Learning from Rational Behavior: Predicting Solutions to Unknown Linear Programs

Shahin Jabbari, Ryan Rogers, Aaron Roth, Steven Wu

University of Pennsylvania

Motivating Example: Learning from Revealed Preferences

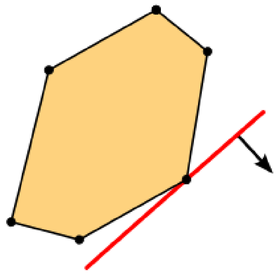
Unknown Constraints Problem:

- Store owner sees the same customer whose utility over d goods $u_i : \mathbb{R}^d \to \mathbb{R}$ is $u_i(\mathbf{x}) = \mathbf{v} \cdot \mathbf{x}$ is known, but the customer has a set of unknown constraints.
- At each day t, known prices are posted over the d goods, $\mathbf{p}^t \in \mathbb{R}^d_+$ and we know the budget of the customer b^t.
- Predict the feasible bundle of goods that maximizes the customer's utility.

Unknown Objective Problem:

- Each day t a subset of buyers $S^t \subseteq [n]$ enter a store, where each person has an unknown utility over d goods $u_i : \mathbb{R}^d \to \mathbb{R}$ where $u_i(\mathbf{x}) = \mathbf{v}^i \cdot \mathbf{x}$.
- Predict the feasible bundle of goods in known $\mathcal{P}^t$ that maximizes welfare of S^t.

General Problem: Learning from Rational Behavior



- At each round t, an agent optimizes a linear objective function subject to linear constraints (i.e. solves a linear program)
- Predict agent's response: the solution of an unknown linear program given partial feedback either about the constraints or the objective

Mathematical Model

Unknown Constraints Problem:
$\mathbf{v}, (\mathbf{p}^t, b^t)$ known, $(A, \mathbf{b})$ unknown

$$\mathbf{x}^{(t)} = \operatorname*{argmax}_{\mathbf{x}} \mathbf{v} \cdot \mathbf{x}$$
$$\text{s.t. } A\mathbf{x} \leq \mathbf{b}$$
$$\mathbf{p}^t \cdot \mathbf{x} \leq b^t$$

Unknown Objective Problem:
S^t, $\mathcal{P}^t$ known, $(\mathbf{v}^1, \cdots, \mathbf{v}^n)$ unknown

$$\mathbf{x}^{(t)} = \operatorname*{argmax}_{\mathbf{x}} \sum_{i \in S^t} \mathbf{v}^i \cdot \mathbf{x}$$
$$\text{s.t. } \mathbf{x} \in \mathcal{P}^t$$

Mistake Model:

For day $t = 1, 2, \ldots$

- Predict $\hat{\mathbf{x}}^{(t)}$, then see optimal $\mathbf{x}^{(t)}$
- If $\hat{\mathbf{x}}^{(t)} \neq \mathbf{x}^{(t)}$ then mistake at round t.
- Want to minimize total # of mistakes.

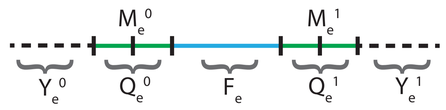
Assumptions

- Coefficients of the constraint matrix A can be represented with N bits of precision
- Constraint matrix A satisfies standard non-degeneracy assumptions

Unknown Constraint Problem: Main Algorithm - LearnEdge

We give an algorithm LearnEdge with mistake bound and running time polynomial in the *number of edges* of polytope $A\mathbf{x} \leq \mathbf{b}$, dimension d and precision N

- Intuition: Learn the feasible region on the edges of the polytope $A\mathbf{x} \leq \mathbf{b}$



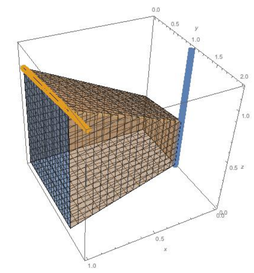
F_e: feasible region
Q_e: questionable region
Y_e: identified infeasible region

- Apply a *halving algorithm* on each edge of the unknown polytope - with each mistake, make progress towards learning the unknown feasible region
- Mistake bound that is polynomial in d when the number of rows of A is $d + O(1)$

Unknown Constraints Problem: Lower Bound and Extension

Lower Bound: Dependence on Bits of Precision N

- A lower bound of N on the number of mistakes of any algorithm when $d \geq 3$
- For $d = 1$ and 2 there exists efficient algorithms with finite mistake bounds independent of N

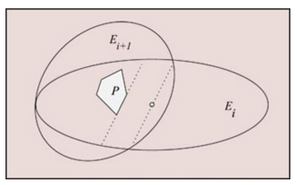


Extension: Stochastic Setting

- What if the constraints are chosen from an unknown distribution, rather than adversarially?
  - ▷ We give an efficient algorithm LearnHull with a mistake bound polynomial in the dimension d and the number of edge of polytope $A\mathbf{x} \leq \mathbf{b}$, but with *no dependence on the precision N*
  - ▷ Simply predict the point with the highest objective value from the convex hull of previously observed solutions, so it never makes infeasible guesses and the feasible region will always expand upon making a mistake

Unknown Objective Problem: Main Algorithm

We give an algorithm LearnEllipsoid with mistake bound and running time polynomial in the dimension d and precision N



- We leverage the ELLIPSOID algorithm.
- Start with an ellipsoid containing the set of all possible vectors v that parametrize the unknown objective function
- Each mistake provides a separating hyperplane for the ELLIPSOID to update
- Number of mistakes LearnEllipsoid makes is bounded by the number of updates of the ELLIPSOID algorithm.