Nash-Bargaining-Based Models for Matching Markets: One-Sided and Two-Sided; Fisher and Arrow-Debreu

Mojtaba Hosseini[1] ✉ 🏠 🆔
The Paul Merage School of Business, University of California, Irvine, CA, USA

Vijay V. Vazirani ✉ 🏠 🆔
Computer Science Department, University of California, Irvine, CA, USA

──── **Abstract** ────

This paper addresses two deficiencies of models in the area of matching-based market design. The first arises from the recent realization that the most prominent solution that uses cardinal utilities, namely the Hylland-Zeckhauser (HZ) mechanism [22], is intractable; computation of even an approximate equilibrium is PPAD-complete [32, 8]. The second is the extreme paucity of models that use cardinal utilities, in sharp contrast with general equilibrium theory.

Our paper addresses both these issues by proposing Nash-bargaining-based matching market models. Since the Nash bargaining solution is captured by a convex program, efficiency follow; in addition, it possesses a number of desirable game-theoretic properties. Our approach yields a rich collection of models: for one-sided as well as two-sided markets, for Fisher as well as Arrow-Debreu settings, and for a wide range of utility functions, all the way from linear to Leontief.

We also give very fast implementations for these models which solve large instances, with $n = 2000$, in one hour on a PC, even for a two-sided matching market. A number of new ideas were needed, beyond the standard methods, to obtain these implementations.

1 Introduction

Within the area of matching-based market design, the most prominent solution that uses cardinal utilities[2] is the Hylland-Zeckhauser (HZ) mechanism [22]. It is based on creating parity between demand and supply, i.e., it uses the power of a pricing mechanism, which gives it attractive properties: the allocations produced satisfy Pareto optimality and envy-freeness [22] and the mechanism is incentive compatible in the large [21].

A serious drawback of HZ, from the viewpoint of practical applicability, is lack of computational efficiency: the recent papers [32] and [8] show that the problem of computing even an approximate equilibrium is PPAD-complete. More precisely, [32] showed membership

---

[1] Corresponding author

[2] For a brief comparison of cardinal and ordinal utilities for matching markets, see Section 1.4.

in PPAD and remarked that it will not be surprising if intractability sets in even for the highly special case in which utilities of agents come from a trivalued set, say $\{0, \frac{1}{2}, 1\}$; for bivalued sets, they gave an efficient algorithm. Next, [8] showed PPAD-hardness even for the case that utilities of agents come from a four-valued set; the trivalued case is open.

The second issue addressed by this paper is a deficiency of the area of matching-based market design itself, namely the extreme paucity of models that use cardinal utilities. This stands in sharp contrast with general equilibrium theory, which has defined and extensively studied several fundamental market models to address a number of specialized and realistic situations. HZ can be seen as corresponding to the most elementary model in that theory, namely the linear Fisher model. A model corresponding to the linear Arrow-Debreu market model was also studied by Hylland and Zeckhauser [22]; however, they ended their investigation on finding instances that do not admit an equilibrium. Considering these difficulties, studying further generalizations made little sense. In particular, we are not aware of any two-sided matching market models that use cardinal utilities.

Our paper addresses both these issues by proposing Nash-bargaining-based matching market models. As is well known, the Nash bargaining solution is captured as in optimal solution to a convex program. Therefore, if for a specific game, a separation oracle can be implemented in polynomial time, then using the ellipsoid algorithm, one can get as good an approximation as desired in time that is polynomial in the number of bits of accuracy required [20, 33]. For all models defined in this paper, the constraints of the convex program are linear, thereby ensuring zero duality gap and easy solvability.

The game-theoretic properties of the the Nash bargaining solution include: it satisfies Pareto optimality and symmetry, and since it maximizes the product of the utilities of agents, the allocations it produces are remarkably fair. The latter has been noted by several researchers [7, 2, 27] and has been further explored under the name of Nash Social Welfare [10, 9]. Compared to HZ, we have sacrificed envy-freeness for this fairness property – the two are incomparable, with neither dominating the other. We have also sacrificed incentive compatibility in the large, but have gained efficient solvability. Clearly, without the latter, the nice properties of HZ have little meaning, since the mechanism is unusable except for extremely small instances, perhaps not exceeding $n = 10$.

Another major gain from the move to Nash bargaining is that it yields a plethora of matching market models, not only one-sided but also two-sided; for the Fisher as well as the Arrow-Debreu settings, with the latter being not much harder than the former. Furthermore, our models cover a large range of utility functions, all the way from linear to Leontief.

For the two reasons given above, namely computational efficiency and richness of models, we have proposed a shift from a pricing mechanism to a Nash-bargaining-based mechanism for matching market models. The following two questions arise: Is this shift a principled one, i.e., is there a fundamental connection between the two types of models? Is either type of mechanism reducible to the other? Section 1.1 provides answers to these questions.

We note that the origins of the idea of operating markets via Nash bargaining go back to [31]. For the linear case of the Arrow-Debreu market model, instead of seeking allocations via a pricing mechanism, [31] formulated it as a Nash bargaining game and gave a combinatorial, polynomial time algorithm for solving the underlying convex program.

As is well known, polynomial time solvability is often just the beginning of the process of obtaining an "industrial grade" implementation. Towards this end, we give very fast implementations as well as experimental results for all five of our one-sided market models and the most basic two-sided model; the more general two-sided markets are analogous to the rest of the one-sided markets. In particular, our implementation can solve very large

instances, with $n = 2000$, in one hour even for a two-sided matching market. In Section 1.3 we have described how the standard methods needed to be adapted to the special intricacies of our settings, in order to obtain these very fast implementations.

In contrast, an HZ equilibrium, in particular, the equilibrium price, is not captured by any known mathematical construct, regardless of its computational complexity. The only known method for conducting an exhaustive search for obtaining an HZ equilibrium is algebraic cell decomposition [4]; its use for computing HZ equilibria was studied in [3]. Each iteration of this method is time-consuming. This, together with the exhaustive search required, makes it viable for only very small values of n, not exceeding 10.

The recent computer science revolutions of the Internet and mobile computing led to the launching of highly impactful and innovative matching markets such as Adwords, Uber and Airbnb, and in turn led to a major revival of the area of matching-based market design, e.g., see [16]. It is safe to assume that innovations will keep coming in the future and that new models, with good algorithmic properties, may be needed at any time in the future. Our work was motivated by these considerations.

## 1.1 The Connection between HZ and Nash-Bargaining-Based Models

In this section, we answer the two questions raised above by attempting a comparative study of one-sided matching markets under the two types of models. The answer to the second question is "No" since under an affine transformation of the utilities of agents, the Nash bargaining solution and an HZ equilibrium change in fundamentally different ways: Whereas the former solution undergoes the same affine transformation (see Section 2.1), the latter remains unchanged, as shown in [32]. The answer to the first question is "Yes", due to the connection established in [30]. We provide a brief synopsis of the argument below.

First consider the linear Fisher market model defined in Section 2.2. The setup of the *linear Fisher problem (LFP)* is identical, except that the agents don't have any money, so this is not really a market model. The problem is to design a polynomial time mechanism for distributing all the goods among the agents so that the allocation satisfies Pareto optimality.

$$
\begin{aligned}
\max \quad & \sum_{i \in A} \log\left(\sum_{j \in G} u_{ij} x_{ij}\right) \\
\text{s.t.} \quad & \sum_{i \in A} x_{ij} \leq 1 \quad \forall j \in G, \\
& \boldsymbol{x} \geq 0.
\end{aligned}
\tag{1}
$$

[30] give two such mechanisms. The first is to give each agent 1 Dollar, thereby transforming LFP to the linear Fisher market model, and ask for an equilibrium allocation, which satisfies Pareto optimality. This can be obtained in polynomial time, via a combinatorial algorithm [11], or by expressing it as a convex program. The latter is the celebrated Eisenberg-Gale convex program [14], given in (1).

The second is to view LFP as a Nash bargaining problem; Pareto optimality is one of the axioms which it satisfies, see Section 2.1. This is done by defining a convex, compact set $\mathcal{N} \subseteq \mathbb{R}_+^n$, called the feasible set, and a point $\boldsymbol{c} \in \mathcal{N}$, called the disagreement point, see Section 2.1 for details. In this case, $\boldsymbol{c} = 0$, and $\mathcal{N}$ will consist of all possible vectors of utilities to the n agents that can be obtained by partitioning 1 unit each of all m goods among the agents. It is easy to see that the resulting convex program will be precisely Eisenberg-Gale convex program. Therefore, the two mechanisms are identical!
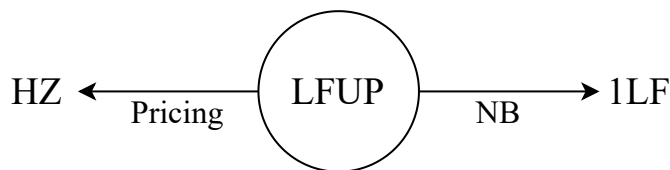
Figure 1 Figure illustrating connection between HZ and NB.

Next, [30] define the *linear Fisher unit demand problem (LFUP)* to be LFP with the additional requirements that $m = n$ and that each agent should get a total of one unit of goods. As a result, every feasible allocation is a fractional perfect matching over the n agents and n goods.

Now it turns out that when LFUP is solved via the pricing mechanism, it is identical to HZ, and when it is solved via the Nash bargaining mechanism it is identical to *1LF*, i.e., our most basic Nash-bargaining-based model, see Section 3.1. This establishes a strong connection between HZ and the Nash-bargaining-based models and is illustrated in Figure 1.

1.2 Our Results

In Section 3.1, we give four basic models for one-sided matching markets covering a wide range of utility functions. For each model, we also give a natural application. In Section 3.2 we give a model for the most basic two-sided matching market. This model can be easily enhanced to four more models in a manner analogous to the other four one-sided matching market models given in Section 3.1.

In Section 4, we give convex programs capturing the Nash-bargaining-based solution for all the models mentioned above. These convex programs can be solved to ϵ precision in time that is polynomial in the size of the input and $\log 1/\epsilon$ via ellipsoid-based methods [18, 33].

In Section 5, we present two solution schemes for solving these convex programs. Our methods, namely cutting-plane method [24, 33] and Frank-Wolfe method [17, 23], rely on linear approximations of the convex programs. We present enhancement techniques as well as an overview of the way structural properties of these problems can be exploited.

To demonstrate the effectiveness of these methods in handling large-scale instances of the problems, we performed extensive computational experiments in Section 6 and tested the algorithms on instances of up to 2000 agents/goods and 10 segments for the piecewise linear utility functions. In particular, the Frank-Wolfe algorithm is well-suited for matching market models with linear utilities, and is capable of producing sparse optimal solutions. The cutting-plane algorithm is able to produce optimal or near-optimal solutions for the more challenging problems of one-sided market models with non-linear utility functions.

1.3 Ideas Needed beyond Standard Methods

Our solution methods, namely cutting-plane algorithm and Frank-Wolfe (FW) algorithm, rely on iterative linear approximations of convex programs for the one-sided and two-sided market models. For efficient implementation of these algorithms, one needs to pay attention to the structural properties of these models as described below.

We implement a central cutting-plane algorithm (CCP), which not only guarantees a linear convergence rate, but also produces more effective cuts, since central points are more likely to be in the relative interior of the feasible region. Additionally, straightforward implementations of CCP are often prone to numerical instabilities. For instance, if the cut

coefficients are of different scales, the solvers may not handle the cuts properly. We avoid this by choosing proper scales for the cut coefficients. Secondly, since the objective function of the convex programs involves the logarithm function, we require positive utilities for each agent at each iteration of CCP. However, since CCP is an outer-approximation algorithm, it is possible that in an iteration of CCP, the utilities of some agents may become zero, which makes the solution unbounded, and one cannot extract a cut based on this solution. We resolve this issue by taking a convex combination of the current point and some feasible interior point. The latter point is obtained by choosing the closest feasible point to the current point on the line segment from current point to the interior point.

We also implement a Frank-Wolfe algorithm for solving instances of the matching markets with linear utilities. An interesting property of these models is that once the nonlinear objective function of the respective convex programs are replaced by linear functions, the resulting problems can be solved as matching problems. The solution produced by FW is therefore a sparse convex combination of a set of integral perfect matchings.

1.4 Related Results

Recently [32] gave the first comprehensive study of the computational complexity of HZ. They gave an example which has only irrational equilibria; as a consequence, this problem is not in PPAD. They showed membership of the exact equilibrium computation problem in FIXP and approximate equilibrium in PPAD. They also gave a combinatorial, strongly polynomial time algorithm for computing an equilibrium for the case of dichotomous utilities, i.e., 0/1 utilities, and they extended this result to the case of bivalued utilities, i.e., each agent's utility for individual goods comes from a set of cardinality two, though the sets may be different for different agents.

Next, [8] showed PPAD-hardness even for the case that utilities of agents come from a four-valued set; the trivalued case is open.

The success of our implementations, using available solvers, naturally raises the question of finding efficient combinatorial algorithms for our proposed market models. The subsequent paper [30] has given such algorithms, based on the techniques of multiplicative weights update (MWU) and conditional gradient descent (CGD), for several of our one-sided and two-sided models. They also defined and developed algorithms for the non-bipartite matching market model which has applications to the roommate problem. Lastly, they gave the connection between HZ and the Nash-bargaining-based models stated in Section 1.1.

The extension of one-sided matching markets to the setting in which agents have initial endowments of goods, called the Arrow-Debreu setting, has several natural applications beyond the original Fisher setting, e.g., allocating students to rooms in a dorm for the next academic year, assuming their current room is their initial endowment. The issue of obtaining such an extension of the HZ scheme, was studied by Hylland and Zeckhauser. However, this culminated in an example which inherently does not admit an equilibrium [22].

As a recourse, [12] introduced the notion of an *α-slack Walrasian equilibrium*. This is a hybrid between the Fisher and Arrow-Debreu settings in which agents have initial endowments of goods and for a fixed $\alpha \in (0, 1]$, the budget of each agent, for given prices of goods, is $\alpha + (1 - \alpha) \cdot m$, where m is the value for her initial endowment. Via a non-trivial proof, using the Kakutani Fixed Point Theorem, they proved that an α-slack equilibrium always exists. A pure Arrow-Debreu model was proposed in [19] by suitably relaxing the notion of an equilibrium to an *ε-approximate equilibrium*. Their proof of existence of equilibrium follows from that of [12].

An interesting recent paper [2] defines the notion of a random partial improvement mechanism for a one-sided matching market. This mechanism truthfully elicits the cardinal preferences of the agents and outputs a distribution over matchings that approximates every agent's utility in the Nash bargaining solution.

In recent years, several researchers have proposed Hylland-Zeckhauser-type mechanisms for a number of applications, e.g., see [6, 21, 25, 26]. The basic scheme has also been generalized in several different directions, including two-sided matching markets, adding quantitative constraints, and to the setting in which agents have initial endowments of goods instead of money, see [12, 13].

Ordinal vs cardinal utilities. Under ordinal utilities, the agents provide a total preference order over the goods and under cardinal utilities, they provide a non-negative real-valued function. Both forms have their own pros and cons and neither dominates the other. Whereas the former is easier to elicit from agents, the latter is far more expressive, enabling an agent to not only report if she prefers good A to good B but also by how much. [1] exploit this greater expressivity of cardinal utilities to give mechanisms for school choice which are superior to ordinal-utility-based mechanisms.

Example 1, taken from [19], provides a very vivid illustration of the advantage of cardinal utilities over ordinal ones in one-sided matching markets.

▶ **Example 1.** The following example illustrates the advantage of cardinal vs ordinal utilities. The instance has three types of goods, T_1, T_2, T_3, and these goods are present in the proportion of (1%, 97%, 2%). Based on their utility functions, the agents are partitioned into two sets A_1 and A_2, where A_1 constitute 1% of the agents and A_2, 99%. The utility functions of agents in A_1 and A_2 for the three types of goods are (1, ϵ, 0) and (1, $1 - \epsilon$, 0), respectively, for a small number $\epsilon > 0$. The main point is that whereas agents in A_2 marginally prefer T_1 to T_2, those in A_1 overwhelmingly prefer T_1 to T_2.

Clearly, the ordinal utilities of all agents in $A_1 \cup A_2$ are the same. Therefore, a mechanism based on such utilities will not be able to make a distinction between the two types of agents. On the other hand, the HZ mechanism, which uses cardinal utilities, will fix the price of goods in T_3 to be zero and those in T_1 and T_2 appropriately so that by-and-large the bundles of A_1 and A_2 consist of goods from T_1 and T_2, respectively.

2   Preliminaries

2.1   The Nash Bargaining Game

An *n-person Nash bargaining game* consists of a pair $(\mathcal{N}, \boldsymbol{c})$, where $\mathcal{N} \subseteq \mathbb{R}^n_+$ is a compact, convex set and $\boldsymbol{c} \in \mathcal{N}$. The set $\mathcal{N}$ is called the *feasible set* – its elements are vectors whose components are utilities that the n players can simultaneously accrue. Point $\boldsymbol{c}$ is the *disagreement point* – its components are utilities which the n players accrue if they decide not to participate in the proposed solution.

The set of n agents will be denoted by A and the agents will be numbered $1, 2, \ldots n$. Instance $(\mathcal{N}, \boldsymbol{c})$ is said to be *feasible* if there is a point in $\mathcal{N}$ at which each agent does strictly better than her disagreement utility, i.e., $\exists \boldsymbol{v} \in \mathcal{N}$ such that $\forall i \in A$, $v_i > c_i$, and *infeasible* otherwise. In game theory it is customary to assume that the given Nash bargaining problem $(\mathcal{N}, \boldsymbol{c})$ is feasible; we will make this assumption as well.

The solution to a feasible instance is the point $\boldsymbol{v} \in \mathcal{N}$ that satisfies the following four axioms:

1. *Pareto optimality:* No point in $\mathcal{N}$ weakly dominates $\boldsymbol{v}$.
2. *Symmetry:* If the players are renumbered, then a corresponding renumber the coordinates of $\boldsymbol{v}$ is a solution to the new instance.
3. *Invariance under affine transformations of utilities:* If the utilities of any player are redefined by multiplying by a scalar and adding a constant, then the solution to the transformed problem is obtained by applying these operations to the particular coordinate of $\boldsymbol{v}$.
4. *Independence of irrelevant alternatives:* If $\boldsymbol{v}$ is the solution to $(\mathcal{N}, \boldsymbol{c})$, and $\mathcal{S} \subseteq \mathbb{R}_+^n$ is a compact, convex set satisfying $\boldsymbol{c} \in \mathcal{S}$ and $\boldsymbol{v} \in \mathcal{S} \subseteq \mathcal{N}$, then $\boldsymbol{v}$ is also the solution to $(\mathcal{S}, \boldsymbol{c})$.

Via an elegant proof, Nash proved:

▶ **Theorem 2** (Nash [28])**.** *If the game $(\mathcal{N}, \boldsymbol{c})$ is feasible then there is a unique point in $\mathcal{N}$ satisfying the axioms stated above. Moreover, this point is obtained by maximizing $\Pi_{i \in A}(v_i - c_i)$ over $\boldsymbol{v} \in \mathcal{N}$.*

Nash's solution to his bargaining game involves maximizing a concave function over a convex domain, and is therefore the optimal solution to the following convex program.

$$
\begin{aligned}
\max \quad & \sum_{i \in A} \log(v_i - c_i) \\
\text{s.t.} \quad & \boldsymbol{v} \in \mathcal{N}
\end{aligned}
\tag{2}
$$

As a consequence, if for a specific game, a separation oracle can be implemented in polynomial time, then using the ellipsoid algorithm one can get as good an approximation to the solution of this convex program as desired in time polynomial in the number of bits of accuracy needed [20, 33].

2.2 Fisher Market Model

The *Fisher market model* consists of a set $A = \{1, 2, \ldots n\}$ of agents and a set $G = \{1, 2, \ldots, m\}$ of infinitely divisible goods. By fixing the units for each good, we may assume without loss of generality that there is a unit of each good in the market. Each agent i has money $m_i \in \mathbb{Q}_+$.

Let x_{ij}, $1 \le j \le m$ represent a *bundle of goods allocated to agent i*. Each agent i has a utility function $u : \mathbb{R}_+^m \to \mathbb{R}_+$ giving the utility accrued by i from a bundle of goods. We will assume that u is concave and weakly monotonic. Each good j is assigned a non-negative price, p_j. Allocations and prices, $\boldsymbol{x}$ and $\boldsymbol{p}$, are said to form an *equilibrium* if each agent obtains a utility maximizing bundle of goods at prices $\boldsymbol{p}$ and the *market clears*, i.e., each good is fully sold to the extent of one unit and all money of agents is fully spent. We will assume that each agent derives positive utility from some good and for each agent, there is a good which gives her positive utility; clearly, otherwise we may remove that agent or good from consideration.

2.3 Arrow-Debreu Market Model

The Arrow-Debreu market model, also known as the *exchange model* differs from Fisher's model in that agents come to the market with initial endowments of good instead of money. The union of all goods in initial endowments are all the goods in the market. Once again, by redefining the units of each good, we may assume that there is a total of one unit of each good in the market. The utility functions of agents are as before. The problem now is to

find non-negative prices for all goods so that if each agent sells her initial endowment and buys an optimal bundle of goods, the market clears. Clearly, if p is equilibrium prices then so is any scaling of p by a positive factor.

2.4   Hylland-Zeckhauser Scheme

Let $A = \{1, 2, \ldots n\}$ be a set of n agents and $G = \{1, 2, \ldots, n\}$ be a set of n indivisible goods. The goal of the HZ scheme is to allocate exactly one good to each agent. However, in order to use the power of a pricing mechanism, which endows the HZ scheme with the properties of Pareto optimality and incentive compatibility in the large, it casts this one-sided matching market in the mold of a linear Fisher market as follows.

Goods are rendered divisible by assuming that there is one unit of probability share of each good, and utilities u_{ij}s are defined as in a linear Fisher market. Let x_{ij} be the allocation of probability share that agent i receives of good j. Then, $\sum_j u_{ij} x_{ij}$ is the *expected utility* accrued by agent i. Each agent has 1 dollar for buying these probability shares and each good j has a price $p_j \geq 0$.

Beyond a Fisher market, an additional constraint is that the total probability share allocated to each agent is one unit, i.e., the entire allocation must form a *fractional perfect matching* in the complete bipartite graph over vertex sets A and G. Subject to these constraints, each agent buys a utility maximizing bundle of goods. Another point of departure from a linear Fisher market is that in general, an agent's optimal bundle may cost less than one dollar, i.e., the agents are not required to spend all their money. Since each good is fully sold, the market clears. Hence these are defined to be *equilibrium allocation and prices*.

Clearly, an equilibrium allocation can be viewed as a doubly stochastic matrix. The Birkhoff-von Neumann procedure then extracts a random underlying perfect matching in such a way that the expected utility accrued to each agent from the integral perfect matching is the same as from the fractional perfect matching. Since *ex ante* Pareto optimality implies *ex post* Pareto optimality, the integral allocation will also be Pareto optimal.

3   Nash-Bargaining-Based Models

3.1   One-Sided Matching Markets

We will define four one-sided matching market models based on our Nash bargaining approach. For each model, we will also give a standard application. For the case of linear utilities, we have singled out the Fisher and Arrow-Debreu versions, namely *1LF* and *1LAD*, since we will study both in some detail later in the paper. For more general utility functions we have defined only the Arrow-Debreu version; the Fisher version is obtained by setting disagreement utilities to zero. It is easy to see that the fourth one generalizes the first three; however, the earlier ones involve less notation and have an independent standing of their own, hence necessitating all four definitions.

Our one-sided matching market models consist of a set $A = \{1, 2, \ldots n\}$ of agents and a set $G = \{1, 2, \ldots, n\}$ of infinitely divisible goods; observe that there is an equal number of agents and goods. There is one unit of each good and each agent needs to be allocated a total of one unit of goods. Hence the allocation needs to be a fractional perfect matching, as defined next.

▶ **Definition 3.** *Let us name the coordinates of a vector $\boldsymbol{x} \in \mathbb{R}_+^{n^2}$ by pairs i, j for $i \in A$ and $j \in G$. Then $\boldsymbol{x}$ is said to be a* fractional perfect matching *if*

$$\forall i \in A : \sum_j x_{ij} = 1 \quad and \quad \forall j \in G : \sum_i x_{ij} = 1.$$

As mentioned in Section 2, an equilibrium allocation can be viewed as a doubly stochastic matrix, and the Birkhoff-von Neumann procedure [5, 34] can be used to extract a random underlying perfect matching in such a way that the expected utility accrued to each agent from the integral perfect matching is the same as from the fractional perfect matching.

1) Under the *linear Fisher Nash bargaining one-sided matching market*, abbreviated *1LF*, each agent $i \in A$ has a linear utility function, as defined in Section 2.2. Corresponding to each fractional perfect matching $\boldsymbol{x}$, there is a vector v_x in the feasible set $\mathcal{N}$; its components are the utilities derived by the agents under the allocation given by $\boldsymbol{x}$. The disagreement point is the origin. Observe that the setup of *1LF* is identical to that of the HZ mechanism; the difference lies in the definition of the solution to an instance. Its standard application is matching agents to goods.

2) Under the *linear Arrow-Debreu Nash bargaining one-sided matching market*, abbreviated *1LAD*, each agent $i \in A$ has a linear utility function, as above. Additionally, we are specified an initial fractional perfect matching $\boldsymbol{x}_I$ which gives the initial endowments of the agents. Each agent has one unit of initial endowment over all the goods and the total endowment of each good over all the agents is one unit, as given by $\boldsymbol{x}_I$. These two pieces of information define the utility accrued by each agent from her initial endowment; this is her disagreement point c_i. As stated in Section 2.1, we will assume that the problem is feasible, i.e., there is a fractional perfect matching, defining a redistribution of the goods, under which each agent i derives strictly more utility than c_i. Each vector $\boldsymbol{v} \in \mathcal{N}$ is as defined in *1LF*. Henceforth, we will consider the slightly more general problem in which are specified the disagreement point $\boldsymbol{c}$ and not the initial endowments $\boldsymbol{x}_I$. There is no guarantee that $\boldsymbol{c}$ comes from a valid fractional perfect matching of initial endowments. However, we still want the problem to be feasible. This model is applicable when agents start with an initial endowment of goods and exchange them to improve their happiness.

3) The *separable, piecewise-linear concave Arrow-Debreu Nash bargaining one-sided matching market*, abbreviated *1SAD*, is analogous to 1LAD, with the difference that each agent has a separable, piecewise-linear concave utility function, hence generalizing the linear utility functions specified in 1LAD. Economists model *diminishing marginal utilities* via concave utility functions. Since we are in a fixed-precision model of computation, we have considered separable, piecewise-linear concave (SPLC) utility functions.

We next define these functions in detail. For each agent i and good j, function $f_i^j : \mathbb{R}_+ \to \mathbb{R}_+$ gives the utility derived by i as a function of the amount of good j she receives. Each f_i^j is a non-negative, non-decreasing, piecewise-linear, concave function. The overall utility of buyer i, $u_i(\boldsymbol{x})$, for bundle $\boldsymbol{x} = (x_1, \ldots, x_n)$ of goods, is additively separable over the goods, i.e., $u_i(\boldsymbol{x}) = \sum_{j \in G} f_i^j(x_j)$.

We will call each piece of f_i^j a *segment*. Number the segments of f_i^j in order of decreasing slope; throughout we will assume that these segments are indexed by k and that S_{ij} is the set of all such indices. Let σ_{ijk}, $k \in S_{ij}$, denote the kth segment, l_{ijk} denote the amount of good j represented by this segment; we will assume that the last segment in each function is of unbounded length. Let u_{ijk} denote the rate at which i accrues utility per unit of good j received, when she is getting an allocation corresponding to this segment. Clearly, the maximum utility she can receive corresponding to this segment is $u_{ijk} \cdot l_{ijk}$.

We will assume that u_{ijk} and l_{ijk} are rational numbers. Finally, let S_σ^i be the set of all indices (j, k) corresponding to the segments in all utility functions of agent i under the given instance, i.e.,

$$S_\sigma^i = \{(j, k) \mid j \in G, \ k \in S_{ij}\}.$$

4) The *non-separable piecewise-linear concave Arrow-Debreu Nash bargaining one-sided matching market*, abbreviated *1NAD*, differs from *1SAD* in that agents' utility functions are now assumed to be non-separable, piecewise-linear concave. These utility functions are very general and can be used to capture whether goods are complements or substitutes and much more.

These functions are defined next. For each agent i, the parameter $l(i)$ specifies the number of hyperplanes used for defining the utility of i. The latter, $u_i(\boldsymbol{x})$, for bundle $\boldsymbol{x} = (x_1, \ldots, x_n)$ of goods is defined to be

$$u_i(\boldsymbol{x}) = \min_{k \leq l(i)} \left\{ \sum_{j \in G} a_{ij}^k x_{ij} + b_i^k \right\},$$

where a_{ij}^k and b_i^k are non-negative rational numbers. Furthermore, $b_i^k = 0$ for at least one value of k so that the utility derived by i from the empty bundle is zero.

Leontief utilities is a fundamental special case of non-separable piecewise-linear concave utilities under which agents want goods in specified ratios. It is used for modeling utilities when goods are complements. In this case, for each agent i, we are specified a set $S_i \subseteq G$ of goods she is interested in, and

$$u_i(\boldsymbol{x}) = \min_{j \in S_i} \left\{ \frac{x_{ij}}{a_{ij}} \right\},$$

where $a_{ij} > 0$ are rational numbers.

In Section 4 we prove that each of the matching markets defined above admits a convex program.

▶ Remark 4. Throughout this paper, we will index elements of A, G and S_{ij} by i, j and k, respectively. When the domain of i, j or k is not specified, especially in summations, it should be assumed to be A, G and S_{ij}, respectively.

3.2   Two-Sided Matching Markets

Our two-sided matching market model consist of a set $A = \{1, 2, \ldots n\}$ of workers and a set $J = \{1, 2, \ldots, n\}$ of firms. For uniformity, we have assumed that there is an equal number of workers and firms, though the model can be easily enhanced and made more general. Our goal is to find an integral perfect matching between workers and firms. In this setting, it is natural to assume that each side has a utility function over the other side, making this a two-sided matching market.

As before, we will relax the problem to finding a fractional perfect matching, $\boldsymbol{x}$, followed by rounding as described above. We will explicitly define only the simplest case of two-sided markets; more general models follow along the same lines as one-sided markets.

Under the *linear Fisher Nash bargaining two-sided matching market*, abbreviated *2LF*, the utility accrued by agent $i \in A$ under allocation $\boldsymbol{x}$,

$$u_i(\boldsymbol{x}) = \sum_{j \in J} u_{ij} x_{ij},$$

where u_{ij} is the utility accrued by i if she were assigned job j integrally. Analogously, the utility accrued by job $j \in J$ under allocation $\boldsymbol{x}$,

$$w_j(x) = \sum_{i \in A} w_{ij} x_{ij},$$

where w_{ij} is the utility accrued by j if it were assigned to i integrally.

In keeping with the axiom of symmetry under Nash bargaining, we will posit that the desires of agents and jobs are equally important and we are led to defining the feasible set in a $2n$ dimensional space, i.e., $\mathcal{N} \subseteq \mathbb{R}_+^{2n}$. The first n components of feasible point $v \in \mathcal{N}$ represent the utilities derived by the n agents, i.e., $u_i(\boldsymbol{x})$s, and the last n components the utilities derived by the n jobs, i.e., $w_j(\boldsymbol{x})$, under a fractional perfect matching $\boldsymbol{x}$. Under *2LF*, the disagreement point is the origin, and we seek the Nash bargaining point. A convex program of *2LF* is given in (7).

4 Convex Programs for the Models

We start by presenting convex programs for *1LF* and *1LAD*, namely (3) and (4). These differ only in that the latter has the parameters c_i in the objective function. For convenience, we define $\mathcal{X}$ to be the set of feasible fractional perfect matchings as defined in Definition 3.

$$
\begin{aligned}
\max \quad & \sum_{i \in A} \log(v_i) \\
\text{s.t.} \quad & v_i = \sum_j u_{ij} x_{ij} \quad \forall i \in A, \\
& \boldsymbol{x} \in \mathcal{X}
\end{aligned}
\tag{3}
$$

$$
\begin{aligned}
\max \quad & \sum_{i \in A} \log(v_i - c_i) \\
\text{s.t.} \quad & v_i = \sum_j u_{ij} x_{ij} \quad \forall i \in A, \\
& \boldsymbol{x} \in \mathcal{X}
\end{aligned}
\tag{4}
$$

Program (5) is a convex program for *1SAD*.

$$
\begin{aligned}
\max \quad & \sum_{i \in A} \log(v_i - c_i) \\
\text{s.t.} \quad & v_i = \sum_j \sum_k u_{ijk} x_{ijk} \quad && \forall i \in A, \\
& \sum_j \sum_k x_{ijk} = 1 \quad && \forall i \in A, \\
& \sum_i \sum_k x_{ijk} = 1 \quad && \forall j \in G, \\
& x_{ijk} \le l_{ijk} \quad && \forall i \in A, \forall j \in G, \forall k \in S_{ij}, \\
& x_{ijk} \ge 0 \quad && \forall i \in A, \forall j \in G, \forall k \in S_{ij}
\end{aligned}
\tag{5}
$$

Program (6) is a convex programs for *1NAD*.

$$
\begin{aligned}
\max \quad & \sum_{i \in A} \log(v_i - c_i) \\
\text{s.t.} \quad & v_i \le \sum_j a_{ij}^k x_{ij} + b_i^k \quad \forall i \in A, \forall k \le l(i), \\
& \boldsymbol{x} \in \mathcal{X}, \boldsymbol{v} \in \mathbb{R}_+^n
\end{aligned}
\tag{6}
$$

Program (7) is a convex program for *2LF*.

$$
\begin{aligned}
\max \quad & \sum_{i \in A} \log(v_i) \; + \sum_{j \in J} \log(v_j) \\
\text{s.t.} \quad & v_i = \sum_j u_{ij} x_{ij} \quad \forall i \in A, \\
& v_j = \sum_i w_{ij} x_{ij} \quad \forall j \in J, \\
& \boldsymbol{x} \in \mathcal{X}
\end{aligned}
\tag{7}
$$

5   Solution Methods

We present two solution methods for solving instances of the convex programs given in Section 4: (a) Cutting-plane algorithm, and (b) Frank-Wolfe algorithm. Both algorithms rely on linear approximations of these problems and converge to the optimal solution in polynomial time. For simplicity of exposition, we focus on the simpler models *1LAD* (and *1LF*) and *2LF* to describe the algorithms. We will explain how these algorithms can be extended to other models.

5.1   Cutting-plane Algorithm

The underlying principle in the cutting-plane method for convex programs with nonlinear objective function is to outer-approximate the epigraph of the objective function through a series of linear programs [24, 33]. Let $f(\boldsymbol{v}) = \sum_{i \in A} \log(v_i - c_i)$ be the objective function in *1LAD*. Since f is concave in $\boldsymbol{v}$, for a given solution $\hat{\boldsymbol{v}}$ we have:

$$
f(\boldsymbol{v}) \le f(\hat{\boldsymbol{v}}) + \nabla f(\hat{\boldsymbol{v}})^\top (\boldsymbol{v} - \hat{\boldsymbol{v}}) = f(\hat{\boldsymbol{v}}) - n + \sum_{i \in A} \frac{v_i - c_i}{\hat{v}_i - c_i}
\tag{8}
$$

Therefore, we can rewrite *1LAD* as the following semi-infinite linear program (SILP):

$$
\begin{aligned}
\max \quad & \eta \\
\text{s.t.} \quad & \eta \le f(\hat{\boldsymbol{v}}) + \nabla f(\hat{\boldsymbol{v}})^\top (\boldsymbol{v} - \hat{\boldsymbol{v}}) \quad \forall \hat{\boldsymbol{v}} \in \mathcal{N}, \\
& (\boldsymbol{x}, \boldsymbol{v}) \in \mathcal{S},
\end{aligned}
\tag{9}
$$

where $\mathcal{N}$ is the set of vectors $\hat{\boldsymbol{v}}$ such that $\hat{v}_i > c_i$, and $\mathcal{S}$ is the set of feasible assignments. Observe that replacing $\mathcal{N}$ with $\hat{\mathcal{N}} \subset \mathcal{N}$ in (9) yields an LP which is a relaxation of the SILP (9). A natural way of solving SILP (9) is to start with a manageable subset $\hat{\mathcal{N}}$ and grow this set until the upper bound produced by the LP is sufficiently close to the optimal solution [24]. However, instead of solving such relaxed LPs and obtaining optimal corner points of the hypograph approximations, it is customary to solve modified forms of these LPs to find

Algorithm 1 Central cutting-plane algorithm for solving *1LAD*.

---
1 Find an initial solution $(\boldsymbol{v}^{(0)}, \boldsymbol{x}^{(0)})$;
2 Initialize $\hat{\mathcal{N}} \leftarrow \{\boldsymbol{v}^{(0)}\}$; $\underline{f} \leftarrow f(\boldsymbol{v}^{(0)})$; $t \leftarrow 1$;
3 $(\boldsymbol{v}^*, \boldsymbol{x}^*) \leftarrow (\boldsymbol{v}^{(0)}, \boldsymbol{x}^{(0)})$;
4 **while** *not converged* **do**
5   Solve LP (10) to obtain the center $(\boldsymbol{v}^{(t)}, \boldsymbol{x}^{(t)}, \eta^{(t)})$ and radius $\sigma^{(t)}$;
6 $\hat{\mathcal{N}} \leftarrow \hat{\mathcal{N}} \cup \{\boldsymbol{v}^{(t)}\}$;
7   **if** $\underline{f} < f(\boldsymbol{v}^{(t)})$ **then**
8 $\underline{f} \leftarrow f(\boldsymbol{v}^{(t)})$; $(\boldsymbol{v}^*, \boldsymbol{x}^*) \leftarrow (\boldsymbol{v}^{(t)}, \boldsymbol{x}^{(t)})$;
9 $t \leftarrow t + 1$;
---

the center of the hypograph approximations. Let $\underline{f}$ be a lower bound on the optimal value of f (e.g., obtained using a feasible allocation). As described by [15], we may construct a cutting plane through the center of the hypograph approximation by solving

$$
\begin{aligned}
\max \quad & \sigma \\
\text{s.t.} \quad & \eta \geq \underline{f} + \sigma, \\
& \eta \leq f(\hat{\boldsymbol{v}}) + \nabla f(\hat{\boldsymbol{v}})^{\top}(\boldsymbol{v} - \hat{\boldsymbol{v}}) - \sigma\|(1, \nabla f(\hat{\boldsymbol{v}}))\|_2 \quad \forall \hat{\boldsymbol{v}} \in \hat{\mathcal{N}}, \\
& (\boldsymbol{x}, \boldsymbol{v}) \in \mathcal{S},
\end{aligned} \tag{10}
$$

which yields radius σ and center $(\boldsymbol{v}, \boldsymbol{x}, \eta)$ of the largest ball that can be inscribed inside the hypograph approximation [29]. Algorithm 1 describes the proposed *Central Cutting-Plane* (CCP) algorithm for solving instances of *1LAD*. As the algorithm iterates, we improve the lower bound $\underline{f}$ and add new cuts to tighten the hypograph approximation. Consequently, the inscribed ball shrinks (i.e., the sequence of hypresphere radii $\{\sigma^{(t)}\}_{t=0}^{\infty}$ converges to 0), and $\{(\boldsymbol{v}^{(t)}, \boldsymbol{x}^{(t)})\}_{t=0}^{\infty}$ converges to the optimal solution with a linear rate as described in Theorem 5 below.

▶ **Theorem 5.** *Central Cutting Plane Algorithm 1 converges to the optimal solution of 1LAD with linear rate.*

Proof. Strict concavity of the objective function in *1LAD* implies existence of a unique optimal solution. This guarantees a linear convergence rate as described in Theorem 7 in [15]. ◀

To assess the convergence of Algorithm 1 numerically, we use the optimality gap in (11) and terminate the algorithm once this gap falls below a given optimality gap threshold.

$$
\text{Gap} = \frac{\sigma^{(t)}}{|\eta^{(t)}|}. \tag{11}
$$

5.1.1 Enhancement techniques

Cut generation. To produce effective cuts and to improve the lower bound quickly, instead of cutting off the current solution $(\boldsymbol{v}^{(t)}, \boldsymbol{x}^{(t)})$, we cut off an intermediate point $(\tilde{\boldsymbol{v}}, \tilde{\boldsymbol{x}}) = \tilde{\alpha}(\boldsymbol{v}^{(t)}, \boldsymbol{x}^{(t)}) + (1 - \tilde{\alpha})(\boldsymbol{v}^*, \boldsymbol{x}^*)$, where $\tilde{\alpha} \in (0, 1]$ is an appropriately-chosen scalar and $(\boldsymbol{v}^*, \boldsymbol{x}^*)$ is the current incumbent solution. To guarantee convergence, $\tilde{\alpha}$ must be chosen such

that the produced cut cuts off $(\tilde{v}, \tilde{x})$, that is $\eta^{(t)} > f(\tilde{v}) - n + \sum_{i \in A} \frac{v_i^{(t)} - c_i}{\tilde{v}_i - c_i}$. At each iteration of Algorithm 1, we initialize $\tilde{\alpha}$ via line search between $v^{(t)}$ and v^*, that is $\tilde{\alpha} = \arg\max_{\alpha \in [0,1]} f(\alpha v^{(t)} + (1 - \alpha) v^*)$.

Avoiding unboundedness. Since the objective function in the convex programs is of the form $\sum_{i \in A} \log(v_i - c_i)$, we require $v_i - c_i > 0$ for each agent i at each iteration of CCP to be able to produce a cut (note that cut coefficients are $1/(\hat{v}_i - c_i)$). However, since CCP is an outer-approximation algorithm, it is possible that in an iterate of CCP, $\hat{v}_i - c_i = 0$ for some agent i, which makes the solution $\hat{v}$ unbounded, and we cannot extract a cut based on this solution. We resolve this issue by taking the convex combination of $(\hat{v}, \hat{x})$ and some feasible interior point $(\bar{v}, \bar{x})$. We do this by choosing the closest feasible point to the $(\hat{v}, \hat{x})$ on the line segment from $(\hat{v}, \hat{x})$ to $(\bar{v}, \bar{x})$, that is by choosing smallest α such that $\alpha \bar{v}_i + (1 - \alpha)\hat{v}_i - c_i \geq \epsilon$ for each i and a small ϵ, which yields $\alpha = \min_{i \in A : \hat{v}_i = c_i} \{\frac{\epsilon}{\bar{v}_i - c_i}\}$.

Scaling of η. Another important aspect in stabilizing CCP is choosing comparable coefficients for the variables. For a given solution $\hat{v}$, coefficient of η in a cut of the form (9) is 1, while the coefficients of the v-variables are $(\frac{1}{\hat{v}_i - c_i})_{i \in A}$, which can be much larger than 1 depending on the value of $\hat{v}$. For instance, when entries of the utility matrices are binary and $c_i > 0$, then $\hat{v}_i - c_i < 1$, and it is possible that $\frac{1}{\hat{v}_i - c_i} \gg 1$ for some agents, making the cut coefficients unbalanced. An LP solver using floating point arithmetic might not handle unbalanced cuts properly. To balance the cut, we replace η with $\eta = \theta\gamma$, where $\theta > 0$ is a fixed scalar and γ acts as the new variable in place of η. With this change of variable, coefficient of σ in the cuts becomes $\|(\theta, \nabla f(\hat{v}))\|_2$. In our implementation, we choose θ as the largest coefficient of the v-variables in the first cut produced, that is $\theta = 1/\min_{i \in A}\{\hat{v}_i - c_i\}$. Note that we may dynamically change θ, but we use the same initial θ for stabilizing all subsequent cuts.

Reoptimization. At each iteration of Algorithm 1, we add a single constraint of the form (8) to the current LP approximation of *1LAD*. Using the Dual Simplex algorithm, we can reuse the information obtained in the previous iteration (e.g. the basis), and thus avoid solving the LPs from scratch at each iteration.

5.1.2 Extension to other models

Algorithm 1 extends to *2LF*, *1SAD*, and *1NAD* easily by replacing the objective function and the constraints with the suitable function and constraints, respectively. For instance, for *2LF*, the cutting planes take the form of

$$\eta \leq \sum_{i \in A} \log(\hat{v}_i) + \sum_{j \in G} \log(\hat{v}_j) - 2n + \sum_{i \in A} \frac{v_i}{\hat{v}_i} + \sum_{j \in G} \frac{v_j}{\hat{v}_j}.$$

Note that, in *2LF*, we may eliminate the x_{ij} variables such that both u_{ij} and w_{ij} are zero. In *1SAD* and *1NAD* the constraints that define $\mathcal{S}$ are updated accordingly.

5.2 Frank-Wolfe Algorithm

Frank-Wolfe (FW) method [17, 23] is one of the simplest and earliest known iterative algorithms for solving non-linear convex optimization problems of the form

$$\max_{x \in \mathcal{X}} f(x),$$

◼ **Algorithm 2** Frank-Wolfe algorithm for solving *1LAD*.

---

1 Set $t \leftarrow 0$ and find an initial perfect matching $\boldsymbol{x}^{(0)}$;

2 **while** *not converged* **do**

3   Compute $g_{ij} = \frac{\partial}{\partial x_{ij}} f(\boldsymbol{x}^{(t)}) = \frac{u_{ij}}{v_i^{(t)} - c_i}$, where $v_i^{(t)} = \sum_{j \in G} u_{ij} x_{ij}^{(t)}$;

4   Compute perfect matching $\hat{\boldsymbol{x}}^{(t)}$ by solving the following problem:

$$\hat{\boldsymbol{x}}^{(t)} = \arg \max_{\boldsymbol{x} \in \mathcal{X}} \sum_{i \in A} \sum_{j \in G} g_{ij} x_{ij}$$

5   Compute the step-size $\gamma^{(t)}$ using the following line search

$$\gamma^{(t)} = \arg \max_{\gamma \in [0,1]} f\left((1-\gamma)\boldsymbol{x}^{(t)} + \gamma \hat{\boldsymbol{x}}^{(t)} \right)$$

6   Update $\boldsymbol{x}^{(t+1)} = (1-\gamma^{(t)})\boldsymbol{x}^{(t)} + \gamma^{(t)} \hat{\boldsymbol{x}}^{(t)}$;

7 $t \leftarrow t+1$;

---

where f is a concave function and $\mathcal{X}$ is a compact convex set. The underlying principle in Frank-Wolfe method is to replace the non-linear objective function f with its linear approximation $\tilde{f}(\boldsymbol{x}) = f(\boldsymbol{x}^{(0)}) + \nabla f(\boldsymbol{x}^{(0)})^\top (\boldsymbol{x} - \boldsymbol{x}^{(0)})$ at a trial point $\boldsymbol{x}^{(0)} \in \mathcal{X}$, and solve a simpler problem

$$\max_{\boldsymbol{x} \in \mathcal{X}} \tilde{f}(\boldsymbol{x}),$$

to produce an "atom" solution $\hat{\boldsymbol{x}}$. The algorithm then iterates by performing line search between $\boldsymbol{x}^{(0)}$ and $\hat{\boldsymbol{x}}$ to produce the next trial point $\boldsymbol{x}^{(1)}$ as a convex combination of $\boldsymbol{x}^{(0)}$ and $\hat{\boldsymbol{x}}$. Algorithm 2 presents the FW algorithm for solving instances of *1LAD*, in which the objective function f is defined as $f(\boldsymbol{x}) = \sum_{i \in A} \log(\sum_{j \in G} u_{ij} x_{ij} - c_i)$ and the feasible region is defined as

$$\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}_+^{n^2} : \sum_{j \in G} x_{ij} = 1 \quad \forall i \in A, \sum_{i \in A} x_{ij} = 1 \quad \forall j \in G\}.$$

Producing an atom. Frank-Wolfe method is particularly useful when $\mathcal{X}$ is a polyhedron and one can exploit its combinatorial properties. In the case of *1LAD* (also *1LF* and *2LF*), the feasible region $\mathcal{X}$ corresponds to a matching polyhedron. Hence, at each iteration of Algorithm 2, the *atom* is an integral perfect matching produced by solving a matching problem. The optimal solution produced by FW is therefore a convex combination of these integral perfect matchings.

Convergence. In general, Frank-Wolfe algorithm admits a sublinear convergence rate [17, 23], that is, after $\mathcal{O}(\frac{1}{\epsilon})$ many iterations, the iterate $\boldsymbol{x}^{(t)}$ is an ϵ-approximate solution to problem *1LAD*.

▶ **Theorem 6** (Jaggi [23]). *Frank-Wolfe Algorithm 1 converges to the optimal solution of 1LAD with sublinear rate.*

Furthermore, as discussed in [23], concavity of f implies that at iteration t of Algorithm 2, $\sum_{i,j} g_{ij}(\hat{x}_{ij}^{(t)} - x_{ij}^{(t)}) \geq f(\boldsymbol{x}^*) - f(\boldsymbol{x}^{(t)})$, where $\boldsymbol{x}^*$ is the optimal solution. Therefore, $\sum_{i,j} g_{ij}(\hat{x}_{ij}^{(t)} - x_{ij}^{(t)})$ provides an upper bound on the optimality gap of iterate $x^{(t)}$, and we may numerically assess convergence of the FW algorithm using

$$\text{Gap} = \frac{\sum_{i,j} g_{ij}(\hat{x}_{ij}^{(t)} - x_{ij}^{(t)})}{|f(\boldsymbol{x}^{(t)})|}. \tag{12}$$

Extension to other models.   As in the cutting-plane method, Algorithm 2 may be extended to other models. For *2LF*, it suffices to compute the gradient g_{ij} as $\frac{\partial}{\partial x_{ij}} f(\boldsymbol{x}^{(t)}) = u_{ij}/v_i^{(t)} + w_{ij}/v_j^{(t)}$. For *1SAD* and *1NAD*, however, while the general framework can be extended, since the feasible region no longer defines a matching polyhedron, finding a new atom $\hat{\boldsymbol{x}}^{(t)}$ is not straightforward. Our primary computational experiments show that a naïve implementation of Frank-Wolfe algorithm does not scale for large instances of these problems.

6   Computational Results

To assess the scalability of the proposed algorithms, we conducted extensive computational experiments on instances of various difficulty levels for each matching market model. We coded our algorithms in `C#` and solved the LPs using the `ILOG Concert` library and `CPLEX 12.10` solver. All experiments were conducted on a Dell desktop equipped with Intel(R) Xeon(R) CPU E5-2680 v3 at 2.50GHz with 8 Cores and 32 GB of memory running a 64-bit Windows 10 operating system. We used the Dual Simplex method for solving the LPs in Algorithm 1 by setting the `RootAlgorithm` parameter to `Cplex.Algorithm.Dual`. Although the matching problems in Algorithm 2 can be solved by specialized algorithms, after primary experiments, we found that using a general-purpose LP solver such as the Primal Simplex method benefits from better warm-start mechanism making the overall implementation simpler. We used the primal simplex method by setting the parameter `RootAlgorithm` to `Cplex.Algorithm.Primal`. We terminated Algorithms 1 and 2 upon reaching either an optimality gap of 10^{-7}, running time of 3600 seconds, or after 1000 iterations.

6.1   Computational Results for *1LAD, 1LF* and *2LF*

We start by presenting the results for matching market models with linear utility functions. We performed computational experiments on *1LAD, 1LF* and *2LF* by producing random utility matrices u (and w in *2LF*) as follows. We considered two general scenarios: (a) **binary**, in which the entries of matrices u and w were drawn from $\{0, 1\}$, and (b) **nonbinary**, in which entries of matrices u and w were general integer values. In both scenarios, u_{ij} was set to 0 with probability $1 - \rho$, where $\rho \in \{\frac{1}{20}, \frac{1}{3}, \frac{2}{3}\}$ represents the density of the utility matrix. For the nonbinary case, positive values of u_{ij} were drawn uniformly from the set $\{1, 2, \ldots, 20\}$. In *1LAD*, the parameters c_i were uniformly chosen from the set $\{\frac{\bar{u}}{3}, \frac{\bar{u}}{4}, 0\}$, where $\bar{u} = \frac{1}{4} \max_{ij}\{u_{ij}\}$ to ensure feasibility.

Tables 1, 2 and 3 present the computational results for models *1LAD, 1LF* and *2LF*, respectively. In these tables, values under columns "Time", "Gap" and "Iter." represent the running time (in seconds), optimality gap (as per equations (11) and (12)) and the number of iterations, respectively. Each entry represents average value over 5 randomly generated instances for each pair of n (number of agents/goods) and ρ (density of the utility matrices). In these tables, whenever a column is missing, it means the corresponding values were 0 across all experiments.

Table 1 Computational results for *1LAD*.

		binary		nonbinary			
n	ρ	Time (CCP)	Time (FW)	Time (CCP)	Iter. (CCP)	Time (FW)	Iter. (FW)
10	0.33	0.008	0.000	0.008	15.5	0.000	3.0
	0.67	0.003	0.000	0.000	4.2	0.003	1.6
20	0.33	0.006	0.000	0.006	15.2	0.006	7.8
	0.67	0.006	0.000	0.003	4.6	0.003	2.2
50	0.33	0.003	0.000	0.013	14.0	0.013	3.2
	0.67	0.013	0.003	0.013	5.4	0.025	2.2
100	0.33	0.037	0.009	0.053	14.0	0.081	6.6
	0.67	0.060	0.000	0.065	5.2	0.116	8.6
200	0.05	0.079	0.094	0.120	70.0	0.210	11.2
	0.33	0.106	0.116	0.097	16.4	0.359	6.0
	0.67	0.081	0.141	0.204	1.0	0.134	1.0
500	0.05	0.125	0.025	0.421	31.0	0.610	7.5
	0.33	0.881	0.669	1.480	1.0	0.833	1.0
	0.67	3.489	0.960	1.422	1.0	1.120	1.0
1000	0.05	0.609	0.088	2.728	29.5	16.555	11.0
	0.33	9.802	3.422	3.437	1.0	3.463	1.0
	0.67	32.402	11.756	8.964	1.0	15.104	1.0
2000	0.05	3.726	0.438	9.468	5.0	9.525	11.0
	0.33	82.599	27.088	24.307	1.0	24.703	1.0
	0.67	274.854	90.766	65.958	1.0	74.131	1.0

Both CCP and FW are able to solve all the *1LAD* and *1LF* instances and the majority of the *2LF* instances to optimality within the given time/iteration limits. We observe that even the largest instances of one-sided market models *1LAD* and *1LF* are solved in less than two minutes, while instances of *2LF* prove to be computationally more challenging; still, the optimality gaps are negligible for large instances of *2LF*. FW outperforms CCP in larger instances in terms of computation time, particularly in *1LF* and *1LAD*, and the lower computation time of FW suggests its capacity for handling even larger instances.

Table 2 Computational results for *1LF*.

		binary				nonbinary			
		CCP		FW		CCP		FW	
n	ρ	Time	Iter.	Time	Iter.	Time	Iter.	Time	Iter.
10	0.33	0.008	28.0	0.000	3.6	0.000	15.0	0.027	205.3
	0.67	0.003	1.0	0.000	1.0	0.000	3.2	0.000	3.6
20	0.33	0.009	1.0	0.000	1.0	0.006	13.6	0.012	69.0
	0.67	0.003	1.0	0.000	1.0	0.003	4.4	0.003	4.8
50	0.33	0.012	1.0	0.003	1.0	0.009	14.6	0.022	36.6
	0.67	0.025	1.0	0.003	1.0	0.012	8.4	0.013	11.0
100	0.33	0.056	1.0	0.003	1.0	0.069	18.6	0.094	16.8
	0.67	0.078	1.0	0.009	1.0	0.125	17.0	0.106	16.6
200	0.05	0.083	1.0	0.008	1.0	1.352	67.2	0.218	84.2
	0.33	0.136	1.0	0.101	1.0	1.753	26.2	0.267	17.0
	0.67	0.278	1.0	0.144	1.0	0.422	1.0	0.410	1.0
500	0.05	0.149	1.0	0.025	1.0	0.397	29.0	0.758	21.5
	0.33	0.755	1.0	0.687	1.0	1.354	1.0	0.844	1.0
	0.67	2.849	1.0	0.994	1.0	1.380	1.0	1.120	1.0
1000	0.05	0.562	1.0	0.087	1.0	5.961	49.5	11.633	18.5
	0.33	5.843	1.0	3.400	1.0	3.844	1.0	3.568	1.0
	0.67	16.917	1.0	11.844	1.0	9.250	1.0	15.208	1.0
2000	0.05	4.196	1.0	0.425	1.0	14.500	7.0	12.103	15.5
	0.33	38.568	1.0	27.181	1.0	27.557	1.0	24.630	1.0
	0.67	143.406	1.0	90.941	1.0	73.198	1.0	73.724	1.0

🟨 **Table 3** Computational results for *2LF*.

		binary						nonbinary					
		CCP			FW			CCP			FW		
n	ρ	Time	Gap	Iter.	Time	Gap	Iter.	Time	Gap	Iter.	Time	Gap	Iter.
10	0.33	0.023	0.00%	1.0	0.070	0.00%	396.0	0.016	0.00%	38.3	0.151	0.02%	1001.0
	0.67	0.012	0.00%	1.0	0.009	0.00%	1.0	0.009	0.00%	24.2	0.038	0.00%	348.6
20	0.33	0.028	0.00%	1.0	0.141	0.00%	684.0	0.041	0.00%	54.6	0.317	0.01%	939.8
	0.67	0.009	0.00%	1.0	0.000	0.00%	1.0	0.022	0.00%	38.0	0.056	0.00%	175.8
50	0.33	0.037	0.00%	1.0	0.003	0.00%	3.0	0.350	0.00%	134.6	1.342	0.01%	622.8
	0.67	0.053	0.00%	1.0	0.003	0.00%	1.0	0.078	0.00%	50.8	0.141	0.00%	120.6
100	0.33	0.134	0.00%	1.0	0.010	0.00%	1.0	1.381	0.00%	196.6	2.743	0.01%	348.0
	0.67	0.228	0.00%	1.0	0.016	0.00%	1.0	0.265	0.00%	33.4	0.537	0.00%	37.6
200	0.05	2501.869	0.02%	1000.0	57.796	0.07%	1000.0	224.755	0.00%	1000.0	257.177	0.04%	1000.0
	0.33	0.275	0.00%	1.0	0.109	0.00%	1.0	6.143	0.00%	188.0	8.663	0.00%	148.8
	0.67	0.303	0.00%	1.0	0.128	0.00%	1.0	1.573	0.00%	54.8	9.025	0.00%	38.0
500	0.05	3608.480	0.00%	388.5	166.305	0.01%	1000.0	2638.611	0.00%	1000.0	1090.482	0.03%	1000.0
	0.33	1.146	0.00%	1.0	0.797	0.00%	1.0	23.548	0.00%	150.0	17.095	0.00%	59.8
	0.67	3.198	0.00%	1.0	1.625	0.00%	1.0	12.360	0.00%	66.0	39.471	0.00%	24.6
1000	0.05	3616.122	0.00%	262.0	773.132	0.11%	1000.0	3678.832	0.00%	656.0	3393.494	0.02%	913.0
	0.33	7.692	0.00%	1.0	9.244	0.00%	1.0	153.357	0.00%	131.3	363.582	0.00%	37.0
	0.67	22.061	0.00%	1.0	17.431	0.00%	1.0	105.834	0.00%	56.0	412.022	0.00%	20.2
2000	0.05	3636.086	0.00%	154.0	1781.018	0.06%	895.6	3608.648	0.00%	227.5	3617.831	0.08%	120.0
	0.33	40.645	0.00%	1.0	68.122	0.00%	1.0	923.272	0.00%	109.0	2540.135	0.00%	26.7
	0.67	172.478	0.00%	1.0	126.075	0.00%	1.0	811.562	0.00%	44.0	3384.356	0.00%	18.3

6.2   Computational Results for *1SAD* and *1NAD*

We generated random instance for *1SAD* by constructing piece-wise linear concave utility functions each with K segments of equal size $\frac{1}{K}$. To ensure that the slopes of the segments for each pair (i,j) (i.e., u_{ijk}) are non-decreasing (i.e., $u_{ij1} > u_{ij2} > \cdots > u_{ijK}$), we first generated K random values σ_{ijk} uniformly drawn from the set $\{1, \ldots, 20\}$, and then set $u_{ijk} = \sum_{l=k}^{K} \sigma_{ijl}$. For compatibility of experiments, we scaled the u_{ijk} values such that the area below the utility function is equal to $\frac{1}{2}\tilde{v}$, where $\tilde{v}$ is uniformly drawn from the set $\{1, \ldots, 20\}$.

For *1NAD*, we considered K hyperplanes of the form $\sum_{j \in G} a_{ij}^k x_{ij} + b_i^k$ for each $i \in A$, and generated the coefficients a_{ij}^k by multiplying $\frac{2}{3}$ with a value uniformly drawn from the set $\{0, 1, \ldots, 20\}$, and generated the intercept b_i^k by multiplying $\frac{1}{3}$ with a value uniformly drawn from the set $\{0, 1, \ldots, 20\}$. If $b_i^k > 0$ for all k, then we randomly set one of them to 0.

Tables 4 and 5 present the computational results respectively for *1SAD* and *1NAD* across different choices of n and K using CCP Algorithm 1. As expected, in both models, as K increases the problems become more challenging, yet the CCP algorithm is able to find the optimal solution or yield a small optimality gap in both cases. Interestingly, *1SAD* instances are significantly easier to solve than *1NAD* instances, and CCP is able to solve all *1SAD* instances up to 2000 agents and 10 segments to optimality within less than 10 minutes. This highlights the capacity of CCP for solving even larger instance of *1SAD*.

References

1   Atila Abdulkadiroğlu, Yeon-Koo Che, and Yosuke Yasuda. Expanding" choice" in school choice. *American Economic Journal: Microeconomics*, 7(1):1–42, 2015.

2   Rediet Abebe, Richard Cole, Vasilis Gkatzelis, and Jason D Hartline. A truthful cardinal mechanism for one-sided matching. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2096–2113. SIAM, 2020.

■ **Table 4** Computational results for *1SAD*.

	$K = 5$			$K = 10$		
n	Time	Gap	Iter.	Time	Gap	Iter.
10	0.012	0.00%	7.1	0.011	0.00%	7.6
20	0.014	0.00%	7.7	0.033	0.00%	8.1
50	0.088	0.00%	7.0	0.158	0.00%	7.1
100	0.386	0.00%	7.0	0.661	0.00%	6.9
200	2.328	0.00%	8.3	2.764	0.00%	6.5
500	11.756	0.00%	7.8	18.540	0.00%	6.0
1000	69.812	0.00%	8.0	97.465	0.00%	6.0
2000	365.448	0.00%	7.6	553.306	0.00%	6.0

■ **Table 5** Computational results for *1NAD*.

	$K = 5$			$K = 10$		
n	Time	Gap	Iter.	Time	Gap	Iter.
10	0.008	0.00%	6.9	0.008	0.00%	6.8
20	0.012	0.00%	7.3	0.025	0.00%	5.2
50	0.111	0.00%	6.2	0.244	0.00%	5.2
100	0.789	0.00%	5.4	1.759	0.00%	5.0
200	4.528	0.00%	5.1	14.313	0.00%	5.0
500	83.479	0.00%	5.6	393.963	0.00%	5.0
1000	1498.531	0.00%	6.0	>3600.000	3.02%	2.0
2000	2816.649	0.00%	6.0	—	—	—

3   Saeed Alaei, Pooya Jalaly Khalilabadi, and Eva Tardos. Computing equilibrium in matching markets. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 245–261, 2017.

4   Saugata Basu, Richard Pollack, and MF Roy. A new algorithm to find a point in every cell defined by a family of polynomials. *Quantifier Elimination and Cylindrical Algebraic Decomposition, B. Caviness and J. Johnson eds., Springer-Verlag, to appear*, 1995.

5   Garrett Birkhoff. Tres observaciones sobre el algebra lineal. *Univ. Nac. Tucuman, Ser. A*, 5:147–154, 1946.

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

7   Ioannis Caragiannis, David Kurokawa, Herve Moulin, Ariel D Procaccia, Nisarg Shah, and Junxing Wang. The unreasonable fairness of maximum Nash welfare. *ACM Transactions on Economics and Computation (TEAC)*, 7(3):1–32, 2019.

8   Thomas Chen, Xi Chen, Binghui Peng, and Mihalis Yannakakis. Computational hardness of the hylland-zeckhauser scheme. *arXiv preprint arXiv:2107.05746*, 2021.

9   Richard Cole, Nikhil Devanur, Vasilis Gkatzelis, Kamal Jain, Tung Mai, Vijay V Vazirani, and Sadra Yazdanbod. Convex program duality, Fisher markets, and Nash social welfare. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 459–460, 2017.

10   Richard Cole and Vasilis Gkatzelis. Approximating the Nash social welfare with indivisible items. *SIAM Journal on Computing*, 47(3):1211–1236, 2018.

11   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 ACM (JACM)*, 55(5):22, 2008.

12   Federico Echenique, Antonio Miralles, and Jun Zhang. Constrained pseudo-market equilibrium. *arXiv preprint arXiv:1909.05986*, 2019.

13   Federico Echenique, Antonio Miralles, and Jun Zhang. Fairness and efficiency for probabilistic allocations with endowments. *arXiv preprint arXiv:1908.04336*, 2019.

14   E. Eisenberg and D. Gale. Consensus of subjective probabilities: the Pari-Mutuel method. *The Annals of Mathematical Statistics*, 30:165–168, 1959.

15   Jack Elzinga and Thomas G Moore. A central cutting plane algorithm for the convex programming problem. *Mathematical Programming*, 8(1):134–145, 1975.

16   Simons Institute for the Theory of Computing. Online and matching-based market design, 2019. URL: `https://simons.berkeley.edu/programs/market2019`.

17   Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Res. Logis. Quart.*, 3(1-2):95–110, 1956.

18   David Gale and Lloyd S Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.

19   Jugal Garg, Thorben Tröbst, and Vijay V Vazirani. An Arrow-Debreu extension of the hylland-zeckhauser scheme: Equilibrium existence and algorithms. *arXiv preprint arXiv:2009.10320*, 2020.

20   Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

21   Yinghua He, Antonio Miralles, Marek Pycia, and Jianye Yan. A pseudo-market approach to allocation with priorities. *American Economic Journal: Microeconomics*, 10(3):272–314, 2018.

22   Aanund Hylland and Richard Zeckhauser. The efficient allocation of individuals to positions. *Journal of Political economy*, 87(2):293–314, 1979.

23   Martin Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *International Conference on Machine Learning*, pages 427–435. PMLR, 2013.

24   James E Kelley, Jr. The cutting-plane method for solving convex programs. *SIAM Journal on Applied Mathematics*, 8(4):703–712, 1960.

25   Phuong Le. Competitive equilibrium in the random assignment problem. *International Journal of Economic Theory*, 13(4):369–385, 2017.

26   Andy McLennan. Efficient disposal equilibria of pseudomarkets. In *Workshop on Game Theory*, volume 4, page 8, 2018.

27   Hervé Moulin. Fair division in the age of internet. *Annual Review of Economics*, 2018.

28   John Nash. Two-person cooperative games. *Econometrica: Journal of the Econometric Society*, pages 128–140, 1953.

29   George L Nemhauser and WB Widhelm. A modified linear program for columnar methods in mathematical programming. *Operations Research*, 19(4):1051–1060, 1971.

30   Ioannis Panageas, Thorben Tröbst, and Vijay V Vazirani. Combinatorial algorithms for matching markets via nash bargaining: One-sided, two-sided and non-bipartite. *arXiv preprint arXiv:2106.02024*, 2021.

31   V. V. Vazirani. The notion of a rational convex program, and an algorithm for the Arrow-Debreu Nash bargaining game. *Journal of the ACM*, 59(2), 2012.

32   Vijay V Vazirani and Mihalis Yannakakis. Computational complexity of the hylland-zeckhauser mechanism for one-sided matching markets. In *Innovations in Theoretical Computer Science*, 2021.

33   Nisheeth Vishnoi. *Algorithms for Convex Optimization*. Cambridge University Press, 2021. To appear.

34   John Von Neumann. A certain zero-sum two-person game equivalent to the optimal assignment problem. *Contributions to the Theory of Games*, 2(0):5–12, 1953.