and allow the algorithm to see data prior to taking any actions. We will let the algorithm wait a given l rounds of the auction, setting the reserve price to be zero as it waits, so that it may learn about the underlying distribution prior to taking any action. We will then test the following scenarios:

1. Variation 1.7: $m = 2$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$ and $l = 2$
2. Variation 1.8: $m = 2$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$ and $l = 4$

3. Variation 1.9: $m = 10$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$ and $l = 2$
4. Variation 2.0: $m = 10$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$ and $l = 4$

We anticipate that by allowing our algorithm to learn before attempting to take any action, it will be able to converge on the optimal reserve price faster and be more accurate in the long run. In addition, we hypothesize that this strategy will work better when there are less bidders, as we expect the bids in the first round to be more representative of the underlying distribution when there are more of them.

For part 2 we analyzed the performance of EW in learning the optimal allocation rule for a bilateral exchange scenario. This scenario involves a single buyer with value v drawn from distribution F and a single seller with cost c drawn from distribution G. The expected revenue of the broker is equal to the expected virtual welfare which is the virtual value of the buyer minus the virtual cost of the seller. The virtual value of the buyer is defined per usual

$$\phi(v) = v - \frac{1 - F(v)}{f(v)}$$

where $F(v)$ is the cummulative distribution function and $f(v)$ is the density function of F. The virtual cost of the seller on the other hand is defined as follows

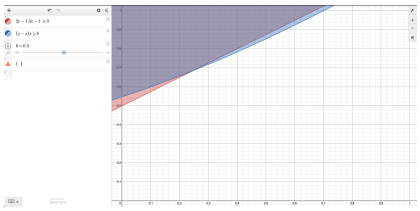
$$\phi(c) = c + \frac{G(c)}{g(c)}$$

where $G(c)$ is the cumulative distribution function and $g(c)$ is the density function of G.
The optimal allocation rule in this context will be the one such that an exchange happens if the virtual welfare is positive, or whenever $\phi(v) - \phi(c) \geq 0$.
Unfortunately, our algorithm won't know the distributions F and G, so while we know we need at least $v \geq c$ to justify an exchange, we don't immediately know for which such cases it is true that $\phi(v) \geq \phi(c)$.
To solve this, let's analyze the above equation further. Note that both $\phi(v)$ and $\phi(c)$ are monotonically increasing ($\frac{1-F(z)}{f(z)}$ decreases as v increases by definition). Now imagine we have a discrete set of pairs (v, c) and we sorted them according to the following metric:

$$(v - c) * v$$

The motivation behind this ordering is to create an ordering such that everything below a certain point should not be have a sale and everything above a certain point should. You would think that you can just use $(v - c)$ for this ordering, but clearly, $0.95 - 0.9$ and $0.05 - 0.0$ are going to be different situations because $\frac{1-F(z)}{f(z)}$ and $\frac{G(c)}{g(c)}$ vary. We found that by multiplying this metric by v we were able to create an ordering that could approximate our goal for any linear

constraint that we could see within this problem, and by definition of virtual values in this situation, all equations of the form $v - \frac{1-F(v)}{f(v)} - c - \frac{G(c)}{g(c)}$ must be linear. In addition, any action that acts on this ordering, by definition of their being a singular cutoff point, will be monotonically non-decreasing with respect to our allocation function. The actions in this case will be the possible cutoff points within our ordering, so action 0 says "let all deals with $v - c \geq 0$ go through" and action $j > 0$ says "let all deals whose metric is greater than the jth auction in our metric go through".
Note that this structure implies the number of actions will increase each round. This isn't that big of a deal though; we simply dynamically define the optimal learning rate in terms of this increasing value and calculate the payoffs of each new action according to how it would've performed in past rounds.
To test this strategy, we will plot the average revenue our algorithm obtains per round and compare it to the theoretically optimal expected revenue. We will be examining the situation where the buyer's distribution is the uniform distribution on $[0, 1]$ and the seller's distribution is the distribution with quadratic cumulative distribution function $G(z) = z^2$ on $[0, 1]$. In this situation, the optimal allocation rule can be calculated to be 1 if $2v-1-\frac{3}{2}c \geq 0$ and 0 if not. Note that this strategy is the optimal truthful mechanism because both $\phi(v) = 2v-1$ and $\phi(c) = \frac{3}{2}c$ are monotonically increasing. The above figure shows our our

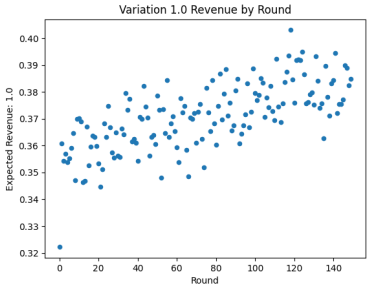


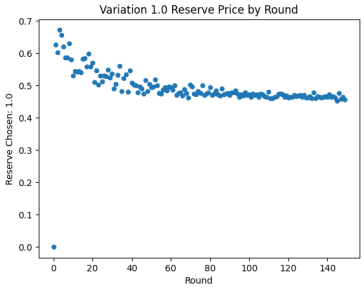
ordering will approximate the optimal allocation rule. We accordingly expect our algorithm to do decently well, but we do not expect it to get the theoretically optimal expected revenue of $\int_{0.5}^{1} \int_{0}^{\frac{4}{3}y-\frac{2}{3}} (y - x)dxdy \approx 0.101$ (the region described by this integral is the area where $2y - 1 - 1.5x \geq 0$ on $[x, y]$ where $y, x \in [0, 1]$)

3 Results

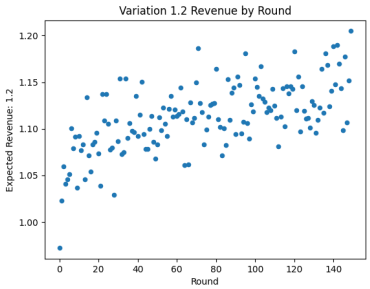
As described above, for part 1, we tested the performance of our algorithm using different variations, and comparing its observed revenue and reserve price to the optimal expected revenue and reserve price. We used the known distributions and virtual values to calculate optimal expected revenue and reserve price. For each variation, we list the optimal reserve price, optimal expected revenue, final observed reserve price, and final observed revenue, and show two plots of our algorithm's revenue by round, and reserve price by round. The following were the results of our tests for each scenario:

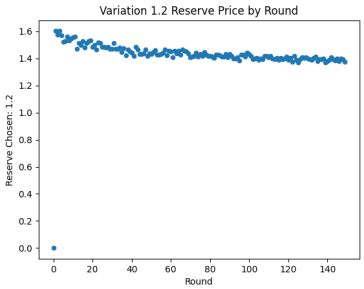
1. Baseline: $m = 2$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$
    - Optimal reserve price: 0.5
    - Optimal expected revenue: 0.41667
    - Final observed reserve price: 0.45659
    - Final observed revenue: 0.38494
    - Plots of observed revenue by auction round and observed reserve price by auction round:





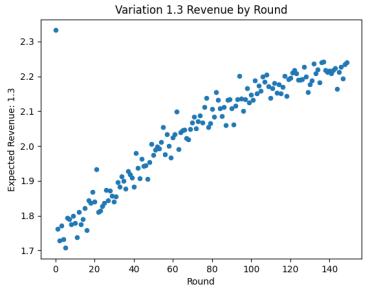
2. Variation 1.2: $m = 2$ bidders arrive each round with $F(z) = z$ for $z \in [0, 3]$
    - Optimal reserve price: 1.5
    - Optimal expected revenue: 1.25
    - Final observed reserve price: 1.42980
    - Final observed revenue: 1.12785
    - Plots of observed revenue by auction round and observed reserve price by auction round:

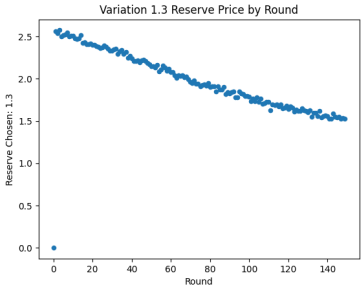




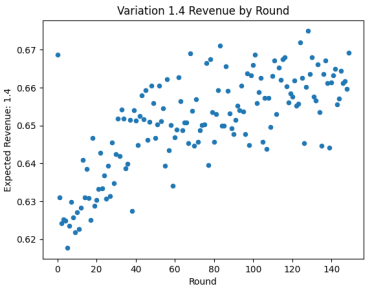
3. Variation 1.3: $m = 2$ bidders arrive each round with $F(z) = z$ for $z \in [2, 3]$

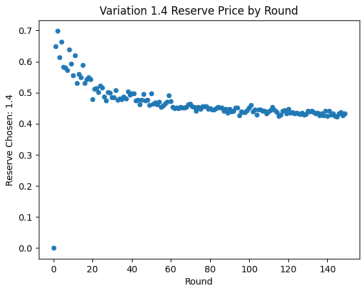
- Optimal reserve price: 2
- Optimal expected revenue: 2.33333
- Final observed reserve price: 1.57970
- Final observed revenue: 2.24247
- Plots of observed revenue by auction round and observed reserve price by auction round:





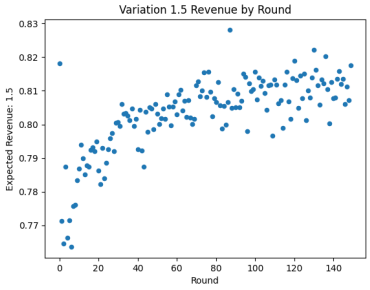
4. Variation 1.4: $m = 5$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$
   - Optimal reserve price: 0.5
   - Optimal expected revenue: 0.58646
   - Final observed reserve price: 0.42812
   - Final observed revenue: 0.66112
   - Plots of observed revenue by auction round and observed reserve price by auction round:

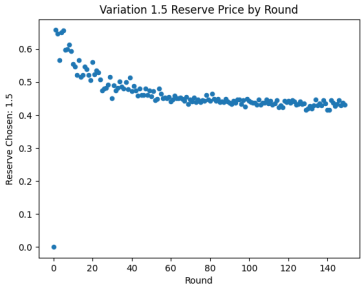




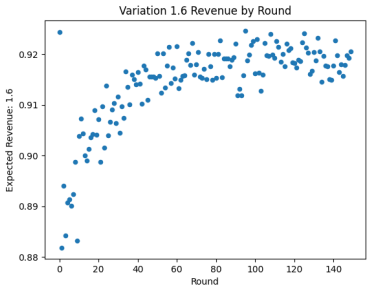
5. Variation 1.5: $m = 10$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$
   - Optimal reserve price: 0.5

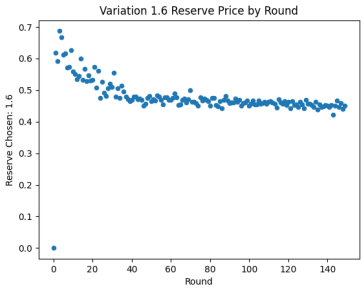
- Optimal expected revenue: ≈ 0.90909
- Final observed reserve price: 0.44053
- Final observed revenue: 0.81037
- Plots of observed revenue by auction round and observed reserve price by auction round:





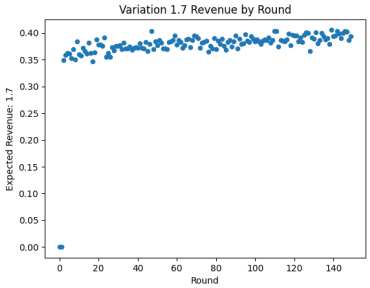
6. Variation 1.6: $m = 25$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$
   - Optimal reserve price: 0.5
   - Optimal expected revenue: ≈ 0.96154
   - Final observed reserve price: 0.46344
   - Final observed revenue: 0.92015
   - Plots of observed revenue by auction round and observed reserve price by auction round:

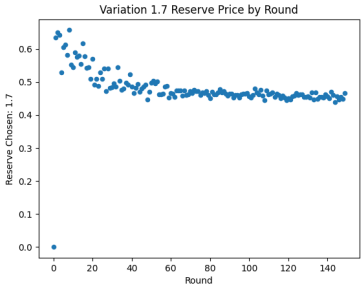




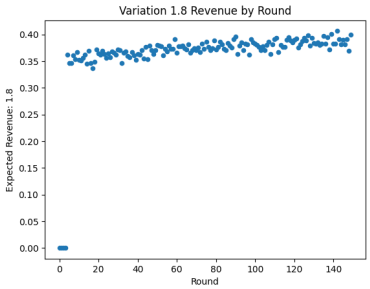
7. Variation 1.7: $m = 2$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$ and $l = 2$
   - Optimal reserve price: 0.5

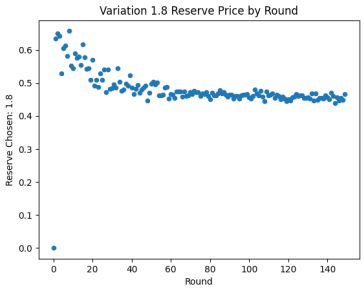
- Optimal expected revenue: 0.41667
- Final observed reserve price: 0.46539
- Final observed revenue: 0.39387
- Plots of observed revenue by auction round and observed reserve price by auction round:





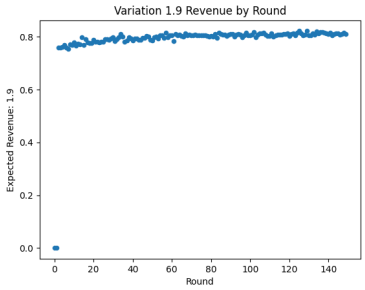
8. Variation 1.8: $m = 2$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$ and $l = 4$
   - Optimal reserve price: 0.5
   - Optimal expected revenue: 0.41667
   - Final observed reserve price: 0.46539
   - Final observed revenue: 0.39959
   - Plots of observed revenue by auction round and observed reserve price by auction round:

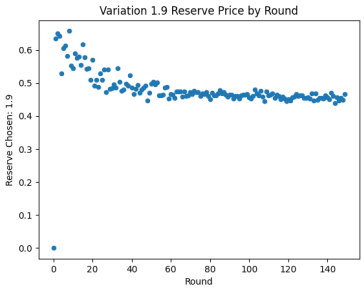




9. Variation 1.9: $m = 10$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$ and $l = 2$
   - Optimal reserve price: 0.5

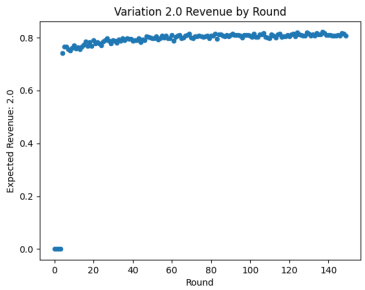
- Optimal expected revenue: ≈ 0.90909
- Final observed reserve price: 0.46539
- Final observed revenue: 0.81044
- Plots of observed revenue by auction round and observed reserve price by auction round:

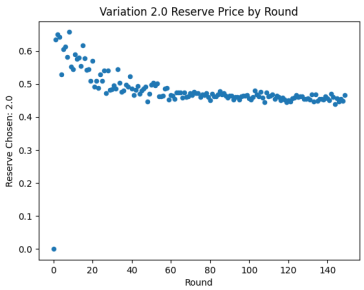




10. Variation 2.0: $m = 10$ bidders arrive each round with $F(z) = z$ for $z \in [0, 1]$ and $l = 4$

    - Optimal reserve price: 0.5
    - Optimal expected revenue: ≈ 0.90909
    - Final observed reserve price: 0.46539
    - Final observed revenue: 0.80813
    - Plots of observed revenue by auction round and observed reserve price by auction round:

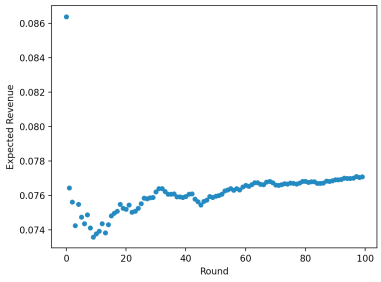


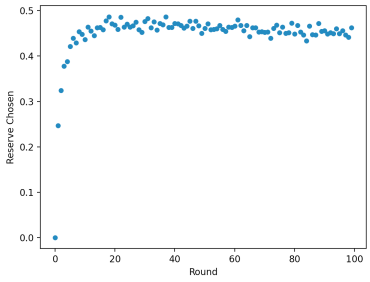


    In general, it our algorithm performed slightly below the theoretical expected revenue.

Part 2: Buyer with uniform distribution on $[0, 1]$ and Seller with the distribution with quadratic cumulative distribution function z^2 on $[0, 1]$

- Theoretical Optimal Revenue = ≈ 0.101





4 Conclusions

For part 1 of this study, our algorithm performed well with the task of optimizing revenue and reserve price in a SPA with reserve. As seen in the charts in our results section, our algorithm quickly converges to near the theoretically optimal values of revenue and reserve price. This performance trend remained through all scenarios and variations as we changed the distributions from which values were drawn and the number of bidders participating in the auction. There was no significant change in the performance of our algorithm when the distributions were altered or when the number of bidders increased. Additionally, our algorithm learned relatively quickly, as it converged on the optimal revenue and optimal reserve price at about round 100 (or earlier) of the auction. Another interesting discovery made during this part of the study was that increasing the amount of k actions that the algorithm can choose from increases its performance with respect to both it's rate of convergence and final accuracy. This follows logically, as a higher number of actions indicates more options for reserve price choice, which means one of them will likely be closer to the theoretically optimal choice. This phenomena was not what we intended to study in this project, but we think it would be interesting for future studies.

We hypothesized for part 1, that allowing our algorithm to wait a given l rounds of the auction before taking any action would make our algorithm more successful in the long run. This hypothesis was inspired by the Secretary Algorithm from lecture. We saw that waiting allowed our algorithm's final reserve price to be more accurate. This makes sense given this strategy allows our algorithm to have a better understanding of the distribution before it discretizes it's action space.

One way to further test our algorithm for this part of the study would be to increase the amount of items being sold (i.e. selling 3 items and running a

truthful 4^{th} price auction). We would hypothesize that our algorithm would still perform well in this context. Additionally, a further exploration of this algorithm could be to optimize the amount of rounds l that it waits before taking any action. In our variations, we tested $l = 2$ and $l = 4$.
Moving on to part 2, we saw that our algorithm performed well, but not quite up to the theoretically optimal expected revenue. This is exactly as we expected as our algorithm's possible best allocation strategy does not perfectly represent the true optimal allocation strategy. That said, it got about 75% of the way there. In addition, we can see the average "reserve" chosen in each round. This value is the actual action taken divided by the round number so it represents the percent of observed auctions that were excluded from allocation. According to our desmos graph in the preliminaries section, we would have expected this value to converge to about $\frac{2}{3}$, which is an approximation of the area our algorithm's best choice excludes. As we can see, this also got fairly close, but not quite there. In the future, we would like to see if we can come up with a general strategy that can get even closer to optimal expected revenue.

5 Appendix

https://github.com/patrickdwyer33/project4.git