MS&E 111/MS&E211 Suggested Course Project III: Online Linear Programming and Resource Allocation

December 11, 2023

Group Members: Stephanie Szpylka, Mark Teffeteller, Aishwarya Vaidhyanathan, Yuer Zhou

Project Solutions:

For Question 1, the algorithm outputs the below result (rounded to 2 decimal places):

Table 1: SLPM and Offline Revenue

k	SLPM Revenue
50	8314.43
75	8330.53
100	8399.01
150	8758.69
200	8830.13
250	8515.76
350	9181.86
500	8915.80
1000	9038.67
5000	4912.98

Offline Revenue
9956.98

Based on these results, we have found that there is a bell curve to the sensitivity ratio. We sacrifice valuable bids when the "sweet spot" of information is not considered, i.e. too little or too much information. The point that maximizes revenue in this algorithm is k=350, which produces a result of 9181.86. Due to the randomness of this experiment, each simulation brings slightly different results. We used the High Performance Optimization Solver to help solve this linear program. [5]

There are various trade-offs between choosing large and small k. A couple of pros of increasing k include more accurate estimation prices and reduced regret. When k is higher, there are more bids that the algorithm can learn from and increase revenue with more accurate and more efficient representation of prices. Following that is reduced regret because the difference between revenue generated with the online algorithm and the optimal revenue is decreased with a better understanding of the market.

A couple of the cons of choosing a larger k is higher computational cost and potential for overfitting. Solving the linear program with more bids requires more computational resources, which can be slow and resource-intensive for very large k values. Also, with too many bids, the algorithm may overfit to the specific data points and not generalize well to unseen scenarios.

On the flip side, choosing a smaller k allows for lower computational cost and reduced sensitivity to noise. The fewer bids allows for more efficiency with use of less computational resources, bringing a better use for real-time applications. Less bids additionally prevents individual outliers or noisy data points from affecting the algorithm.

With a smaller k comes the cons of less accurate estimates of prices and higher regret. With fewer bids, the algorithm may have less accurate estimates of the true prices, leading to potential revenue loss. A smaller k value can increase regret as the algorithm has a less complete picture of the market dynamics.

An offline algorithm with 100% information about all bids will always achieve the optimal revenue. This is because it can make perfect allocation decisions based on the known prices, without any uncertainty or error, but that is not realistic for the real world. Therefore, comparing the revenue generated by SLPM with k to the offline optimal revenue provides a measure of the algorithm's performance and insight into the potential revenue loss due to limited information. The results show that even with a relatively small k value (e.g., 50), SLPM can achieve a significant portion of the offline optimal revenue. This suggests that it can be a viable and efficient approach for online revenue maximization in situations with limited information.

A comparison of algorithms brings an interesting conclusion - Algorithms for consideration:

Highs Simplex: This algorithm iteratively explores the vertices of the feasible region until it finds the optimal solution Highs Dual Simplex: This algorithm focuses on the dual problem and iterates through its vertices to find the optimal solution. Highs Interior Point: This algorithm takes advantage of barrier functions to find the optimal solution within the interior of the feasible region.

Output:
highs: [86, 84, 90, 93, 88, 87, 91, 95, 96, 54]
highs ds: [67, 84, 89, 93, 87, 86, 91, 94, 94, 54]
highs ipm: [67, 83, 89, 93, 86, 86, 90, 93, 94, 53]

After running the simulation on the same n=10,000 dataset across 3 different algorithms (highs "simplex", highs-ds "dual simplex", high-ipm "interior point") we can observe that the highs simplex method consistently achieves revenue that best approaches the optimal result (the offline revenue). Highs-ds and highs-ipm algorithms exhibit similar performance; while both highs-ds and highs-ipm algorithms achieve lower revenue than the highs simplex method, their performance is relatively consistent with each other. This indicates that both algorithms may be suitable for scenarios where achieving the absolute highest revenue is not critical, but speed or other factors might be more important.

For Question 2, the dual prices are as follows:

i	dual
50	None
100	$[1.00509794, 1.01171952, \ldots, 0.99891171]$
200	$[1.00274409, 1.01171952, \ldots, 1.00147182]$
400	$[1.00677922, 1.0062498, \ldots, 1.0041374]$
800	$[1.00529389, 1.00744263, \ldots, 1.00329182]$
1600	$[1.00651467, 1.00699794, \ldots, 1.00147182]$
3200	$[1.00623907, 1.00594589, \ldots, 1.0041374]$
6400	$[1.00348096, 1.00475593, \ldots, 1.00174985]$

Dynamic SLPM revenue: 9921.21

Offline revenue: 10058.21

The dynamic learning algorithm performs better than the static SLPM algorithm. This is shown by the higher average revenue generated by updating the dual prices at defined intervals. The dual price vectors do gradually approach the ground truth vector. In our output, the defined ground truth price is 1 and we can see movement towards it.

For Question 3, the output from the algorithm is seen below (vector transformed to table for comparison):

Iteration	Incremental AHD	Incremental Algorithm
1	1.005784	0.0
2	3.017352	3.014647
3	6.034704	9.021489
4	10.057840	12.022022
5	15.086760	15.021675
6	21.121464	16.975648
7	28.161952	23.968791
8	36.208224	29.958686
9	41.279352	32.933830
10	47.356265	35.908982

Both algorithms display a rising performance trend with increasing bids, effectively utilizing accepted ones. AHDLA initially exhibits a slightly higher performance, indicating its ability to capitalize on early bids and more frequently updated dual prices effectively. It maintains a slight performance advantage at later stages, potentially attributed to its dynamic decision updates. It dynamically adjusts decision-making

based on action history, potentially leading to better responsiveness to changing bid quality. SLPM relies on a static decision rule based on a subset of bids, resulting in less adaptability to evolving bid patterns.

Both algorithms demonstrate strong performance in terms of revenue generation and value capture compared to the optimal solution. Selecting between them depends on factors like if bid quality fluctuates significantly over time, AHDLA's dynamic approach might be advantageous or that SLPM's simpler logic might be preferable in resource-constrained situations.

For Question 4, the formulation is as follows:

$$\begin{aligned} &\text{minimize}_{\bar{\mathbf{y}}} && \mathbf{d}^T\bar{\mathbf{y}} + \mathbb{E}\left(\pi - \mathbf{a}^T\bar{\mathbf{y}}\right)^+, \\ &\text{s.t.} && \bar{\mathbf{y}} \geq 0, \end{aligned} \tag{1}$$

where $d = \frac{b}{n}$ and $(\cdot)^+ = max\{\cdot, 0\}$. To identify whether (3) is a convex optimization problem or not, we can analyze each part of (3) separately. For the first part $d^T\bar{y}$ of (3), $d^T\bar{y}$ is linear in $\bar{y}$. We know that linear functions are both convex and concave. For the second part $\mathbb{E}\left(\pi - \mathbf{a}^T\bar{\mathbf{y}}\right)^+$, since $\pi - \mathbf{a}^T\bar{\mathbf{y}}$ is linear in $\bar{y}$. What $(\cdot)^+$ does is to get maximum value between this linear function + and zero, since both of them are convex, $(\pi - \mathbf{a}^T\bar{\mathbf{y}})^+$is also convex. The last part we need to consider is the expectation.

From Jensen's inequality:

$\mathbb{E}[f(X)] \geq f(\mathbb{E}[X])$ we know that if $f(x)$ is convex, then $\mathbb{E}[f(x)]$ is also convex. Therefore, $\mathbb{E}(\pi - \mathbf{a}^T\bar{\mathbf{y}})^+$ is convex. Since both parts of (3) are convex, we know that (3) is a convex optimization problem.

The dual problem of (1) is:

$$\begin{aligned} &min \sum_{i=1}^{m} b_i y_i + \sum_{j=1}^{n} \beta_j \\ s.t. \sum_{i=1}^{m} a_{ij} y_i + \beta_j \geq \pi_j, j = 1, ..., n. \\ &\beta_j \geq 0 \ \forall i, j \end{aligned}$$

From previous questions, we know that the primal optimal solution satisfies:

$$x_j^* = \begin{cases} 1, & \text{if } \pi_j \geq a_j^T y_k^* \\ 0, & \text{if } \pi_j \leq a_j^T y_k^* \end{cases}$$

The optimal solution $\mathbf{x_j}$ may take non-integer values. That means the optimal solution of primary problem (1) highly depends on $\mathbf{y}$. So by plugging the constraints $\sum_{i=1}^{m} a_{ij} y_i + \beta_j \geq \pi_j$ into the objective function, an equivalent form of the dual problem can be obtained as:

$$\sum_{i=1}^{m} b_i y_i + \sum_{j=1}^{n} (\pi_j - \sum_{i=1}^{m} a_{ij} y_i)^+$$
$$\text{s.t.} y_i \geq 0, i = 1, ..., m.$$

And if we divide it by n, from above we know that $\mathbf{d} = \frac{\mathbf{b}}{n}$, so we now have:

$$min f_n(\bar{y}) = \sum_{i=1}^{m} d_i y_i + \frac{1}{n} \sum_{j=1}^{n} (\pi_j - \sum_{i=1}^{m} a_{ij} y_i)^+$$
$$\text{s.t.} y_i \geq 0, i = 1, ..., m.$$

Since π, α is a sequence of i.i.d. random vectors, we can also write this formulation as:

$$\begin{aligned} &\text{minimize}_{\bar{\mathbf{y}}} \quad \mathbf{d}^T \bar{\mathbf{y}} + \mathbb{E}\left(\pi - \mathbf{a}^T \bar{\mathbf{y}}\right)^+, \\ &\text{s.t.} \qquad\qquad \bar{\mathbf{y}} \geq 0, \end{aligned} \tag{2}$$

where the expectation is taken with respect to (π, α).

For Question 5, Stochastic Gradient Decent (SGD) method is suggested, which can be particularly useful in large scale problems. SGD tries to lower the computation required per iteration (although the number of iterations may increase for convergence of the problem). In SGD we can select a random time step k where we can carry out the learning algorithm to correct the dual price vector y. The optimization problem is,

$$\begin{aligned} &\text{minimize}_{\bar{\mathbf{y}}} \quad \mathbf{d}^T \bar{\mathbf{y}} + \mathbb{E}\left(\pi - \mathbf{a}^T \bar{\mathbf{y}}\right)^+, \\ &\text{s.t.} \qquad\qquad \bar{\mathbf{y}} \geq 0, \end{aligned} \tag{3}$$

The derivative of the function is:

$\mathbf{f}(\bar{\mathbf{y}}) = \mathbb{E}\left(\mathbf{d} - \mathbf{a}\mathbb{I}_{\{\pi > \mathbf{a}^T \bar{\mathbf{y}}\}}\right)$.

Algorithm:

1. For k=1, initialize the dual price vector y as 0 (as mentioned in the problem).
2. For k=2, use $\beta = 1/\sqrt{2}$. M is the time step chosen for calculating dual price.

$$y^{k+1} = y^m - \beta^{-1} \nabla f(y^m)$$
$$y^{k+1} = 0\text{–}1/\sqrt{2}(\nabla f(0))$$

This will provide the next set of dual prices. Decision for next step can be made with the help of this dual price. Use this to calculate the revenue and compare this information with ground truth price.

3. For k=3, use $\beta = 1/\sqrt{3}$. For calculating the dual prices, pick out any time step information m between k=2 and k=3 randomly and use that to calculate the next set of dual prices with the formula mentioned in step 2.
4. For all following time steps k chose random time step m between 1 and k and chose $\beta = 1/\sqrt{k}$. Use the formula $y^{k+1} = y^m - \beta^{-1}\nabla f(y^m)$. Like if k = 50, chose any time step between 2-50 and calculate the dual prices with $\beta = 1/\sqrt{50}$. Rather than updating the dual price at every time step we can also update in every 10 or 20 or 30 time steps (using the mini batch concept will reduce the variance of the stochastic gradient).
5. The true minimizer is approached when $f(x^k) - f(x^*) \leq 2\beta||x^0 - x^*||/k$.
The step size that we pick up in SGD decides the problems convergence to the optimal solution. The smaller the step size the problem appears less stochastic, and it will lead to huge number of iterations with very less speed of convergence. Picking up the step size is crucial for the problem. As the stochastic model gives a quick convergence in the beginning hence it is particularly useful for big data problems, which works great on unseen data. As the Stochastic approach to gradient method helps in rapid convergence in the beginning, it fluctuates a lot near the optimal solution. But, the amount of noise can be controlled by the variance of the stochastic gradient; the smaller the variance the more the stochastic gradient is close to the true gradient.

References

[1] Shipra Agrawal and Nikhil R. Devanur. Fast Algorithms for Online Stochastic Convex Programming http://arxiv.org/abs/1410.7596, SODA2015

[2] S. Agrawal, E. Delage, M. Peters, Z. Wang and Y. Ye. A Unified Framework for Dynamic Prediction Market Design. Operations Research, 59:3 (2011) 550-568.

[3] S. Agrawal, Z. Wang and Y. Ye. A Dynamic Near-Optimal Algorithm for Online Linear Programming. http://arxiv.org/abs/0911.2974; to appear in Operations Research.

[4] Anupam Gupta and Marco Molinaro How the Experts Algorithm Can Help Solve LPs Online. https://arxiv.org/abs/1407.5298, 2014.

[5] Q. Huangfu and J. A. J. Hall. Parallelizing the dual revised simplex method. Mathematical Programming Computation, 10 (1), 119-142. DOI: 10.1007/s12532-017-0130-5, 2018.

[6] Thomas Kesselheim, Klaus Radke, Andreas Tönnis, Berthold Vöcking. Primal Beats Dual on Online Packing LPs in the Random-Order Model. https://arxiv.org/abs/1311.2578, STOC 2014.

[7] X. Li and Y. Ye. Online Linear Programming: Dual Convergence, New Algorithms, and Regret Bounds. https://arxiv.org/abs/1909.05499

[8] M. Peters, A. M-C. So and Y. Ye. Pari-mutuel Markets: Mechanisms and Performance. *The 3rd International Workshop On Internet And Network Economics*, 2007.