The Price is (Probably) Right: Learning Market Equilibria from Samples

Omer Lev
Ben-Gurion University of the Negev
omerlev@bgu.ac.il

Neel Patel
University of Southern California
neelbpat@usc.edu

Vignesh Viswanathan
University of Massachusetts, Amherst
vviswanathan@umass.edu

Yair Zick
University of Massachusetts, Amherst
yzick@umass.edu

ABSTRACT

Equilibrium computation in markets usually considers settings where player valuation functions are known. We consider the setting where player valuations are *unknown*; using a PAC learning-theoretic framework, we analyze some classes of common valuation functions, and provide algorithms which output direct PAC equilibrium allocations, not estimates based on attempting to learn valuation functions. Since there exist trivial PAC market outcomes with an unbounded worst-case efficiency loss, we lower-bound the efficiency of our algorithms. While the efficiency loss under general distributions is rather high, we show that in some cases (e.g., unit-demand valuations), it is possible to find a PAC market equilibrium with significantly better utility.

KEYWORDS

Fisher Markets; PAC Learning; Market Equilibria

ACM Reference Format:
Omer Lev, Neel Patel, Vignesh Viswanathan, and Yair Zick. 2021. The Price is (Probably) Right: Learning Market Equilibria from Samples. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 9 pages.

1 INTRODUCTION

Do markets admit equilibrium allocations? This question has been extensively studied for many years [37]; more recently, the econ/CS community devoted significant effort to understanding when one can *efficiently compute* market equilibria. Much of this literature assumes that one has full access to player valuations over bundles of goods, an unrealistic assumption in many instances: combinatorial valuations are often difficult to elicit (especially for large markets), precluding any possibility of running full-information market algorithms. Machine learning techniques offer a compromise – assuming access to a *partial* dataset, we can learn player valuations, and use the learned valuations as a proxy. However, this approach raises several issues too: market valuations are often complex, and require a large number of samples to learn without overfitting. Moreover, even if we assume that player valuations have a simple structure, it is not immediately obvious that an *exact* equilibrium for the approximate valuations acts as an *approximate* equilibrium for the exact valuations; as we shall show, this may not be the case.

Our work explores a relatively new paradigm: instead of learning valuations, we focus on directly learning *market equilibria* from data. We build upon the framework of Jha and Zick [26], and adopt the *PAC solution learning* framework. Jha and Zick [26] show that in order to ensure that a market outcome (i.e. an allocation of items to players, as well as item prices) is likely to be a market equilibrium, it suffices to show that it is consistent with the data. That is, the prices and item allocation they induce are such that no player has a sample in the data they can afford and would rather have over their allocation. Finding a consistent market outcome is trivial: setting the price of all the goods to infinity would ensure consistency. However, this outcome would be very inefficient. Our goal is thus to learn *approximately efficient* PAC market equilibria.

Our Contribution. We study Fisher markets with indivisible goods under different classes of valuation functions, and propose algorithms which output an efficient PAC market equilibrium. That is, each player receives, with high probability, their most preferred affordable bundle of goods. We examine a variety of valuation classes: *unit-demand* (Section 4), *single minded* (Section 5), *additive* (Section 6) and *submodular* (Section 7) valuations. For each class, we provide a tight, distribution-independent, efficiency bound. We also show that, under more favorable distributions, we can achieve far better efficiency guarantees for unit-demand and additive valuations.

1.1 Related Work

There is a rich body of classical literature on market equilibria with indivisible goods [20, 23, 27, 32, 37], exchange economies [11] and Fisher markets [4, 12–14, 34]. In recent years there is a significant renewed interest in computing market outcomes, such as fair allocation [21, 29], optimal pricing [24, 35], approximate equilibria [10, 16] and markets with divisible goods [18]. However, the above do not address learning approximately efficient market solutions from data.

There exists a fast-growing body of literature on learning game-theoretic solutions from data, in cooperative games [6, 9, 25, 26, 36], auctions [15, 17, 19, 30] and optimization [7, 8, 33]. Some of this literature propose methods to learn market outcomes as well: Murray et al. [31] and Kroer et al. [28] examine the simpler case with divisible goods and additive valuations, Shen et al. [35] examine markets with a single item and Viqueira and Greenwald [38] propose a method to learn market outcomes indirectly from noisy

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.
© 2021 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

valuations. However, to the best of our knowledge, there exists no prior work that attempts to learn market outcomes in combinatorial markets from samples.

2 MODEL AND PRELIMINARIES

We study the *Fisher market* model; there is a set of *players*, $N = \{1, 2, \ldots, n\}$ and a set of *goods*, $G = \{g_1, g_2, \ldots, g_k\}$. Each player i has a *budget* $b_i \in \mathbb{R}^+$ and a *valuation function* $v_i : 2^G \to \mathbb{R}_+ \cup \{0\}$ which assigns a value $v_i(S)$ for each bundle of goods $S \subseteq G$. We assume that no two players have the same budget, and that $b_1 > \cdots > b_n$. This is a standard assumption, and is not a significant loss of generality: it is mostly done to induce some priority order among players, and ensure that equilibria exist. When budgets are equal, one can introduce small perturbations (this is the method used by Budish [16]). An *allocation* in such a market is a tuple $(\mathcal{A}, \vec{p})$, where $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$ is the allocation vector and $\vec{p} = \{p_1, p_2, \ldots, p_k\}$ is the price vector. Some A_is may be empty i.e. if $A_i = \emptyset$, then the player receives nothing. We define the *affordable set* $D_i(\vec{p}, b_i)$ as the set of affordable bundles for player i given a price vector $\vec{p}$:

$$D_i(\vec{p}, b_i) = \{S \subseteq G : \sum_{g_j \in S} p_j \leq b_i\}$$

An allocation is a *Walrasian* equilibrium (or simply an equilibrium) if all players are allocated the best possible set of goods they can afford i.e. $A_i \in D_i(\vec{p}, b_i)$ for all $i \in N$, and for all $i \in N$:

$$v_i(A_i) \in \arg\max\{v_i(S) : S \in D_i(\vec{p}, b_i)\}$$

Jha and Zick [26] define a learning-theoretic equilibrium notion based on the *probably approximately correct* (PAC) framework [2] called *PAC Equilibria*. An allocation $(\mathcal{A}, \vec{p})$ is a PAC Equilibrium if it is unlikely that a sample from a distribution $\mathcal{D}$ (over bundles in G), under the same prices, is both better than the current allocation and affordable for any player i. It is often easier to discuss learning-theoretic notions in terms of their *expected loss*; here, the loss is a function of player valuations v and budgets $\vec{b}$, a bundle $S \subseteq G$, and the proposed outcome $(\mathcal{A}, \vec{p})$:

$$L_{v,\vec{b}}(S, \mathcal{A}, \vec{p}) = \begin{cases} 1 & \text{if } \exists i \in N, v_i(A_i) < v_i(S) \\ & \wedge S \in D_i(\vec{p}, b_i) \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

We omit the v and $\vec{b}$ subscripts when they are clear from context. An allocation is an ϵ-PAC equilibrium with respect to $\mathcal{D}$ if its expected loss, denoted $L_{\mathcal{D}}(\mathcal{A}, \vec{p})$, is lower than ϵ

$$L_{\mathcal{D}}(\mathcal{A}, \vec{p}) \triangleq \mathbb{E}_{S\sim\mathcal{D}}[L_{v,\vec{b}}(S, A, \vec{p})] < \epsilon \quad (2)$$

ϵ-PAC equilibria are somewhat similar to ϵ-PAC approximations [2]: given a function $u : 2^G \to \mathbb{R}$, we say that $\bar{u}$ is an ϵ-PAC approximation of u w.r.t. $\mathcal{D}$ if $\Pr_{S\sim\mathcal{D}}[u(S) \neq \bar{u}(S)] < \epsilon$.

We follow a standard model of learning from samples: we are given players' budgets $b_1, b_2, \ldots, b_n$, m input samples $S_1, S_2, \ldots, S_m$ drawn i.i.d. from a distribution $\mathcal{D}$, and player valuations over the samples: $v_i(S_j)$ for all $i \in N$ and $j \in [m]$. Our goal is to find algorithms, whose input is a set of i.i.d. sampled bundles and valuations over them, that output a PAC Equilibrium (as per Equation (2)) with probability $\geq 1 - \delta$ (over the randomization of sampling m i.i.d. samples from $\mathcal{D}$). In other words, if $(\mathcal{A}, \vec{p})$ is the output of some learning algorithm, then the PAC guarantee is

$$\Pr_{S_1,\ldots,S_m \overset{\text{iid}}{\sim} \mathcal{D}} \left[L_{\mathcal{D}}(\mathcal{A}, \vec{p}) < \epsilon\right] \geq 1 - \delta$$

The number of samples needed, m, should be polynomial in the number of players, the number of goods, and in $\frac{1}{\epsilon}, \log\frac{1}{\delta}$. As mentioned in Section 1, PAC equilibria are not guaranteed to be efficient; in what follows we explore market *stability*, *envy* and allocative *efficiency*.

An allocation $(\mathcal{A}, \vec{p})$ is said to be *envy free* if all the players prefer their bundle to every other player's bundle that they can afford i.e. an allocation is envy free if for all $i, j \in N$:

$$v_i(A_i) \geq v_i(A_j) \vee A_j \notin D_i(\vec{p}, b_i) \quad (3)$$

Note that if $(\mathcal{A}, \vec{p})$ is a Walrasian equilibrium, then (3) is trivially true. The *efficiency ratio* of an allocation $\mathcal{A}$ is the ratio of the total welfare (or utility) of $\mathcal{A}$ to that of the optimal equilibrium allocation i.e.

$$ER_v(\mathcal{A}) = \frac{\sum_{i=1}^n v_i(A_i)}{\sum_{i=1}^n v_i(A_i^*)} \quad (4)$$

where $\mathcal{A}^*$ is a welfare-maximizing equilibrium allocation; Unlike simpler settings (e.g. rent division [22]), market outcomes need not maximize social welfare in Fisher markets with indivisible goods.

Due to space constraints, some of the proofs have been omitted or replaced by proof sketches. The full proofs can be found in the appendix.

3 COMPUTING PAC EQUILIBRIA

We first discuss some sufficient conditions for finding a PAC Equilibrium from samples, starting with a simple observation: if we are able to approximate player valuation functions v using an underestimate $\bar{v}$, then any *exact* equilibrium for $\bar{v}$ is a PAC equilibrium for v.

Proposition 3.1. *Let $v_1, \ldots, v_n : 2^G \to \mathbb{R}$ be a player valuation profile; let $(\bar{v}_i)_{i\in N}$ be $\frac{\epsilon}{n}$-PAC approximations of $(v_i)_{i\in N}$ w.r.t. $\mathcal{D}$, such that for all $i \in N$ and all $S \subseteq G$, $\bar{v}_i(S) \leq v_i(S)$. If $(\mathcal{A}, \vec{p})$ is a market equilibrium under $\bar{v}$, then $(\mathcal{A}, \vec{p})$ is an ϵ PAC equilibrium for v w.r.t. $\mathcal{D}$.*

Jha and Zick [26] prove that a PAC equilibrium can be directly learned using only $O(k)$ samples if one can efficiently compute a *consistent solution*, that is, a market outcome that has zero loss on the samples. More precisely, we say that a mechanism $\mathcal{M}$ outputs a *consistent solution* if for any given set of samples $\mathcal{S} = \{S_1, \ldots, S_m\}$, $\mathcal{M}$ outputs $(\mathcal{A}, \vec{p})$ such that the *empirical loss* $\hat{L}(\mathcal{A}, \vec{p})$ is 0:

$$\hat{L}(\mathcal{A}, \vec{p}) \triangleq \frac{1}{m}\sum_{j=1}^m L(S_j, \mathcal{A}, \vec{p}) = 0$$

Theorem 3.2 (Jha and Zick [26]). *Suppose that an algorithm $\mathcal{M}$ takes as input a set of m samples of goods $\mathcal{S}$ drawn i.i.d. from an unknown distribution $\mathcal{D}$, and outputs a consistent equilibrium allocation. If $m \in O\left(\frac{1}{\epsilon}\left(k\log\frac{1}{\epsilon} + \log\frac{1}{\delta}\right)\right)$ then the allocation output by $\mathcal{M}$ is an ϵ-PAC Equilibrium w.p. $\geq 1 - \delta$.*

Proposition 3.1 and Theorem 3.2 provide two paths to computing PAC market equilibria: either compute an equilibrium for a PAC

underestimate, or directly learn market outcomes from samples. As we mention above, our objective is finding market outcomes with provable social welfare guarantees, with respect to the *true* player valuation profile.

4 UNIT DEMAND MARKETS

We begin our exploration with a fundamental class of market valuations: unit-demand markets. In a unit demand market, the value of each bundle $S \subseteq G$ is the value of the most valuable good in S, i.e. for all $i \in N$, $v_i(S) = \max_{g \in S} v_i(\{g\})$. We make the standard assumption that players have distinct values for goods, i.e. that $v_i(\{g\}) \neq v_i(\{g'\})$ if $g \neq g'$; this is mostly done to break ties (see Budish [16]). Unit demand markets correspond to room/housing allocation scenarios where each tenant can only stay in a single room/buy a single home [1, 3, 22], or to gaming "loot boxes", in which players care mainly about the most valuable item.

The standard data-driven approach is to PAC learn the valuation functions, and output an equilibrium allocation for the learned valuations. We refer to this method as *indirect learning*, and to outcomes computed in this manner as indirectly learned outcomes. For unit demand markets this can be done quite easily, by estimating the value of each item as the value of the least valuable sample that contains it (creating a PAC approximation for the valuations), and then allocating the items first to the player with the largest budget, who gets their most valued item; then the player with the second largest budget, who gets their most valued item which is still available, and so on .

Such an algorithm has a simple guarantee of efficiency:

Proposition 4.1. *If $(\mathcal{A}, \vec{p})$ is the output of an algorithm calculating PAC approximations of unit demand valuations and then allocating goods in decreasing order of player budgets, then $ER_{\mathcal{V}}(\mathcal{A}) \geq \frac{1}{\sigma}$ where $\sigma = \max_{i \in N} \frac{\max_{g \in G} v_i(\{g\})}{\min_{g \in G} v_i(\{g\})}$, the maximal ratio between a players valuation for two different items.*

The main drawback with such an algorithm is that it does not output a PAC Equilibrium. Consider the example below:

Example 4.1. Consider a setting where $N = \{1, 2\}$ and $G = \{g_1, g_2, g_3\}$. Player budgets are $b_1 = 2, b_2 = 1$. Player valuations satisfy

$$v_1(\{g_1, g_2\}) = 5; v_1(\{g_3\}) = 3$$
$$v_2(\{g_1, g_2\}) = 4; v_2(\{g_3\}) = 2$$

We observe a distribution $\mathcal{D}$ which samples uniformly at random two sets: $\{g_1, g_2\}$ and $\{g_3\}$. We can thus reasonably assume that we observe both bundles with high probability after a small number of i.i.d. samples. Approximating preferences would yield:

$$\bar{v}_1(\{g_1\}) = \bar{v}_1(\{g_2\}) = 5; \bar{v}_1(\{g_3\}) = 3$$
$$\bar{v}_2(\{g_1\}) = \bar{v}_2(\{g_2\}) = 4; \bar{v}_2(\{g_3\}) = 2$$

A valuation-approximating algorithm allocates one item from g_1, g_2 to player 1 and the other to player 2, and allocates g_3 to player 2. We set the price of g_1 to 2 and the price of g_2 to 1. Assume w.l.o.g. that g_1 is assigned to player 1; it is possible that $v_1(\{g_1\}) = 0$ and $v_1(\{g_2\}) = 5$, in which case player 1 demands g_3. In that case, the probability of observing a sample (namely $\{g_3\}$) which player 1 demands is $\frac{1}{2}$, not an arbitrarily low $\epsilon > 0$, so this approach does not yield an ϵ-PAC equilibrium.

The bad behavior in Example 4.1 is not due to some intrinsic failure of the valuation-approximating algorithm; it is impossible to learn a consistent underestimate of a unit demand valuation. Consider again the setting in Example 4.1: it is impossible to determine whether $v_1(\{g_1\}) = 5$ or $v_1(\{g_2\}) = 5$; indeed, the only viable underestimate sets both items' values to 0. However, doing so yields $\bar{v}_1(\{g_1, g_2\}) = 0 < v_1(\{g_1, g_2\})$, an inconsistency. To conclude, the indirect approach does not yield a PAC Equilibrium. Let us turn our attention to directly learning PAC market outcomes from samples. We refer to this method as *direct solution learning*, and any outcome computed from this method as a directly learned equilibrium. Algorithm 1 directly learns a PAC equilibrium in the unit-demand setting.

Algorithm 1 iterates over all players in decreasing order of budget, and allocates the smallest bundle of goods from all available goods with the highest possible value. We use two properties of unit demand valuations, formalized in the following lemma.

Lemma 4.2. *Given two bundles of goods $S, T \subseteq G$ and some player $i \in N$ with unit demand valuations, if no two goods have the same value for i then*

(1) If $v_i(S) = v_i(T) = c$ then $v_i(S \cap T) = c$ as well.
(2) If $v_i(S) > v_i(T)$ then $v_i(S) = v_i(S \setminus T)$

Using Lemma 4.2, we identify the smallest most valued bundle B_i^1 for player i, and allocate it to the player if it contains no previously allocated items; otherwise, we remove all such samples from $\mathcal{S}$, since we know such items are already priced out of their budget by previous players, and we cannot use them to get information on the next most valued set of goods for this player. We continue to identify the next most valued bundle of minimal size for player i. We repeat this process until we identify the smallest subset of most valued items among unallocated items. If we allocate a bundle to i after t steps, then player i receives a bundle containing their t-th most valuable good – denoted B_i^t; we then price the items in B_i^t such that their total price is b_i. Note that all samples that contain B_i^t have a price of $\geq b_i$, which guarantees that no player $i' > i$ can afford them.

We repeat this procedure for all players. At the end of the ***for loop*** (Algorithm 1, line 3), we allocate any leftover goods to player n for free, and assign any good which is not present in the sample set to player 1 at a price of 0.

We first show that Algorithm 1 outputs a consistent outcome.

Theorem 4.3. *Algorithm 1 outputs a consistent market outcome.*

Proof. Let the output of Algorithm 1 be $(\mathcal{A}, \vec{p})$. Let us assume there is a sample $S \in \mathcal{S}$ such that for some player i, $v_i(S) > v_i(A_i)$; we need to show that $S \notin D_i(\vec{p}, b_i)$. Consider the items not available to player i when it is their turn to select a bundle, referred to as *Alloc* in Algorithm 1. If $S \cap Alloc \neq \emptyset$, then S must contain some previously allocated bundle $A_{i'}$, where $b_{i'} > b_i$; thus the price of S is greater than b_i, and S is not demanded by i. If S can be allocated to player i and is one of the most valued bundles at the time, player i selects their bundle (i.e. $S \in \mathcal{L}_i^t$), then $B_i^t \subseteq S$; in particular, $v_i(S) = v_i(B_i^t)$. Otherwise, $v_i(B_i^t) > v_i(S)$ therefore $v_i(A_i) \geq v_i(S)$ and i would not demand S. □

Algorithm 1: Directly Learning Equilibria for Unit Demand Valuations

Input: A set of samples $\mathcal{S}$, player valuations and budgets $b_1 > \cdots > b_n$

Alloc $\leftarrow \emptyset$
Allocate unobserved goods to player 1 at price 0
for $i \leftarrow 1$ ***to*** n **do**
  $\mathcal{S}_i^1 \leftarrow \mathcal{S}$; $c \leftarrow$ *False*; $t \leftarrow 1$
  while $c =$ *False* **do**
    $C_i^t \leftarrow$ some set in $\arg\max_{T \in \mathcal{S}_i^t} v_i(T)$
    $\mathcal{L}_i^t \leftarrow \{T \in \mathcal{S}_i^t \,|\, v_i(T) = v_i(C_i^t)\}$
    $B_i^t \leftarrow \bigcap_{T \in \mathcal{L}_i^t} T$
    $B_i^t = B_i^t \setminus \bigcup_{T \in \mathcal{S} | v_i(T) < v_i(C_i^t)} T$
    if $B_i^t \cap$ *Alloc* $\neq \emptyset$ **then**
      $t \leftarrow t+1$; $\mathcal{S}_i^t \leftarrow \mathcal{S}_i^{t-1} \setminus \mathcal{L}_i^t$
    end
    else
      $c \leftarrow$ *True*; *Alloc* $\leftarrow$ *Alloc* $\cup B_i^t$
      $A_i \leftarrow B_i^t$ and price of each $g \in B_i^t$ is $\frac{b_i}{|B_i|}$
    end
  end
end
Allocate the leftover goods to player n at price 0

While Algorithm 1 outputs a consistent outcome, it offers an efficiency guarantee of $\frac{1}{\min\{n,k\}}$, under the minor assumption that player valuations are normalised with respect to their budget.

Proposition 4.4. *If for all $i \in N$, $\max_{g \in G} v_i(\{g\}) = b_i$, Algorithm 1 outputs an allocation $(\mathcal{A}, \vec{p})$ with $ER_v(\mathcal{A}) \geq \frac{1}{\min\{n,k\}}$.*

Proposition 4.4 offers a rather weak bound: the same efficiency ratio can be achieved by allocating all goods to the player with the highest budget. However, the bound is tight, and is an outcome of "bad" distributions. We show that there exists sample sets for which no allocation can guarantee an efficiency greater than $\frac{1}{\min\{n,k\}}$.

Theorem 4.5. *Let $\mathcal{S}, v(\mathcal{S})$ be a set of samples along with its valuations; let $\mathcal{V}$ be the set of unit demand valuation profiles consistent with the set of samples and are budget normalised i.e. $\max_{g \in G} v_i(\{g\}) = b_i$ for all the players $i \in N$, and $\mathcal{B} \subset \mathbb{R}_+^n$ be the set of all feasible budgets i.e. the set of all budgets in $\mathbb{R}_+^n$ such that $b_1 > b_2 > \cdots > b_n$. Then, we have*

$$\min_{v \in \mathcal{V}} \max_{\mathcal{A}} \min_{\mathcal{S} \subseteq 2^G, \vec{b} \in \mathcal{B}} ER_v(\mathcal{A}) \leq \frac{1}{\min\{n,k\} - \delta}$$

for any $\delta \in (0, n)$ where $\mathcal{A}$ is a consistent allocation with respect to the samples.

Proof Sketch. Suppose the only sample you have is the set of goods G. Then the only consistent allocation which can guarantee a non-zero efficiency is one which allocates the entire set of goods to player 1. Any allocation which partitions the set of goods and allocates it to multiple players cannot offer any efficiency guarantees. This leaves us with allocating all goods to player 1. There exist allocations which provide a total utility of $\geq (\min\{n,k\} - \delta) \times b_1$; however, allocating the set of goods to player 1 guarantees a utility of b_1, which yields the upper bound $\frac{1}{\min\{n,k\}-\delta}$. □

While Algorithm 1 offers no reasonable welfare guarantees for general distributions, its performance guarantees improve significantly under certain distributional assumptions. Specifically, this holds true if $\mathcal{D}$ is a product distribution with a bounded probability of sampling each good. Recall that $\mathcal{D}$ is a product distribution over G if there exist values $p_1, \ldots, p_k \in [0,1]$ such that for every $S \subseteq G$, $\Pr_{\mathcal{D}}[S] = \prod_{g_j \in S} p_j$. Product distributions offer more amenable welfare guarantees for two reasons: first, by definition, the presence of a particular good in the sample is independent of the presence of any other good (offering us a better chance of observing players' valuations for individual items); second, goods are sampled with non-zero probability (thus we observe all goods in some bundle with high probability). Theorem 4.8 shows that Algorithm 1 outputs a PAC equilibrium with an efficiency ratio of 1 with exponentially high probability, when samples are drawn i.i.d. from a product distribution; the proof requires that player preference orders over items are sufficiently distinct. Before we prove Theorem 4.8, we present two technical results – Lemma 4.6 and Lemma 4.7 – which we use to prove Theorem 4.8.

Lemma 4.6. *In unit demand markets with unequal budgets and strict preferences over items, any equilibrium allocation assigns player i the best possible available good, i.e. $\{g_i^*\}$ equals $\arg\max_{g \in \mathcal{G}_i} v_i(g)$ ($\mathcal{G}_1 = \arg\max_{g \in G} v_1(g)$ and for $i > 1$, $\mathcal{G}_i = G \setminus \{\bigcup_{l=1}^{i-1} \mathcal{G}_l\}$). Moreover, all equilibria have the same social welfare $\sum_i v_i(g_i^*)$.*

In Lemma 4.6, we show that the social welfare for any equilibrium for unit demand players is unique and each player i gets the good g_i^*. Therefore to show that the efficiency of Algorithm 1 is 1 with high probability, it is sufficient to show that Algorithm 1 assigns g_i^* for all i with high probability.

We now present Lemma 4.7, in which we prove that for any player i, if $\mathcal{S}_i^t$ at t−th iteration of the ***while loop*** in Algorithm 1 contains more than k^2 samples then the corresponding B_i^t contains only the best available good for player i in $\bigcup_{S \in \mathcal{S}_i^t} S$, with high probability.

Lemma 4.7. *Suppose that $\mathcal{D}$ is a product distribution such that for all $g \in G$, $1 - \sqrt{2e^{-1/k} - 1} < \Pr_{S \in \mathcal{D}}(g \in S) < \frac{1}{2} + \frac{\sqrt{2e^{-1/k}-1}}{2}$. If $|\mathcal{S}_i^t| \geq k^2$ (at the t−th iteration of the **while loop** in Algorithm 1 for player i), the corresponding B_i^t equals $\{\hat{g}_i\}$ to player i with at least $1 - e^{-\frac{k}{2}}$ probability, where*

$$\hat{g}_i \in \arg\max\{v_i(\{g\}) : g \in \bigcup_{S \in \mathcal{S}_i^t} S\}$$

We are now ready to prove Theorem 4.8. We show that when we assume agent preferences sufficiently differ – no two agents have exactly the same favorite $O(\log(\max(n,k)))$ goods – Algorithm 1 is optimal with high probability..

Theorem 4.8. *Suppose that $\mathcal{D}$ is a product distribution, such that $\Pr_{S \sim \mathcal{D}}[g \in S] \in [\alpha, \beta]$. Assume that for every agent i, $|\{g \in G : v_i(g) > v_i(g_i^*)\}| < \frac{\max\{\log n, \log k\}}{\log(\frac{1}{1-\beta})}$* [1].

[1] g_i^* is defined as in Lemma 4.6: $\{g_i^*\} = \arg\max_{g \in \mathcal{G}_i} v_i(g)$ ($\mathcal{G}_1 = \arg\max_{g \in G} v_1(g)$ and for $i > 1$, $\mathcal{G}_i = G \setminus \{\bigcup_{l=1}^{i-1} \mathcal{G}_l\}$.

If $k > 3$, $1 - \sqrt{2e^{-1/k} - 1} \leq \alpha$ and $\beta \leq \frac{1}{2} + \frac{\sqrt{2e^{-1/k}-1}}{2}$, the output of Algorithm 1, $(\mathcal{A}, \vec{p})$, satisfies

$$\Pr[ER_v(\mathcal{A}) = 1] \geq 1 - \frac{2n \max\{\log n, \log k\}}{\log\left(\frac{1}{(1-\beta)}\right)} e^{-\frac{\max\{k,n\}}{4}}$$

Proof Sketch. When we have $\max\{k^4, n^2k^2\}$ samples, if the condition on the valuation functions is satisfied, then for every player i, there is some t for which $|S_i^t| \geq k^2$ and $g_i^* \in \arg\max\{v_i(\{g\}) : g \in \bigcup_{S \in \mathcal{S}_i^t} S\}$. Therefore, using induction and Lemma 4.7, every player gets allocated g_i^* w.h.p., resulting in an efficiency ratio of 1. □

As β decreases (provided $\beta > 1 - \sqrt{2e^{-1/k} - 1}$), the condition in Theorem 4.8 on the difference between players' preferences becomes less stringent. Moreover, if either n or k is large, the exponential term in the probability guarantee dominates, and Algorithm 1 is highly likely to output an efficient outcome. However, if β is smaller, the efficiency guarantee is less likely to hold. Note that when $\beta = 1$, i.e., there is a good g that appears in all samples, the performance of Algorithm 1 depends on which player gets g. If the most preferred good for all players is g, Algorithm 1 allocates g to player 1 and will not be able to continue: it is impossible to identify the second preferred good (and beyond). Therefore, Algorithm 1 has an efficiency $\geq \frac{1}{\rho n}$ since we can only guarantee that the highest budget player will receive their optimal equilibrium allocation.

We can generalize the efficiency bound in Theorem 4.8 for any preference order over the items for all players. We observe that with at least $\max(k^4, n^2k^2)$ samples, the first $\sim \max(\log k, \log n)$ players will be assigned g_i^* with high probability. We show the efficiency guarantee for algorithm 1 for any preference order in Proposition 4.9, and its connection to the disparity in valuation functions between agents.

Proposition 4.9. *If $\mathcal{D}$ is a product distribution such that for all $g_j \in G$, $1 - \sqrt{2e^{-1/k} - 1} < \Pr_{S \in \mathcal{D}}(g \in S) < \frac{1}{2} + \frac{\sqrt{2e^{-1/k}-1}}{2}$ and $k > 3$. Then, with exponentially high probability, Algorithm 1 allocates goods with an efficiency ratio $ER_v(\mathcal{A}) \geq \frac{\log n}{\rho n \log\left(\frac{1}{1-\beta}\right)}$ where $\rho = \max_{g \in G} \frac{\max_{i \in N} v_i(g)}{\min_{i \in N} v_i(g)}$ and $\beta = \max_{g_j \in G} \Pr_{S \in \mathcal{D}}(g \in S)$.*

Furthermore, in Corollary 4.10, we show the efficiency bound when each good is sampled i.i.d. w.p. $\frac{1}{2}$.

Corollary 4.10. *If the distribution $\mathcal{D}$ is uniform over the set 2^G and $k > 3$, with exponentially high probability. Algorithm 1 allocates goods with an efficiency $ER_v(\mathcal{A}) > \frac{\log n}{\rho n}$ where*

$$\rho = \max_{g \in G} \frac{\max_{i \in N} v_i(g)}{\min_{i \in N} v_i(g)}$$

using a polynomial number of samples.

5 SINGLE MINDED MARKETS

In single minded markets, each player has a particular bundle of goods, $D_i \subseteq G$ they desire; every bundle that does not contain D_i has no value i.e.

$$v_i(S) = \begin{cases} 1 & D_i \subseteq S \\ 0 & \text{otherwise.} \end{cases}$$

We show that a PAC underestimate for single-minded valuations can be efficiently learned, and an equilibrium for single-minded valuations can be efficiently computed. Therefore, using Proposition 3.1, a PAC Equilibrium is computable in polynomial time.

Proposition 5.1. *The class of single minded valuation functions can be efficiently PAC learned, such that the learned valuation function weakly underestimates players' true valuations.*

Proof. From a given set of samples $\mathcal{S}$, set $\bar{D}_i = \bigcap_{S \in \mathcal{S}: v_i(S) > 0} S$. If, for a player $i \in N$, no sample has $v_i(S) > 0$, then set $\bar{D}_i = G$. This learned valuation is consistent and weakly lower than the actual valuations since $D_i \subseteq \bar{D}_i$ (i.e., a sample containing a set of items that is in D_i but not all of $\bar{D}_i$ will be given a value 0 instead of 1).

The total number of possible valuation functions, i.e., size of the hypothesis class, is 2^k (the number of possible choices for D_i). Thus, in order to PAC-learn D_i, we need a number of samples polynomial in $\frac{1}{\epsilon}$, $\log \frac{1}{\delta}$ and $\log |\mathcal{H}| \in O(k)$ (a classic learning result for finite hypothesis classes, see Anthony and Bartlett, 1999). □

Brânzei et al. [14] present an algorithm to compute equilibria under equal budgets. We extend this algorithm to settings where all budgets are different.

Theorem 5.2. *Algorithm 2 outputs a market equilibrium for single minded players with all different budgets.*

Proof. Algorithm 2 iteratively allocates goods while keeping track of players' remaining budgets. If a good is demanded by multiple players, it is priced such that only one player can afford it, and allocated to that player. The `SetPrice` function ensures that no two players have the same remaining budget, by slightly increasing the price; this ensures that there are no ties when selecting the next player to allocate a good to.

All players either get their desired set or a subset of their desired set if it is unaffordable. Thus the resulting allocation is an equilibrium: players who do not receive their desired set are not able to afford it. □

The key difference between our approach and that of Brânzei et al. [14] is how over-demanded goods are priced. Brânzei et al. [14] assign the good to the player with the smallest desired set at a price equal to their budget. In our case, player budgets differ and therefore, ties cannot be broken by desired set size; rather, we instead break ties by remaining budgets.

Computing an equilibrium with total welfare at least K has been shown to be NP-hard by Brânzei et al. [13] when players have equal budgets. In Theorem 5.3, we show this for our setting as well.

Theorem 5.3. *It is NP-Complete to decide if a single minded market has an equilibrium with total welfare at least K*

Theorem 5.5 shows that despite this, it is possible to compute a PAC equilibrium with an efficiency ratio $\geq \frac{1}{\min\{n,k\}}$. We now turn to establishing the efficeincy bounds of the algorithm.

Lemma 5.4. *Algorithm 2 assigns at least one player its desired set.*

Theorem 5.5. *Let $(\mathcal{A}, \vec{p})$ be the output of Algorithm 2 on valuations learned as in Proposition 5.1; then $ER_v(\mathcal{A}) \geq \frac{1}{\min\{n,k\}}$.*

Algorithm 2: Competitive Equilibrium for Single Minded Valuations and Different Budgets

Notation: b_i^* is the remaining budget for player i; prices are represented by $\vec{p}$.

$\vec{p} = \vec{0}$; $\vec{b}^* = \{b_1, b_2, \ldots, b_n\}$
$\mathcal{B} = \{D_1, D_2, \ldots, D_n\}$
for *each* $g_j \in G$ **do**
  if g_j *is only demanded by one player* **then**
    Allocate g_j to that player at $p_j = 0$
  end
  else if g_j *is demanded by multiple players* **then**
    $p_j \leftarrow$ `SetPrice`$(g_j, \vec{b}^*, \mathcal{B})$
    Allocate g_j to the player that can afford it at price p_j
    `UpdateDemand` $(\mathcal{B}, \vec{p}, \vec{b}^*)$
  end
end
Allocate all unallocated goods to player n at price 0
Function `SetPrice` $(g_j, \vec{b}^*, \mathcal{B})$**:**
  $s = \arg\max_{i \in N \wedge g_j \in D_i} b_i^*$
  $t = \arg\max_{i \in N \setminus s \wedge g_j \in D_i} b_i^*$
  $p_j = b_t^* + \frac{b_s^* - b_t^*}{n^2}$
  $b_s^* = b_s^* - p_j$
  while $\exists i \neq s : b_i^* = b_s^*$ **do**
    $b_s^* = b_s^* - \frac{b_s^* - b_t^*}{n^2}$, $p_j = p_j + \frac{b_s^* - b_t^*}{n^2}$
  end
  return p_j
Function `UpdateDemand` $(\mathcal{B}, \vec{p}, \vec{b}^*)$**:**
  for $i \in N$ **do**
    if $(D_i \neq \emptyset) \wedge (p(D_i) > b_i^*)$ **then**
      $D_i = \emptyset$
    end
  end

Proof. From Lemma 5.4, we get that at least one player will receive his desired set. This desired set is the learned desired set which is a superset of the actual desired set (see Proposition 5.1). Therefore, the player who receives his learned desired set also receives his actual desired set. This means that the total welfare obtained is at least 1. The maximum welfare any allocation can obtain is $\min\{n, k\}$ since the total number of players getting their desired set is upper bounded by k and n. Thus, the efficiency of the computed PAC Equilibrium is $\geq \frac{1}{\min\{n,k\}}$ □

Similar to unit demand markets, we show that our result in Theorem 5.5 is tight and no algorithm can guarantee a better efficiency.

Theorem 5.6. *Let $\mathcal{S}, v(\mathcal{S})$ be a set of samples along with its valuations, $\mathcal{V}$ be the set of single minded valuation function profiles which are consistent with the set of samples and $\mathcal{B} \subset \mathbb{R}_+^n$ be the set of all feasible budgets i.e. the set of all budgets in $\mathbb{R}_+^n$ such that $b_1 > b_2 > \cdots > b_n$. Then, we have*

$$\min_{v \in \mathcal{V}} \max_{\mathcal{A}} \min_{\mathcal{S} \subseteq 2^G, \vec{b} \in \mathcal{B}} ER_v(\mathcal{A}) \leq \frac{1}{\min\{n, k\}}$$

where $\mathcal{A}$ is a consistent allocation with respect to the samples.

We also show that our learned allocations are envy free.

Proposition 5.7. *Let $(\mathcal{A}, \vec{p})$ be the output of Algorithm 2 on valuations learned as in Proposition 5.1; then $(\mathcal{A}, \vec{p})$ is envy free.*

6 ADDITIVE MARKETS

In additive markets, each player has additive valuations. The valuation of a bundle is equal to the sum of the valuations of every good in that bundle: $v_i(S) = \sum_{g \in S} v_i(\{g\})$. While additive valuations are PAC-Learnable, we cannot use Proposition 3.1 to learn a PAC-Equilibrium since in a lot of cases, we cannot learn an underestimate of the valuations. This can be seen using Example 4.1.

Although additive Fisher markets with indivisible goods have recently received a lot of attention, there are still many open questions regarding the efficient computation of a market clearing equilibrium. Babaioff et al. [4] examine the specific case where there are only two players and Brânzei et al. [13] show that it is computationally intractable to decide if a market has a competitive equilibrium when budgets are equal. This dearth of positive algorithmic results means that even if we could accurately learn the valuation of each good (which is not guaranteed and depends on the samples), we may not be able to compute an equilibrium in polynomial time. In this paper, we take a different approach and attempt to learn an equilibrium directly (using Theorem 3.2); however, our outcome is not necessarily market clearing.

Our approach is described in Algorithm 3. The algorithm has three steps. First, we pre-process the samples to ensure that there are no proper subsets in the samples. This is done to ensure that no sample which is a superset of another sample is allocated. We can remove the supersets and replace them by the set difference between the superset and the subset: we can derive the value of this bundle under additive valuations, as executed in the function PreProcess.

The second step allocates samples to players. To each player, the algorithm allocates the favourite sample among all the unallocated samples. Here, a sample is unallocated if no good in the sample has been allocated. It then prices each good equally such that the total price is equal to the budget of the player.

The last step ensures consistency, it checks each of the original samples to see if a player prefers it over their own sample and can afford it. If there exists such a player, the algorithm proceeds to set the price of one of the goods in the sample to infinity to ensure that no player can afford it. This good is chosen as follows: if the sample has an unallocated good, then the unallocated good is chosen. If the sample does not have an unallocated good, the algorithm takes away a good from the sample which belonged to the player with the least budget and then sets its price to infinity. We refer to the act of setting the price of a good to infinity as burning a good.

It is easy to see because of the third step that the algorithm is always consistent. It also worth noting that as long as we can underestimate the valuation in Line 25 in Algorithm 3, we will always end up with a consistent outcome. This means that this algorithm could be modified for any class of valuations to output a consistent outcome.

We now prove two efficiency bounds for our algorithm. These bounds hold only for additive valuations. To start with, we show that when the valuations are budget normalised, then the efficiency

Algorithm 3: Consistent Allocation For Additive Markets

Input: A set of samples $\mathcal{S}$, player valuations for these samples $v(\mathcal{S})$ and budgets $b_1 > b_2 > \cdots > b_n$

$\mathcal{S}', \tilde{v}(\mathcal{S}')$ = PreProess$(\mathcal{S}, v(\mathcal{S}))$
for $i \leftarrow 1$ *to* n **do**
  $B_i =$ some set in $\arg\max_{T \in \mathcal{S}'} \tilde{v}_i(T)$
  Allocate B_i to player i i.e. $A_i = B_i$, $\tilde{v}_i(A_i) = \tilde{v}_i(B_i)$
  Set $p_g = \frac{b_i}{|B_i|}$ $\forall g \in B_i$
  $\mathcal{S}' = \mathcal{S}' \setminus \bigcup_{S \in \mathcal{S}': S \cap B_i \neq \emptyset} S$
end
while $\exists i \in N, S \in \mathcal{S}$ *s.t* $\tilde{v}_i(A_i) < v_i(S) \wedge \sum_{g \in S} p_g \leq b_i$ **do**
  if $\exists g \in S$ *s.t.* $g \notin \bigcup_{i \in N} A_i$ **then**
    Set $p_g = \infty$
  end
  else
    $j = \arg\min_{i \in N: S \cap A_i \neq \emptyset} b_i$
    $g =$ any good in $A_j \cap S$
    $p_g = \infty$
    $A_j = A_j \setminus \{g\}$
    $\tilde{v}_j(A_j) = 0$
  end
end
Allocate all leftover goods to player 1 at price 0
Function PreProess$(\mathcal{S}, v(\mathcal{S}))$**:**
  $\mathcal{S}' = \mathcal{S}$
  $\tilde{v}(\mathcal{S}') = v(\mathcal{S})$
  while $\exists S', S'' \in \mathcal{S}'$ *s.t.* $S' \subsetneq S''$ **do**
    $\tilde{v}_i(S'') = \tilde{v}_i(S'') - \tilde{v}_i(S')$ $\quad \forall i \in N$
    $S'' = S'' \setminus S'$
  end
  return $\mathcal{S}', \tilde{v}(\mathcal{S}')$

is inversely related to the number of goods. Before that, we show that no good in player 1's initially allocated sample gets taken away in Lemma 6.1.

Lemma 6.1. *In Algorithm 3, no good in player 1's initially allocated sample gets taken away.*

Theorem 6.2. *When* $\forall i \in N, \max_{g \in G} v_i(\{g\}) = b_i$*, then Algorithm 3 outputs an allocation with* $ER_v(\mathcal{A}) \geq \frac{1}{k}$

Proof. Algorithm 3 always ensures the first player has a bundle with valuation at least b_1. If the first player's favourite good is not present in any sample, he receives at price 0 resulting in a valuation of at least b_1.

If the first player's favourite good is present in the samples, then there exists a sample (with the first player's favourite good in it) which is valued at at least b_1 by the first player. Since the first player is allocated his favourite sample, he is given a bundle whose value is at least b_1. By Lemma 6.1, none of these goods are taken away and their final utility is at least b_1.

Since the largest amount of value a good can give a player is b_1. The total utility of any allocation is upper bounded by kb_1. This gives us the following bound:

$$ER_v(\mathcal{A}) = \frac{\sum_{i \in N} v_i(A_i)}{\sum_{i \in N} v_i(A_i^*)} \geq \frac{v_1(A_1)}{\sum_{i \in N} v_i(A_i^*)} \geq \frac{b_1}{kb_1} = \frac{1}{k}$$

□

We now show that this bound is tight for general distributions.

Theorem 6.3. *Let* $\mathcal{S}, v(\mathcal{S})$ *be a set of samples along with its valuations,* $\mathcal{V}$ *be the set of additive valuation function profiles which are consistent with the set of samples and are budget normalised i.e.* $\max_{g \in G} v_i(\{g\}) = b_i$ *for all the players* $i \in N$ *and* $\mathcal{B} \subset \mathbb{R}^n_+$ *be the set of all feasible budgets i.e. the set of all budgets in* $\mathbb{R}^n_+$ *such that* $b_1 > b_2 > \cdots > b_n$*. Then, we have*

$$\min_{v \in \mathcal{V}} \max_{\mathcal{A}} \min_{\mathcal{S} \subseteq 2^G, \vec{b} \in \mathcal{B}} ER_v(\mathcal{A}) \leq \frac{1}{k - \delta}$$

for any $\delta \in (0, k)$ *where* $\mathcal{A}$ *is a consistent allocation with respect to the samples.*

Our next bound does not require the valuations to be normalised but imposes conditions on the samples and depends on the disparity in the valuations of goods.

In Proposition 6.4, we show that when samples are disjoint, the efficiency varies inversely with the disparity in valuations

Proposition 6.4. *When all the samples in* $\mathcal{S}$ *are pairwise disjoint, then Algorithm 3 outputs an allocation with* $ER_v(\mathcal{A}) \geq \frac{1}{\rho}$ *where* $\rho = \max_{g \in G} \frac{\max_{i \in N} v_i(\{g\})}{\min_{i \in N} v_i(\{g\})}$

7 SUBMODULAR MARKETS

In submodular markets, each player has monotone submodular valuations i.e. each player's valuation function $v_i : 2^G \mapsto \mathbb{R}^+ \cup \{0\}$ satisfies the following three conditions:

(a) $v_i(\emptyset) = 0$
(b) For any two $S, T \subseteq G$ such that $S \subseteq T$, $v_i(S) \leq v_i(T)$.
(c) For any two $S, T \subseteq G$,

$$v_i(S) + v_i(T) \geq v_i(S \cup T) + v_i(S \cap T) \tag{5}$$

The class of monotone submodular valuations contains the class of additive valuations, as well as many others. This increase in complexity comes with an even greater dearth of positive algorithmic results. In addition to this, monotone submodular valuations cannot be efficiently PAC learned [5]. So, we cannot use Proposition 3.1 to learn a PAC Equilibrium.

We, instead, use a direct learning approach similar to that of additive markets but modify our algorithm slightly due to two reasons. First, the pre-process step that worked for additive valuations will not work for submodular valuations since we cannot accurately determine the value of the bundle that results when you remove a subset from a set. However, we can underestimate it using equation (5) as follows: given two sets $A, B \subseteq G$ such that $A \subseteq B$, then by substituting $S = B \setminus A$ and $T = A$ in equation (5) we get

$$v_i(B \setminus A) \geq v_i(B) - v_i(A)$$

Therefore, $v_i(B) - v_i(A)$ gives us an underestimate of $v_i(B \setminus A)$. Furthermore, the inequality does not change if we replace $v_i(B)$ with an underestimate of $v_i(B)$.

Second, because we have to underestimate valuations, our efficiency guarantee may not hold. In order to prevent this, we modify our algorithm so that it can use extra information about the valuations. This is done using an additional input parameter c_i for all $i \in N$ which specifies an underestimate of the value of the highest

valued good i.e., for all $i \in N$: $c_i \leq \max_{g \in G} v_i(\{g\})$. Note that when there is no available information about the value of c_i, we can set $c_i = 0$.

The algorithm has been described in Algorithm 4. The algorithm has the same three steps as that of Algorithm 3 but the first two steps are modified to work for submodular valuations.

The PreProcess step removes any supersets from the set $\mathcal{S}$ and replaces them with the set difference between the superset and the subset. It also computes the set of goods which could have a value $\geq c_i$ and stores it in the set F_i. Note that F_i is never empty and has a value of at least c_i to player i. The following lemma proves it.

Lemma 7.1. *In the set $\mathcal{F}$ output by the PreProcess function of Algorithm 4, $F_i \neq \emptyset$ and $v_i(F_i) \geq c_i \quad \forall i \in N$.*

We then use this in the second step to give a player a bundle of value at least c_i when no other sample guarantees a value of at least c_i. Of course, this is not applicable when an element of F_i has been allocated to some other player.

The third step remains the same and ensures consistency since $\tilde{v}_i$ is an underestimate of v_i. So, if for any $S \in \mathcal{S}$, $v_i(S) > v_i(A_i)$, then, $v_i(S) > \tilde{v}_i(A_i)$.

We now show that when valuations are budget normalised, then the algorithm has an efficiency of at least $\frac{1}{k}$. But before we do that, we show that even in this algorithm, none of player 1's goods get taken away.

Lemma 7.2. *In Algorithm 4, none of player 1's goods get taken away.*

This brings us to our final proof. When we have budget normalised valuations, then Algorithm 4 gives us an allocation with efficiency at least $\frac{1}{k}$

Theorem 7.3. *When $\max_{g \in G} v_i(\{g\}) = b_i$, then Algorithm 4 outputs an allocation with efficiency $ER_v(\mathcal{A}) \geq \frac{1}{k}$*

Since additive valuations are a subset of monotone submodular valuations, Theorem 6.3 applies in this case as well. This means the bound in Theorem 7.3 is tight.

Algorithm 4: Submodular Markets Consistent Allocation

Input: A set of samples $\mathcal{S}$, player valuations for these samples $v(\mathcal{S})$, budgets $b_1 > b_2 > \cdots > b_n$ and $c_i \leq \max_{g \in G} v_i(\{g\}) \forall i \in N$

$\mathcal{S}', \tilde{v}(\mathcal{S}'), \mathcal{F} =$ `PreProess`$(\mathcal{S}, v(\mathcal{S}))$
for $i \leftarrow 1$ *to* n **do**
  if $\tilde{v}_i(S') < c_i \ \forall S' \in \mathcal{S}' \wedge F_i \cap \bigcup_{j=1}^{i-1} A_j = \emptyset$ **then**
    Allocate F_i to player i i.e. $A_i = F_i$, $\tilde{v}_i(A_i) = c_i$
  end
  else
    $B_i =$ some set in $\arg\max_{T \in \mathcal{S}'} \tilde{v}_i(T)$
    Allocate B_i to player i i.e. $A_i = B_i$, $\tilde{v}_i(A_i) = \tilde{v}_i(B_i)$
  end
  Set $p_g = \frac{b_i}{|A_i|} \ \forall g \in A_i$
  $\mathcal{S}' = \mathcal{S}' \setminus \bigcup_{S \in \mathcal{S}' : S \cap A_i \neq \emptyset} S$
end
while $\exists i \in N, S \in \mathcal{S}$ *s.t* $\tilde{v}_i(A_i) < v_i(S) \wedge \sum_{g \in S} p_g \leq b_i$ **do**
  if $\exists g \in S$ *s.t.* $g \notin \bigcup_{i \in N} A_i$ **then**
    Set $p_g = \infty$
  end
  else
    $j = \arg\min_{i \in N : S \cap A_i \neq \emptyset} b_i$
    $g =$ any good in $A_j \cap S$
    $p_g = \infty$
    $A_j = A_j \setminus \{g\}$
    $\tilde{v}_j(A_j) = 0$
  end
end
Allocate all leftover goods to player 1 at price 0
Function `PreProess`$(\mathcal{S}, v(\mathcal{S}))$**:**
  $\mathcal{S}' = \mathcal{S}$
  $\tilde{v}(\mathcal{S}') = v(\mathcal{S})$
  for $S' \in \mathcal{S}'$ **do**
    while $\exists S \in \mathcal{S}$ *s.t.* $S \subsetneq S'$ **do**
      $\tilde{v}_i(S') = \tilde{v}_i(S') - v_i(S) \quad \forall i \in N$
      $S' = S' \setminus S$
    end
  end
  if $\exists S', S'' \in \mathcal{S}'$ *s.t.* $S' \subsetneq S''$ **then**
    Remove S' from $\mathcal{S}'$
  end
  $\mathcal{F} = \{F_1, F_2, \ldots, F_n\}$
  $F_i = \left(\bigcup_{S \in \mathcal{S} : v_i(S) \geq c_i} S \setminus \bigcup_{S' \in \mathcal{S} : v_i(S') < c_i} S' \right) \cup \left(G \setminus \bigcup_{S \in \mathcal{S}} S \right) \quad \forall i \in N$
  return $\mathcal{S}', \tilde{v}(\mathcal{S}'), \mathcal{F}$

8 CONCLUSIONS AND FUTURE WORK

This work shows the benefit of directly learning equilibrium states, instead of learning utility functions, and calculating equilibria states from them. We deal with several valuation function families, and in all of them show algorithms to produce a PAC-approximation, with our results being tight, i.e., no better approximation can be guranteed.

We believe that this work is the tip of the iceberg in showing how PAC learning can help in reaching economic, game-theoretic results, directly from the data, without using the data to construct intermediate steps (such as learning utility functions). Plenty of problems are still open – from expanding results to a larger family of functions (XOS, gross substitutes), to further type of results (e.g., other desirable states beyond equilibria).

ACKNOWLEDGEMENTS

Lev, Patel and Zick were supported by the Singapore NRF Research Fellowship #R-252-000-750-733. Patel and Zick were also supported by the AI Singapore Award #AISG-RP-2018-009. Viswanathan was supported by the IITKGP Foundation Award. Most of the work was done while all the authors were at the National University of Singapore. The authors would also like to thank the anonymous reviewers of AAAI 2020, AAMAS 2020 and AAMAS 2021 for their informative comments.

REFERENCES

[1] Ahmet Alkan, Gabrielle Demange, and David Gale. Fair allocation of indivisible goods and criteria of justice. *Econometrica*, 59(4):1023–1039, 1991.

[2] M. Anthony and P. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999.

[3] Enriqueta Aragones. A derivation of the money rawlsian solution. *Social Choice and Welfare*, 12(3):267–276, 1995.

[4] M. Babaioff, N. Nisan, and I. Talgam-Cohen. Competitive equilibria with indivisible goods and generic budgets. *arXiv preprint arXiv:1703.08150*, 2017.

[5] Maria-Florina Balcan and Nicholas J.A. Harvey. Learning submodular functions. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC)*, 2011.

[6] M.F. Balcan, A.D. Procaccia, and Y. Zick. Learning cooperative games. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 475–481, 2015.

[7] Eric Balkanski and Yaron Singer. Minimizing a submodular function from samples. In *Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NIPS)*, pages 814–822, 2017.

[8] Eric Balkanski and Yaron Singer. The sample complexity of optimizing a convex function. In *Proceedings of the 30th Conference on Computational Learning Theory (COLT)*, pages 275–301, 2017.

[9] Eric Balkanski, Umar Syed, and Sergei Vassilvitskii. Statistical cost sharing. In *Proceedings of the 30th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 6221–6230, 2017.

[10] Siddharth Barman and Sanath Krishnamurthy. On the proximity of markets with integral equilibria. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI)*, pages 1748–1755, 2019.

[11] S. Bikhchandani and John W. Mamer. Competitive equilibrium in an exchange economy with indivisibilities. *Journal of Economic Theory*, 74(2):385 – 413, 1997.

[12] Allan Borodin, Omer Lev, and Tyrone Strangway. Budgetary effects on pricing equilibrium in online markets. In *Proceedings of the 15th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 95–103, Singapore, 2016.

[13] Simina Brânzei, Hadi Hosseini, and Peter Bro Miltersen. Characterization and computation of equilibria for indivisible goods. In *Algorithmic Game Theory*, pages 244–255, 2015.

[14] Simina Brânzei, Yuezhou Lv, and Ruta Mehta. To give or not to give: Fair division for single minded valuations. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 123–129, 2016.

[15] Gianluca Brero, Benjamin Lubin, and Sven Seuken. Combinatorial auctions via machine learning-based preference elicitation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 128–136, 2018.

[16] Eric Budish. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economics*, 119(6):1061–1103, 2011.

[17] Richard Cole and Tim Roughgarden. The sample complexity of revenue maximization. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 243–252, 2014.

[18] Nikhil R Devanur, Christos H Papadimitriou, Amin Saberi, and Vijay V Vazirani. Market equilibrium via a primal–dual algorithm for a convex program. *Journal of the Association for Computing Machinery*, 55(5):22, 2008.

[19] Nikhil R. Devanur, Zhiyi Huang, and Christos-Alexandros Psomas. The sample complexity of auctions with side information. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing (STOC)*, pages 426–439, 2016.

[20] Edmund Eisenberg. Aggregation of utility functions. *Management Science*, 7(4): 337–350, 1961.

[21] Alireza Farhadi, Mohammad Ghodsi, MohammadTaghi Hajiaghayi, Sébastien Lahaie, David Pennock, Masoud Seddighin, Saeed Seddighin, and Hadi Yami. Fair allocation of indivisible goods to asymmetric agents. *Journal of Artificial Intelligence Research*, 64(1):1–20, 2019. ISSN 1076-9757.

[22] Ya'akov (Kobi) Gal, Moshe Mash, Ariel D. Procaccia, and Yair Zick. Which is the fairest (rent division) of them all? *Journal of the Association for Computing Machinery*, 64(6):39:1–39:22, 2017.

[23] F. Gul and E. Stacchetti. Walrasian equilibrium with gross substitutes. *Journal of Economic Theory*, 87(1):95–124, 1999.

[24] Venkatesan Guruswami, Jason D. Hartline, Anna R. Karlin, David Kempe, Claire Kenyon, Frank McSherry, and Frank McSherry. On profit-maximizing envy-free pricing. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1164–1173, 2005.

[25] Ayumi Igarashi, Jakub Sliwinski, and Yair Zick. Forming probably stable communities with limited interactions. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI)*, pages 2053–2060, 2019.

[26] Tushant Jha and Yair Zick. A learning framework for distribution-based game-theoretic solution concepts. In *Proceedings of the 21st ACM Conference on Economics and Computation (EC)*, pages 355–377, 2020.

[27] Alexander S. Kelso and Vincent P. Crawford. Job matching, coalition formation, and gross substitutes. *Econometrica*, 50(6):1483–1504, 1982. ISSN 00129682, 14680262. URL http://www.jstor.org/stable/1913392.

[28] Christian Kroer, Alexander Peysakhovich, Eric Sodomka, and Nicolas E Stier-Moses. Computing large market equilibria using abstractions. In *Proceedings of the 19th ACM Conference on Economics and Computation (EC)*, 2019.

[29] David Kurokawa, Ariel D. Procaccia, and Junxing Wang. When can the maximin share guarantee be guaranteed? In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI)*, pages 523–529, 2016.

[30] Jamie H Morgenstern and Tim Roughgarden. On the pseudo-dimension of nearly optimal auctions. In *Proceedings of the 28th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 136–144, 2015.

[31] Riley Murray, Christian Kroer, Alex Peysakhovich, and Parikshit Shah. Robust market equilibria with uncertain preferences. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[32] Renato Paes-Leme and Sam Chiu-Wai Wong. Computing walrasian equilibria: Fast algorithms and structural properties. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 632–651, 2017.

[33] Nir Rosenfeld, Eric Balkanski, Amir Globerson, and Yaron Singer. Learning to optimize combinatorial functions. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 4374–4383, 2018.

[34] Erel Segal-Halevi. Competitive equilibrium for almost all incomes. In *Proceedings of the 16th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 1267–1275, 2017.

[35] Weiran Shen, Sebastien Lahaie, and Renato Paes-Leme. Learning to clear the market. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 5710–5718, 2019.

[36] J. Sliwinski and Y. Zick. Learning hedonic games. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2730–2736, 2017.

[37] H. Varian. Equity, envy, and efficiency. *Journal of Economic Theory*, 9:63–91, 1974.

[38] Enrique Areyan Viqueira and Amy Greenwald. Learning competitive equilibria in noisy combinatorial markets. In *Proceedings of the 2nd Games, Agents, and Incentives Workshop (GAIW@AAMAS 2020)*, 2020.