Polynomial Time Algorithms to Find an Approximate Competitive Equilibrium for Chores

Shant Boodaghians[*1], Bhaskar Ray Chaudhury[2], and Ruta Mehta[†1]

[1]*University of Illinois Urbana-Champaign,* {boodagh2,rutamehta}@illinois.edu
[2]*Max Planck Institute for Informatics, Saarland Informatics Campus,*
braycha@mpi-inf.mpg.de

Abstract

Competitive equilibrium with equal income (CEEI) is considered one of the best mechanisms to allocate a set of items among agents fairly and efficiently. In this paper, we study the computation of CEEI when items are chores that are disliked (negatively valued) by agents, under 1-homogeneous and concave utility functions which includes linear functions as a subcase. It is well-known that, even with linear utilities, the set of CEEI may be non-convex and disconnected, and the problem is PPAD-hard in the more general exchange model. In contrast to these negative results, we design FPTAS: A polynomial-time algorithm to compute ε-approximate CEEI where the running-time depends polynomially on $\frac{1}{\varepsilon}$.

Our algorithm relies on the recent characterization due to Bogomolnaia et al. (2017) of the CEEI set as exactly the KKT points of a non-convex minimization problem that have all coordinates non-zero. Due to this *non-zero* constraint, naïve gradient-based methods fail to find the desired local minima as they are attracted towards zero. We develop an *exterior-point method* that alternates between guessing *non-zero* KKT points and maximizing the objective along supporting hyperplanes at these points. We show that this procedure must converge quickly to an approximate KKT point which then can be mapped to an approximate CEEI; this exterior point method may be of independent interest.

When utility functions are linear, we give explicit procedures for finding the exact iterates, and as a result show that a stronger form of approximate CEEI can be found in polynomial time. Finally, we note that our algorithm extends to the setting of un-equal incomes (CE), and to mixed manna with linear utilities where each agent may like (positively value) some items and dislike (negatively value) others.

1 Introduction

Allocating a set of items among agents in a non-wasteful *(efficient)* and agreeable *(fair)* manner is an age old problem extensively explored within economics, social choice, and computer science. An allocation based on *competitive equilibria (CE)* has emerged as one of the best mechanisms for this problem due its remarkable fairness and efficiency guarantees [AD54, Var74, BMSY17]. The existence and computation of competitive equilibria has seen much work when all the items are goods, *i.e.* liked (positively valued) by agents. However, when items are chores, *i.e.* disliked (negatively valued) by agents, the problem is relatively less explored even though it is as relevant in every day life; for example dividing teaching load among faculty, job shifts among workers, and daily household chores among tenants.

In this paper, we study the problem of computing competitive equilibria with *equal income* (CEEI) [Var74, BMSY17] for chore division, where a set of m *divisible* chores has to be allocated among a set of agents. Agents receive *payments* for doing chores, and are required to earn a minimum amount, and under *equal income*, these amounts are the same.[1] A competitive equilibrium (CE) for chores consists of a payment per-unit for each chore, and an allocation of chores to agents such that every agent gets her *optimal bundle, i.e.,* the disutility-minimizing bundle subject to fulfilling her earning requirement. Typically, agent preferences are represented by a monotone and concave *utility function* [AD54, BMSY17], that is negative and decreasing in case of chores. Equivalently, we consider *disutility functions*, namely $D_i : \mathbb{R}_+^m \to \mathbb{R}_+$ for agent i, that is monotone increasing and convex. We assume disutility functions to be 1-homogeneous as otherwise the problem is known to be intractable [CT09, CGMM20]. We note that 1-homogeneous functions form a rich class that includes the well-studied linear and CES functions as special cases.

The computational complexity of CE is well-understood when items are goods, *e.g.,* [DPSV08, CDDT09, CPY17, VY11, CDG+17, Rub18] (see Section 2 for a detailed discussion): for 1-homogeneous utilities, the famous Eisenberg-Gale [EG59] convex programming formulation and its dual are known to give equilibrium allocation and prices respectively. As a consequence the set of CE is convex, and the ellipsoid and/or interior point methods would find an approximate CE in polynomial-time, assuming utility functions are *well-behaved*. When utility functions are further restricted to be linear, there are many (strongly) polynomial time combinatorial algorithms known [DPSV08, Orl10], even for the more general *exchange model* where agents want to *exchange* items they own to optimize their utilities [DM15, DGM16, GV19].

Although goods and chores problems seem similar, results for chores are surprisingly contrasting: Even in the restricted case of linear disutilities, the set of CEEI can be non-convex and disconnected [BMSY17, BMSY19], and in the exchange model computing a CE is PPAD-hard [CGMM20]. No polynomial time algorithms are known to find CEEI with chores, except for when number of agents or number of chores is a constant [BS19, GM20].[2] We note that the combinatorial approaches known for the goods case [DPSV08, Orl10, Vég12] seem to fail due to disconnectedness of the CEEI set (see Remark 1 for further explanation). In light of these results, computing exact CEEI may turn out to be hard even with linear disutilities, but what about an approximate CEEI?

We resolve the above question by designing an FPTAS for the more general class of 1-homogeneous disutilities. Specifically, we design an algorithm to find ε-approximate CEEI in time polynomial in $\frac{1}{\varepsilon}$ and bit-size of the input instance parameters. We remark that many of the above bottlenecks

---

[1]The earning requirement of an agent can also be thought of as her importance/weight compared to others, and thereby under *equal income* all agents have the same weight.

[2]These algorithms are based on enumeration from a cleverly designed set of candidates. Similar approaches are known for goods manna when the number of items or agents is a constant [DK08, GMSV15], while the general case is PPAD-hard even to approximate [CT09, Rub18]

exist even when we focus on approximate CEEI. In particular, the set of approximate CEEI can be non-convex and disconnected. And the fundamental bottleneck in generalizing the combinatorial algorithm explained in Remark 1 still persists. Despite these challenges, we are able to design an FPTAS to find an approximate CEEI, and extend it to more general valuations than linear which includes CES valuation functions.

Our algorithm crucially builds on the characterization of Bogomolnaia et al. [BMSY17], which states that the set of CEEI is exactly the *strictly positive* local-minima (KKT points) of a non-convex formulation, namely minimize the *product of disutilities (equivalently $\sum_i \log d_i$)* over the space of *feasible* disutility vectors. The set of feasible disutility vectors may not be convex, but they can be made convex by allowing overallocation. Unfortunately, standard interior-point methods for finding local optimum, such as gradient descent, will fail at ensuring the strict positivity constraint, since the gradient of the objective is attracted towards the minimum disutility coordinate. This difficulty is not alleviated by barrier function methods either. A possible fix is to introduce additional constraints to avoid zeros, but then we loose the CEEI characterization.

The above issues would not arise if we *maximize $\sum_i \log d_i$* instead of minimizing it. Motivated from this observation, we design an *exterior-point method* that tries to maximize the objective *outside* of the feasible region, starting from an outside point that is below the lower-hull. However, we are faced with two crucial difficulties: (*i*) now the *outside region* is truly non-convex, and (*ii*) we must ensure that we do find a desired local minimum from the inside.

Our exterior-point method handles the above issues by repeatedly guessing candidate solutions, and checking if they are local minima for the problem inside the feasible region by verifying if the gradient is parallel to some supporting hyperplane. If not, it goes on to try another such candidate, while ensuring it is always increasing along the objective function. Thus, the objective acts as a potential function, and we can bound convergence rates by the size of objective improvement at each step. This method may be of independent interest. We terminate search when the supporting hyperplane direction is *approximately* equal to the gradient, in a multiplicative sense, and argue that such an approximate KKT point suffices to guarantee an approximate CEEI.

The crucial step in each iteration of this procedure is to find the nearest feasible point in the disutility space, which allows us to find a boundary point along with a supporting hyperplane at it. When disutility functions are linear, we argue that both distance minimization and supporting hyperplane computation can be solved exactly, leading to a stronger form of approximate CEEI.

For the case of general 1-homogeneous and convex disutility functions, the nearest feasible point must be found by interior point methods. We assume black-box access to the disutility functions' value and partial derivatives. This approximate nearest-point computation introduces errors in the local optimum and the supporting hyperplane both, that are tricky to handle. We show how to handle these extra errors by modifying the algorithm, and argue that a slight weakening of approximately competitive equilibria can still be guaranteed. As expected, these guarantees, including successful application of the interior point method, rely on the disutility functions being "well-behaved", and the running time of the algorithm depends logarithmically on continuity parameters of the disutility functions, namely, the Lipschitz constants for lower-bounding and upper-bounding the partial derivatives.

Extensions. Finally, we argue how our algorithm easily extends to the setting of un-equal income (CE) when max to min income/weight ratios is polynomially bounded. Another natural extension we consider is to mixed manna, where each agent may like some items and dislike others. Again, using the characterization of [BMSY17], every instance can be put into one of the three categories, namely positive, negative, and null. We argue that the instance in the positive category can be solved using the Eisenberg-Gale convex program [EG59], and those in null have a trivial solution. For instances in the negative category, we discuss how our algorithm can be extended with simple

modifications.

Linear Disutilities with Infinities. We note that, [CGMM20] that shows PPAD-hardness for the linear exchange model allows an agent to have infinite disutility for some chores indicating they do not have skills to do the chore in a reasonable amount of time. Our algorithm extends to this model as well, since their sufficiency conditions to ensure existence of equilibrium dictates that every component of the bipartite graph between agents and chores with finite disutility edges should be a complete bipartite graph. They show that even CEEI may not exist without this condition, and checking if it exists is NP-hard. Under this condition, it suffices to find CEEI for each of the connected component separately where there are no agent-chore pairs with infinite disutility.

In order to convey the main ideas cleanly we mainly focus on CEEI with chores in what follows, and discuss the extensions to CE and mixed manna at the end of the paper.

1.1 Model and Our Results

In the chore division problem, a set of m *divisible* chores $[m] := \{1, \ldots, m\}$ is to be allocated to a set of n agents $[n] := \{1, \ldots, n\}$. It is without loss of generality to assume that exactly one unit of each chore needs to be allocated. Agent i's preferences (over chores) is represented by a non-negative, non-decreasing, and convex disutility function $D_i \colon \mathbb{R}_{\geq 0}^m \to \mathbb{R}_{\geq 0}$.[3] We denote by x_{ij} the fraction of item j that is allocated to agent i, and we denote $\boldsymbol{x}_i := (x_{i1}, \ldots, x_{im})$. We assume that D_i's are 1-homogeneous, *i.e.*

$$D_i(a \cdot \boldsymbol{x}_i) = a \cdot D_i(\boldsymbol{x}_i) \quad \text{for all } \boldsymbol{x}_i, \quad \text{and all } a \geq 0. \tag{1}$$

If $D_i(\cdot)$ is linear, then it is represented by $D_i(\boldsymbol{x}_i) = \sum_{j \in [m]} D_{ij} \cdot x_{ij}$ where $D_{ij} \in (0, \infty)$ is the disutility of agent i per unit of chore j.[4] Equivalently, we write $D_i(\boldsymbol{x}_i) = \langle \boldsymbol{D}_i, \boldsymbol{x}_i \rangle$ where $\boldsymbol{D}_i = (D_{i1}, D_{i2}, \ldots, D_{im})$. We also use $\overrightarrow{D}(\boldsymbol{x})$ to denote the disutility vector $(D_1(\boldsymbol{x}_1), D_2(\boldsymbol{x}_2), \ldots, D_n(\boldsymbol{x}_n))$.

Competitive equilibrium with equal income (CEEI) At a CE with chores, payments are linear, and the j-th chore pays p_j per unit of the chore assigned. Let $\boldsymbol{p} = (p_1, \ldots, p_m)$ denote the vector of payments, and then the payment to agent i is $\langle \boldsymbol{p}, \boldsymbol{x}_i \rangle$. Each agent seeks to minimize their disutility subject to being paid at least 1 unit. We note that, under *equal income*, the exact value being paid is immaterial so long as all agents get paid the same amount. Prices $\boldsymbol{p}$ and allocation $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$ are said to be at CEEI if all the chores are fully allocated when every agent consumes her least-disliked bundle with payment at least 1, *i.e.,* an optimal bundle. Formally [Var74, BMSY17]

(E1) *(equal payments)* for all agents i and i' we have $\langle \boldsymbol{x}_i, \boldsymbol{p} \rangle = \langle \boldsymbol{x}_{i'}, \boldsymbol{p} \rangle$, and

(E2) *(optimal bundle)* for all $i \in [n]$, we have $D_i(\boldsymbol{x}_i) \leq D_i(\boldsymbol{y}_i)$ for all $\boldsymbol{y}$ s.t. $\langle \boldsymbol{y}_i, \boldsymbol{p} \rangle \geq \langle \boldsymbol{x}_i, \boldsymbol{p} \rangle$, and

(E3) *(feasible allocation)* for all $j \in [m]$, we have $\sum_{i \in [n]} x_{ij} = 1$.

It is known that the set of CEEI may be nonconvex, or even disconnected [BMSY17]. In light of this fact, and the PPAD-hardness of CE in the linear-exchange model [CGMM20], we turn our attention to approximately competitive equilibria. We formalize the notion of ε-CEEI as follows:

---

[3]Typically, agents' preferences for chores are represented by non-positive, non-increasing, and concave utility functions since agents dislike chores [BMSY17]. By taking the negation of these utility functions we get non-negative, non-decreasing, convex disutility functions that agents want to minimize.

[4]If for some (i, j) pair $D_{ij} = 0$ then chore j can be freely allocated to agent i, and can be removed. Infinite disutilities can be handled as discussed in the introduction.

Definition 1. *Prices $\boldsymbol{p}$ and allocation $\boldsymbol{x}$ are termed a ε-CEEI for an $\varepsilon \geq 0$, if and only if*

    (1) for all agents i and i', we have $(1 - \varepsilon) \cdot \langle \boldsymbol{x}_i, \boldsymbol{p} \rangle \leq \langle \boldsymbol{x}_{i'}, \boldsymbol{p} \rangle$, and

    (2) for all $i \in [n]$, we have and $(1 - \varepsilon) \cdot d_i(\boldsymbol{x}_i) \leq d_i(\boldsymbol{y}_i)$ for all $\boldsymbol{y}$ such that $\langle \boldsymbol{y}_i, \boldsymbol{p} \rangle \geq \langle \boldsymbol{x}_i, \boldsymbol{p} \rangle$, and

    (3) for all $j \in [m]$, we have $1 - \varepsilon \leq \sum_{i \in [n]} x_{ij} \leq 1 + \varepsilon$.

It is well known that CEEI satisfy well-sought-after fairness and efficiency notions of *envy-freeness* and *Pareto-optimality* respectively. An allocation $\boldsymbol{x}$ is said to be envy-free (EF) if every agent prefers their own bundle over that of any other agent. And it is said to be Pareto-optimal (PO) if no other allocation Pareto-dominates it, *i.e.,* there is no feasible allocation $\boldsymbol{y}$ such that $D_i(\boldsymbol{y}_i) \leq D_i(\boldsymbol{x}_i)$ for all i, and for some agent k, $D_i(\boldsymbol{y}_i) < D_i(\boldsymbol{x}_i)$. In Appendix B we show that an ε-CEEI allocation approximately guarantees these properties.

Our main contribution in this paper is an FPTAS – a polynomial time algorithm to find an ε-CEEI where the running time depends polynomially on $\frac{1}{\varepsilon}$; proved formally in Section 5. Informally, lets say that function D_i is *L-well-behaved* if it satisfies Assumption 16 regarding it's derivatives.

Theorem. *Given black-box access to 1-homogeneous and convex disutilities D_1, ..., D_n that are L-well-behaved, and also to their partial derivatives, Algorithm 2, finds an ε-CEEI in time polynomial in n, m, $1/\varepsilon$, and $\log(L)$.*

We note that our result holds under a weaker assumption than of Assumption 16; discussed briefly in Remark 42. For linear disutilities, we show the following stronger guarantee in Section 4.

Theorem. *Given an instance I with linear disutility functions represented by $D_{11}, \ldots, D_{nm} > 0$, and an $\varepsilon > 0$, a stronger ε-CEEI can be computed in time $\text{poly}\left(n, m, \log\left(\frac{\max_{ij} D_{ij}}{\min_{ij} D_{ij}}\right), \frac{1}{\varepsilon}\right)$ where no error is incurred in the last two conditions, i.e., $(\boldsymbol{x}, \boldsymbol{p})$ that satisfies (1), (E2), and (E3).*

More importantly, our algorithm is an *exterior point method* that builds on tools from continuous optimization to find an *approximate KKT point*, which may be of independent interest. Next we give an overview of this method and it's analysis.

1.2 Overview of the Algorithm and Analysis

Our algorithm builds on the following characterization of CEEI due to Bogomolnaia et al. [BMSY17]: Analogous to the convex program of Eisenberg and Gale [EG59], the CEEI in the case of bads are characterized as local minima to the product of disutilities. However, this optimization program is over *disutility space*, rather than *allocation space*.

Formally, we let $\mathcal{F}$ denote the set of feasible allocations, namely

$$\mathcal{F} := \{\boldsymbol{x} \in \mathbb{R}^{nm} \mid \sum_i x_{ij} = 1 \;\; \forall j, \; x_{ij} \geq 0 \;\; \forall i, j\} \; . \tag{2a}$$

The disutility space $\mathcal{D}$ will be the set of all disutility profiles which can be attained over $\mathcal{F}$, or

$$\mathcal{D} := \{\boldsymbol{d} = (d_1, \ldots, d_n) \in \mathbb{R}^n \mid \exists \, \boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) \in \mathcal{F} : \; D_i(\boldsymbol{x}_i) = d_i \; \forall i\} \; . \tag{2b}$$

In all that follows, we will distinguish between disutilities as functions and as variables by the upper- and lower-case symbols respectively. When disutility functions are linear, $\mathcal{D}$ is a polytope. However, for more general convex disutility functions, $\mathcal{D}$ may not be a convex set. We will remedy this by working instead with the extended feasible region $\mathcal{D} + \mathbb{R}^n_{\geq 0}$, the Minkowski sum, which we
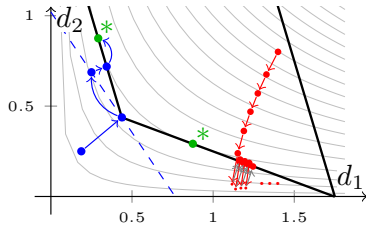
Figure 1: *A representation of both the exterior point method (in blue), and the pitfalls of gradient descent (in red) in minimizing the objective inside the feasible region. The light gray lines denote the level sets of the objective, orthogonal to the gradient.* **Blue:** *For the exterior point method, we start outside the region, find a nearest point and supporting hyperplane, jump to a new exterior point, and repeat until we find one of the two on-face local optima (green).* **Red:** *For the gradient descent, we start inside the region and quickly accelerate towards the $d_2 \geq 0$ boundary, which we wish to avoid.*

show is convex (Claim 3). This is the set of all disutility profiles which are *at least as bad* as some feasible profile, *i.e.,* disutility profiles attainable at over-allocations of the chores.

The characterization of Bogomolnaia et al. [BMSY17] states that any disutility profile $\boldsymbol{d}$ which is a local minimum (KKT point) to the following non-convex minimization program is the disutility profile of *some* CEEI, and the prices and allocation of this CEEI can be found by understanding $\boldsymbol{d}$ in allocation-space.

$$\min_{\boldsymbol{d}\in\mathcal{D}} \ \textstyle\prod_{i=1}^{n} d_i \quad \text{s.t. } d_i > 0 \ \forall\, i \ .$$

Note that minimizing over the set $\mathcal{D}$ is equivalent to minimizing over the extended set $\mathcal{D} + \mathbb{R}_{\geq 0}^{n}$. And the KKT points to this program are equivalent to the KKT points for the minimization of the logarithm of objective, $\mathcal{L}(\boldsymbol{d}) := \sum_{i=0}^{n} \log(d_i)$. Hence the above program can equivalently stated as,

$$\min_{\boldsymbol{d}\in\mathcal{D}+\mathbb{R}_{\geq 0}^{n}} \ \mathcal{L}(\boldsymbol{d}) = \textstyle\sum_{i=0}^{n} \log(d_i) \quad \text{s.t. } d_i > 0 \ \forall\, i \ . \tag{3}$$

Primary difficulty. The open constraints $d_i > 0$ are both fundamental to the above characterization, and the source of the main difficulty of the problem. Any disutility profile $\boldsymbol{d}$ with a zero coordinate are trivial optima to these minimization problems, but are economically meaningless since no fairness or efficiency properties can be guaranteed. Furthermore, any naïve interior-point attempt at finding local minima are attracted by these constraints: the gradient of the objective $\mathcal{L}$ at $\boldsymbol{d}$ is inversely proportional to $\boldsymbol{d}$ componentwise, since $\frac{\partial}{\partial d_i}\mathcal{L} = 1/d_i$. This has the effect of accelerating gradient descent towards the $d_i \geq 0$ constraint for the smallest d_i value. See Figure 1 (red) for an illustration. This effect is robust to barrier methods at the boundaries, and thus gradient-following methods are not helpful in this task. The same problem afflicts attempts at strengthening the constraint to $d_i > \eta$ for some small η, as the dual variables for these constraint break the CEEI characterization. We circumvent this issue by designing an iterative *exterior-point method* that always increases the log-sum $\mathcal{L}$.

Remark 1. A natural question is if the combinatorial methods known for computing CE in linear Fisher (exchange) model with goods, *e.g.,* [DPSV08, Orl10], extend to chores with linear disutilities? Unfortunately, they do not. In particular, the non-convexity and disconnectedness of the CEEI set is a primary difficulty in extending any algorithm from the goods setting to the chores setting. For example, these methods rely on the fact that CE allocation and prices changes continuously with the the (money) endowments of the agents [MV07], which is not true with chores. A chore division instance may have multiple disconnected equilibria some of which may disappear as we change these parameters, and as a result the said methods may get stuck. This fundamental bottleneck persists even if we restrict ourselves to approximate-CEEI.

In the rest of this section, we will outline our approach first for linear disutility functions, and then afterwards in the general case. In the linear setting, many sub-routines can be solved exactly.

Thus, it requires less technical detail to present, and serves as a good intuition for the more involved general case that we address later.

1.2.1 Linear Disutilities: Relating Approximate KKT to Approximate CEEI

The main insights of our result are that (1) approximate KKT points allow for approximately competitive equilibria to be constructed, and (2) approximate KKT points can be found, despite the difficulty described above about ensuring strict positivity constraints. We begin here by formally defining the approximate KKT conditions.

Recall, KKT points are local optima where the cone of normal vectors of the tight constraints contains the function's gradient. Equivalently, there exists a supporting hyperplane at the local optimum whose normal vector is parallel to the function's gradient. Formally, $\boldsymbol{d}$ is a KKT point for the problem $\min_{\boldsymbol{d} \in \mathcal{D}} \mathcal{L}(\boldsymbol{d})$ if there exists a normal vector $\boldsymbol{a}$ such that: $\{\boldsymbol{y} \in \mathbb{R}^n | \langle \boldsymbol{a}, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}, \boldsymbol{d} \rangle\}$ is a supporting hyperplane for $\mathcal{D} + \mathbb{R}^n_{\geq 0}$, and there exists some $c > 0$ such that $\boldsymbol{a} = c \cdot \nabla \mathcal{L}(\boldsymbol{d})$, i.e. $a_i = c/d_i$ for all i.

Since our procedure is iterative, it will converge in the limit to a KKT point, but only approximately after finitely many iterations. We show that after a polynomial number of iterations, it finds an *approximate KKT point*, defined below, with inverse-polynomial error.

Definition 2 (γ-Approximate KKT)**.** *For $\gamma \geq 1$, we say a point $\boldsymbol{d}$ along with the normal direction $\boldsymbol{a}$ is a γ-KKT point for problem* (3) *if*

(1) $\boldsymbol{d} \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$, (2) $\gamma_i^{-1} \leq a_i \cdot d_i \leq \gamma_i$ for all i, and (3) $\mathcal{D} + \mathbb{R}^n_{\geq 0} \subseteq \{\boldsymbol{y} \in \mathbb{R}^n | \langle \boldsymbol{a}, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}, \boldsymbol{d} \rangle\}$.

Informally, each entry of $\boldsymbol{a}$ is a γ-approximation of $\boldsymbol{1}/\boldsymbol{d}$, the gradient of $\mathcal{L}$, and $\boldsymbol{a}$ is normal to a supporting hyperplane for $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ at $\boldsymbol{d}$. Furthermore, we say $\boldsymbol{d}$ is a γ-KKT point if there exists a vector $\boldsymbol{a}$ such that $(\boldsymbol{d}, \boldsymbol{a})$ satisfy the above conditions.

Recall, when disutilities are linear, $\mathcal{D}$ is a linear polytope and is therefore convex. Hence, the above definition need not be defined over $\mathcal{D} + \mathbb{R}^n_{\geq 0}$, but we introduce it as it will be necessary later.

We outline here the first insight of our result, that approximate local minima give approximate equilibria. In the original analysis of the Eisenberg-Gale program [EG59] for goods, and more notably in the proof of [BMSY17] for chores, the relationship between competitive equilibria and local maxima hinges on the gradient being inversely proportional to the marginal (dis)utility-per-dollar incurred. Intuitively, the KKT conditions enforce that the payment to each agent (in the chores setting) is perfectly balanced by their disutility incurred, and their payment is equal to that of any other player. It can be shown that if some player is paid more, then the KKT conditions are violated. Thus, we can conclude condition (1) of Definition 1, with $\varepsilon = 0$. Condition (2) is argued using the fact that an agent is only allocated her minimum disutility-per-dollar chores, and (3) is true by definition since the allocation lies in $\mathcal{F}$.

To extend this argument to the approximate setting, it suffices to observe that when multiplicative error is introduced in the gradient direction, then this argument suffers only multiplicatively. A γ-sized error bound in the gradient direction allows for some player to be paid γ less than the unit, and another γ more, which allows us to show that γ-KKT points satisfy condition (1) of Definition 1 with $\varepsilon = (1 - \gamma^2)$. As above, condition (2) is argued similarly with the same ε, and condition (3) holds with $\varepsilon = 0$, again by feasibility. Formally, by extending the argument of Bogomolnaia et al. [BMSY17], we show the following.

Theorem. *Let $(\boldsymbol{d}, \boldsymbol{a})$ be a $(1+\varepsilon)$-KKT point for the problem of minimizing $\mathcal{L}(\boldsymbol{d})$ subject to $\boldsymbol{d} \in \mathcal{D}$, and $\mathcal{L}(\boldsymbol{d}) > -\infty$. Let $\boldsymbol{x} \in \mathcal{F}$ be any allocation that realizes $\boldsymbol{d}$, i.e. $D_i(\boldsymbol{x}_i) = d_i$ for all i. Then*

6

there exists payments $\boldsymbol{p} = (p_1, \ldots, p_m)$ such that $(\boldsymbol{x}, \boldsymbol{p})$ form a stronger 2ε-CEEI, where no error is incurred in the last two conditions, i.e., $(\boldsymbol{x}, \boldsymbol{p})$ satisfies (1), (E2), and (E3).

Furthermore, when disutilities are linear, the allocation $\boldsymbol{x}$ and payments $\boldsymbol{p}$ can be computed exactly in polynomial time from the disutility profile $\boldsymbol{d}$ and normal vector $\boldsymbol{a}$.

This theorem is proven in Section 4.1, Theorem 4, and the first part of it does not require that the disutilities be linear. However, as we will see below, γ-KKT points can only be guaranteed when disutilities are linear, and the definitions will need to be modified for the general case. The allocation and prices can be efficiently computed when disutilities are linear because they are the solutions to linear feasibility problems. With this theorem in hand, it remains therefore to compute $(1 + \varepsilon)$-KKT points, discussed next.

1.2.2   Linear Disutilities: Exterior Point Methods for Approximate KKT Points.

Here we discuss our approach to find approximate-KKT point in polynomial time; formal details are presented in Section 4.2. As discussed above, it is tempting to hope that interior-point methods will find local minima efficiently, but they will not work in this setting. Instead, we will rely on the geometry of the feasible space and objective function to allow us to repeatedly make guesses at KKT points, all the while increasing along the objective $\mathcal{L}(\boldsymbol{d}) = \sum_i \log(d_i)$, which we treat as a potential function. This potential will ensure that if we do not find approximate KKT points, then we make significant progress, dependent on the degree of precision γ needed. By bounding the values that the potential can take, this will suffice to show that the procedure is an FPTAS. Refer to Figure 1 (blue) for a pictorial representation of the algorithm.

Our "guesses" at KKT points are made by starting with an exterior, infeasible point $\boldsymbol{d}$, and finding the nearest feasible point $\boldsymbol{d}_*$ to it. Formally, $\boldsymbol{d}_*$ is the solution to $\min_{\boldsymbol{y} \in \mathcal{D}} \|\boldsymbol{y} - \boldsymbol{d}\|_2^2$. Using the fact that it is the nearest point to $\boldsymbol{d}$ in the ℓ_2 sense, we show that $\boldsymbol{a} = \boldsymbol{d}_* - \boldsymbol{d}$ is normal to a supporting hyperplane for $\mathcal{D}$ at $\boldsymbol{d}_*$ (Lemma 8). Notice that, so long as $\boldsymbol{d}_* \geq \boldsymbol{d}$ componentwise, then this all still holds when replacing $\mathcal{D}$ with $\mathcal{D} + \mathbb{R}_{\geq 0}^n$. Furthermore, this will ensure that we are increasing along the potential $\mathcal{L}$.

It remains to find the start of the next iterate, while ensuring that we are increasing in the $\mathcal{L}$ direction. Note that we have that the hyperplane $\{\boldsymbol{y} \in \mathbb{R}^n | \langle \boldsymbol{a}, \boldsymbol{y} \rangle = \langle \boldsymbol{a}, \boldsymbol{d}_* \rangle\}$ is supporting for $\mathcal{D}$, and therefore none of the points on this hyperplane are in the interior of $\mathcal{D}$. Thus, we can choose our next starting point to be the $\mathcal{L}$-maximizing point on this hyperplane. Since $\boldsymbol{d}_*$ is also feasible, this will ensure that we are increasing in the $\mathcal{L}$ direction, and that we are starting from a new exterior, infeasible point. This $\mathcal{L}$-maximizer on the hyperplane can be found efficiently, since we have a closed form for it: the maximizer on the hyperplane will be the point at which $\nabla \mathcal{L}$ is proportional to $\boldsymbol{a}$, and we show that it is exactly a rescaling of $(1/a_1, \ldots, 1/a_n)$ (Claim 11).

Thus, the algorithm is iterative, and each round $k \geq 0$ proceeds as follows:

0. $\boldsymbol{d}^k$ is the infeasible point "lying below" $\mathcal{D}$ starting the round. $\boldsymbol{d}^0$ is any infeasible point.

1. Set $\boldsymbol{d}_*^k$ to be the nearest feasible point to $\boldsymbol{d}^k$, *i.e.* the solution to $\min_{\boldsymbol{d} \in \mathcal{D} + \mathbb{R}_{\geq 0}^n} \|\boldsymbol{d} - \boldsymbol{d}^k\|_2^2$.

2. Set $\boldsymbol{a}^k \propto \boldsymbol{d}_*^k - \boldsymbol{d}_k$, rescaled so that $\langle \boldsymbol{a}^k, \boldsymbol{d}_*^k \rangle = n$.

3. Define $\boldsymbol{d}^{k+1}$ to be $(1/a_1, \ldots, 1/a_n)$, the maximizer of $\mathcal{L}(\boldsymbol{d})$ subject to $\langle \boldsymbol{a}^k, \boldsymbol{d} \rangle = n$.

4. Stop if $\boldsymbol{d}^{k+1}$ is "close enough" to $\boldsymbol{d}_*^k$, otherwise repeat.

We initialize the procedure at any infeasible point "lying below" the feasible region. When disutilities are linear, this can be found by noticing that we can lower-bound the disutilities over

the feasible region, and picking an allocation which assigns half of the lower bound to each agent (Claim 9). The normalization in Step 2 ensures that if $\boldsymbol{a}^k$ is approximately parallel to the gradient $\nabla \mathcal{L}(\boldsymbol{d}_*^k)$, then it is also of the right magnitude. The notion of "close enough" in Step 4 is multiplicative, as it measures increase in the potential function $\sum_{i=1}^n \log(d_i)$.

The Potential Function, and Convergence Rates. As discussed above, we wish to use $\mathcal{L}(\boldsymbol{d}) = \sum_{i=1}^n \log(d_i)$ as a potential function to measure the progress of the algorithm. For each iteration $k \geq 0$, we will have $\mathcal{L}(\boldsymbol{d}^k) \leq \mathcal{L}(\boldsymbol{d}_*^k) \leq \mathcal{L}(\boldsymbol{d}^{k+1})$ (Claim 11). This first inequality is due to the observation that the nearest feasible point to $\boldsymbol{d}^k$ Pareto dominates it, and $\mathcal{L}$ is monotone increasing in each coordinate. The second inequality is by construction, as we show that $\boldsymbol{d}^{k+1}$ maximizes $\mathcal{L}$ on a hyperplane that contains $\boldsymbol{d}_*^k$.

It remains then to argue that progress along $\mathcal{L}$ is rapid, relative to its range. We noted above that the stopping condition in Step 4 is multiplicative. Formally, we stop when the ℓ_1 norm of the logarithmic difference, *i.e.* $\sum_{i=1}^n \left| \log \left((\boldsymbol{d}_*^k)_i / (\boldsymbol{d}^{k+1})_i \right) \right|$ is at most ε. When this log-distance is more than ε, we show that the objective $\mathcal{L}$ increases by at least $\Omega(\varepsilon^2/n^2)$ (Lemma 12).

Conversely, when this log-distance is upper-bounded by ε, we will show that $(\boldsymbol{d}_*^k, \boldsymbol{a}^k)$ form a $(1+\varepsilon)$-KKT point (Lemma 10). Thus, since it is reasonable to bound $\log(D_i(\boldsymbol{x}_i))$ over the feasible region, we will be able to bound the maximum number of iterations as a polynomial in $1/\varepsilon$, n, and $-\mathcal{L}(\boldsymbol{d}^0)$. This allows us to argue that an approximate equilibrium may be found in polynomially many iterations.

Implementing Iterations in Polynomial Time. We have argued above that an approximate KKT point, and therefore an approximate equilibrium, can be found in polynomially many iterates of the exterior point method. However, it remains to show that each step can be solved efficiently.

With the exception of Step 1 above, the rest of the algorithm is arithmetic, which can be easily performed. The minimization problem in Step 1 may pose a problem in general, if we expect an exact minimum. This is the source of the extra care needed in the general case. However, in the case of linear disutilities, we show that the minimization problem is actually a quadratic program with a semidefinite bi-linear form over $\mathcal{F}$ space (Lemma 14), and methods for finding exact solutions to such programs have long been known [KTK80].

Finally, we note that although each step in our algorithm generates polynomial sized rational numbers wrt it's parameters, one needs to be careful about how their bit-sizes grow. This can be taken care of by rounding down the $\boldsymbol{d}^k$ to a nearest rational vector with polynomial bit-size at the end of each iteration. Note that this step will ensure that $\boldsymbol{d}^k$ lies below $\mathcal{D}$ and we also argue why the bound on the iterations still hold: Since at every iteration k of our algorithm, the value of each d_i^k can be lower bounded using the value of the potential at $\boldsymbol{d}^k$ and the upper bound on the maximum disutility values in $\boldsymbol{d}^k$,[5] such a rounding is possible without hitting the $d_i \geq 0$ boundary, and while ensuring at least $\Omega(\varepsilon^2/n^2)$ increase in the potential $\mathcal{L}(\cdot)$. However, to convey the main important technical ideas, in Section 4 we focus on bounding number of arithmetic operations. And we note that the analysis of Section 5 for the general case is robust to such a rounding.

Putting all the above together, we get an FPTAS to compute stronger approximate CEEI where the last two conditions of ε-CEEI are satisfied with $\varepsilon = 0$ (Theorem 15).

---

[5]Note that we start with a $\boldsymbol{d}^0$ where each agent has a non-negligible disutility, and at any point in time, the disutilities of the agents in $\boldsymbol{d}^k$ are upper-bounded (as the disutility vector lies below $\mathcal{D}$), implying that there cannot be a significant increase in the disutility of any agent throughout the algorithm. Also, since the sum of logs of the disutilities $\mathcal{L}(\cdot)$ is increasing throughout the algorithm, we can conclude that there cannot be a significant decrease in the disutility of any agent throughout the algorithm, implying that the disutilities in $\boldsymbol{d}^k$ are also lower bounded.

1.2.3 General 1-Homogeneous Disutilities

In general, the disutility functions are 1-homogeneous and convex, and are given as a *value oracle* black-box, along with a value oracle for their partial derivatives. In this section we outline the new issues that arise in extending our algorithm, and their resolutions; see Section 5 for formal details.

At a high-level the issues are as follows: First, to find the nearest points we need to employ *interior point* methods which returns approximate solutions, and in turn we incur error in the hyperplane as well as the gradient. Secondly, in order to use the interior point method, we will have to work in the allocation space and can not work with the disutility space directly. This causes problems as convex constraints in disutility space need not be convex in allocation space. We elaborate these two issues and also highlight how we overcome them. Finally, we give an overview of the entire algorithm by putting everything together.

Finding Approximate Nearest Point. Recall, we have defined $\overrightarrow{D}(\boldsymbol{x}) := (D_1(\boldsymbol{x}_1), \ldots, D_n(\boldsymbol{x}_n))$. The natural program to find the nearest point in $\mathcal{D} + \mathbb{R}_{\geq 0}^n$ to a point $\boldsymbol{d}$ below $\mathcal{D}$ (or equivalently outside $\mathcal{D} + \mathbb{R}_{\geq 0}^n$) requires finding a $\boldsymbol{x} \in \mathcal{F}'$ that minimizes $||\overrightarrow{D}(\boldsymbol{x}) - \boldsymbol{d}||_2^2$, where $\mathcal{F}' := \{\boldsymbol{y} \in \mathbb{R}_{\geq 0}^{nm} \mid \sum_{i \in [n]}(\boldsymbol{y})_{ij} \geq 1 \text{ for all } j \in [m]\}$. Unfortunately the objective function is not necessarily convex[6]. One way to ensure it's convexity is to put additional constraints of the form $D_i(\boldsymbol{x}_i) \geq \boldsymbol{d}_i$ for all $i \in [n]$. But again, since the disutility functions $D_i(\cdot)$ are convex, these constraints create non-convex feasible region.

We come up with an alternative formulation for finding the approximate nearest point which is convex. The crucial observation is the fact that given any point $\boldsymbol{d}$ outside $\mathcal{D} + \mathbb{R}_{\geq 0}^n$ there exists no point $\boldsymbol{d}' \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$, such that $\boldsymbol{d}$ *Pareto-dominates*(coordinate-wise larger or equal) $\boldsymbol{d}'$, i.e., $\boldsymbol{d}'$ does not belong in the negative orthant centered at $\boldsymbol{d}$. Therefore, a point $\boldsymbol{d} \in \mathbb{R}_{\geq 0}^n$ can Pareto-dominate any point $\boldsymbol{d}' \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$ if and only if $\boldsymbol{d} \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$. We now show how to use this fact to come up with a convex program to find the nearest point in $\mathcal{D} + \mathbb{R}_{\geq 0}^n$. Our goal is to find a vector $\boldsymbol{\beta} \in \mathbb{R}^n$ of smallest magnitude and a point $\boldsymbol{d}' \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$ such that the point $\boldsymbol{d} + \boldsymbol{\beta}$ Pareto-dominates $\boldsymbol{d}'$: Note that this is only possible when $\boldsymbol{d} + \boldsymbol{\beta} \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$. Since $||\boldsymbol{\beta}||_2^2$ is minimum, $\boldsymbol{d} + \boldsymbol{\beta}$ is the nearest point in $\mathcal{D} + \mathbb{R}_{\geq 0}^n$ to $\boldsymbol{d}$. Formally,

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i \in [n]} ((\boldsymbol{\beta})_i)^2 \\
\text{subject to} \quad & \sum_{i \in [n]} z_{ij} \geq 1, && \forall j \in [m] \\
& z_{ij} \geq 0, && \forall i \in [n], \forall j \in [m] \\
& D_i(\boldsymbol{z}_i) - (\boldsymbol{d})_i - (\boldsymbol{\beta})_i \leq 0, && \forall i \in [n],
\end{aligned}
$$

It is easy to verify that the above program minimizes a convex function over a convex domain. The above convex program returns point $\boldsymbol{z} \in \mathcal{F}'$ such that $\overrightarrow{D}(\boldsymbol{z})$ is the nearest point in $\mathcal{D} + \mathbb{R}_{\geq 0}^n$ to $\boldsymbol{d}$. Unfortunately, this program cannot be solved exactly in polynomial time and therefore we need to argue about how to extract an approximate-CEEI given an approximate nearest neighbour.

Approximate Supporting Hyperplane and $(\lambda, \gamma, \delta)$-KKT Points. In polynomial time, we can only find an approximate nearest neighbour of a point $\boldsymbol{d}$ in $\mathcal{D} + \mathbb{R}_{\geq 0}^n$. Therefore, our supporting hyperplanes will also be approximate, and therefore we need to redefine the approximate KKT

---

[6]The natural sufficient condition for composition of two convex functions to be convex is if the outer function is monotone in the variables. We do not have this with our current objective function.

points that we can compute. Let $\overrightarrow{D}(\boldsymbol{z}^*)$ be the nearest point in $\mathcal{D}+\mathbb{R}_{\geq 0}^n$ to $\boldsymbol{d}$. Then, $\overrightarrow{D}(\boldsymbol{z}^*) - \boldsymbol{d}$ is normal to a supporting hyperplane of $\mathcal{D}+\mathbb{R}_{\geq 0}^n$ at $\overrightarrow{D}(\boldsymbol{z}^*)$, i.e., $\langle \overrightarrow{D}(\boldsymbol{z}^*) - \boldsymbol{d}, \boldsymbol{y} \rangle = \langle \overrightarrow{D}(\boldsymbol{z}^*) - \boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}^*) \rangle$ is a supporting hyperplane of $\mathcal{D}+\mathbb{R}_{\geq 0}^n$ at $\overrightarrow{D}(\boldsymbol{z}^*)$. Since we have access only to an approximate nearest neighbour of $\boldsymbol{d}$, say $\overrightarrow{D}(\boldsymbol{z}')$, we wish to have $\langle \overrightarrow{D}(\boldsymbol{z}') - \boldsymbol{d}, \boldsymbol{y} \rangle = \langle \overrightarrow{D}(\boldsymbol{z}') - \boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}') \rangle$ as an approximate supporting hyperplane, i.e. $\langle \overrightarrow{D}(\boldsymbol{z}') - \boldsymbol{d}, \boldsymbol{y} \rangle \geq \langle \overrightarrow{D}(\boldsymbol{z}') - \boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}') \rangle - \delta$ for all $\boldsymbol{y} \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$ for a sufficiently small δ.

With this, we introduce the notion of $(\lambda, \gamma, \delta)$-KKT points.

Definition 3 $((\lambda, \gamma, \delta)$-Approximate KKT)**.** *We say $(\boldsymbol{a}, \boldsymbol{d}, \boldsymbol{x})$, i.e., a point $\boldsymbol{d}$ along with the normal direction $\boldsymbol{a}$ and a pre-image $\boldsymbol{x}$ is a $(\lambda, \gamma, \delta)$-KKT point with $\lambda \geq 1$, $\gamma \geq 1$, and $\delta > 0$, for the minimization problem on $\mathcal{D} + \mathbb{R}_{\geq 0}^n$ if*

1. *$x_{ij} \geq 0$ for all $i \in [n]$ and $j \in [m]$, and $\lambda^{-1} \leq \sum_{i \in [n]} x_{ij} \leq \lambda$ for all $j \in [m]$,*

2. *$\boldsymbol{d} = \overrightarrow{D}(\boldsymbol{x})$ and $\gamma_i^{-1} \leq a_i \cdot d_i \leq \gamma_i$ for all $i \in [n]$, and*

3. *and $\mathcal{D} + \mathbb{R}_{\geq 0}^n \subseteq \{\boldsymbol{y} \in \mathbb{R}^n | \langle \boldsymbol{a}, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}, \boldsymbol{d} \rangle - \delta = n - \delta\}$.*

Informally, all chores are almost fully allocated, each entry of $\boldsymbol{a}$ is a γ-approximation of $\mathbf{1}/\boldsymbol{d}$, the gradient of $\mathcal{L}$, and $\boldsymbol{a}$ is a δ-approximately-supporting hyperplane for $\mathcal{D} + \mathbb{R}_{\geq 0}^n$.

In Section 5.1, we show that a $(\lambda, \gamma, \delta)$-KKT point with where $\lambda = 1 + \varepsilon/2^{\mathrm{poly}(n,m)}$, $\gamma = 1 + \varepsilon$ and $\delta = \varepsilon/2^{\mathrm{poly}(n,m)}$ can be mapped to a $\varepsilon^{1/6}$-CEEI. The proof emulates the proof in [BMSY17], and consequently matches the proof in the linear case.

However, some subtle problems arise when generalizing the algorithm from the linear case to determine a $(\lambda, \gamma, \delta)$-KKT point. Firstly, the convergence of the entire algorithm relies crucially on the fact that the potential $\mathcal{L}(\boldsymbol{d})$ never decreases at any point. For this, we require that we have $\overrightarrow{D}(\boldsymbol{z}')$ Pareto-dominate $\boldsymbol{d}$. We can ensure this by first computing an arbitrary approximate nearest point $\boldsymbol{z}''$ and then increase the consumption of certain chores in $\boldsymbol{z}''$ to get $\boldsymbol{z}'$ such that $\overrightarrow{D}(\boldsymbol{z}')$ Pareto-dominates $\boldsymbol{d}$. Since we know that $\overrightarrow{D}(\boldsymbol{z}^*)$ Pareto-dominates $\boldsymbol{d}$, and $||\overrightarrow{D}(\boldsymbol{z}'') - \overrightarrow{D}(\boldsymbol{z}^*)||_2$ is small, the increase in consumption of the chores will also be small (Observations 23 and 24).

Secondly, the hyperplane $\langle \overrightarrow{D}(\boldsymbol{z}') - \boldsymbol{d}, \boldsymbol{y} \rangle = \langle \overrightarrow{D}(\boldsymbol{z}') - \boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}') \rangle$ can be a good approximation of the hyperplane $\langle \overrightarrow{D}(\boldsymbol{z}^*) - \boldsymbol{d}, \boldsymbol{y} \rangle = \langle \overrightarrow{D}(\boldsymbol{z}^*) - \boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}^*) \rangle$ (or equivalently δ is inverse-exponentially small) only if $||\boldsymbol{d} - \overrightarrow{D}(\boldsymbol{z}^*)||_2$ is significantly larger than $||\overrightarrow{D}(\boldsymbol{z}^*) - \overrightarrow{D}(\boldsymbol{z}')||_2$. Therefore, if at any point in our algorithm, we have $||\boldsymbol{d} - \overrightarrow{D}(\boldsymbol{z}')||_2 \leq M\varepsilon$ for a sufficiently large M, where $\varepsilon \geq ||\overrightarrow{D}(\boldsymbol{z}') - \overrightarrow{D}(\boldsymbol{z}^*)||_2$, then we stop and return a pre-image of $\boldsymbol{d}$ (note that as the disutility functions are 1-homogeneous, this can be done by appropriately scaling the consumption of chores for each agent).

Finally, and most importantly, we need to ensure that the approximate supporting hyperplanes do not introduce point with excessive over-allocation. Let $\langle \boldsymbol{a}^*, \boldsymbol{y} \rangle = n$ and $\langle \boldsymbol{a}', \boldsymbol{y} \rangle = n$ represent the hyperplanes $\langle \overrightarrow{D}(\boldsymbol{z}^*) - \boldsymbol{d}, \boldsymbol{y} \rangle = \langle \overrightarrow{D}(\boldsymbol{z}^*) - \boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}^*) \rangle$ and $\langle \overrightarrow{D}(\boldsymbol{z}') - \boldsymbol{d}, \boldsymbol{y} \rangle = \langle \overrightarrow{D}(\boldsymbol{z}') - \boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}') \rangle$ respectively after appropriate scaling, i.e., $\boldsymbol{a}^{\ell} = \left(n/\langle \overrightarrow{D}(\boldsymbol{z}^{\ell}) - \boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}^{\ell}) \rangle\right) \cdot (\overrightarrow{D}(\boldsymbol{z}^{\ell}) - \boldsymbol{d})$ for $\ell \in \{*, '\}$. Since we are dealing with approximate supporting hyperplane[7], the point maximizing $\mathcal{L}$, say $\boldsymbol{d}'$ on $\langle \boldsymbol{a}', \boldsymbol{y} \rangle = n$, maybe contained in the strict interior of $\mathcal{D} + \mathbb{R}_{\geq 0}^n$. Also note that in this case, the nearest point in $\mathcal{D}+\mathbb{R}_{\geq 0}^n$ to $\boldsymbol{d}'$ is $\boldsymbol{d}'$ itself, and therefore the distance between $\boldsymbol{d}'$ and its approximate nearest point in $\mathcal{D} + \mathbb{R}_{\geq 0}^n$ is significantly smaller than $M\varepsilon$ and our algorithm will return the point $\boldsymbol{d}'$, the normal to the hyperplane $\boldsymbol{a}'$ and its pre-image, say $\boldsymbol{x}'$ in the very next iteration. Now note

---

[7]$\langle \boldsymbol{a}', \boldsymbol{y} \rangle \geq n - \delta'$ for all $\boldsymbol{y} \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$, where $\delta' = \frac{\delta \langle \overrightarrow{D}(\boldsymbol{z}') - \boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}') \rangle}{n}$.

that while conditions (2) and (3) in Definition 3 are satisfied, condition (1) may not be satisfied. In particular, there could be chores that are significantly over-allocated! At first this may seem to be counter-intuitive as the hyperplane $\langle \boldsymbol{a}', \boldsymbol{y} \rangle = n$ is a good approximation of the exact supporting hyperplane $\langle \boldsymbol{a}^*, \boldsymbol{y} \rangle = n$, and, the point $\boldsymbol{d}^*$ that maximizes $\mathcal{L}$ on $\langle \boldsymbol{a}^*, \boldsymbol{y} \rangle = n$ lies outside $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ and as a result no chores are over allocated in a pre-image of $\boldsymbol{d}^*$. However, we show that the disutility profiles of the point $\boldsymbol{d}^*$ maximizing $\mathcal{L}$ on the hyperplane $\langle \boldsymbol{a}^*, \boldsymbol{y} \rangle = n$ and the point $\boldsymbol{d}'$ maximizing $\mathcal{L}$ on the hyperplane $\langle \boldsymbol{a}', \boldsymbol{y} \rangle = n$ can be very far apart even if $||\boldsymbol{a}' - \boldsymbol{a}^*||_2$ is small[8]. This is primarily due to the fact that $\boldsymbol{d}' = \left(\frac{1}{a'_1}, \frac{1}{a'_2}, \ldots, \frac{1}{a'_n} \right)$ and $\boldsymbol{d}^* = \left(\frac{1}{a^*_1}, \frac{1}{a^*_2}, \ldots, \frac{1}{a^*_n} \right)$, and even though $|a'_i - a^*_i| \leq \varepsilon$ for all $i \in [n]$, $1/a'_i$ and $1/a^*_i$ can be very far apart. We circumvent this issue by showing that if there are some chores that are significantly over-allocated in $\boldsymbol{x}'$, then we can find an allocation $\boldsymbol{x}''$ from $\boldsymbol{x}'$ by reducing consumption of the over-allocated chores and re-allocating some of the not-over-allocated chores such that $\overrightarrow{D}(\boldsymbol{x}'') \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$ and $\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{x}'') \rangle < n - \delta'$, which is a contradiction to the fact that $\langle \boldsymbol{a}, \boldsymbol{y} \rangle = n$ is an approximate supporting hyperplane to $\mathcal{D} + \mathbb{R}^n_{\geq 0}$. This is where the bulk of the error analysis is required (summarized in Lemmas 28 and 34).

We now outline the entire procedure.

Putting it Together. Similar to the case with linear disutilities, the algorithm is iterative. In each iteration $k \geq 0$,

0. $\boldsymbol{d}^k$ is the infeasible point "lying below" $\mathcal{D}$ at the start of round k. $\boldsymbol{d}^0$ is any infeasible point.

1. Find $\boldsymbol{x}^k_+$ such that $\overrightarrow{D}(\boldsymbol{x}^k_+)$ is an ε-approximate nearest feasible point to $\boldsymbol{d}^k$ in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$, s.t. $(\boldsymbol{d}^k_+)_i \geq (\boldsymbol{d}^k)_i$ for all $i \in [n]$ and then *round up* $\boldsymbol{d}^k_+$ to the nearest rational point with polynomial bit size.

2. If $||\boldsymbol{d}^k_+ - \boldsymbol{d}^k||_2 \leq M \cdot \varepsilon$, then return $(\boldsymbol{a}^{k-1}, \boldsymbol{d}^k, \boldsymbol{x}^k)$ where $\boldsymbol{x}^k$ is a pre-image of $\boldsymbol{d}^k$ obtained by rescaling $\boldsymbol{x}^k_+$ appropriately, i.e., $(\boldsymbol{x}^k)_i \leftarrow (\boldsymbol{x}^k_+)_i \cdot \frac{(\boldsymbol{d}^k)_i}{(\boldsymbol{d}^k_+)_i}$ for all $i \in [n]$.

3. Set $\boldsymbol{a}^k \propto \boldsymbol{d}^k_+ - \boldsymbol{d}_k$, rescaled so that $\langle \boldsymbol{a}^k, \boldsymbol{d}^k_+ \rangle = n$.

4. Define $\boldsymbol{d}^{k+1}$ to be $(1/a_1, \ldots, 1/a_n)$, the maximizer of $\mathcal{L}(\boldsymbol{d})$ subject to $\langle \boldsymbol{a}^k, \boldsymbol{d} \rangle = n$.

5. Return $(\boldsymbol{a}^k, \boldsymbol{d}^k_+, \boldsymbol{x}^k_+)$ if $\boldsymbol{d}^{k+1}$ is "close enough" to $\boldsymbol{d}^k_*$, otherwise repeat.

The algorithm has polynomially many iterations, since similar to the case when agents have linear disutilities, if it does not terminate in iteration k, then the potential $\mathcal{L}$ increases by at least $\Omega(\varepsilon^2/n^2)$. And $\mathcal{L}$ is upper bounded. By arguing that every iteration can be done in polynomial time in Section 5.3, we get an FPTAS in Theorem 41.

1.3 Organization

We give a brief road map of the rest of the paper. In what follows, we first discuss some related work on CE in Section 2 and state some fundamental results from [BMSY17] that we use crucially for our algorithm design in Section 3 . Thereafter, we present the FPTAS when agents have linear disutilities in Section 4 so that the reader gets a good idea of the meta-level algorithm. Finally, in Section 5 we discuss the FPTAS when agents have general 1-homogeneous disutilities. In section 6, we discuss the extensions of our results to the setting when the items to be divided contain both goods and bads (*mixed manna*) with linear valuations, and when agents have unequal income needs (*CE in Fisher model*).

---

[8]In fact $||\boldsymbol{a}' - \boldsymbol{a}^*||_2$ will be small as $||\boldsymbol{d}' - \boldsymbol{d}^*||_2$ is significantly small.

2 Related Work

Competitive equilibrium (CE) has been a fundamental concept in several economic models since the time of Léon Walras [Wal74] in the 19th century. In this paper, we primarily focus on CEEI, which is a special case of CE in Fisher markets, which again is a special case of CE in exchange markets (also referred to as Arrow-Debreu markets). The existence of CE under some mild assumption was proved in the exchange setting by Arrow and Debreu [AD54] and independently by Mackenzie [McK54, McK59]. However, the proofs of existence used fixed point theorems and were non-constructive. In the last few decades, there has been substantial contribution from the computer science community in coming up with constructive algorithms to determine a CE. As mentioned in the introduction, there has been a long line of convex programs, interior point and combinatorial polynomial time algorithms for determining CE with goods in both Fisher and the exchange setting [CDG$^+$17, DGV16, NP83, DPSV08, Orl10, Vég12, DM15, DGM16, GV19, CCD13]. There are also hardness results known when agents have more general utility functions [CPY17, CDDT09, CT09, Rub18]. The existence and computational complexity of CE and its relaxations have been studied in discrete settings (with indivisible objects) as well [FGL16].

The study of CE with chores/ bads has not received similar extensive investigation. One plausible reason could be that this does not capture a natural market and such a setting is interesting only from a fair division perspective. Nevertheless, the CE with bads exhibits far less structure than the CE with goods as explained in the introduction. There are polynomial time enumerative algorithms known only when there are constant number of agents or chores [BS19, GM20]. Quite recently, [CGMM21] gave an LCP formulation for determining CEEI with *mixed manna* (goods and bads) when the utility functions are separable piecewise-linear and concave (SPLC) which includes linear.

3 Preliminaries

Recall the chore division problem formalized in Section 1.1 above: We seek to divide m divisible chores among n agents with convex, 1-homogeneous disutility functions $D_1, \ldots, D_n$, through the mechanism of *competitive equilibrium with equal income (CEEI)*. In this section we state a characterization of CEEI and certain properties of the disutility space that are crucial for our results.

In the case of dividing goods, the seminal work of Eisenberg and Gale [EG59] shows that any allocation that maximizes the Nash welfare — or equivalently the geometric mean of the utilities — is at a CEEI. Since the Nash welfare maximization is a convex program, an approximate CEEI can be determined by an ellipsoid algorithm. Unfortunately, in the case of dividing bads, the set of equilibria could be non-convex and therefore one cannot hope for convex program formulation that captures equilibria [BMSY17]. However, a recent result by Bogomolnaia et al. [BMSY17] show a similar, but non-convex formulation for an exact CEEI (Definition 1, with $\varepsilon = 0$) with chores. In particular, [BMSY17] show that the conditions of an exact CE hold if and only if the disutility profile is a critical point for the Nash welfare on the boundary of the feasible region. Formally:

Theorem 2 ([BMSY17])**.** *Let $\mathcal{F}$ and $\mathcal{D}$ be the feasible space of allocations and disutility profiles as defined in* (2)*. For some $\boldsymbol{d} \in \mathbb{R}^n$, denote the Nash social welfare as* $\mathrm{NSW}(\boldsymbol{d}) := \prod_{i=1}^n d_i$*. Then $\boldsymbol{d}$ can be achieved by a CEEI if and only if the following conditions all hold: a) $\boldsymbol{d} \in \mathcal{D}$, b) $\mathrm{NSW}(\boldsymbol{d}) > 0$, and c) $\boldsymbol{d}$ satisfies the KKT conditions for the problem of minimizing* NSW *on $\mathcal{D}$. Equivalently, $\boldsymbol{d}$ is on the lower-boundary of $\mathcal{D}$, but not on the boundary of $\mathbb{R}^n_{\geq 0}$, and the gradient $\nabla \mathrm{NSW}(\boldsymbol{d})$ is parallel to some supporting hyperplane normal for $\mathcal{D}$ at the point $\boldsymbol{d}$.*

Note that when dis-utilities are linear functions, $\mathcal{D}$ is a linear polytope, though it need not have an efficient representation. When dis-utilities are general, 1-homogeneous, convex functions, the set $\mathcal{D}$ need not be convex. However, we next show that $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ is convex, and we will therefore use it as our feasible region in the analysis; see Appendix A for the proof.

Claim 3. $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ *is convex, when the disutility functions* $D_1, \ldots, D_n$ *are convex.*

In the following sections, we extend Theorem 2 to map approximate KKT points to approximate CEEI (Definition 1), and then design an algorithm to find an approximate KKT point.

4 Polynomial-Time Algorithm for ε-CEEI under Linear Disutilities

In this section we present an algorithm to find an ε-CEEI in time polynomial in $\frac{1}{\varepsilon}$ and the size of the input instance, when agents have linear disutility functions. Recall that, the linear function of agent i is represented by $D_i(\boldsymbol{x}_i) = \sum_{j=1}^m D_{ij}x_{ij}$, or equivalently $D_i(\boldsymbol{x}_i) = \langle \boldsymbol{D}_i, \boldsymbol{x}_i \rangle$ where $\boldsymbol{D}_i = (D_{i1}, D_{i2}, \ldots, D_{im})$.

Our algorithm will ensure a stronger notion of approximation where all the chores are exactly allocated, *i.e.*, condition 3 in Definition 1 is satisfied exactly. For this, the algorithm finds a γ-KKT point as defined in Definition 2. Let us first discuss how such a KKT point gives a stronger approximate CEEI in the next section, thereby extending Theorem 2.

4.1 Approximate KKT Suffices to get Approximate CEEI

We begin with some notation: as we often use element-wise inverse of a vector, for any two n-dimensional vectors $\boldsymbol{x} = (x_1, \ldots, x_n)$ and $\boldsymbol{y} = (y_1, \ldots, y_n)$, we denote

$$\boldsymbol{x}/\boldsymbol{y} := (x_1/y_1, \ldots, x_n/y_n) \ .$$

Recall that we are interested in finding local minima for the logarithm of the Nash social welfare

$$\mathcal{L}(\boldsymbol{d}) := \log(\mathrm{NSW}(\boldsymbol{d})) = \sum_{i=1}^n \log(d_i) \ . \tag{4}$$

Observe that $\nabla \mathcal{L}(\boldsymbol{d}) = \boldsymbol{1}/\boldsymbol{d}$. From Definition 2, recall the γ-KKT point, $\gamma \geq 1$, for minimizing $\mathcal{L}$ on $\mathcal{D}$: point $\boldsymbol{d}$ on the boundary of $(\mathcal{D} + \mathbb{R}^n_{\geq 0})$, such that it has $\{\boldsymbol{y} \mid \boldsymbol{a}^\top \cdot \boldsymbol{y} \geq n\}$ as a supporting hyperplane for $\mathcal{D} + \mathbb{R}^n_{\geq 0}$, where $\boldsymbol{a} \in \mathbb{R}^n$ approximates $\nabla \mathcal{L}(\boldsymbol{d})$ coordinate-wise, i.e., $\forall i, \gamma^{-1} \leq \frac{a_i}{1/d_i} \leq \gamma$.

We emulate here the proof of Bogomolnaia et al. [BMSY17] to show that approximate KKT points give approximate CEEI.

As stated in the overview, we wish to show the following.

Theorem 4. *Let* $(\boldsymbol{d}, \boldsymbol{a})$ *be a* $(1 + \varepsilon)$-*KKT point for the problem of minimizing* $\mathcal{L}(\boldsymbol{d})$ *subject to* $\boldsymbol{d} \in \mathcal{D}$, *and* $\mathcal{L}(\boldsymbol{d}) > -\infty$. *Let* $\boldsymbol{x} \in \mathcal{F}$ *be any allocation that realizes* $\boldsymbol{d}$, *i.e.* $D_i(\boldsymbol{x}_i) = d_i$ *for all* i. *Then there exists payments* $\boldsymbol{p} = (p_1, \ldots, p_m)$ *such that* $(\boldsymbol{x}, \boldsymbol{p})$ *form a stronger* 2ε-*CEEI, where no error is incurred in the last two conditions,* i.e., $(\boldsymbol{x}, \boldsymbol{p})$ *satisfies* (1), (E2), *and* (E3).

Furthermore, when disutilities are linear, the allocation $\boldsymbol{x}$ *and payments* $\boldsymbol{p}$ *can be computed exactly in polynomial time from the disutility profile* $\boldsymbol{d}$ *and normal vector* $\boldsymbol{a}$.

Proof. Let $\gamma = (1+\varepsilon)$, then it suffices to show that γ-KKT gives $(1-\gamma^{-2})$-CEEI since $2\varepsilon > (1-\gamma^{-2})$ for $\varepsilon > 0$. Recall we have defined $\overrightarrow{D}(\boldsymbol{x}) := (D_1(\boldsymbol{x}_1), \ldots, D_n(\boldsymbol{x}_n))$, and sets $\mathcal{F}$ and $\mathcal{D}$ are as in (2), namely, the set of feasible allocations and the set of feasible disutility profiles, in general.

13

Defining and Computing the Allocation and Prices. Let $\boldsymbol{d}$ be the disutility profile of the approximate KKT point. Since $\mathcal{D} + \mathbb{R}^n_{\geq 0} \subseteq \{\boldsymbol{y} \mid \boldsymbol{a}^\top \boldsymbol{y} \geq \langle \boldsymbol{a}, \boldsymbol{d} \rangle\}$ and the entries of $\boldsymbol{a}$ are positive, then $\boldsymbol{d} \in \mathcal{D}$, by minimality. Now, consider any allocation $\boldsymbol{z}$ in $\mathcal{F}$, such that $\vec{D}(\boldsymbol{z}) = \boldsymbol{d}$.

For the second part of the statement of the theorem, we must show that $\boldsymbol{z}$ can be computed, as this will be the allocation of the approximate CEEI. In fact, it suffices to find an allocation vector $\boldsymbol{x}$ which simultaneously satisfies the non-negativity constraints of $\mathcal{F}$, and the linear equality constraints of $\mathcal{F}$ along with $\vec{D}(\boldsymbol{x}) = \boldsymbol{d}$. This can be solved by linear programming techniques in polynomial time.

We wish now to compute the prices at the allocation, for which we will need separating hyperplanes. To this end, define the set $S_\lambda := \{\boldsymbol{x} \in \mathbb{R}^{nm} \mid \langle \boldsymbol{a}, \vec{D}(\boldsymbol{x}) \rangle \leq \lambda\}$. As the disutility functions are convex and continuous, we can conclude that the set S_λ is closed, convex, and non-empty for all $\lambda > 0$, since $S_\lambda \ni \boldsymbol{0}$. When disutilities are linear, S_λ is in fact a closed half-space, since

$$\langle \boldsymbol{a}, \vec{D}(\boldsymbol{x}) \rangle \leq n \iff \sum_{i=1}^n a_i \sum_{j=1}^m D_{ij} x_{ij} \leq n \ .$$

Now, because $\langle \boldsymbol{a}, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}, \vec{D}(\boldsymbol{z}) \rangle$ for all $y \in \mathcal{D}$, we can conclude that the S_λ does not intersect $\mathcal{F}$ for any $\lambda < \langle \boldsymbol{a}, \vec{D}(\boldsymbol{z}) \rangle$. Denote $S^* := S_{\langle \boldsymbol{a}, \vec{D}(\boldsymbol{z}) \rangle}$. The set S^* must be only tangent to $\mathcal{F}$, since the D_i's are continuous, but $\boldsymbol{z} \in \mathcal{F} \cap S^*$. See Figure 2 for an illustration. Thus, there exists a half-space $H_{\boldsymbol{c}} := \{\boldsymbol{x} \mid \langle \boldsymbol{c}, \boldsymbol{x} \rangle \geq b\}$ which separates the two sets, *i.e.* $\mathcal{F} \subseteq H_{\boldsymbol{c}}$, and $S^* \subseteq \mathrm{cl}(H_{\boldsymbol{c}}^\complement)$. Also, note that we must have $\langle \boldsymbol{c}, \boldsymbol{z} \rangle = b$. Note that when disutilities are linear, we have $c_{ij} = a_i D_{ij}$, and $b = n$ as the hyperplane separating $\mathcal{F}$ and S^* is $\langle a, \vec{D}(\boldsymbol{x}) \rangle = n$.

Finally, we can define the prices at the allocation. Let $p_j := \min_i c_{ij}$, and let $\boldsymbol{p} := (p_1, \ldots, p_m)$. See Figure 2 for an illustration of the supporting hyperplanes $\langle \boldsymbol{a}, \boldsymbol{y} \rangle = \langle \boldsymbol{a}, \vec{D}(\boldsymbol{z}) \rangle$ in $\mathcal{D}$ and $\langle \boldsymbol{c}, \boldsymbol{x} \rangle = \langle \boldsymbol{c}, \boldsymbol{z} \rangle$ in $\mathcal{F}$.

It remains then to show that the allocation $\boldsymbol{z}$ and the price vector $\boldsymbol{p}$ satisfy the conditions in Definition 1 where the last two are satisfied without any error, since we have argued already that they can be computed efficiently.

Satisfying Condition (1) in Definition 1. We want to show that for all agents i and i', we have $\gamma^{-2} \cdot \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle \leq \langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle$. But first we make some simple but crucial observations about the price vector $\boldsymbol{p}$.

Claim 5. *We have $\sum_{j \in [m]} p_j = \langle \boldsymbol{c}, \boldsymbol{z} \rangle = b$.*

Proof. $\langle \boldsymbol{c}, \boldsymbol{x} \rangle \geq \langle \boldsymbol{c}, \boldsymbol{z} \rangle = b$ for all $\boldsymbol{x} \in \mathcal{F}$ by definition. Also, since $\boldsymbol{z} \in \mathcal{F}$, we can claim that $b = \min_{\boldsymbol{x} \in \mathcal{F}} \langle \boldsymbol{c}, \boldsymbol{x} \rangle$. Observe that $\min_{x \in \mathcal{F}} \langle \boldsymbol{c}, \boldsymbol{x} \rangle$ is obtained by assigning each chore fully to the agent that has the smallest c_{ij} value for it. Therefore, we have that $\min_{\boldsymbol{x} \in \mathcal{F}} \langle \boldsymbol{c}, \boldsymbol{x} \rangle = \sum_{j \in [m]} \min_{i \in [n]} c_{ij} = \sum_{j \in [m]} p_j$ (by the definition of p_j). $\qquad\square$

Now, consider the half-space $H_{\boldsymbol{p}} = \{\boldsymbol{x} \in \mathbb{R}^{nm}_{\geq 0} \mid \sum_{i \in [n], j \in [m]} p_j \cdot x_{ij} \geq b\}$. We first observe that this half-space is entirely contained in $H_{\boldsymbol{c}}$.

Claim 6. *We have $H_{\boldsymbol{p}} \subseteq H_{\boldsymbol{c}}$.*

Proof. Consider any point $\boldsymbol{x} \in H_{\boldsymbol{p}}$. We have $b \leq \sum_{i \in [n]} \sum_{j \in [m]} x_{ij} \cdot p_j$. Since $p_j \leq c_{ij}$ for all $i \in [n]$, we have that $\sum_{i \in [n]} \sum_{j \in [m]} x_{ij} p_j \leq \sum_{i \in [n]} \sum_{j \in [m]} x_{ij} \cdot c_{ij}$, implying that $\sum_{i \in [n]} \sum_{j \in [m]} x_{ij} \cdot c_{ij} \geq b$, i.e., $\langle \boldsymbol{c}, \boldsymbol{x} \rangle \geq b$. Therefore $\boldsymbol{x} \in H_{\boldsymbol{c}}$. $\qquad\square$

Finally, note that every point $\boldsymbol{x} \in \mathcal{F}$ is also contained in $H_{\boldsymbol{p}}$.
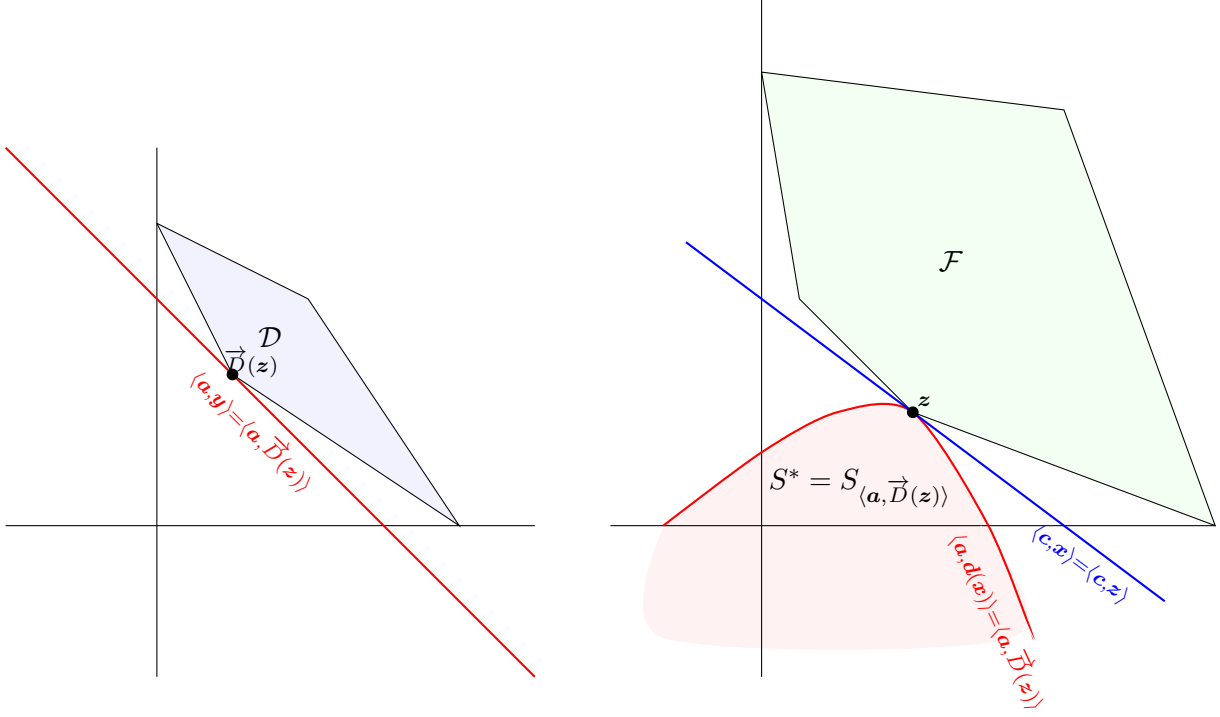
Figure 2: Illustration of the supporting hyperplanes: $\boldsymbol{z} \in \mathcal{F}$ is a point such that $(\vec{D}(\boldsymbol{z}), \boldsymbol{a})$ satisfies the approximate KKT conditions in Definition 3. Thus, we have a supporting hyperplane $\langle \boldsymbol{a}, \boldsymbol{y} \rangle = \langle \boldsymbol{a}, \vec{D}(\boldsymbol{z}) \rangle = n$ of $\mathcal{D}$ such that $\gamma^{-1} \leq a_i \cdot D_i(z_i) \leq \gamma$ (left). The figure on the right describes the set $S^* = S_{\langle \boldsymbol{a}, \vec{D}(\boldsymbol{z}) \rangle}$ and the hyperplane $\langle \boldsymbol{c}, \boldsymbol{x} \rangle = \langle \boldsymbol{c}, \boldsymbol{z} \rangle$ that separates $\mathcal{F}$ from S^*. Note that $\boldsymbol{z} \in \mathcal{F} \cap S^*$ and the curve $\langle \boldsymbol{a}, \vec{D}(\boldsymbol{x}) \rangle = \langle \boldsymbol{a}, \vec{D}(\boldsymbol{z}) \rangle$ coincides with the hyperplane $\langle \boldsymbol{c}, \boldsymbol{x} \rangle = \langle \boldsymbol{c}, \boldsymbol{z} \rangle$ when the disutility functions are linear.

Claim 7. *Consider any $\boldsymbol{x} \in \mathcal{F}$. Then $\boldsymbol{x} \in H_{\boldsymbol{p}}$.*

Proof. Consider any $\boldsymbol{x} \in \mathcal{F}$. We have

$$\sum_{i \in [n], j \in [m]} x_{ij} \cdot p_j = \sum_{j \in [m]} p_j \cdot \sum_{i \in [n]} x_{ij}$$

$$= \sum_{j \in [m]} p_j \qquad (\sum_{i \in [n]} x_{ij} = 1 \text{ as } \boldsymbol{x} \in \mathcal{F})$$

$$= b \qquad \text{(by Claim 5)}$$

Therefore $\boldsymbol{x} \in H_{\boldsymbol{p}}$. $\qquad \square$

Now, we are ready to show that $\gamma^{-2} \cdot \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle \leq \langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle$. Assume otherwise and say we have $\gamma^{-2} \cdot \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle > \langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle$. Then we could replace the allocation as follows: Construct $\hat{\boldsymbol{z}}$ by setting $\hat{\boldsymbol{z}}_{i'} = \frac{1}{2} \boldsymbol{z}_{i'}$, and $\hat{\boldsymbol{z}}_i = \left(1 + \frac{\langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle}{2 \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle} \right) \boldsymbol{z}_i$. Since $\boldsymbol{z} \in \mathcal{F}$, we have $\sum_{i \in [n], j \in [m]} p_j z_{ij} = b$. Also note that

$$b = \sum_{i \in [n], j \in [m]} p_j z_{ij} = \sum_{i \in [n], j \in [m]} p_j \hat{z}_{ij} \;,$$

since the payment subtracted from agent i' is equal to the payment added to agent i and so $\hat{\boldsymbol{z}} \in H_p$.

By Claim 6, we have that $\hat{z} \in H_{\boldsymbol{c}}$. Recall that $H_{\boldsymbol{c}}$ is a separating half-space between S^* and $\mathcal{F}$, *i.e.*, $\mathcal{F} \subseteq H_{\boldsymbol{c}}$ and $S^* \subseteq \mathrm{cl}(H_{\boldsymbol{c}}^{\complement})$, implying that for every point $\boldsymbol{x} \in H_{\boldsymbol{c}}$ we have $\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{x}) \rangle \geq \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle$. Since $\hat{z} \in H_{\boldsymbol{c}}$, we have $\langle \boldsymbol{a}, \overrightarrow{D}(\hat{z}) \rangle \geq \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle$. However,

$$
\begin{aligned}
\langle \boldsymbol{a}, \overrightarrow{D}(\hat{z}) \rangle - \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle &= -\frac{1}{2} a_{i'} D_{i'}(\boldsymbol{z}_{i'}) + \frac{\langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle}{2 \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle} a_i D_i(\boldsymbol{z}_i) \\
&\leq -\frac{1}{2} \gamma^{-1} + \frac{\langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle}{2 \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle} \cdot \gamma \\
&< -\tfrac{1}{2} \gamma^{-1} + \tfrac{1}{2} \gamma^{-1} = 0 , && (\text{as } \langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle / \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle < \gamma^{-2})
\end{aligned}
$$

which is a contradiction. The first inequality is due to the definition of γ-approximate KKT, which dictates that $\gamma^{-1} \leq a_\ell \cdot D_\ell(\boldsymbol{z}_\ell) \leq \gamma$ for all $\ell \in [n]$.

Satisfying Condition (2) in Definition 1 Exactly (i.e., Condition 1.1). We want to show that for all $i \in [n]$, we have $D_i(\boldsymbol{z}_i) \leq D_i(\boldsymbol{y})$ for all y such that $\langle \boldsymbol{y}, \boldsymbol{p} \rangle \geq \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle$. Let us assume that there exists a $\boldsymbol{y}$ such that $D_i(\boldsymbol{z}_i) > D_i(\boldsymbol{y})$ and $\langle \boldsymbol{y}, \boldsymbol{p} \rangle \geq \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle$. We define a new allocation $\boldsymbol{z}' = (\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{i-1}, \boldsymbol{y}, \boldsymbol{z}_{i+1}, \ldots, \boldsymbol{z}_n)$. First note that $\sum_{i \in [n], j \in [m]} z'_{ij} \cdot p_j \geq \sum_{i \in [n], j \in [m]} z_{ij} \cdot p_j = b$ as $\langle \boldsymbol{y}, \boldsymbol{p} \rangle \geq \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle$. Therefore $\boldsymbol{z}' \in H_{\boldsymbol{p}}$. By Claim 6, we have that $\boldsymbol{z}' \in H_{\boldsymbol{c}}$. Recall that $H_{\boldsymbol{c}}$ is a separating half-space between S^* and $\mathcal{F}$, *i.e.*, $\mathcal{F} \subseteq H_{\boldsymbol{c}}$ and $S^* \subseteq \mathrm{cl}(H_{\boldsymbol{c}}^{\complement})$, implying that for every point $\boldsymbol{x} \in H_{\boldsymbol{c}}$ we have $\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{x}) \rangle \geq \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle$. Since $\boldsymbol{z}' \in H_{\boldsymbol{c}}$, we have $\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}') \rangle \geq \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle$. However, since $D_i(\boldsymbol{y}) < D_i(\boldsymbol{z}_i)$ and $a_i \geq \gamma^{-1} / D_i(\boldsymbol{z}_i) > 0$ (by the definition of approximate KKT point), we have that $\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}') \rangle < \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle$, which is a contradiction.

Satisfying Condition (3) in Definition 1 Exactly (i.e., Condition 1.1). Since $\boldsymbol{z} \in \mathcal{F}$, we have that $\sum_{i \in [n]} z_{ij} = 1$ for all $i \in [n]$. $\qquad \square$

This concludes the proof that an approximate-CEEI can be determined from approximate-KKT points in polynomial time. In the next subsection, we outline a polynomial time algorithm that determines an approximate-KKT point.

4.2 Algorithm, and Convergence Guarantees

We show that approximate-KKT points can be found in polynomial time. We begin with an overview of the procedure, and later show how the steps are implemented. The idea is to perform an exterior-point procedure outside of the feasible region, which produces a sequence of guesses for approximate KKT points, while increasing along the objective. Due to the nature of the objective function, we alternate between finding supporting hyperplanes, and finding NSW-maximizing points on these hyperplanes, until we find a point whose gradient is approximately in line with the supporting hyperplane.

To be precise, our algorithm starts from a point $\boldsymbol{d}^0$ very close to $\boldsymbol{0}$. Note that this point lies below $\mathcal{D}$. Then, we find the nearest point $\boldsymbol{d}_*^0$ in $\mathcal{D} + \mathbb{R}_{\geq 0}^n$ to $\boldsymbol{d}^0$. We will address how to find this nearest point, and explain how to robustly handle approximation errors in finding this nearest point. In doing so, it will be helpful to find nearest points in the convex region $\mathcal{D} + \mathbb{R}_{\geq 0}^n$, but keeping in mind that the true optimum $\boldsymbol{d}_*^0$ lies in $\mathcal{D}$: to see this, note that $\boldsymbol{d}_*^0$ has to lie on the lower envelope of $\mathcal{D}$, and since it is the closest point in $\mathcal{D}$ to $\boldsymbol{d}^0$, it follows that $(\boldsymbol{d}_*^0 - \boldsymbol{d}^0)$ is normal to a supporting hyperplane of $\mathcal{D}$ at $\boldsymbol{d}_*^0$. Furthermore, we show that $\boldsymbol{d}_*^0$ Pareto-dominates $\boldsymbol{d}^0$, thereby implying that the Nash welfare at $\boldsymbol{d}_*^0$ is larger than the Nash welfare at $\boldsymbol{d}_*^0$.

Let $\langle \boldsymbol{a}, \boldsymbol{y} \rangle = n$ be the supporting hyperplane of $\mathcal{D}$ at $\boldsymbol{d}_*^0$, where $\boldsymbol{a} \propto (\boldsymbol{d}_*^0 - \boldsymbol{d}^0)$. Let $\boldsymbol{d}^1$ be a point on this hyperplane with maximum Nash welfare. Observe that at $\boldsymbol{d}^1$, we should have $\nabla \mathcal{L}$ proportional to $\boldsymbol{a}$, i.e., $\boldsymbol{a} = \boldsymbol{1}/\boldsymbol{d}^1$, implying that $\boldsymbol{d}^1 = \boldsymbol{1}/\boldsymbol{a}$. Since $\langle \boldsymbol{a}, \boldsymbol{y} \rangle = n$ is a supporting hyperplane of $\mathcal{D}$ at $\boldsymbol{d}_*^0$ (a point on the lower envelope of $\mathcal{D}$), we have that $\boldsymbol{d}^1$ also lies below the lower envelop of $\mathcal{D}$. We prove that if the distance between $\boldsymbol{d}_*^0$ and $\boldsymbol{d}^1$ is small, then $\boldsymbol{d}_*^0$ is our approximate KKT-point, otherwise we have a new point $\boldsymbol{d}^1$ below $\mathcal{D}$, which has significantly higher Nash welfare than $\boldsymbol{d}^0$. We run the exact same steps from $\boldsymbol{d}^1$. We argue that such a procedure should eventually give us an approximate KKT point as there is significant increase in Nash welfare with every iteration of the algorithm whenever no approximate KKT point is found. The full description of the algorithm is given in Algorithm 1.

In what follows, define $\mathrm{RelDist}(\boldsymbol{x}, \boldsymbol{y}) := \sum_i |\log(x_i/y_i)|$. Notice that if $\mathrm{RelDist}(\boldsymbol{x}, \boldsymbol{y}) \leq \varepsilon$, then $(1 + \varepsilon)^{-1} \leq x_i/y_i \leq (1 + \varepsilon)$ for all i, since $\log(1 + a) \leq a$ for all $a > -1$. We will find a point which is a $(1 + \varepsilon)$-approximate KKT point following Algorithm 1.

Algorithm 1 Finding Approximate KKT

1: Let $\boldsymbol{d}^0$ be any infeasible, strictly positive, disutility profile, near $\boldsymbol{0}$
2: **while true do**
3:     Set $\boldsymbol{d}_*^k$ to be the nearest dominating point in $\mathcal{D}$ to $\boldsymbol{d}^k$, i.e.

$$\arg\min \left\{ \|\boldsymbol{y} - \boldsymbol{d}^k\|_2^2 \;\middle|\; \boldsymbol{y} \in \mathcal{D} + \mathbb{R}_{\geq 0}^n, \; \boldsymbol{y} \geq \boldsymbol{d}^k \right\}$$

4:     **Set $\boldsymbol{a}^k \leftarrow (\boldsymbol{d}_*^k - \boldsymbol{d}^k)$**, the direction from $\boldsymbol{d}^k$ to $\mathcal{D}$
5:     **Rescale $\boldsymbol{a}^k$** so that $\langle \boldsymbol{a}^k, \boldsymbol{d}_*^k \rangle = n$
6:     **Set $\boldsymbol{d}^{k+1} \leftarrow \boldsymbol{1}/\boldsymbol{a}^k$**
7:     **if $\mathrm{RelDist}(\boldsymbol{d}^{k+1}, \boldsymbol{d}_*^k) < \varepsilon$ then**
8:        **Return $(\boldsymbol{d}_*^k, \boldsymbol{a}^k)$**

Correctness. We begin by proving here that the algorithm truly returns an approximate KKT point and we will later show that (i) it will terminate in polynomially many iterations, (ii) each iteration can be implemented in polynomial time. To this end, we will need the following technical results, about the steps of the algorithm.

Lemma 8. *Regardless of the geometry of $\mathcal{D}$, so long as $\mathcal{D} + \mathbb{R}_{\geq 0}^n$ is convex, we have that for each iteration $k \geq 0$ of Algorithm 1:*

1. *The hyperplane defined as $\{\boldsymbol{y} \in \mathbb{R}^n | \langle \boldsymbol{a}^k, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}^k, \boldsymbol{d}_*^k \rangle\}$ is supporting for $\mathcal{D} + \mathbb{R}_{\geq 0}^n$, at $\boldsymbol{d}_*^k$.*

2. *If $\boldsymbol{d}^k$ has strictly positive entries and does not lie in $\mathcal{D} + \mathbb{R}_{\geq 0}^n$, then $\boldsymbol{d}_*^k$, $\boldsymbol{a}^k$, and $\boldsymbol{d}^{k+1}$ have strictly positive entries, and $\boldsymbol{d}_*^k \in \mathcal{D}$.*

We show these results in Appendix A, as the proofs are mostly technical. Informally, these hold due to the geometry of the feasible region, and ensure that each iterate is well-defined, and economically meaningful. To complete the proof of correctness, we show that we can efficiently find a starting point $\boldsymbol{d}^0$ which is strictly positive in every entry, and is infeasible. Thus, Lemma 8 will inductively show that every point is positive and well-defined.

Claim 9. *The point $\boldsymbol{d}^0 = \frac{m\delta}{2n}\boldsymbol{1}$ where $\delta = \min_{ij} D_{ij}$ is a strictly positive infeasible disutility profile.*

Proof. Since $D_{ij} \geq \delta$ for all i and j, any feasible dis-utility profile must assign disutility at least $m\delta/n$ to some agent. Therefore, it is impossible for every agent to have disutility $\frac{1}{2}m\delta/n$ at a feasible point. $\qquad\square$

We now show that in the stopping condition, Algorithm 1 returns an approximate KKT point. Intuitively, this holds because the RelDist function in the stopping condition is designed to correctly captures the multiplicative error needed in the definition of approximate KKT.

Lemma 10. *Algorithm 1 returns a $(1+\varepsilon)$-KKT point for minimizing $\mathcal{L}$ on $\mathcal{D}$.*

Proof. Suppose the algorithm terminates and returns $(\boldsymbol{d}_*^k, \boldsymbol{a}^k)$ on line 8. Note that we have $\boldsymbol{d}^{k+1} = \boldsymbol{1}/\boldsymbol{a}^k$ and $\text{RelDist}(\boldsymbol{d}^{k+1}, \boldsymbol{d}_*^k) < \varepsilon$, implying that $\text{RelDist}(\boldsymbol{1}/\boldsymbol{a}^k, \boldsymbol{d}_*^k) < \varepsilon$. Then, we have $(1+\varepsilon)^{-1} \leq a_i^k \cdot (d_*^k)_i \leq 1 + \varepsilon$ for all i. Also by Lemma 8, we have that $\langle \boldsymbol{a}^k, \boldsymbol{y} \rangle = n$ is a supporting hyperplane of $\mathcal{D}$ passing through $\boldsymbol{d}_*^k$. Therefore, the point $\boldsymbol{d}_*^k$ is a $(1+\varepsilon)$-KKT point as in Definition 2. $\qquad\square$

In the rest of this section, we will argue that the number of iterations must be polynomial, and that each iteration can be solved in polynomial time, which will allow us to conclude the correctness and efficiency of the algorithm.

Polynomially Many Iterations. We show that in polynomially many iterations the algorithm finds an approximate KKT point. In particular, we show that (a) the log-NSW $\mathcal{L}$ is always increasing throughout Algorithm 1, and (b) it increases additively by $\text{poly}(n, 1/\varepsilon)$ every time $\text{RelDist}(\boldsymbol{d}^{k+1}, \boldsymbol{d}_*^k) \geq \varepsilon$. Bounding the range of $\mathcal{L}$ over the course of the iteration will then give our desired bound.

Claim 11. *Steps 3. and 6. always increase $\mathcal{L}$, the log-product of disutilities. Formally, $\mathcal{L}(\boldsymbol{d}^{k+1}) \geq \mathcal{L}(\boldsymbol{d}_*^k) \geq \mathcal{L}(\boldsymbol{d}^k)$ for all $k \geq 0$.*

Proof. By Lemma 8, $\boldsymbol{d}_*^k \geq \boldsymbol{d}^k$, coordinate-wise. Thus, since $\mathcal{L}$ is monotone increasing in each coordinate direction, $\mathcal{L}(\boldsymbol{d}_*^k) \geq \mathcal{L}(\boldsymbol{d}^k)$.

We prove that Step 6 is an improvement by showing that $\boldsymbol{d}^{k+1}$ is the maximizing point on the hyperplane $\langle \boldsymbol{a}^k, \boldsymbol{y} \rangle = n$, and therefore $\mathcal{L}(\boldsymbol{d}^{k+1}) \geq \mathcal{L}(\boldsymbol{d}_*^k)$.

Since $\mathcal{L}$ is a concave function, it is maximized on this hyperplane when $\nabla \mathcal{L}$ is proportional to $\boldsymbol{a}^k$, *i.e.* when $a_i^k = c/d_i$ for some $c > 0$, for all i. Since we need $\langle \boldsymbol{a}^k, \boldsymbol{d} \rangle = n$, it suffices to set $c = 1$. Thus, $\boldsymbol{d}^{k+1}$ is the $\mathcal{L}$-maximizing point on the supporting hyperplane which contains $\boldsymbol{d}_*^k$, and so this move is an $\mathcal{L}$-improvement. $\qquad\square$

Using the above claims, next we show that $\mathcal{L}$ increases significantly in each iteration of our algorithm.

Lemma 12. *If Algorithm 1 does not return at step 8, then the logarithm of the Nash social welfare increases by at least $\frac{1}{16}(\varepsilon/n)^2$, i.e., $\mathcal{L}(\boldsymbol{d}^{k+1}) - \mathcal{L}(\boldsymbol{d}^k) \geq \frac{1}{16}(\varepsilon/n)^2$.*

Proof. Since $\mathcal{L}(\boldsymbol{d}_*^k) > \mathcal{L}(\boldsymbol{d}^k)$ by Claim 11, it suffices to show that if $\text{RelDist}(\boldsymbol{d}^{k+1}, \boldsymbol{d}_*^k) > \varepsilon$, then $\mathcal{L}(\boldsymbol{d}^{k+1}) - \mathcal{L}(\boldsymbol{d}_*^k)$ is large. Let $A = \text{diag}(\boldsymbol{a}^k)$, and note that $\langle \boldsymbol{1}, A\boldsymbol{d} \rangle = \langle \boldsymbol{a}^k, \boldsymbol{d} \rangle$, and furthermore, $A\boldsymbol{d}^{k+1} = \boldsymbol{1}$. Let $\boldsymbol{\Delta} = A\boldsymbol{d}_*^k - \boldsymbol{1}$, and notice that

$$\langle \boldsymbol{1}, \boldsymbol{\Delta} \rangle = \langle \boldsymbol{1}, A(\boldsymbol{d}_*^k - \boldsymbol{d}^{k+1}) \rangle = 0$$

18

Note that $\boldsymbol{d}_*^k = (\boldsymbol{1} + \boldsymbol{\Delta})/\boldsymbol{a}^k$, where we take the quotient componentwise as is defined at the start of Section 4.1. With $\boldsymbol{d}^{k+1} = \boldsymbol{1}/\boldsymbol{a}^k$, this gives $\mathrm{RelDist}(\boldsymbol{d}_*^k, \boldsymbol{d}^{k+1}) = \mathrm{RelDist}((\boldsymbol{1} + \boldsymbol{\Delta}), \boldsymbol{1})$. Therefore, we know that $\sum_{i=1}^n |\log(1 + \Delta_i)| > \varepsilon$. We also get

$$\mathcal{L}(\boldsymbol{d}^{k+1}) - \mathcal{L}(\boldsymbol{d}_*^k) = \sum_{i=1}^n \log(1/a_i^k) - \log((1 + \Delta_i)/a_i^k) = -\sum_{i=1}^n \log(1 + \Delta_i)$$

Define:

$$F(z) := \begin{cases} \frac{1}{4}z^2 & \text{if } -1 < z \le 1 \\ \frac{1}{2}z - \frac{1}{4} & \text{if } z \ge 1 \\ +\infty & \text{otherwise} \end{cases}$$

At $z = 0$, we have that $-z + F(z) = 0 = \log(1 + z)$ and $\frac{\mathrm{d}}{\mathrm{d}z}(-z + F(z)) = -1 = \frac{\mathrm{d}}{\mathrm{d}z}(-\log(1 + z))$. By comparing derivatives for the other values of $z > -1$, we can show that $-\log(1 + z) \ge -z + F(z)$ for all z. Thus,

$$\mathcal{L}(\boldsymbol{d}^{k+1}) - \mathcal{L}(\boldsymbol{d}_*^k) = -\sum_{i=1}^n \log(1 + \Delta_i) \ge \sum_{i=1}^n -\Delta_i + \sum_{i=1}^n F(\Delta_i) = \sum_{i=1}^n F(\Delta_i)$$

Now, since we have $\sum_{i=1}^n |\log(1 + \Delta_i)| > \varepsilon$, there must be some i such that $|\log(1 + \Delta_i)| > \varepsilon/n$. If $\Delta_i > 0$, then $\Delta_i \ge \log(1 + \Delta_i) \ge \varepsilon/n$. Conversely, if $\Delta_i < 0$, we being by noting that for $|z| < 0.5$, we have $-\log(1 + z) \le -z + z^2$ for reasons similar to the above. Thus, we get

$$\varepsilon/n < -\log(1 + \Delta_i) \le -\Delta_i + \Delta_i^2$$

We must have $\Delta_i > -1$, since the argument can't be negative, so we have $2|\Delta_i| > \Delta_i^2 - \Delta_i > \varepsilon/n$, or $\Delta_i < -\frac{1}{2}\varepsilon/n$. Noting that $F(z) \ge 0$ for all z, we can then conclude

$$\mathcal{L}(\boldsymbol{d}^{k+1}) - \mathcal{L}(\boldsymbol{d}_*^k) \ge \sum_i F(\Delta_i) \ge \max_i F(\Delta_i) \ge \frac{1}{16}\varepsilon^2/n^2$$

as desired. $\qquad\square$

Finally, to bound the number of iterations Algorithm 1 would take we need to bound the log-NSW value at the starting point, namely $\mathcal{L}(\boldsymbol{d}^0)$, where $\boldsymbol{d}^0 := \boldsymbol{1} \cdot \frac{m}{2n} \min_{i,j} D_{ij}$, as in Claim 9. We show the following.

Lemma 13. *Starting at $\boldsymbol{d}^0 := \boldsymbol{1} \cdot \frac{m}{2n} \min_{i,j} D_{ij}$, Algorithm 1 finds a $(1 + \varepsilon)$-KKT point in*

$$O\left(\frac{n^3}{\varepsilon^2} \cdot \log\left(\frac{n \cdot \max_{i,j} D_{ij}}{\min_{i,j} D_{ij}}\right)\right)$$

many iterations.

Proof. If we can bound the range of the log-NSW objective, then the proof follows using Lemmas 10 and 12. Let M be such that $D_i(\boldsymbol{x}_i) \le M$ for every agent i, at every feasible $\boldsymbol{x} \in \mathcal{F}$. Note that $M \le m \cdot \max_{i,j} D_{ij}$.

Then we have that for any feasible $\boldsymbol{x}$, $\mathcal{L}(\boldsymbol{d}(\boldsymbol{x})) \le n \log M$. Since each round of the above algorithm that doesn't terminate increases the log-NSW by at least $\frac{1}{16}(\varepsilon/n)^2$, then the total number of rounds possible is at most

$$16 \cdot \frac{n^2}{\varepsilon^2} \cdot (n \log(M) - \mathcal{L}(\boldsymbol{d}^0)) \le \frac{16n^3}{\varepsilon^2} \cdot \left(\log(m \cdot \max_{i,j} D_{ij}) - \log(\tfrac{m}{2n} \min_{i,j} D_{ij})\right),$$

which gives the desired bound. $\qquad\square$

Now that we have shown there are polynomially many iterations in our algorithm, it suffices to show that each iteration can be implemented in polynomial time to establish that Algorithm 1 is indeed polynomial time.

Implementing Each Iteration in Polynomial Time. To show that each iteration can be implemented in polynomial time, it suffices to show that the *nearest neighbour search* (step 3 in Algoritm 1) can be implemented in polynomial time.

Lemma 14. *Each iteration of Algorithm 1 can be computed exactly in time polynomial in n, m, and the description complexity of the D_{ij}'s.*

Proof. Let $\overrightarrow{D}(\boldsymbol{x}) := (D_1(\boldsymbol{x}_1), \ldots, D_n(\boldsymbol{x}_n))$ as defined previously. Recall that disutility functions are linear, with $D_i(\boldsymbol{x}_i) := \sum_{j=1}^{m} D_{ij} x_{ij}$.

Let $\widehat{D}$ be the $n \times nm$ block-diagonal matrix such that $\widehat{D}\boldsymbol{x} = \overrightarrow{D}(\boldsymbol{x})$. To find the nearest-feasible disutility profiles, we will find the allocation $\boldsymbol{x}$ which minimizes the following convex quadratic program:

$$\min_{\boldsymbol{x} \in \mathcal{F}} \left\| \overrightarrow{D}(\boldsymbol{x}) - \boldsymbol{d}^k \right\|_2^2 = \min_{\boldsymbol{x} \in \mathcal{F}} \boldsymbol{x}^\top \left(\widehat{D}^\top \widehat{D} \right) \boldsymbol{x} - 2(\boldsymbol{d}^k)^\top \widehat{D}\boldsymbol{x} + (\boldsymbol{d}^k)^\top \boldsymbol{d}^k .$$

It was shown by Khachiyan et al. [KTK80] that this program can be solved exactly, with running time polynomial in the description complexity of the system. Thus, so long as $\widehat{D}$ and $\boldsymbol{d}^k$ have rational entries with polynomial description complexity (polynomial-sized numerators and denominators), the problem can be solved exactly in polynomial time, and the solution will have small description complexity.

The matrix $\widehat{D}$ consists of the D_{ij}'s and our running time is assumed to depend on their description complexity. □

Final Result. We now have all the ingredients to conclude that an approximate CEEI (Definition 1) can be computed in polynomial time. Lemma 13 bounds the number of iterations as a polynomial in n, $1/\varepsilon$, and the description complexity of the instance, Claim 9 shows how to find a good starting point, Lemma 14 shows that each iteration can be computed in polynomial time, with the same arguments, and Theorem 4 shows how to compute a ε-CEEI in polynomial time given the output of Algorithm 1. Thus, we conclude that Algorithm 1 is an FPTAS for finding 3ε-CEEI.

Theorem 15. *Given linear disutility values $D_{11}, \ldots, D_{nm}$, Algorithm 1, along with 4, finds an ε-CEEI in time polynomial in n, m, $1/\varepsilon$, $\log(\frac{\max D_{ij}}{\min D_{ij}})$, and the description complexity of the D_{ij}'s.*

5 1-Homogeneous Disutilities: Computing ε-CEEI in Polynomial-Time

In this section, we show how to extend the results of the previous section when agents' disutility functions are general 1-homogeneous and convex. Access to the disutility functions are through value oracle. For ease of notation, throughout this section, we refer to the ith coordinate of a disutility vector $\boldsymbol{d}$ as $\boldsymbol{d}_i$ (or equivalently $(\boldsymbol{d})_i$). Similarly, given an allocation $\boldsymbol{x}$, we refer to agent i's bundle as $\boldsymbol{x}_i$ (or equivalently $(\boldsymbol{x})_i$) and the amount of chore j allocated to agent i as $\boldsymbol{x}_{ij}$ (or equivalently $(\boldsymbol{x})_{ij}$). We first discuss the two main roadblocks in generalizing the approach in Section 4. The convex program $\min_{\boldsymbol{x} \in \mathcal{F}} \|\overrightarrow{D}(\boldsymbol{x}) - \boldsymbol{d}^k\|_2^2$ for finding the nearest neighbour is not necessarily convex when agents have general 1-homogeneous and convex disutilities. We design an

alternative formulation that returns the nearest feasible point, and is convex. However, the domain of the new convex program is not defined by a set of linear inequalities and as such one can only find approximate nearest neighbours, *e.g.*, via interior point methods [Bub14]. In turn, the supporting hyperplanes $\langle \boldsymbol{a}, \boldsymbol{y} \rangle = n$ in Algorithm 1 are now approximate supporting hyperplanes. To allow this extra error, we extend the notion of approximate KKT to that given in Definition 3, namely

1. $(\boldsymbol{z})_{ij} \geq 0$ for all $i \in [n]$, and $j \in [m]$, and $\lambda^{-1} \leq \sum_{i \in [n]} (\boldsymbol{z})_{ij} \leq \lambda$ for all $j \in [m]$,

2. $\boldsymbol{d} = \overrightarrow{D}(\boldsymbol{z})$ and $\langle \boldsymbol{a}, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}, \boldsymbol{d} \rangle - \delta = n - \delta$ for all $\boldsymbol{y} \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$, and

3. for each $i \in [n]$, we have $\gamma^{-1} \leq (\boldsymbol{a})_i \cdot (\boldsymbol{d})_i \leq \gamma$.

In the previous section, with linear disutilities we have $\lambda = 1$ and $\delta = 0$ in the above definition, and this was crucially used to map approximate KKT to stronger approximate CEEI. We show in Section 5.1 that the claim follows even with $\lambda > 1, \delta > 0$.

For all of these to work, the disutility functions have to be *well-behaved*. To this end, we make the following assumptions about the rate of growth of the disutility functions.

Assumption 16. We assume that the disutility functions have Lipschitz-style lower- and upper-bounds. Formally, for some constant $L > 0$, we assume that for all $i \in [n]$ and for all $j \in [m]$, we have $|D_i(\boldsymbol{x} + \delta \cdot \boldsymbol{e}_j) - D_i(\boldsymbol{x})| \geq \delta/L$; furthermore, for all $i \in [n]$, and all $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$, we assume $|D_i(\boldsymbol{x}) - D_i(\boldsymbol{y})| \leq L \cdot ||\boldsymbol{x} - \boldsymbol{y}||_2$.

The running time of our algorithm will be polynomial in $\log(L)$. However, we believe that we can also handle cases with a weaker lower-Lipschitz condition: for all $i \in [n]$, $|D_i(\boldsymbol{x} + \delta \boldsymbol{e}_j) - D_i(\boldsymbol{x})| \geq \frac{1}{L} \cdot min(\delta, \delta^k)$ for $k \in \text{poly}(n, m)$. Towards the end of this section, we briefly mention what changes would be required to Algorithm 2 to make it work with the weaker assumption. For simplicity, we stick to Assumption 16 for the rest of this section.

Analogously to the linear case, we begin by showing in Section 5.1 that approximate KKT points will constructively yield approximate CEEI, and show in Section 5.2 a refinement of Algorithm 1 to find these approximate KKT points in the general setting. Finally, in Section 5.3, we bound the number of iterations of this new algorithm, and show how to compute each iteration efficiently.

5.1 $(\lambda, \gamma, \delta)$-KKT Gives Approximate CEEI.

In this Subsection, we show the following strengthening of Theorem 4.

Theorem 17. *Let* $(\boldsymbol{a}, \boldsymbol{d}, \boldsymbol{x})$ *be a* $(\lambda, \gamma, \delta)$-*KKT point for the problem of minimizing* $\mathcal{L}(\boldsymbol{y})$ *subject to* $\boldsymbol{y} \in \mathcal{D}$, *and* $\mathcal{L}(\boldsymbol{y}) > -\infty$. *Then there exists payments* $\boldsymbol{p} = (p_1, \ldots, p_m)$ *such that* $(\boldsymbol{x}, \boldsymbol{p})$ *form a* ε-*CEEI, as in Definition 1, where* $\varepsilon = \max\{3(\gamma - 1) + 5\delta, \ \lambda - 1\}$.

The whole of this subsection constitutes the proof of the above theorem. Let $(\boldsymbol{a}, \boldsymbol{d}, \boldsymbol{z})$ be a $(\lambda, \gamma, \delta)$-KKT point, as in Definition 3. Since $\boldsymbol{a}_i \geq 0$ for all $i \in [n]$, let $\boldsymbol{d}^*$ be any point in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ such that $\{\boldsymbol{y} \in \mathbb{R}^n | \langle \boldsymbol{a}, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}, \boldsymbol{d}^* \rangle\}$ is a supporting hyperplane for $\mathcal{D} + \mathbb{R}^n_{\geq 0}$. Since $\boldsymbol{d}^* \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$ and $\langle \boldsymbol{a}, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}, \boldsymbol{d} \rangle - \delta$ for all $\boldsymbol{y} \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$, we have $\langle \boldsymbol{a}, \boldsymbol{d}^* \rangle \geq \langle \boldsymbol{a}, \boldsymbol{d} \rangle - \delta$, or equivalently, $\langle \boldsymbol{a}, \boldsymbol{d} \rangle - \langle \boldsymbol{a}, \boldsymbol{d}^* \rangle \leq \delta$.

From here on, our proof emulates the proof of Theorem 4, and consequently the proof of Bogomolnaia et al. [BMSY17]. Recall that we have defined

$$\mathcal{F}' := \left\{ \boldsymbol{y} \in \mathbb{R}^{nm}_{\geq 0} \ \middle| \ \sum_{i \in [n]} \boldsymbol{y}_{ij} \geq 1 \text{ for all } j \in [m] \right\} ,$$

the pre-image of $\mathcal{D} + \mathbb{R}_{\geq 0}^n$ under $\overrightarrow{D}$. As before, let $S_\lambda = \{\boldsymbol{x} \in R^{mn} \mid \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{x}) \rangle \leq \lambda\}$. S_λ is non-empty, closed and convex for all $\lambda > 0$ as the disutilities are convex, and $\boldsymbol{0}$ is feasible. Since $\langle \boldsymbol{a}, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}, \boldsymbol{d}^* \rangle$ for all $\boldsymbol{y} \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$, we have $S_\lambda \cap \mathcal{F}' = \emptyset$ for all $\lambda < \langle \boldsymbol{a}, \boldsymbol{d}^* \rangle$. Thus, $S^* = S_{\langle \boldsymbol{a}, \boldsymbol{d}^* \rangle}$ is tangent to $\mathcal{F}'$ at the point $\boldsymbol{z}^*$ and $\overrightarrow{D}(\boldsymbol{z}^*) = \boldsymbol{d}^*$. Therefore, there exists a supporting hyperplane $\langle \boldsymbol{c}, \boldsymbol{x} \rangle = b$ of $\mathcal{F}'$ at $\boldsymbol{z}^*$, separating $\mathcal{F}'$ from S^*. We define the vector $\boldsymbol{p} \in \mathbb{R}_{\geq 0}^m$ such that $\boldsymbol{p}_j = min_{i \in [n]}(\boldsymbol{c})_{ij}$. Define $H_{\boldsymbol{c}} = \{\boldsymbol{x} \in \mathbb{R}^{nm} \mid \langle \boldsymbol{c}, \boldsymbol{x} \rangle \geq b\}$ and $H_{\boldsymbol{p}} = \{\boldsymbol{x} \in \mathbb{R}^{nm} \mid \sum_{j \in [m]} \sum_{i \in [n]} \boldsymbol{p}_j \boldsymbol{x}_{ij} \geq b\}$. As before, we have $\mathcal{F} \subseteq H_{\boldsymbol{c}}$.

Satisfying Condition (1) in Definition 1. We want to show that for all agents i and i', we have $(\gamma + 2\delta)^{-2} \cdot \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle \leq \langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle$. Note that for γ sufficiently close to 1, and δ sufficiently small, $(\gamma + 2\delta)^2 \leq 1 + 3(\gamma - 1) + 5\delta$. We begin with the following observations.

Claim 18. *We have $\sum_{j \in [m]} \boldsymbol{p}_j = \langle \boldsymbol{c}, \boldsymbol{z}^* \rangle = b$.*

Proof. Since $H_{\boldsymbol{c}}$ is a supporting hyperplane of $\mathcal{F}'$ at $\boldsymbol{z}^*$, we have $b = \min_{\boldsymbol{x} \in \mathcal{F}'} \langle \boldsymbol{c}, \boldsymbol{x} \rangle$. Observe that this minimum is obtained by assigning each chore fully to the agent that has the smallest $\boldsymbol{c}_{ij}$ value for it. Therefore, we have that

$$\min_{\boldsymbol{x} \in \mathcal{F}} \langle \boldsymbol{c}, \boldsymbol{x} \rangle = \sum_{j \in [m]} \min_{i \in [n]} \boldsymbol{c}_{ij} = \sum_{j \in [m]} \boldsymbol{p}_j \ . \qquad \square$$

Claim 19. $H_{\boldsymbol{p}} \subseteq H_{\boldsymbol{c}}$.

Proof. Consider any point $\boldsymbol{x} \in H_{\boldsymbol{p}}$. We have $b \leq \sum_{i \in [n]} \sum_{j \in [m]} \boldsymbol{p}_j \boldsymbol{x}_{ij}$. Since $p_j \leq c_{ij}$ for all $i \in [n]$ and $j \in [m]$,

$$b \leq \sum_{i \in [n]} \sum_{j \in [m]} \boldsymbol{p}_j \boldsymbol{x}_{ij} \leq \sum_{i \in [n]} \sum_{j \in [m]} \boldsymbol{c}_{ij} \boldsymbol{x}_{ij} = \langle \boldsymbol{c}, \boldsymbol{x} \rangle \ ,$$

as desired. $\qquad \square$

Claim 20. $\mathcal{F}' \subseteq H_{\boldsymbol{p}}$, and $\mathcal{F} \subseteq \partial H_{\boldsymbol{p}}$, the boundary.

Proof. Consider any $\boldsymbol{x} \in \mathcal{F}$. Note that this is the original feasible region with equality. We have

$$\sum_{i \in [n]} \sum_{j \in [m]} \boldsymbol{p}_j \boldsymbol{x}_{ij} = \sum_{j \in [m]} \boldsymbol{p}_j = b \ .$$

The first equality holds since we assume $\sum_i \boldsymbol{x}_{ij} = 1$ in $\mathcal{F}$, and the second holds by Claim 18. If instead $\boldsymbol{x} \in \mathcal{F}'$, then the first equality becomes an inequality, concluding the proof. $\qquad \square$

With these three claims, we can show the first condition for CEEI. Assume for a contradiction that $(\gamma + 2\delta)^{-2} \cdot \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle > \langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle$ for some i, i'. Then we could replace the allocation as follows: Construct $\hat{\boldsymbol{z}}$ by setting $\hat{\boldsymbol{z}}_{i'} = \frac{1}{2} \boldsymbol{z}_{i'}$, and $\hat{\boldsymbol{z}}_i = \left(1 + \frac{\langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle}{2 \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle} \right) \boldsymbol{z}_i$. Since $\boldsymbol{z} \in \mathcal{F}$, we have $\sum_{i \in [n], j \in [m]} \boldsymbol{p}_j \boldsymbol{z}_{ij} = b$. Also note that

$$b = \sum_{i \in [n], j \in [m]} \boldsymbol{p}_j \hat{\boldsymbol{z}}_{ij} = \sum_{i \in [n], j \in [m]} \boldsymbol{p}_j \boldsymbol{z}_{ij} \ ,$$

since the payment subtracted from agent i' is equal to the payment added to agent i and so $\hat{\boldsymbol{z}} \in H_{\boldsymbol{p}}$.

By Claim 19, we have that $\hat{\boldsymbol{z}} \in H_{\boldsymbol{c}}$. Recall that $H_{\boldsymbol{c}}$ is a separating half-space between S^* and $\mathcal{F}'$, i.e., $\mathcal{F}' \subseteq H_{\boldsymbol{c}}$ and $S^* \subseteq \mathrm{cl}(H_{\boldsymbol{c}}^{\complement})$, implying that for every point $\boldsymbol{x} \in H_{\boldsymbol{c}}$ we have $\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{x}) \rangle \geq \langle \boldsymbol{a}, \boldsymbol{d}^* \rangle \geq \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle - \delta$. Since $\hat{\boldsymbol{z}} \in H_{\boldsymbol{c}}$, we have $\langle \boldsymbol{a}, \overrightarrow{D}(\hat{\boldsymbol{z}}) \rangle \geq \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle - \delta$. However,

$$\langle \boldsymbol{a}, \overrightarrow{D}(\hat{\boldsymbol{z}}) \rangle - \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle = -\frac{\boldsymbol{a}_{i'} \cdot D_{i'}(\boldsymbol{z}_{i'})}{2} + \boldsymbol{a}_i \cdot D_i(\boldsymbol{z}_i) \cdot \frac{\langle \boldsymbol{p}, \boldsymbol{z}_{i'} \rangle}{2\langle \boldsymbol{p}, \boldsymbol{z}_i \rangle}$$

$$\leq -\frac{\gamma^{-1}}{2} + \gamma \cdot \frac{\langle \boldsymbol{p}, \boldsymbol{z}_{i'} \rangle}{2\langle \boldsymbol{p}, \boldsymbol{z}_i \rangle} \qquad\qquad (\gamma^{-1} \leq a_i D_i(\boldsymbol{z}_i) \leq \gamma)$$

$$\leq -\frac{\gamma^{-1}}{2} + \frac{\gamma}{2(\gamma + 2\delta)^2} = -\frac{\gamma^{-1}}{2}\left(1 - \frac{1}{(1+\frac{2\delta}{\gamma})^2}\right)$$

Now, we have $1 - \frac{1}{(1+p)^2} > 2p - 3p^2$ for $p \geq 0$, and therefore

$$-\frac{\gamma^{-1}}{2}\left(1 - \frac{1}{(1 + \frac{2\delta}{\gamma})^2}\right) \;<\; -\tfrac{1}{2}\gamma^{-1}(4\delta/\gamma - 8\delta^2/\gamma^2) \;\leq\; -\tfrac{1}{2}\gamma^{-1}(3\delta/\gamma) \;\leq\; -\delta \;,$$

for δ sufficiently close to 0 and γ sufficiently close to 1. This implies that we have $\langle \boldsymbol{a}, \overrightarrow{D}(\hat{\boldsymbol{z}}) \rangle < \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle - \delta = \langle \boldsymbol{a}, \boldsymbol{d} \rangle - \delta$, which is a contradiction.

Satisfying Condition (2) in Definition 1. We want to show that for all $i \in [n]$, we have $(1 - 2\delta) \cdot D_i(\boldsymbol{z}_i) \leq D_i(\boldsymbol{y})$ for all $\boldsymbol{y}$ such that $\langle \boldsymbol{y}, \boldsymbol{p} \rangle \geq \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle$. Let us assume that there exists a $\boldsymbol{y}$ such that $(1 - 2\delta) \cdot D_i(\boldsymbol{z}_i) > D_i(\boldsymbol{y})$ and $\langle \boldsymbol{y}, \boldsymbol{p} \rangle \geq \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle$. We define a new allocation $\boldsymbol{z}' = (\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{i-1}, \boldsymbol{y}, \boldsymbol{z}_{i+1}, \ldots, \boldsymbol{z}_n)$. First note that $\sum_{i \in [n], j \in [m]} z'_{ij} \cdot p_j \geq \sum_{i \in [n], j \in [m]} z_{ij} \cdot p_j = b$ as $\langle \boldsymbol{y}, \boldsymbol{p} \rangle \geq \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle$. Therefore $\boldsymbol{z}' \in H_{\boldsymbol{p}}$. By Claim 19, we have that $\boldsymbol{z}' \in H_{\boldsymbol{c}}$. Recall that $H_{\boldsymbol{c}}$ is a separating half-space between S^* and $\mathcal{F}'$, i.e. $\mathcal{F}' \subseteq H_{\boldsymbol{c}}$ and $S^* \subseteq \mathrm{cl}(H_{\boldsymbol{c}}^{\complement})$. Therefore $\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}') \rangle \geq \langle \boldsymbol{a}, \boldsymbol{d}^* \rangle \geq \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle - \delta$. However,

$$\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}') \rangle - \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle = \boldsymbol{a}_i \cdot (D_i(\boldsymbol{z}'_i) - D_i(\boldsymbol{z}_i))$$

$$< -2\delta \boldsymbol{a}_i D_i(\boldsymbol{z}_i) \leq -2\gamma^{-1}\delta \leq -\delta \;.$$

This implies that $\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}') \rangle < \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle - \delta$, which is a contradiction.

Satisfying Condition (3) in Definition 1. By definition of $(\lambda, \gamma, \delta)$-KKT point, we have $(\boldsymbol{z})_{ij} \geq 0$ for all $i \in [n]$, and $j \in [m]$ and also $\lambda^{-1} \leq \sum_{i \in [n]}(\boldsymbol{z})_{ij} \leq \lambda$ for all $j \in [m]$.

5.2 Exterior-Point Methods for $(\lambda, \gamma, \delta)$-KKT Points

We introduce here a refinement of Algorithm 1 which allows us to handle the extra errors, and finds $(\lambda, \gamma, \delta)$-KKT points in polynomial time. The algorithm takes as input three error terms, ε_1, ε_2, and ε_3, and we will later see how to set these to attain polynomial running time. We encourage the reader to go through Subsection 1.2.3 to get an overview of the entire algorithm and the challenges it handles compared to the setting where all agents have linear disutilities.

Recall that $\mathcal{F}' := \mathcal{F} + \mathbb{R}_{\geq 0}^{nm} = \{\boldsymbol{x} \in \mathbb{R}_{\geq 0}^{nm} \mid \sum_{i \in [n]} x_{ij} \geq 1 \;\; \forall i \in [n]\}$, as above, and note that $\mathcal{D} + \mathbb{R}_{\geq 0}^n = \{\boldsymbol{d} \in \mathbb{R}^n \mid \boldsymbol{d} = \overrightarrow{D}(\boldsymbol{y}) \text{ for some } \boldsymbol{y} \in \mathcal{F}'\}$. Our new algorithm is as follows.

Algorithm 2 Finding Approximate KKT for 1-Homogeneous Disutilities

1: Let $(\boldsymbol{d}^0) \leftarrow$ INITIALIZE(), be any infeasible, strictly positive, disutility profile near $\boldsymbol{0}$, and $k = 1$
2: **while** true **do**
3:      **Set** $(\boldsymbol{x}_+^k, \boldsymbol{d}_+^k) \leftarrow$ NEAREST-POINT$(\boldsymbol{d}^k, \varepsilon_1)$
4:      **Set** $(\boldsymbol{x}_+^k)_{ij} \leftarrow (\boldsymbol{x}_+^k)_{ij} + L \cdot (2 \cdot \varepsilon_1)$ for all $i \in [n]$ and $j \in [m]$ and $\boldsymbol{d}_+^k \leftarrow \overrightarrow{D}(\boldsymbol{x}_+^k)$
5:      **if** $||\boldsymbol{d}_+^k - \boldsymbol{d}^k||_2 \leq \varepsilon_2$ **then**
6:          **Set** $(\boldsymbol{x}_+^k, \boldsymbol{d}_+^k) \leftarrow$ ADJUST-COORDINATES$(\boldsymbol{x}_+^k, \boldsymbol{d}_+^k, \boldsymbol{d}^k)$.
7:          **Return** $(\boldsymbol{a}^{k-1}, \boldsymbol{d}_+^k, \boldsymbol{x}_+^k)$
8:      **Set** $\boldsymbol{a}^k = \boldsymbol{d}_+^k - \boldsymbol{d}^k$.
9:      **Rescale** $\boldsymbol{a}^k$ so that $\langle \boldsymbol{a}^k, \boldsymbol{d}_+^k \rangle = n$.
10:     **Set** $\boldsymbol{d}^{k+1} \leftarrow \boldsymbol{1}/\boldsymbol{a}^k$
11:     **if** RelDist$(\boldsymbol{d}^{k+1}, \boldsymbol{d}_+^k) < \varepsilon_3$ **then**
12:         **Return** $(\boldsymbol{a}^k, \boldsymbol{d}_+^k, \boldsymbol{x}_+^k)$
13:     **Set** $k \leftarrow k + 1$

Algorithm 3 INITIALIZE()

1: **Fix** an arbitrary $a \in [n]$ and $c \in [m]$.
2: **Set** $(\boldsymbol{x}^0)_{ac} \leftarrow 1/2nL^2$ and $(\boldsymbol{x})_{ij} = 0$ for all other $i \in [n]$ and $j \in [m]$.
3: **Set** $\boldsymbol{d}^0 \leftarrow \overrightarrow{D}(\boldsymbol{x}^0)$.
4: **Return** $\boldsymbol{d}^0$

Algorithm 4 NEAREST-POINT$(\boldsymbol{d}^k, \varepsilon_1)$

Returns the point $(\boldsymbol{x}_+^k, \boldsymbol{d}_+^k)$ such that $\boldsymbol{d}_+^k = \overrightarrow{D}(\boldsymbol{x}_+^k)$ and $||\boldsymbol{x}_+^k - \boldsymbol{x}_*^k||_2 \leq \varepsilon_1$ and $||\boldsymbol{d}_+^k - \boldsymbol{d}_*^k||_2 \leq \varepsilon_1$ where $\boldsymbol{x}_*^k \in \mathcal{F}'$ such that $\boldsymbol{d}_*^k = \overrightarrow{D}(\boldsymbol{x}_*^k)$ is the nearest point to $\boldsymbol{d}^k$ in $\mathcal{D} + \mathbb{R}_{\geq 0}^n$. Also every coordinate of $\boldsymbol{x}_+^k$ and $\boldsymbol{d}_+^k$ is an integral multiple of $\varepsilon_1/2^{\mathrm{poly}(n,m)}$ for some poly(n, m). The details of this algorithm will be presented in Section 5.3.

Algorithm 5 ADJUST-COORDINATES$(\boldsymbol{x}_+^k, \boldsymbol{d}_+^k, \boldsymbol{d}^k)$

1: **Set** $y_{ij} = (\boldsymbol{x}^k)_{ij} \cdot \frac{(\boldsymbol{d}^k)_i}{(\boldsymbol{d}_+^k)_i}$ for all $i \in [n]$, $j \in [m]$.
2: **Return** $(\boldsymbol{y}, \overrightarrow{D}(\boldsymbol{y}))$.

We begin by showing that the above algorithm will correctly return a $(\lambda, \gamma, \delta)$-KKT point for the appropriate values of ε_1, ε_2, ε_3, which will be chosen later. For now, the reader should think of the ε's as being related as follows: $\varepsilon_3 \gg \varepsilon_2 \gg \varepsilon_1$. In particular, we have $\varepsilon_3 \geq 2^{\mathrm{poly}(n,m)} \cdot \varepsilon_2$ and $\varepsilon_2 \gg n^3 m^3 L^3 \varepsilon_1$.

The proof of correctness for Algorithm 2 will be significantly more involved than that of Algorithm 1. Notably, the error in the computation of the nearest point $\boldsymbol{d}^k$ on line 3 will introduce many sources of additive error, which will need to be handled along with Assumption 16 above, to ensure that we can recover multiplicative error guarantees.

We will need to argue that an approximate equilibrium can be found whether the algorithm returns in either of Steps 7 or 12. The latter was the stopping condition for the original Algorithm 1, and its proof will be simpler.

Let $\boldsymbol{d}_*^k$ be the true nearest point to $\boldsymbol{d}^k$ in $\mathcal{D} + \mathbb{R}_{\geq 0}^n$ and let $\boldsymbol{x}_*^k \in \mathcal{F}'$ be such that $\overrightarrow{D}(\boldsymbol{x}_*^k) = \boldsymbol{d}_*^k$.

5.2.1 Stopping on Line 12 of Algorithm 2

We first show that the $\boldsymbol{a}^k$ vectors are indeed normals to approximately supporting hyperplanes.

Lemma 21. *In each iteration k of Algorithm 2, after step 9, we have $\langle \boldsymbol{a}^k, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}^k, \boldsymbol{d}_+^k \rangle - \delta$ for all $y \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$ with $\delta = 9n^5 mL^2 \frac{\varepsilon_1}{\varepsilon_2^2}$.*

Before proving this results, we first need some technical claims.

Observation 22. In each iteration k of Algorithm 2, after step 9, we have $\boldsymbol{d}_*^k \in \mathcal{D}$, $\boldsymbol{x}_*^k \in \mathcal{F}$, $||\boldsymbol{d}_*^k||_2 \leq nL$.

Proof. Note that since the algorithm has constructed the vector $\boldsymbol{a}^k$, in iteration k, we have $\boldsymbol{d}^k \notin \mathcal{D} + \mathbb{R}_{\geq 0}^n$. Otherwise, by the choice of ε's, the algorithm would have terminated since $\boldsymbol{d}_+^k$ would have been too close to $\boldsymbol{d}^k = \boldsymbol{d}_*^k$. Therefore, we have $||\boldsymbol{d}_+^k - \boldsymbol{d}^k||_2 \ll \varepsilon_2$ and our algorithm will terminate before step 9. Therefore, $\boldsymbol{d}^k$ lies outside $\mathcal{D} + \mathbb{R}_{\geq 0}^n$, *i.e.*, below the lower envelop of $\mathcal{D} + \mathbb{R}_{\geq 0}^n$. By Lemma 8, $\boldsymbol{d}_*^k \in \mathcal{D}$. This implies that $\boldsymbol{d}_*^k$ has a pre-image $\boldsymbol{x}_*^k \in \mathcal{F}$ under $\overrightarrow{D}$. As a result, we can bound $||\boldsymbol{d}_*^k||_2 \leq \sqrt{n} \cdot max_{i \in [n]} D_i(\boldsymbol{1})$. Note that by Assumption 16, we have $D_i(\boldsymbol{1}) = D_i(\boldsymbol{1}) - D_i(\boldsymbol{0}) \leq L \cdot ||\boldsymbol{1} - \boldsymbol{0}||_2 \leq L\sqrt{n}$ for all $i \in [n]$. Therefore, we have $||\boldsymbol{d}_*^k||_2 \leq nL$. □

Observation 23. After step 4 of Algorithm 2, we have $(\boldsymbol{x}_+^k)_{ij} \geq (\boldsymbol{x}_*^k)_{ij}$ for all $i \in [n]$, $j \in [m]$ and that $(\boldsymbol{d}_+^k)_i \geq (\boldsymbol{d}_*^k)_i \geq (\boldsymbol{d}^k)_i$ for all $i \in [n]$, implying that $\boldsymbol{x}_+^k \in \mathcal{F}'$ and $\boldsymbol{d}_+^k \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$

Proof. By construction, after step 3, we have $|(\boldsymbol{x}_+^k)_{ij} - (\boldsymbol{x}_*^k)_{ij}| \leq \varepsilon_1$ for all $i \in [n]$ and $j \in [m]$ and $|(\boldsymbol{d}_*^k)_i - (\boldsymbol{d}_+^k)_i| \leq \varepsilon_1$ for all $i \in [n]$. Then in step 4, we increase all $(\boldsymbol{x}_+^k)_{ij}$ by $2L\varepsilon_1$ and thus we have $(\boldsymbol{x}_+^k)_{ij} \geq (\boldsymbol{x}_*^k)_{ij}$. Since our disutility functions have lower-bounded partial derivatives, we have for each $i \in [n]$, the disutility of agent i increases by at least $1/L \cdot 2L\varepsilon_1 = 2\varepsilon_1$ and thus we have $(\boldsymbol{d}_+^k)_i \geq (\boldsymbol{d}_*^k)_i$ for all $i \in [n]$. We complete the proof by showing that $(\boldsymbol{d}_*^k)_i \geq (\boldsymbol{d}^k)_i$ for all $i \in [n]$. To this end, first observe that if $\boldsymbol{d}^k \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$, then $\boldsymbol{d}_*^k = \boldsymbol{d}^k$ and the claim holds trivially. If $\boldsymbol{d}^k \notin \mathcal{D} + \mathbb{R}_{\geq 0}^n$, then by the same argument in Lemma 8, we can prove $(\boldsymbol{d}_*^k)_i \geq (\boldsymbol{d}^k)_i$ for all $i \in [n]$. □

We also show that $\boldsymbol{d}_+^k$ and $\boldsymbol{x}_+^k$ are in the $O(nmL^2 \varepsilon_1)$ neighbourhood of $\boldsymbol{d}_*^k$ and $\boldsymbol{x}_*^k$ respectively.

Observation 24. After step 4 of Algorithm 2, we have

- $||\boldsymbol{x}_+^k - \boldsymbol{x}_*^k||_2 \leq 3nmL\varepsilon_1$, and

- $||\boldsymbol{d}_+^k - \boldsymbol{d}_*^k||_2 \leq 3nmL^2\varepsilon_1$.

Proof. After step 3, we have $||\boldsymbol{x}_+^k - \boldsymbol{x}_*^k||_2 \leq \varepsilon_1$. Then, in step 4, we increased each $(\boldsymbol{x}_+^k)_{ij}$ by $2L\varepsilon_1$. Therefore, after step 4, we have $||\boldsymbol{x}_+^k - \boldsymbol{x}_*^k||_2 \leq \varepsilon_1 + 2nmL\varepsilon_1 \leq 3nmL\varepsilon_1$. Observe that,

$$||\boldsymbol{d}_+^k - \boldsymbol{d}_*^k||_2^2 = \sum_{i \in [n]} |D_i((\boldsymbol{x}_+^k)_i) - D_i((\boldsymbol{x}_*^k)_i)|^2$$

(by Assumption 16) $\leq L^2 \cdot \sum_{i \in [n]} |(\boldsymbol{x}_+^k)_i - (\boldsymbol{x}_*^k)_i|^2 \leq L^2 \cdot ||\boldsymbol{x}_+^k - \boldsymbol{x}_*^k||_2^2 \leq L^2 \cdot (3nmL\varepsilon)^2.$

This implies that $||\boldsymbol{d}_+^k - \boldsymbol{d}_*^k||_2 \leq 3nmL^2\varepsilon_1$. □

We can now prove the lemma.

Proof of Lemma 21. Recall, by Lemma 8, the direction $\boldsymbol{d}_*^k - \boldsymbol{d}^k$ is normal to a supporting hyperplane for $\mathcal{D} + \mathbb{R}_{\geq 0}^n$ at $\boldsymbol{d}_*^k$. Thus, for any $\boldsymbol{y} \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$, we can conclude from Observation 23 above, that

$$\langle \boldsymbol{d}_+^k - \boldsymbol{d}^k, \boldsymbol{y} \rangle \geq \langle \boldsymbol{d}_*^k - \boldsymbol{d}^k, \boldsymbol{y} \rangle \geq \langle \boldsymbol{d}_*^k - \boldsymbol{d}^k, \boldsymbol{d}_*^k \rangle \tag{5}$$

Now, we wish to bound $\langle \boldsymbol{d}_*^k - \boldsymbol{d}^k, \boldsymbol{d}_*^k \rangle$ so that we can control the hyperplane error. We have

$$\begin{aligned}
\langle \boldsymbol{d}_+^k - \boldsymbol{d}^k, \boldsymbol{d}_+^k \rangle - \langle \boldsymbol{d}_*^k - \boldsymbol{d}^k, \boldsymbol{d}_*^k \rangle &= \langle \boldsymbol{d}_+^k - \boldsymbol{d}^k, \boldsymbol{d}_+^k \rangle - \langle \boldsymbol{d}_+^k - \boldsymbol{d}^k, \boldsymbol{d}_*^k \rangle + \langle \boldsymbol{d}_+^k - \boldsymbol{d}^k, \boldsymbol{d}_*^k \rangle - \langle \boldsymbol{d}_*^k - \boldsymbol{d}^k, \boldsymbol{d}_*^k \rangle \\
&= \langle \boldsymbol{d}_+^k - \boldsymbol{d}^k, \boldsymbol{d}_+^k - \boldsymbol{d}_*^k \rangle + \langle \boldsymbol{d}_+^k - \boldsymbol{d}_*^k, \boldsymbol{d}_*^k \rangle \\
&\leq \|\boldsymbol{d}_+^k - \boldsymbol{d}_*^k\|_2 \cdot \left(\|\boldsymbol{d}_+^k - \boldsymbol{d}^k\|_2 + \|\boldsymbol{d}_*^k\|_2 \right) \\
&\leq 3nmL\varepsilon_1 \cdot ((nL + 3nmL\varepsilon_1) + 0 + nL) \leq 9n^2 m L^3 \varepsilon_1 \ ,
\end{aligned}$$

where the last row follows from the fact that $\|\boldsymbol{d}_*^k - \boldsymbol{d}_+^k\|_2 \leq 3nmL^2\varepsilon_1$ (Observation 24) and that $\|\boldsymbol{d}_*^k\|_2 \leq nL$. Thus,

$$\begin{aligned}
\langle \boldsymbol{d}_+^k - \boldsymbol{d}^k, \boldsymbol{d}_+^k \rangle &\leq \langle \boldsymbol{d}_*^k - \boldsymbol{d}^k, \boldsymbol{d}_*^k \rangle + 9n^2 m L^3 \varepsilon_1 \\
&\leq \langle \boldsymbol{d}_+^k - \boldsymbol{d}^k, \boldsymbol{y} \rangle + 9n^2 m L^3 \varepsilon_1 \quad \forall \ \boldsymbol{y} \in \mathcal{D} + \mathbb{R}_{\geq 0}^n \qquad \text{by (5)}
\end{aligned}$$

Since $\boldsymbol{a}^k = (n / \langle \boldsymbol{d}_+^k - \boldsymbol{d}^k, \boldsymbol{d}_+^k \rangle) \cdot (\boldsymbol{d}_+^k - \boldsymbol{d}^k)$, we have for for all $y \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$,

$$\langle \boldsymbol{a}^k, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}^k, \boldsymbol{d}_+^k \rangle - \frac{n}{\langle \boldsymbol{d}_+^k - \boldsymbol{d}^k, \boldsymbol{d}_+^k \rangle} \cdot 9n^2 m L^3 \varepsilon_1 \tag{6}$$

Observe that since we are step 9 of Algorithm 2, our algorithm did not terminate in step 7, and we have $\|\boldsymbol{d}_+^k - \boldsymbol{d}^k\|_2 > \varepsilon_2$. Again, since $(\boldsymbol{d}_+^k)_i \geq (\boldsymbol{d}_*^k)_i \geq (\boldsymbol{d}^k)_i$ for all $i \in [n]$ (by Observation 23), there exists an $i' \in [n]$, such that $(\boldsymbol{d}_+^k)_i \geq (\boldsymbol{d}_+^k)_i - (\boldsymbol{d}^k)_i \geq \varepsilon_2/n$, implying that $\langle \boldsymbol{d}_+^k - \boldsymbol{d}^k, \boldsymbol{d}_+^k \rangle \geq \varepsilon^2/n^2$. Substituting this lower bound in 6, we have,

$$\langle \boldsymbol{a}^k, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}^k, \boldsymbol{d}_+^k \rangle - 9n^5 m L^3 (\varepsilon_1/\varepsilon_2^2). \qquad \square$$

With the bound of Lemma 21, we can now show that if the algorithm stops on Step 12, *i.e.* if $\boldsymbol{d}^{k+1}$ is too close to $\boldsymbol{d}_+^k$, then we have an approximate KKT point.

Lemma 25. *Let $(\boldsymbol{a}^k, \boldsymbol{x}_+^k, \boldsymbol{d}_+^k)$ be the point returned by Algorithm 2 in Step 12. Then $(\boldsymbol{a}^k, \boldsymbol{x}_+^k, \boldsymbol{d}_+^k)$ is a $(\lambda, \gamma, \delta)$-KKT point with $\lambda = 1 + n^2 L \varepsilon_1$, $\gamma = 1 + \varepsilon_3$ and $\delta = 9n^5 m L^3 \frac{\varepsilon_1}{\varepsilon_2^2}$.*

Proof. We first show that for each $i \in [n]$, we have $\gamma^{-1} \leq (\boldsymbol{a}^k)_i \cdot (\boldsymbol{d}_+^k)_i \leq \gamma$. Since our algorithm returns this point in step 12, we have $\text{RelDist}(\boldsymbol{d}_+^k, 1/\boldsymbol{a}^k) \leq \varepsilon_3$. This implies that for each $i \in [n]$, we have $(1+\varepsilon_3)^{-1} \leq (\boldsymbol{d}_+^k)_i \cdot (\boldsymbol{a}^k)_i \leq (1+\varepsilon_3)$. Also, by Lemma 21, we have $\langle \boldsymbol{a}^k, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}^k, \boldsymbol{d}_+^k \rangle - \delta = n - \delta$ for all $\boldsymbol{y} \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$ and $\delta = 9n^5 m L^3 \varepsilon_1/(\varepsilon_2^2)$.

It remains to show that $(\boldsymbol{x}_+^k)_{ij} \geq 0$ for all $i \in [n]$, $j \in [m]$ and $\lambda^{-1} \leq \sum_{i \in [n]} (\boldsymbol{x}_+^k)_{ij} \leq \lambda$. We have that after step 4 of Algorithm 2, $(\boldsymbol{x}_+^k)_{ij} \geq (\boldsymbol{x}_*^k)_{ij} \geq 0$ for all i, j. This is because by construction, after step 3 of Algorithm 2, $(\boldsymbol{x}_+^k)_{ij} \geq (\boldsymbol{x}_*^k)_{ij} - \varepsilon_1$. Furthermore after step 4, $(\boldsymbol{x}_+^k)_{ij}$ is increased by an additive factor of $2L\varepsilon_1$ and thus it becomes larger than $(\boldsymbol{x}_*^k)_{ij}$. Since $\boldsymbol{d}_*^k \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$ and $\boldsymbol{x}_*^k \in \mathcal{F}'$, we have $(\boldsymbol{x}_*^k)_{ij} \geq 0$, further implying that $(\boldsymbol{x}_+^k)_{ij} \geq 0$ as well.

Now, as our algorithm is returning in step 12 and not step 7, we do not change $\boldsymbol{x}_+^k$ after step 4. Therefore, we have $(\boldsymbol{x}_+^k)_{ij} \geq 0$ for all $i \in [n]$ and $j \in [m]$ after step 12 also.

We now show $\lambda^{-1} \leq \sum_{i \in [n]} (\boldsymbol{x}_+^k)_{ij} \leq \lambda$. To this end, first observe that after step 4, $(\boldsymbol{x}_*^k)_{ij} \leq (\boldsymbol{x}_+^k)_{ij} \leq (\boldsymbol{x}_*^k)_{ij} + \varepsilon_1 + 2L\varepsilon_1 \leq (\boldsymbol{x}_*^k)_{ij} + 3L\varepsilon_1$ for all $i \in [n]$ and $j \in [m]$. Furthermore, by Observation 22, we have that $\boldsymbol{d}_*^k \in \mathcal{D}$ and $\boldsymbol{x}_*^k \in \mathcal{F}$. Therefore, we have $\sum_{i \in [n]} (\boldsymbol{x}_*^k)_{ij} = 1$ for all $j \in [m]$. Since, $(\boldsymbol{x}_*^k)_{ij} \leq (\boldsymbol{x}_+^k)_{ij} \leq (\boldsymbol{x}_*^k)_{ij} + 3L\varepsilon_1$, we have $(\lambda)^{-1} \leq 1 - 3nL\varepsilon_1 \leq \sum_{i \in [n]} (\boldsymbol{x}_+^k)_{ij} \leq 1 + 3nL\varepsilon_1 \leq \lambda$. $\qquad\square$

5.2.2 Stopping on Line 7 of Algorithm 2

We have shown that if Algorithm 2 stops in step 12, then we have an approximate KKT point. It remains to show that this holds if we stop in step 7. We start by addressing a small subtlety. The normal vector $\boldsymbol{a}^{k-1}$ is only well defined from $k \geq 2$. Therefore, we need to show that Algorithm 2 never returns at Line 7 at $k = 1$ (the first iteration). This follows from the fact that the distance between $\boldsymbol{d}^0$ and any point in $\mathcal{D} + \mathbb{R}_{\geq 0}^n \gg \varepsilon_2$. To see this, note that for any disutility vector $\boldsymbol{d} \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$, there is one agent who gets at least $1/n$ fraction of some chore and as a result his disutility will be at least $1/nL$ (by Assumption 16). However, the disutility of any agent in $\boldsymbol{d}^0$ is at most $L \cdot \|(\boldsymbol{x}^0)_i - \boldsymbol{0}\|_2 \leq 1/2nL$ (by Assumption 16 and Algorithm 3). Therefore $\|\boldsymbol{d} - \boldsymbol{d}^0\|_2 \geq \|\boldsymbol{d} - \boldsymbol{d}^0\|_\infty \geq 1/2nL \gg \varepsilon_2$. We now focus on the main proof.

Note that this stopping condition is an additive error, and we will need to be more careful. The bulk of the proof will lie in showing that the allocation $\boldsymbol{x}_+^k$ is neither an over-allocation nor an under-allocation of any of the chores. The remaining conditions will be relatively straightforward, as they are a consequence of Lemma 21 on the previous iteration.

Lemma 26. *Let $(\boldsymbol{a}^{k-1}, \boldsymbol{x}_+^k, \boldsymbol{d}_+^k)$ be the point returned by Algorithm 2 in Step 7. Then $(\boldsymbol{a}^{k-1}, \boldsymbol{x}_+^k, \boldsymbol{d}_+^k)$ is a $(\lambda, \gamma, \delta)$-approximate KKT point with $\gamma = 1 + \varepsilon_3$, $\delta = 9n^5 m L^3 \frac{\varepsilon_1}{\varepsilon_2^2}$, and some λ.*

Proof. We have that (i) $\boldsymbol{d}^k$ lies on the hyperplane $\langle \boldsymbol{a}^{k-1}, \boldsymbol{y} \rangle = n$ where $\boldsymbol{a}^{k-1} = \boldsymbol{1}/\boldsymbol{d}^k$, by construction, and (ii) $\langle \boldsymbol{a}^{k-1}, \boldsymbol{y} \rangle = n$ is a δ-approximate supporting hyperplane of $\mathcal{D} + \mathbb{R}_{\geq 0}^n$, by Lemma 21). Therefore, to show that $(\boldsymbol{x}_+^k, \boldsymbol{d}_+^k)$ is a $(\lambda, \gamma, \delta)$-approximate KKT point with no λ bound, it suffices to show that $(\boldsymbol{x}_+^k)_{ij} \geq 0$ for all $i \in [n]$ and $j \in [m]$.

We have already shown in the proof of Lemma 25 that after line 4 of Algorithm 2, $(\boldsymbol{x}_+^k)_{ij} \geq 0$ for all $i \in [n]$ and $j \in [m]$. The returned allocation, after the application of ADJUST-COORDINATES, is a positively re-scaling of this vector. Thus, we satisfy the conditions of a $(\lambda, \gamma, \delta)$-KKT point assuming no λ bounds. $\qquad\square$

It remains then to show that $\sum_{i=1}^n (\boldsymbol{x}_+^k)_{ij}$ is not too far from 1 in any direction. Notice that our supporting hyperplane is approximately supporting the set $\mathcal{D} + \mathbb{R}_{\geq 0}^n$. Thus, it is relatively straightforward to argue, as we do here, that under-allocations are unlikely, but any arbitrary over-allocation will need to be controlled. We begin by ruling out under-allocations.

Recall, if the algorithm stops on line 7, then it will have applied ADJUST-COORDINATES to the allocation. In what follows, let $\boldsymbol{z}_+^k$ denote the value of $\boldsymbol{x}_+^k$ *before* the application of ADJUST-COORDINATES, *i.e.* the value of $\boldsymbol{x}_+^k$ on line 4, and let $\boldsymbol{x}_+^k$ be the returned allocation. Formally $(\boldsymbol{x}_+^k, \overrightarrow{D}(\boldsymbol{x}_+^k)) = \text{ADJUST-COORDINATES}(\boldsymbol{z}_+^k, \overrightarrow{D}(\boldsymbol{z}_+^k), \boldsymbol{d}^k)$.

Claim 27. *If we stop on line 7, then $\boldsymbol{d}^k = \boldsymbol{d}_+^k$.*

Proof. Note that $(\boldsymbol{x}_+^k)_i = (\boldsymbol{z}_+^k)_i \cdot \frac{(\boldsymbol{d}^k)_i}{D_i((\boldsymbol{z}_+^k)_i)}$. As the disutility functions are 1-homogeneous, we have for each $i \in [n]$, $(\boldsymbol{d}_+^k)_i = D_i((\boldsymbol{x}_+^k)_i) = D_i((\boldsymbol{z}_+^k)_i) \cdot \frac{(\boldsymbol{d}^k)_i}{D_i((\boldsymbol{z}_+^k)_i)} = (\boldsymbol{d}^k)_i$. $\qquad\square$

Lemma 28. *Let $(\boldsymbol{a}^{k-1}, \boldsymbol{x}_+^k, \boldsymbol{d}_+^k)$ be the point returned by Algorithm 2 in Step 7. Then we have that $\sum_{i \in [n]} (\boldsymbol{x}_+^k)_{ij} \geq 1 - nL\varepsilon_2$ for all items $j \in [m]$.*

Proof. We begin by showing that for all $i \in [n]$ and $j \in [m]$, $(\boldsymbol{x}_+^k)_{ij} \geq (\boldsymbol{z}_+^k)_{ij} - L\varepsilon_2$. Note that by Observation 23, we have $D_i((\boldsymbol{z}_+^k)_i) \geq (\boldsymbol{d}_*^k)_i \geq (\boldsymbol{d}^k)_i = D_i((\boldsymbol{x}_+^k)_i)$ for all $i \in [n]$. Therefore, $\frac{(\boldsymbol{d}^k)_i}{D_i((\boldsymbol{z}_+^k)_i)} \leq 1$ for all $i \in [n]$, implying that no agent increases their consumption of any chore, *i.e.*, $(\boldsymbol{x}_+^k)_{ij} \leq (\boldsymbol{z}_+^k)_{ij}$ for all $i \in [n]$ and $j \in [m]$. Now, assume that there exist an $i \in [n]$ and $j \in [m]$ such that $(\boldsymbol{x}_+^k)_{ij} < (\boldsymbol{z}_+^k)_{ij} - L\varepsilon_2$. Since the disutility functions have lower-bounded partial derivatives (Assumption 16), and agent i does not increase consumption of any other chore from $(\boldsymbol{z}_+^k)_i$ to $(\boldsymbol{x}_+^k)_i$, we have $D_i((\boldsymbol{x}_+^k)_i) < D_i((\boldsymbol{z}_+^k)_i) - L\varepsilon_2/L$, or equivalently $(\boldsymbol{d}^k)_i < D_i((\boldsymbol{z}_+^k)_i) - \varepsilon_2$, contradicting the fact that $\|\boldsymbol{d}^k - \overrightarrow{D}(\boldsymbol{z}_+^k)\|_2 \leq \varepsilon_2$. Therefore, $(\boldsymbol{x}_+^k)_{ij} \geq (\boldsymbol{z}_+^k)_{ij} - L\varepsilon_2$.

Now, by Observation 23, we have that $\boldsymbol{z}_+^k \in \mathcal{F}'$, implying that $\sum_{i \in [n]} (\boldsymbol{z}_+^k)_{ij} \geq 1$ for all $j \in [m]$. Therefore, if we have $(\boldsymbol{x}_+^k)_{ij} \geq (\boldsymbol{z}_+^k)_{ij} - L\varepsilon_2$, then we have $\sum_{i \in [n]} (\boldsymbol{x}_+^k)_{ij} \geq 1 - nL\varepsilon_2$ for all $j \in [m]$. $\quad\square$

No chores are significantly over-allocated. We now show $\sum_{i \in [n]} (\boldsymbol{x}_+^k)_{ij} \leq 1 + \beta$ for all $j \in [m]$, where $\beta = 2mn^2 L^3(\alpha + \varepsilon_2)$ and $\alpha = 48n^7 L^5 \varepsilon_1/\varepsilon_2^3$. The reason behind the exact choice of the upper bound will become explicit by the end of Claims 32 and 33. We start by making some observations on $\boldsymbol{d}^k$ and $\boldsymbol{a}^{k-1}$. Recall, $\boldsymbol{d}^k$ lies on the hyperplane $\langle \boldsymbol{a}^{k-1}, \boldsymbol{y} \rangle = n$, where

$$\boldsymbol{a}^{k-1} = \frac{n}{\langle \boldsymbol{d}_+^{k-1} - \boldsymbol{d}^{k-1}, \boldsymbol{d}_+^{k-1} \rangle} (\boldsymbol{d}_+^{k-1} - \boldsymbol{d}^{k-1}) \ .$$

We start by showing that there is at least one coordinate where $\boldsymbol{a}^{k-1}$ is not small, w.r.t ε_2.

Claim 29. *There exists an $i_0 \in [n]$ such that $(\boldsymbol{a}^{k-1})_{i_0} \geq \varepsilon_2/(4n^2 L^2)$.*

Proof. In the $(k-1)$-st iteration, Algorithm 2 did not return at step 7, and so $\|\boldsymbol{d}_+^{k-1} - \boldsymbol{d}^{k-1}\|_2 > \varepsilon_2$. There must be some $i \in [n]$ such that $|(\boldsymbol{d}_+^{k-1})_i - (\boldsymbol{d}^{k-1})_i| > \varepsilon_2/n$. By Observation 23, $(\boldsymbol{d}_+^{k-1})_i \geq (\boldsymbol{d}_*^{k-1})_i \geq (\boldsymbol{d}^{k-1})_i$, and thus $(\boldsymbol{d}_+^{k-1})_i - (\boldsymbol{d}^{k-1})_i \geq \varepsilon_2/n$. Now,

$$\begin{aligned} \langle \boldsymbol{d}_+^{k-1} - \boldsymbol{d}^{k-1}, \boldsymbol{d}_+^{k-1} \rangle &\leq \|\boldsymbol{d}_+^{k-1}\|_2^2 \\ &\leq (\|\boldsymbol{d}_*^{k-1}\|_2 + 3nmL^2\varepsilon_1)^2 \qquad \text{(by Observation 24)} \end{aligned}$$

Again, since the algorithm did not return at step 7 in the $(k-1)$-st iteration, we have $\|\boldsymbol{d}_*^{k-1}\|_2 \leq nL$ by Observation 21. This implies that $\langle \boldsymbol{d}_+^{k-1} - \boldsymbol{d}^{k-1}, \boldsymbol{d}_+^{k-1} \rangle \leq (2nL)^2 = 4n^2 L^2$.

Now note that $(\boldsymbol{a}^{k-1})_i = \frac{n}{\langle \boldsymbol{d}_+^{k-1} - \boldsymbol{d}^{k-1}, \boldsymbol{d}_+^{k-1} \rangle}((\boldsymbol{d}_+^{k-1})_i - (\boldsymbol{d}^k)_i) \geq \frac{n}{4n^2 L^2} \cdot \frac{\varepsilon_2}{n} = \varepsilon_2/(4n^2 L^2)$. $\quad\square$

We now define a new allocation $\boldsymbol{y}$ from $\boldsymbol{x}_+^k$ such that $\boldsymbol{y} \in \mathcal{F}'$ and consequently $\overrightarrow{D}(\boldsymbol{y}) \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$. This would imply that $\langle \boldsymbol{a}^{k-1}, \overrightarrow{D}(\boldsymbol{y}) \rangle \geq \langle \boldsymbol{a}^{k-1}, \boldsymbol{d}_+^k \rangle - \delta$. By Claim 27, this equals $\langle \boldsymbol{a}^{k-1}, \boldsymbol{d}^k \rangle - \delta = n - \delta$.

We will show in the following that if any chore is significantly over-allocated in $\boldsymbol{x}_+^k$, then $\langle \boldsymbol{a}^{k-1}, \overrightarrow{D}(\boldsymbol{y}) \rangle < n - \delta$ which is a contradiction. Let us therefore assume that there is a $j' \in [m]$ which is over-allocated, *i.e*, $\sum_{\ell \in [n]} (\boldsymbol{x}_+^k)_{\ell j'} \geq 1 + \beta$. This implies that there is some $i' \in [n]$ such that $(\boldsymbol{x}_+^k)_{i'j'} > 1/n + \beta/n$. Furthermore, for all j, we denote by r_j, the excess amount of chore j left undone in $\boldsymbol{x}_+^k$, *i.e.*, $r_j = \max\{0, \ 1 - \sum_{\ell \in [n]} (\boldsymbol{x}_+^k)_{\ell j}\}$.

As a corollary of Lemma 28, we can bound $r_{j'}$.

Claim 30. *For all $j \in [m]$, we have $r_j \leq nL\varepsilon_2$.*

To define the allocation $\boldsymbol{y}$, we distinguish two cases. Recall that the agent i_0 has $(\boldsymbol{a}^{k-1})_{i_0} \geq \varepsilon_2/(4n^2 L^2)$, by Claim 29. Furthermore, recall that we have chosen $\alpha := 48n^7 L^5 \varepsilon_1/\varepsilon_2^3$ for our desired λ bound.

Case 1. For some item j, $(\boldsymbol{x}_+^k)_{i_0 j} \geq \alpha$. In this case, agent i_0 consumes a non-negligent amount of some chore j w.r.t. the ε's. Define $\boldsymbol{y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)$ as follows:

$$\boldsymbol{y}_\ell := \begin{cases} (\boldsymbol{x}_+^k)_{i_0} - \alpha \cdot \boldsymbol{e}_j & \text{if } \ell = i_0, \\ (\boldsymbol{x}_+^k)_{i'} + \alpha \cdot \boldsymbol{e}_j + \sum_{q \in [m] \setminus \{j, j'\}} r_q \cdot \boldsymbol{e}_q - (\beta/n) \cdot \boldsymbol{e}_{j'} & \text{if } \ell = i', \\ (\boldsymbol{x}_+^k)_\ell & \text{otherwise.} \end{cases}$$

This has the following effects: (i) we decrease i_0's consumption of j by α units and increase i''s consumption of j by α units (so the total consumption of j remains unchanged), then (ii) increase the consumption of every under-consumed chore for agent i' until their total consumption becomes 1 and finally (iii) decrease the i''s consumption of j' (the overallocated chore) by β/n units.

Case 2. If instead, $(\boldsymbol{x}_+^k)_{i_0 j} < \alpha$ for all items j, define $\boldsymbol{y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)$ as follows:

$$\boldsymbol{y}_\ell := \begin{cases} \boldsymbol{0} & \text{if } \ell = i_0, \\ (\boldsymbol{x}_+^k)_{i'} + (\boldsymbol{x}_+^k)_{i_0} + \sum_{q \in [m] \setminus \{j'\}} r_q \cdot \boldsymbol{e}_q - (\beta/n) \cdot \boldsymbol{e}_{j'} & \text{if } \ell = i', \\ (\boldsymbol{x}_+^k)_\ell & \text{otherwise.} \end{cases}$$

(i) We decrease i_0's consumption of each chore to zero and increase i''s consumption of item q by $(\boldsymbol{x}_+^k)_{iq}$ units (so the total consumption of each chore q remains unchanged), then (ii) we increase the consumption of every under-consumed chore for agent i' until their total consumption becomes 1 and finally (iii) we decrease i''s consumption of j' (the overallocated chore) by β/n units.

We first show that $\boldsymbol{y} \in \mathcal{F}'$.

Claim 31. *We have $\boldsymbol{y} \in \mathcal{F}'$.*

Proof. In both Case 1 and 2, we have increased agent i''s consumption of each the under-allocated chores ℓ ($r_\ell > 0$) by r_ℓ, with the exception of j' and j if applicable. Furthermore, the decrease in consumption of any chore for agent i_0 is matched by an increase for i', before subtracting the j' term. Therefore, $\sum_{i \in [n]} (\boldsymbol{y})_{i\ell} \geq 1$ for all $\ell \in [m] \setminus j'$. Finally, the consumption of chore j' is decreased by β/n, but since the total consumption of j' in $\boldsymbol{x}_+^k$ is at least $1 + \beta$, the total consumption of j' in $\boldsymbol{y}$ is at least 1. Thus $\boldsymbol{y} \in \mathcal{F}'$ and $\overrightarrow{D}(\boldsymbol{y}) \in \mathcal{D} + \mathbb{R}_{\geq 0}^n$. $\qquad \square$

We next argue that the disutility values of all agents have not increased.

Claim 32. *For all $\ell \in [n]$, we have $D_\ell((\boldsymbol{y})_\ell) \leq D_\ell((\boldsymbol{x}_+^k)_\ell)$.*

Proof. Note that we have only reduced the consumption of chores for all agents except i'. Therefore, the disutility values for all agents in $[n] \setminus \{i'\}$ decreases. It suffices to show that agent i''s disutility also decreases. We now argue that $D_{i'}((\boldsymbol{y})_{i'}) < D_{i'}((\boldsymbol{x}_+^k)_{i'})$.

Let $\boldsymbol{\Delta}$ be defined to equal $\alpha \boldsymbol{e}_j$ in Case 1, and $(\boldsymbol{x}_+^k)_{i'}$ in Case 2. In both cases, we are subtracting $\boldsymbol{\Delta}$ from $(\boldsymbol{x}_+^k)_{i0}$ and adding it to $(\boldsymbol{x}_+^k)_{i'}$. We recall that 1-homogeneous and convex functions are sub-additive: $D(\boldsymbol{p} + \boldsymbol{q}) = 2D(\frac{1}{2}\boldsymbol{p} + \frac{1}{2}\boldsymbol{q}) \leq 2\frac{1}{2}(D(\boldsymbol{p}) + D(\boldsymbol{q})) = D(\boldsymbol{p}) + D(\boldsymbol{q})$. Therefore,

$$
\begin{aligned}
D_{i'}&(\boldsymbol{y}_{i'}) \\
&= D_{i'}\left((\boldsymbol{x}_+^k)_{i'} + \boldsymbol{\Delta} + \sum_{\ell \in [m] \setminus \{j,j'\}} r_\ell \cdot \boldsymbol{e}_\ell - (\beta/n) \cdot \boldsymbol{e}_{j'}\right) \\
&\leq D_{i'}\left((\boldsymbol{x}_+^k)_{i'} - (\beta/n) \cdot \boldsymbol{e}_{j'}\right) + D_{i'}(\boldsymbol{\Delta}) + \sum_{\ell \in [m] \setminus \{j,j'\}} D_{i'}(r_\ell \cdot \boldsymbol{e}_\ell) \\
&\leq \underbrace{D_{i'}\left((\boldsymbol{x}_+^k)_{i'}\right) - \tfrac{1}{L}(\beta/n)}_{\text{Assumption 16}} + D_{i'}(\boldsymbol{\Delta}) + \sum_{\ell \in [m] \setminus \{j,j'\}} \underbrace{D_{i'}(nL\varepsilon_2 \cdot \boldsymbol{e}_\ell)}_{\text{Claim 30}}
\end{aligned}
$$

Now, in both cases, $\|\boldsymbol{\Delta}\|_\infty \leq \alpha$, and so by Assumption 16, $D_{i'}(\boldsymbol{\Delta}) \leq m \cdot \alpha \cdot L$, and $D_{i'}(\varepsilon_2 \cdot \boldsymbol{e}_\ell) \leq L\varepsilon_2$. Finally, recall $\beta := 2mn^2 L^3(\alpha + \varepsilon_2)$, and so we have

$$
D_{i'}(\boldsymbol{y}_{i'}) \leq D_{i'}\left((\boldsymbol{x}_+^k)_{i'}\right) - 2mnL^2(\alpha + \varepsilon_2) + mL \cdot \alpha + mnL^2(\varepsilon_2) < D_{i'}\left((\boldsymbol{x}_+^k)_{i'}\right) \qquad \square
$$

With these two claims, we can now show that $\langle \boldsymbol{a}^{k-1}, \overrightarrow{D}(\boldsymbol{y}) \rangle < n - \delta$ despite $\boldsymbol{y} \in \mathcal{F}'$, a contradiction.

Claim 33. *We have* $\langle \boldsymbol{a}^{k-1}, \overrightarrow{D}(\boldsymbol{y}) \rangle < n - \delta$.

Proof. The disutility of all agents decreases from $\boldsymbol{x}_+^k$ to $\boldsymbol{y}$ and since every entry in $\boldsymbol{a}^{k-1}$ is strictly positive (Lemma 8 and Observation 23), we have

$$
\langle \boldsymbol{a}^{k-1}, \overrightarrow{D}(\boldsymbol{y}) \rangle - \langle \boldsymbol{a}^{k-1}, \overrightarrow{D}(\boldsymbol{x}_+^k) \rangle \ \leq \ (\boldsymbol{a}^{k-1})_{i_0} \cdot \left(D_{i_0}(\boldsymbol{y}_{i_0}) - D_{i_0}((\boldsymbol{x}_+^k)_{i_0})\right) \ . \tag{7}
$$

We wish to show that this difference is smaller than $-\delta$. We will distinguish the two cases.

Case 1. In this case, we have

$$
D_{i_0}(\boldsymbol{y}_{i_0}) - D_{i_0}((\boldsymbol{x}_+^k)_{i_0}) = D_{i_0}((\boldsymbol{x}_+^k)_{i_0} - \alpha \boldsymbol{e}_j) - D_{i_0}((\boldsymbol{x}_+^k)_{i_0}) \leq -\alpha/L \ ,
$$

by Assumption 16. By Claim 29, $(\boldsymbol{a}^{k-1})_{i_0} \geq \varepsilon_2/(4n^2 L^2)$. Therefore, the right hand side of (7) is at most $-\alpha \varepsilon_2/4n^2 L^3$.

Case 2. In this case, we have

$$
D_{i_0}(\boldsymbol{y}_{i_0}) - D_{i_0}((\boldsymbol{x}_+^k)_{i_0}) = 0 - D_{i_0}((\boldsymbol{x}_+^k)_{i_0}) = -(\boldsymbol{d}_+^k)_{i_0} = -(\boldsymbol{d}^k)_{i_0} \ ,
$$

by Claim 27. Note that $(\boldsymbol{d}^k)_{i_0} = 1/(\boldsymbol{a}^{k-1})_{i_0}$, and so the right hand side of (7) is equal to -1, which is at most $-\alpha \varepsilon_2/4n^2 L^3$.

Substituting $\alpha = 48n^7 m L^6 \varepsilon_1/\varepsilon_2^3$, we have $\langle \boldsymbol{a}^{k-1}, \overrightarrow{D}(\boldsymbol{y}) \rangle - \langle \boldsymbol{a}^{k-1}, \overrightarrow{D}(\boldsymbol{x}_+^k) \rangle \leq -12n^5 m L^3(\varepsilon_1/\varepsilon_2^2) < -9n^5 m L^3(\varepsilon_1/\varepsilon_2^2) = -\delta$. $\qquad \square$

Therefore, we have $\boldsymbol{y} \in \mathcal{F}'$, but $\langle \boldsymbol{a}^{k-1}, \overrightarrow{D}(\boldsymbol{y}) \rangle < n - \delta$, which is a contradiction. This implies that no chores are significantly over-allocated. Thus, we have proven the following:

30

Lemma 34. *For all $j \in [m]$, we have $\sum_{i \in [n]} (\boldsymbol{x}_+^k)_{ij} \leq 1 + \beta$, where $\beta = 2mn^2L^3(\alpha + \varepsilon_2)$ and $\alpha = 48n^7mL^6\varepsilon_1/\varepsilon_2^3$.*

We now have everything we need to prove the following:

Theorem 35. *Algorithm 2 returns a $(\lambda, \gamma, \delta)$-KKT point with $\lambda = 1 + 3mn^2L^3(\alpha + \varepsilon_2)$, where $\alpha = 48n^7mL^6\varepsilon_1/\varepsilon_2^3$, $\gamma = 1 + \varepsilon_3$ and $\delta = 9n^5mL^3\frac{\varepsilon_1}{\varepsilon_2^2}$.*

Proof. If the algorithm returns at step 12, then $(\boldsymbol{a}^k, \boldsymbol{x}_+^k, \boldsymbol{d}_+^k)$ is a $(1 + n^2L\varepsilon_1, 1 + \varepsilon_3, 6n^2L^2\varepsilon_1/\varepsilon_2^2)$-KKT point, by Lemma 25. If the algorithm returns at step 7, then $(\boldsymbol{a}^{k-1}, \boldsymbol{x}_+^k, \boldsymbol{d}_+^k)$ is a $(1 + 3mn^2L^3(\alpha + \varepsilon_2), 1 + \varepsilon_3, 9n^5mL^3\varepsilon_1/\varepsilon_2^2)$-KKT point where $\alpha = 48n^7mL^6\varepsilon_1/\varepsilon_2^3$, by Lemmas 26, 28, and 34. As $n^3m^3L^3\varepsilon_1 \ll \varepsilon_2$, we have $1 + 3mn^2L^3(\alpha + \varepsilon_2) > 1 + n^2L\varepsilon_1$ and thus the point returned by Algorithm 2 is $(1 + 3mn^2L^3(\alpha + \varepsilon_2), 1 + \varepsilon_3, 9n^5mL^3\varepsilon_1/\varepsilon_2^2)$-KKT. $\qquad\square$

5.3 Polynomially Bounding the Number and Running Time of Iterations

In this section, we show that in polynomially many iterations, Algorithm 2 finds the $(1 + 3mn^2L^3(\alpha + \varepsilon_2), 1 + \varepsilon_3, 9n^5mL^3\varepsilon_1/\varepsilon_2^2)$-KKT point of Lemma 35. The proof follows exactly the proof in the setting with linear disutilities (the proof of Lemma 12). One can argue that (a) the log-NSW $\mathcal{L}$ is always increasing throughout Algorithm 1, and (b) it increases additively by $\varepsilon_3/16n^2$ every time $\mathrm{RelDist}(\boldsymbol{d}^{k+1}, \boldsymbol{d}_+^k) \geq \varepsilon_3$. Therefore, the total number of iterations of the algorithm is bounded by $\mathrm{poly}(n, m, 1/\varepsilon_3)$.

Lemma 36. *After $\mathrm{poly}(n, m, 1/\varepsilon_3)$ iterations, Algorithm 2 returns a $(1 + 3mn^2L^3(\alpha + \varepsilon_2), 1 + \varepsilon_3, 9n^5mL^3\varepsilon_1/\varepsilon_2^2)$-KKT point, where $\alpha = 48n^7mL^6\varepsilon_1/\varepsilon_2^3$.*

It remains therefore to show that each iteration can be efficiently computed. The main difficulty lies in finding the approximate nearest point $\boldsymbol{d}_+^k$, or rather $\boldsymbol{x}_+^k$, *i.e.* to explain the algorithm for NEAREST-POINT, from line 3 of Algorithm 2. The remaining steps of the algorithm and of the ADJUST-COORDINATES procedure are polynomial.

Recall that given a scalar ε_1 and a point $\boldsymbol{d}^k$, the subroutine NEAREST-POINT$(\cdot, \cdot)$ returns a point $\boldsymbol{x}_+^k$ such that $||\boldsymbol{x}_+^k - \boldsymbol{x}_*^k||_2 \leq \varepsilon_1$ and $||\overrightarrow{D}(\boldsymbol{x}_+^k) - \boldsymbol{d}_*^k||_2 \leq \varepsilon_1$, where $\boldsymbol{d}_*^k$ is the nearest point in $\mathcal{D} + \mathbb{R}_{\geq 0}^n$ to $\boldsymbol{d}^k$ and $\boldsymbol{x}_*^k \in \mathcal{F}'$ is a pre-image of $\boldsymbol{d}_*^k$. Also note that we have an additional requirement on the nearest point that each coordinate should be an integral multiple of $\varepsilon_1/(\mathrm{poly}(n, m))$. However, this can be implemented by rounding the nearest approximate point that we find in polynomial time without increasing $||\boldsymbol{d}_+^k - \boldsymbol{d}^k||_2$ and $||\boldsymbol{x}_+^k - \boldsymbol{x}^k||_2$ significantly (there will only be additive errors of $\varepsilon_1/(\mathrm{poly}(n, m))$). Therefore, the main bottleneck is in finding the approximate nearest point. We focus mainly on this now.

We will implement this with the following convex program, which returns simultaneously $\boldsymbol{x}_*^k$ and $\boldsymbol{d}_*^k$ as its solution.

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i \in [n]} (\beta_i)^2 \\
\text{subject to} \quad & \sum_{i \in [n]} (\boldsymbol{z})_{ij} \geq 1, && \forall j \in [m] \\
& (\boldsymbol{z})_{ij} \geq 0, && \forall i \in [n], \forall j \in [m] \\
& D_i(\boldsymbol{z}_i) - (\boldsymbol{d}^k)_i - \beta_i \leq 0, && \forall i \in [n],
\end{aligned}
\tag{8}
$$

The objective function is clearly convex, and we show in Claim 44 in Appendix A that the constraints are convex as well.

We now prove that Program 8 is correct, *i.e.* its solution gives the nearest point in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ to d^k.

Lemma 37. *Let (z^*, β^*) be an exact solution to the convex program 8. Then $z^* \in \mathcal{F}'$ and $\overrightarrow{D}(z^*)$ is a nearest point in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ to d^k.*

Proof. Since (z^*, β^*) satisfies the feasibility constraints, we have $(z^*)_{ij} \geq 0$ for all $i \in [n]$, $j \in [m]$ and $\sum_{i \in [n]} (z^*)_{ij} \geq 1$ for all $j \in [m]$, implying that $z^* \in \mathcal{F}'$. Now, it remains to show that $\overrightarrow{D}(z^*) = d^k_*$.

Let OPT be the minimum value of the objective function in (8) achieved by any feasible solution. We wish to show that $OPT = ||d^k_* - d^k||^2_2$. First note that $OPT \leq ||d^k_* - d^k||^2_2$, since it is feasible to set $z \leftarrow x^k_*$ and

$$\beta_i \leftarrow D_i(z_i) - (d^k)_i = D_i((x^k_*)_i) - (d^k)_i = (d^k_*)_i - (d^k)_i$$
$$\implies \sum_{i \in [n]} (\beta_i)^2 = ||d^k_* - d^k||^2_2 \ .$$

Now, we show that $OPT \geq ||d^k_* - d^k||^2_2$. Note that if $d^k \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$, then $d^k_* = d^k$ and thus OPT will be trivially larger than $||d^k_* - d^k||^2_2 = 0$. So we only focus on the case when $d^k \notin \mathcal{D} + \mathbb{R}^n_{\geq 0}$. Suppose for a contradiction that $OPT < ||d^k_* - d^k||^2_2$ and there exists a feasible (z, β) such that $\sum_{i \in [n]} (\beta_i)^2 < ||d^k_* - d^k||^2_2$. This implies that $d^k + \beta \notin \mathcal{D} + \mathbb{R}^n_{\geq 0}$. Therefore, there exists a hyperplane $\langle a, y \rangle = n$ that separates $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ and the point $d^k + \beta$. Since z lies in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$, we have then by definition,

$$\langle a, (d^k + \beta) \rangle < n \leq \langle a, \overrightarrow{D}(z) \rangle$$
$$\implies 0 < \langle a, \overrightarrow{D}(z) - d^k - \beta \rangle \ .$$

By the last constraint in program (8), each coordinate of $\overrightarrow{D}(z) - d^k - \beta$ is non-positive, and so for this inequality to hold, a must have a strictly negative entry. However, since $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ is unbounded in the positive directions, no hyperplane of the form $\langle a, \cdot \rangle \geq n$ can be supporting unless $a \geq 0$, a contradiction.

Therefore, at the optimum, $||\beta^*||^2_2 = ||d^k_* - d^k||^2_2$. Since (z^*, β^*) is a feasible point we have $D_i((z^*)_i) - (d^k)_i \leq (\beta^*)_i$ for all $i \in [n]$, implying that $||\overrightarrow{D}(z^*) - d^k||^2_2 \leq ||d^k_* - d^k||^2_2$. Since d^k_* is a closest point in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ to d^k, and $z^* \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$, we have $||\overrightarrow{D}(z^*) - d^k||^2_2 = ||d^k_* - d^k||^2_2$, implying that $\overrightarrow{D}(z^*)$ is also a closest point in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ to d^k. $\square$

Let (z^*, β^*) be the optimum solution to the program 8. We just need to find an approximate solution to the convex program (z^+, β^+) such that $||(z^+, \beta^+) - (z^*, \beta^*)||_2 \leq \varepsilon_1/L$, as this would give us the desired bounds in the allocation as well as the resulting disutility vector (follows from the Lipschitz condition of the disutility functions mentioned in Assumption 16). This can be determined in polynomial time by interior point algorithms. This brings us to the main lemma of this section.

Lemma 38. *Given a scalar ε_1 and a point d^k, the subroutine NEAREST-POINT$(\cdot, \cdot)$ returns a point x^k_+ such that $||x^k_+ - x^k_*||_2 \leq \varepsilon_1$ and $||\overrightarrow{D}(x^k_+) - d^k_*||_2 \leq \varepsilon_1$, where d^k_* is the nearest point in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ to d^k and $x^k_* \in \mathcal{F}'$ is a pre-image of d^k_*, in time $\mathrm{poly}(n, m, \log(L/\varepsilon_1))$. Additionally, each coordinate of d^k_+ and x^k_+ is an integral multiple of $\varepsilon_1/(\mathrm{poly}(n, m))$.*

Proof. Many interior-point methods exist to solve this program, including the ellipsoid method, which can efficiently find a near-optimal point in $\mathrm{poly}(n, m, \log(1/\varepsilon_1))$ time, given efficient separation oracles [Bub14]. The constraint region needs to be bounded for these methods to work, but we can use the correctness of Algorithm 2 (Theorem 35) to upper-bound the allocations, and Assumption 16 to bound the feasible β values.

We can easily determine which constraint is violated in the program (8), but we need to return a hyperplane if the violated constraint is one of the $D_i(z_i) - (d^k)_i - \beta_i \geq 0$ constraints. We have assumed access to the partial derivatives of the D_i's, and we therefore have access to the gradient of these constraints. A "good enough" candidate point at which to take the supporting hyperplane can be found by approximating the nearest feasible point using unconstrained optimization with barrier functions [Bub14], which gives separation oracles to implement a step of the ellipsoid method. □

To finish arguing that each iteration of Algorithm 2 can be implemented in polynomial time, we need to show that bit size of d^{k+1} computed at the end of each iteration does not grow too much. In particular, there should be at most an additive polynomial increase in the bit-size of d^{k+1} from d^k. To this end, note that $d^{k+1} = \frac{1}{a^k} = \frac{\langle d_+^k - d^k, d_+^k \rangle}{n} \cdot (d_+^k - d^k)$. Note that such an operation can only cause a polynomial additive increase the bit size of d^{k+1} from d^k, as the bit-size of d_+^k is polynomially bounded (recall that we enforce every coordinate of d_+^k to be an integral multiple of $\varepsilon_1/(\mathrm{poly}(n, m)))$. Since the total number of iterations is $\mathrm{poly}(n, m, 1/\varepsilon_3, \log(L))$, we have that the bit-size of d^{k+1} at the end of each iteration is $\mathrm{poly}(n, m, 1/\varepsilon_3, \log(L/\varepsilon_1))$. Therefore, the running time of each iteration is polynomial. We can now bound the running time of the algorithm.

Theorem 39. *In* $\mathrm{poly}(n, m, 1/\varepsilon_3, \log(L/\varepsilon_1))$ *time, we can determine a* $(1 + 3mn^2 L^3(\alpha + \varepsilon_2), 1 + \varepsilon_3, 9n^5 m L^3 \varepsilon_1/\varepsilon_2^2)$*-KKT point where* $\alpha = 48n^7 m L^6 \varepsilon_1/\varepsilon_2^3$.

Proof. Lemma 36 states that in $\mathrm{poly}(n, m, 1/\varepsilon_3)$ many iterations, Algorithm 2 determines a $(1 + 3mn^2 L^3(\alpha + \varepsilon_2), 1 + \varepsilon_3, 9n^5 m L^3 \varepsilon_1/\varepsilon_2^2)$-KKT point where $\alpha = 48n^7 m L^6 \varepsilon_1/\varepsilon_2^3$. Lemma 38, states that each iteration of Algorithm 2 can be implemented in $\mathrm{poly}(n, m, \log(L/\varepsilon_1))$ time. Thus, Algorithm 2 finds the approximate KKT point in $\mathrm{poly}(n, m, 1/\varepsilon_3, \log(L/\varepsilon_1))$ time. □

By setting appropriate values for ε_1, ε_2 and ε_3, we have the following corollary.

Corollary 40. *In* $\mathrm{poly}(n, m, 1/\varepsilon, \log(L))$ *time, we can determine a* $(1 + \frac{\varepsilon}{2^{\mathrm{poly}(n,m)}}, 1 + \varepsilon, \frac{\varepsilon}{2^{\mathrm{poly}(n,m)}})$*-KKT point.*

Proof. We set $\varepsilon_3 = \varepsilon$. Thereafter, we set $\varepsilon_1 = \varepsilon/(2^{f(n) \cdot f(m)} \cdot f(L))$ where $f(\cdot)$ is a polynomial of sufficiently large constant degree. The lower-bound on the degree will become clear by the end of this proof. Then, set $\varepsilon_2 = n^4 m^3 L^3 (\varepsilon_1)^{\frac{1}{6}}$. Note that we have $\varepsilon_2 \gg n^3 m^3 L^3 \varepsilon_1$ and $\varepsilon_3 \geq \frac{\varepsilon_2}{2^{\mathrm{poly}(n,m)}}$ as required by all the proofs so far.

We can show the following bounds in terms of ε:

- $1 + 3mn^2 L^3 \cdot (\alpha + \varepsilon_2) \leq 1 + (\varepsilon^{1/6})/(2^{\mathrm{poly}(n,m)})$: We first note that $\alpha \leq \varepsilon_1^{1/2}$. We have

$$\alpha = 48n^7 m L^6 \cdot \frac{\varepsilon_1}{\varepsilon_2^3} = 48n^7 m L^6 \cdot \frac{\varepsilon_1}{n^{12} \cdot m^9 L^9 \varepsilon_1^{1/2}}$$

$$= \frac{48}{n^5 m^8 L^3} \cdot \varepsilon_1^{1/2} \leq \varepsilon_1^{1/2} \ .$$

Substituting the upper-bound on α in $1 + 3mn^2L^3 \cdot (\alpha + \varepsilon_2)$, we have

$$1 + 3mn^2L^3 \cdot (\alpha + \varepsilon_2) \leq 1 + 3mn^2L^3((\varepsilon_1)^{1/2} + n^4m^3L^3(\varepsilon_1)^{1/6})$$
$$\leq 1 + 3mn^2L^3 \cdot (2n^4m^3L^3(\varepsilon_1)^{1/6})$$
$$\leq 1 + 6n^6m^4L^6 \cdot (\varepsilon_1)^{1/6}$$
$$\leq 1 + \frac{\varepsilon^{1/6}}{2^{\mathrm{poly}(n,m)}} \qquad \text{(for a sufficiently large } f(\cdot)\text{)}.$$

- $1 + \varepsilon_3 = 1 + \varepsilon$.

- $9n^5mL^3 \cdot \varepsilon_1/\varepsilon_2^2 \leq (\varepsilon^{2/3})/(2^{\mathrm{poly}(n,m)})$: We have

$$9n^5mL^3 \cdot \frac{\varepsilon_1}{\varepsilon_2^2} = 9n^5mL^3 \cdot \frac{\varepsilon_1}{n^8m^6L^6\varepsilon_1^{\frac{1}{3}}} = \frac{9}{n^3m^5L^3} \cdot \varepsilon_1^{2/3} \leq \frac{\varepsilon^{2/3}}{2^{\mathrm{poly}(n,m)}}.$$

Substituting these bounds in the statement of Theorem 39, we can conclude that Algorithm 2 returns a $(1 + \frac{\varepsilon^{1/6}}{2^{\mathrm{poly}(n,m)}}, 1 + \varepsilon, \frac{\varepsilon^{2/3}}{2^{\mathrm{poly}(n,m)}})$-KKT point for our choice of ε_1, ε_2 and ε_3. Note that the running time now is

$$\mathrm{poly}(n, m, 1/\varepsilon_3, \log(L/\varepsilon_1)) = \mathrm{poly}\left(n, m, 1/\varepsilon, \log\left(L \cdot f(L) \cdot 2^{(f(n) \cdot f(m))} \cdot 1/\varepsilon\right)\right)$$
$$= \mathrm{poly}(n, m, 1/\varepsilon, f(n) \cdot f(m), \log(L \cdot f(L)))$$
$$= \mathrm{poly}(n, m, 1/\varepsilon, \log(L)) \qquad \text{(as } f(\cdot) \text{ is polynomial)}$$

$\square$

5.4 Finding Approximately Competitive Equilibria in Polynomial Time

We can now combine the results of the previous sections to show that ε-CEEI can be found in polynomial time.

Theorem 41. *Given black-box access to 1-homogeneous and convex disutilities $D_1, \ldots, D_n$ all satisfying Assumption 16 with constant L, and also to their partial derivatives, Algorithm 2, finds an $\varepsilon^{1/6}$-CEEI in time polynomial in n, m, $1/\varepsilon$, and $\log(L)$.*

Proof. From Theorem 17 we know that a $(\lambda, \gamma, \delta)$-KKT point gives a $max(3(\gamma-1)+5\delta, \lambda-1)$- CEEI. Substituting the values of λ, γ and δ as in Corollary 40, we have that in $\mathrm{poly}(n, m, 1/\varepsilon, \log(L))$-time, Algorithm 2, returns a $max(3\varepsilon + \frac{5\varepsilon^{2/3}}{2^{\mathrm{poly}(n,m)}}, \frac{\varepsilon^{1/6}}{2^{\mathrm{poly}(n,m)}})$-CEEI, which is a $\varepsilon^{1/6}$-CEEI. $\square$

We remark that although our overall approximation of CEEI is inverse-polynomial, we satisfy the condition (2) and (3) in Definition 1 with an inverse-exponential ε.

Remark 42. We briefly remark how to make the algorithm work with a weaker assumption than the one in Assumption 16, i.e., for all $i \in [n]$, we have $|D_i(\boldsymbol{x} + \delta \cdot \boldsymbol{e}_j) - D_i(\boldsymbol{x})| \geq \frac{1}{L} \cdot min(\delta, \delta^k)$ for $k \in \mathrm{poly}(n, m)$. In this case, the re-adjustments done in steps 4 such that $\boldsymbol{d}_+^k$ Pareto-dominates $\boldsymbol{d}^k$ should increase $\boldsymbol{x}_+^k$ by $2L\varepsilon_1^{1/k}$ instead of $2L\varepsilon_1$. Similarly, all the bounds for λ and δ in Theorem 39 that are functions of ε_1 and ε_2 will be functions of $\varepsilon_1^{1/k}$ and $\varepsilon_2^{1/k}$. Since, we have the flexibility to choose inverse exponential values for both ε_1 and ε_2 to get an $1/(\mathrm{poly}(n, m))$-CEEI, we can tolerate $k \in \mathrm{poly}(n, m)$.

34

6 Extending to Mixed Linear Disutilities and Un-equal Incomes.

We briefly argue in this section that our algorithms extend to the setting of *mixed manna* with linear disutilities, where some agents likes some items and dislike others, and to un-equal income where agents have different importance/weights modeled as different income requirements. In the latter case the resulting allocation is known as *competitive equilibrium (CE)* allocation.

6.1 Mixed Manna with Linear Disutilities

We say the instance is of *mixed manna* if each agent may value some items positively (goods), and other items negatively (chores). Formally, linear *utility* functions in this setting are represented by $U_i(\boldsymbol{x}_i) = \sum_j U_{ij} x_{ij}$ for agent i, where U_{ij} is positive if j is a good for agent i, and U_{ij} is negative if j is a chore for i, interpreted $U_{ij} = -D_{ij}$.

In this case, [BMSY17] characterized the instanced into three categories, intuitively goods-heavy ("positive"), chores-heavy ("negative"), and null. To formalize this, let us first divide the agents into two categories:

$$N_+ = \{i \mid \exists j, U_{ij} > 0\} , \qquad N_- = \{i \mid \forall j, U_{ij} \leq 0\} .$$

We have the following three cases:

(positive) There exists a feasible allocation $\boldsymbol{x} \in \mathcal{F}$ such that all agents in N_+ get strictly positive utilities, and those in N_- get zero utility. Formally, $U_i(\boldsymbol{x}_i) > 0$, $\forall i \in N_+$ and $U_i(\boldsymbol{x}_i) = 0$, $\forall i \in N_-$.

(null) The positive case is not possible, but there exists a feasible allocation $\boldsymbol{x} \in \mathcal{F}$ such that all agents get zero utility, *i.e.*, $U_i(\boldsymbol{x}_i) = 0$, $\forall i$.

(negative) Every feasible allocation gives strictly negative utility to some agent.

Given an instance it is easy to check which category it belongs to using linear programming. Bogomolnaia et al. [BMSY17] characterize the set of CEEI as follows:

Theorem 43. *[BMSY17] Given an instance $I = (U_1, \ldots, U_n)$ with mixed manna,*

1. *If I is a positive instance, then every agent* spends *at most one unit of money, and an allocation is a CEEI if and only if it maximizes the product $\prod_{i \in N_+} U_i(\boldsymbol{x}_i)$, and agents in N_- get zero utility.*

2. *If I is a null instance, then an allocation is a CEEI if and only if all agents get zero utility.*

3. *If I is a negative instance, then an allocation is a CEEI if and only if it is a local minimum (KKT point) of the $\prod_i |U_i(\boldsymbol{x}_i)|$ with $U_i(\boldsymbol{x}_i) < 0$ for all i.*

Using the above characterization, we note that the positive case can be solved using the Eisenberg-Gale [EG59] convex program, by maximizing $\sum_{i \in N_+} \log(U_i(\boldsymbol{x}_i))$, subject to feasibility and assigning zero to N_-. [9] In the null case, any allocation that gives zero utility to all the agent

---

[9]Since CEEI must be Pareto-optimal, if for some item j, $U_{ij} > 0$ and $U_{i'j} \leq 0$, then item j will never be allocated to agent i'. Therefore, without loss of generality we may eliminate the $x_{i'j}$ variable. After all such modifications, we can divide items into two sets: goods G that are non-negatively valued by all the agents, and chores (bads) B that are non-positively valued by all the agents. Then the feasibility constraints should be $\sum_i x_{ij} \leq 1$, $\forall j \in G$ and $\sum_i x_{ij} \geq 1$, $\forall j \in B$ to ensure correct sign for the prices that comes from the dual variables.

is a CEEI and can be computed through a linear feasibility program. The negative case is similar to the purely-chores setting that we establish in this paper. The rest of this section explains how to extend our algorithms to handle this case.

To follow the notation used in the previous sections, we will return to using disutility functions $D_i(\boldsymbol{x}_i) = -U_i(\boldsymbol{x}_i)$, by setting $D_{ij} = -U_{ij}$ for all i and j. Now as per Theorem 43, in the negative case we want to find a KKT point of $\prod_i \boldsymbol{d}_i$ subject to $\boldsymbol{d} \in \mathcal{D}$ and $d_i > 0$, $\forall i$. This is exactly what we compute in Section 4, but we must now allow for the feasible region $\mathcal{D}$ to contain disutility profiles with negative entries.

We show here that our theorems do not require that all feasible disutility profiles be positive, but instead need only the returned allocation to have positive disutilities. We handle this without any modification by noting that Lemma 8 was written to not require $\mathcal{D}$ to lie in $\mathbb{R}^n_{\geq 0}$. Thus, so long as we can find an initial (infeasible) disutility profile $\boldsymbol{d}^0$ whose entries are all positive and sufficiently large, every other detail will go through with positive disutility.

When disutilities are linear, we must modify Claim 9 to account for the fact that some D_{ij}'s may be negative. Unfortunately, naïve constructions like that of Claim 9 will not work anymore. However, $\boldsymbol{d}^0 = \boldsymbol{0}$ is a valid starting point, but it simply means that we cannot use the objective function at the starting point to bound the number of iterations.

We show that one can always find an infeasible $\boldsymbol{d}^0$ with polynomial bit complexity as follows: We have that $\mathcal{D}$ is a linear polytope and $\boldsymbol{0} \notin \mathcal{D}$. For a $\delta > 0$, check if $\delta \boldsymbol{1} \in \mathcal{D}$ or not. If not then set $\boldsymbol{d}^0 = \delta \boldsymbol{1}$ and then $-\mathcal{L}(\boldsymbol{d}^0) = n \log(1/\delta)$. Otherwise, the line joining $\boldsymbol{0}$ and $\delta \boldsymbol{1}$ must intersect $\mathcal{D}$, and this intersection has to be a point with polynomial bit complexity, so long as δ is sufficiently large. Therefore, we can do a binary search on the line between $\boldsymbol{0}$ and $\delta \boldsymbol{1}$ to find an infeasible $\boldsymbol{d}^0$ of polynomial bit complexity.

The upper bound on $\mathcal{L}(d)$ from Lemma 13 works as is, and thereby we get that the number of iterations remain polynomial.

6.2   Un-equal Incomes: Competitive Equilibrium (CE)

The model with unequal incomes is formally defined as follows: each agent has a disutility function D_i as before, and an income level $\eta_i > 0$. We will show that our algorithm extends to this setting. And the running time remains polynomial as far as $\max_i \eta_i / \min_i \eta_i$ is polynomially bounded.

A competitive equilibrium is an allocation $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) \in \mathcal{F}$, and a payment vector $\boldsymbol{p}$ such that $\langle \boldsymbol{p}, \boldsymbol{x}_i \rangle = \eta_i$ for all i, and every agent minimizes their disutility subject to $\langle \boldsymbol{p}, \boldsymbol{x}_i \rangle \geq \eta_i$. Accordingly the only change in Definition 1 of ε-CEEI is in Condition (1): we now want that all agents' η-rescaled incomes are approximately same, i.e., $(1 - \varepsilon) \cdot \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle / \eta_i \leq \langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle / \eta_{i'}$ for any i, i'.

To take the weights into account the objective functions changes to minimizing $\prod_i d_i^{\eta_i}$, and accordingly define $\mathcal{L}(\boldsymbol{d}; \boldsymbol{\eta}) := \sum_{i=1}^n \eta_i \log(d_i)$. And then the definition of approximate KKT point will need to be modified: instead of requiring that $\gamma^{-1} \leq a_i d_i \leq \gamma$, we instead ask that $\gamma^{-1} \leq a_i d_i / \eta_i \leq \gamma$.

Now, in the proof of Theorems 4 and 17, we make the following changes. When showing condition (1) of Definition 1, we now want to show that $\gamma^{-2} \cdot \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle / \eta_i \leq \langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle / \eta_{i'}$ for any i, i'. Assume otherwise and say we have $\gamma^{-2} \cdot \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle / \eta_i > \langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle / \eta_{i'}$. As before, replace the allocation as follows: Construct $\hat{\boldsymbol{z}}$ by setting $\hat{\boldsymbol{z}}_{i'} = \frac{1}{2} \boldsymbol{z}_{i'}$, and $\hat{\boldsymbol{z}}_i = \left(1 + \frac{\langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle}{2 \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle} \right) \boldsymbol{z}_i$. The same proofs extend,

but the contradiction is attained as follows:

$$\langle \boldsymbol{a}, \overrightarrow{D}(\hat{\boldsymbol{z}})\rangle - \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z})\rangle = -\frac{1}{2} a_{i'} D_{i'}(\boldsymbol{z}_{i'}) + \frac{\langle \boldsymbol{z}_{i'}, \boldsymbol{p}\rangle}{2\langle \boldsymbol{z}_i, \boldsymbol{p}\rangle} a_i D_i(\boldsymbol{z}_i)$$

$$= \eta_{i'} \left(-\frac{1}{2\eta_{i'}} a_{i'} D_{i'}(\boldsymbol{z}_{i'}) + \frac{\eta_i \langle \boldsymbol{z}_{i'}, \boldsymbol{p}\rangle}{2\eta_i \eta_{i'} \langle \boldsymbol{z}_i, \boldsymbol{p}\rangle} a_i D_i(\boldsymbol{z}_i) \right)$$

$$\leq \eta_{i'} \left(-\frac{1}{2}\gamma^{-1} + \frac{\eta_i \langle \boldsymbol{z}_{i'}, \boldsymbol{p}\rangle}{2\eta_{i'} \langle \boldsymbol{z}_i, \boldsymbol{p}\rangle} \cdot \gamma \right)$$

$$< \eta_{i'} \left(-\tfrac{1}{2}\gamma^{-1} + \tfrac{1}{2}\gamma^{-1} \right) = 0 \ ,$$

Conditions (2) and (3) work without modification.

The stopping conditions and update moves of Algorithm 1 must be changed to ensure that the modified definition of KKT points can be met. The first modification is to replace the definition of $\boldsymbol{d}^{k+1}$ on line 6 with $(\eta_1/a_1, \ldots, \eta_n/a_n)$, if we rescale $\boldsymbol{a}$ so that $\langle \boldsymbol{a}^k, \boldsymbol{d}^k_* \rangle = \sum_i \eta_i$. This choice will ensure that small changes between $\boldsymbol{d}^k_*$ and $\boldsymbol{d}^{k+1}$ imply the right η-rescaled KKT conditions. Note that, now the stopping condition on line 7 will indeed ensure that the algorithm returns $(1+\varepsilon)$-KKT point with respect to the new definition.

Finally, it remains to show that the number of iterations is still polynomial, though it will depend on the η_i's. We can use Lemma 12 to lower-bound the improvement at each step: We have $\text{RelDist}(\boldsymbol{d}^k_*, \boldsymbol{d}^{k+1}) > \varepsilon$, and therefore $\mathcal{L}(\boldsymbol{d}^{k+1}, \mathbf{1}) - \mathcal{L}(\boldsymbol{d}^k_*, \mathbf{1}) > \Omega(\varepsilon^2/n^2)$. However, we need to use $\mathcal{L}(\boldsymbol{d}, \boldsymbol{\eta})$, not $\mathcal{L}(\boldsymbol{d}, \mathbf{1})$, since the latter is not a potential function for the modified algorithm. We have, however,

$$\mathcal{L}(\boldsymbol{d}, \mathbf{1}) \cdot \min_i \eta_i \ \leq \ \mathcal{L}(\boldsymbol{d}, \boldsymbol{\eta}) \ \leq \ \mathcal{L}(\boldsymbol{d}, \mathbf{1}) \cdot \max_i \eta_i \ .$$

This allows us to conclude that

$$\mathcal{L}(\boldsymbol{d}^{k+1}, \boldsymbol{\eta}) - \mathcal{L}(\boldsymbol{d}^k_*, \boldsymbol{\eta}) \geq \frac{\varepsilon^2 \cdot \min_i \eta_i}{16n^2} \tag{9}$$

Now, a proof identical to that of Lemma 13 bounds the number of iterations by

$$\frac{\text{poly}(n, 1/\varepsilon)}{\min_i \eta_i} \cdot \left(\max_{\boldsymbol{d} \in \mathcal{D}} \mathcal{L}(\boldsymbol{d}; \boldsymbol{\eta}) - \mathcal{L}(\boldsymbol{d}^0; \boldsymbol{\eta}) \right)$$

$$\leq \frac{\text{poly}(n, 1/\varepsilon)}{\min_i \eta_i} \cdot \left(\sum_i \eta_i \log(m \cdot \max_{i,j} D_{ij}) - \sum_i \eta_i \log(\tfrac{m}{2n} \min_{i,j} D_{ij}) \right)$$

$$= \frac{\text{poly}(n, 1/\varepsilon)}{\min_i \eta_i} \cdot \left(\sum_{i=1}^n \eta_i \right) \cdot \log \left(\frac{2n \cdot \max_{i,j} D_{ij}}{\min_{i,j} D_{ij}} \right) \ ,$$

which is the same running time as the equal-income setting, up to the $\min \eta_i \cdot \sum \eta_i$ term. But the η_i's are dimensionless, and therefore without loss of generality, we can assume $\max_i \eta_i = 1$, which implies $\sum \eta_i \leq n$, and so we can bound the number of iterations by

$$\text{poly}(n, 1/\varepsilon) \cdot \frac{\max_i \eta_i}{\min_i \eta_i} \cdot \log \left(\frac{2n \cdot \max_{i,j} D_{ij}}{\min_{i,j} D_{ij}} \right) \ .$$

Thus, we have shown that when disutilities are linear, the results in this paper extend to CE without too much modification. And the running time guarantees are the same as far as the max to min income ratio is polynomially bounded. It remains to argue that the same is true for general disutilities. The above modifications are in fact the only ones needed for this more general setting: the proofs of Conditions (1), (2), and (3) in the definition of approximate KKT all go through the same as they did in the linear case, and the analysis of approximately supporting hyperplane does not play a role in the unequal income.

References

[AD54]   Kenneth J Arrow and Gerard Debreu. Existence of an equilibrium for a competitive economy. *Econometrica: Journal of the Econometric Society*, pages 265–290, 1954.

[BMSY17]   Anna Bogomolnaia, Hervé Moulin, Fedor Sandomirskiy, and Elena Yanovskaia. Competitive division of a mixed manna. *Econometrica*, 85(6):1847–1871, 2017.

[BMSY19]   Anna Bogomolnaia, Hervé Moulin, Fedor Sandomirskiy, and Elena Yanovskaia. Dividing bads under additive utilities. *Social Choice and Welfare*, 52(3):395–417, 2019.

[BS19]   Simina Branzei and Fedor Sandomirskiy. Algorithms for competitive division of chores. arXiv:1907.01766 (To appear in Mathematics of Operations Research), 2019.

[Bub14]   Sébastien Bubeck. Convex optimization: Algorithms and complexity. *arXiv preprint arXiv:1405.4980*, 2014.

[CCD13]   Yun Kuen Cheung, Richard Cole, and Nikhil Devanur. Tatonnement beyond gross substitutes? Gradient descent to the rescue. In *Proc. 45th Symp. Theory of Computing (STOC)*, pages 191–200, 2013.

[CDDT09] Xi Chen, Decheng Dai, Ye Du, and Shang-Hua Teng. Settling the complexity of Arrow-Debreu equilibria in markets with additively separable utilities. In *Proc. 50th Symp. Foundations of Computer Science (FOCS)*, pages 273–282, 2009.

[CDG+17]   Richard Cole, Nikhil Devanur, Vasilis Gkatzelis, Kamal Jain, Tung Mai, Vijay Vazirani, and Sadra Yazdanbod. Convex program duality, Fisher markets, and Nash social welfare. In *Proc. 18th Conf. Economics and Computation (EC)*, 2017.

[CGMM20] Bhaskar Ray Chaudhury, Jugal Garg, Peter McGlaughlin, and Ruta Mehta. Dividing bads is harder than dividing goods: On the complexity of fair and efficient division of chores. *arXiv preprint arXiv:2008.00285*, 2020.

[CGMM21] Bhaskar Ray Chaudhury, Jugal Garg, Peter McGlaughlin, and Ruta Mehta. Competitive allocation of a mixed manna. In *Proc. 32nd Symp. Discrete Algorithms (SODA)*, 2021.

[CPY17]   Xi Chen, Dimitris Paparas, and Mihalis Yannakakis. The complexity of non-monotone markets. *Journal of the ACM (JACM)*, 64(3):1–56, 2017.

[CT09]   Xi Chen and Shang-Hua Teng. Spending is not easier than trading: on the computational equivalence of fisher and arrow-debreu equilibria. In *International Symposium on Algorithms and Computation*, pages 647–656. Springer, 2009.

[DGM16]   Ran Duan, Jugal Garg, and Kurt Mehlhorn. An improved combinatorial polynomial algorithm for the linear Arrow-Debreu market. In *Proc. 27th Symp. Discrete Algorithms (SODA)*, pages 90–106, 2016.

[DGV16]   Nikhil Devanur, Jugal Garg, and László Végh. A rational convex program for linear Arrow-Debreu markets. *ACM Trans. Econom. Comput.*, 5(1):6:1–6:13, 2016.

[DK08]   Nikhil Devanur and Ravi Kannan. Market equilibria in polynomial time for fixed number of goods or agents. In *Proc. 49th Symp. Foundations of Computer Science (FOCS)*, pages 45–53, 2008.

[DM15]   Ran Duan and Kurt Mehlhorn. A combinatorial polynomial algorithm for the linear Arrow-Debreu market. *Inf. Comput.*, 243:112–132, 2015.

[DPSV08]   Nikhil Devanur, Christos Papadimitriou, Amin Saberi, and Vijay Vazirani. Market equilibrium via a primal–dual algorithm for a convex program. *J. ACM*, 55(5), 2008.

[EG59]   Edmund Eisenberg and David Gale. Consensus of subjective probabilities: The pari-mutuel method. *The Annals of Mathematical Statistics*, 30(1):165–168, 1959.

[FGL16]   Michal Feldman, Nick Gravin, and Brendan Lucier. Combinatorial walrasian equilibrium. *SIAM J. Comput.*, 45(1):29–48, 2016.

[GM20]   Jugal Garg and Peter McGlaughlin. Computing competitive equilibria with mixed manna. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020*, pages 420–428, 2020.

[GMSV15]   Jugal Garg, Ruta Mehta, Milind Sohoni, and Vijay V. Vazirani. A complementary pivot algorithm for market equilibrium under separable, piecewise-linear concave utilities. *SIAM J. Comput.*, 44(6):1820–1847, 2015. Extended abstract appeared in STOC 2012.

[GV19]   Jugal Garg and László A Végh. A strongly polynomial algorithm for linear exchange markets. In *Proc. 51st Symp. Theory of Computing (STOC)*, 2019.

[KTK80]   Mikhail K Kozlov, Sergei P Tarasov, and Leonid G Khachiyan. The polynomial solvability of convex quadratic programming. *USSR Computational Mathematics and Mathematical Physics*, 20(5):223–228, 1980.

[McK54]   Lionel McKenzie. On equilibrium in graham's model of world trade and other competitive systems. *Econometrica*, 22(2):147–161, 1954.

[McK59]   Lionel W. McKenzie. On the existence of general equilibrium for a competitive market. *Econometrica*, 27(1):54–71, 1959.

[MV07]   Nimrod Megiddo and Vijay V. Vazirani. Continuity properties of equilibrium prices and allocations in linear fisher markets. In *WINE*, volume 4858 of *Lecture Notes in Computer Science*, pages 362–367. Springer, 2007.

[NP83]   E I Nenakov and M E Primak. One algorithm for finding solutions of the Arrow-Debreu model. *Kibernetica*, 3:127–128, 1983.

[Orl10]   James Orlin. Improved algorithms for computing Fisher's market clearing prices. In *Proc. 42nd Symp. Theory of Computing (STOC)*, pages 291–300, 2010.

[Rub18]   Aviad Rubinstein. Inapproximability of nash equilibrium. *SIAM J. Comput.*, 47(3):917–959, 2018.

[Var74]   Hal Varian. Equity, envy and efficiency. *J. Econom. Theory*, 29(2):217–244, 1974.

[Vég12]   László Végh. Strongly polynomial algorithm for a class of minimum-cost flow problems with separable convex objectives. In *Proc. 44th Symp. Theory of Computing (STOC)*, pages 27–40, 2012.

[VY11]   Vijay Vazirani and Mihalis Yannakakis. Market equilibrium under separable, piecewise-linear, concave utilities. *J. ACM*, 58(3):10, 2011.

[Wal74]   Léon Walras. *Éléments d'économie politique pure, ou théorie de la richesse sociale (Elements of Pure Economics, or the theory of social wealth)*. English version, Cambridge University Press, Lausanne, Paris, 1874. (1899, 4th ed.; 1926, rev ed., 1954, Engl. transl.).

A Technical Proofs

Claim 3. $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ *is convex, when the disutility functions* $D_1, \ldots, D_n$ *are convex.*

Proof. Let $\boldsymbol{y}, \boldsymbol{y}' \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$, such that $\boldsymbol{y} = \boldsymbol{d} + \boldsymbol{\Delta}$ for some $\boldsymbol{d} \in \mathcal{D}$ and $\boldsymbol{\Delta} \in \mathbb{R}^n_{\geq 0}$. Define $\boldsymbol{d}'$ and $\boldsymbol{\Delta}'$ similarly for $\boldsymbol{y}'$. Let $0 < \lambda < 1$, and let $\bar{\lambda} := 1 - \lambda$. Letting $\vec{D}(\boldsymbol{x}) := (D_1(\boldsymbol{x}_1), \ldots, D_n(\boldsymbol{x}_n))$, let $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{F}$ be such that $\vec{D}(\boldsymbol{x}) = \boldsymbol{d}$, and $\vec{D}(\boldsymbol{x}') = \boldsymbol{d}'$. Since $\mathcal{F}$ is a linear polytope, and is therefore convex, $\lambda \boldsymbol{x} + \bar{\lambda} \boldsymbol{x}' \in \mathcal{F}$, and so $\vec{D}(\lambda \boldsymbol{x} + \bar{\lambda} \boldsymbol{x}') \in \mathcal{D}$. Since the D_i's are convex, we have that $D_i(\lambda \boldsymbol{x}_i + \bar{\lambda} \boldsymbol{x}'_i) \leq \lambda D_i(\boldsymbol{x}_i) + \bar{\lambda} D_i(\boldsymbol{x}'_i)$ for all i. Thus, component-wise, we have

$$\vec{D}(\lambda \boldsymbol{x} + \bar{\lambda} \boldsymbol{x}') \leq \lambda \vec{D}(\boldsymbol{x}) + \bar{\lambda} \vec{D}(\boldsymbol{x}') = \lambda \boldsymbol{d} + \lambda \boldsymbol{d}'$$

Thus, there must exist a $\boldsymbol{\Delta}'' \in \mathbb{R}^n_{\geq 0}$ such that

$$\lambda \boldsymbol{y} + \bar{\lambda} \boldsymbol{y}' = \lambda \boldsymbol{\Delta} + \bar{\lambda} \boldsymbol{\Delta}' + \vec{D}(\lambda \boldsymbol{x} + \bar{\lambda} \boldsymbol{x}') + \boldsymbol{\Delta}'' \in \mathcal{D} + \mathbb{R}^n_{\geq 0}. \qquad \square$$

Lemma 8. *Regardless of the geometry of $\mathcal{D}$, so long as $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ is convex, we have that for each iteration $k \geq 0$ of Algorithm 1:*

1. *The hyperplane defined as $\{\boldsymbol{y} \in \mathbb{R}^n | \langle \boldsymbol{a}^k, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}^k, \boldsymbol{d}^k_* \rangle\}$ is supporting for $\mathcal{D} + \mathbb{R}^n_{\geq 0}$, at $\boldsymbol{d}^k_*$.*

2. *If $\boldsymbol{d}^k$ has strictly positive entries and does not lie in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$, then $\boldsymbol{d}^k_*$, $\boldsymbol{a}^k$, and $\boldsymbol{d}^{k+1}$ have strictly positive entries, and $\boldsymbol{d}^k_* \in \mathcal{D}$.*

Proof. Informally, 1. holds by the KKT conditions of the minimization problem, and the geometry of the ℓ_2 norm, though we give here a more direct proof. The idea is to assume that some feasible point lies on the wrong side of the supporting hyperplane, and contradict the minimality of the distance of $\boldsymbol{d}^k_*$.

Let $\boldsymbol{a} := \boldsymbol{d}^k_* - \boldsymbol{d}^k$ (*i.e.* $\boldsymbol{a}^k$ before rescaling), and let $\boldsymbol{y}$ be any feasible point in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$. Define $\boldsymbol{u} := \boldsymbol{y} - \boldsymbol{d}^k$, which can be decomposed as $\boldsymbol{u} = (1 - \alpha)\boldsymbol{a} + \beta \boldsymbol{v}$, with $\langle \boldsymbol{v}, \boldsymbol{a} \rangle = 0$. Assume without loss of generality $\|\boldsymbol{a}\|_2 = \|\boldsymbol{v}\|_2$.

For any $0 \leq \lambda \leq 1$, the vector $\boldsymbol{d}^k + \lambda \boldsymbol{u} + (1 - \lambda)\boldsymbol{a}$ is feasible, by convexity. Furthermore, its squared distance from $\boldsymbol{d}^k$ is given by

$$\begin{aligned}
\|(\lambda \boldsymbol{y} + (1 - \lambda)\boldsymbol{d}^k_*) - \boldsymbol{d}^k\|_2^2 &= \|\lambda \boldsymbol{u} + (1 - \lambda)\boldsymbol{a}\|_2^2 \\
&= \|(\lambda(1 - \alpha) + 1 - \lambda)\boldsymbol{a} + \lambda \beta \boldsymbol{v}\|_2^2 \\
&= (1 - \lambda \alpha)^2 \|\boldsymbol{a}\|_2^2 + (\lambda \beta)^2 \|\boldsymbol{v}\|_2^2 \\
&= 1 - 2\lambda \alpha + \lambda^2(\alpha^2 + \beta^2)\|\boldsymbol{a}\|_2^2
\end{aligned}$$

Note that $\langle \boldsymbol{y}, \boldsymbol{a} \rangle - \langle \boldsymbol{d}^k_*, \boldsymbol{a} \rangle = ((1 - \alpha) - 1)\|\boldsymbol{a}\|_2^2 = -\alpha \|\boldsymbol{a}\|_2^2$, which is negative if and only if $\alpha > 0$. Thus, assume $\alpha > 0$ for a contradiction, and set $\lambda := \alpha/(\alpha^2 + \beta^2)$, a positive number. Furthermore, note that $\|\boldsymbol{d}^k_* - \boldsymbol{d}^k\|_2^2 = \|\boldsymbol{a}\|_2^2$. Therefore,

$$\frac{\|(\lambda \boldsymbol{y} + (1 - \lambda)\boldsymbol{d}^k_*) - \boldsymbol{d}^k\|_2^2}{\|\boldsymbol{d}^k_* - \boldsymbol{d}^k\|_2^2} = 1 - \frac{2\alpha^2}{\alpha^2 + \beta^2} + \frac{\alpha^2}{\alpha^2 + \beta^2} < 1$$

If $\lambda \leq 1$, this is a contradiction of the minimality of $\boldsymbol{d}^k_*$, since $\lambda \boldsymbol{y} + (1 - \lambda)\boldsymbol{d}^k_*$ must be a feasible point. If $\lambda \geq 1$, we observe that distance from $\boldsymbol{d}^k$ is a convex function, and therefore it must be smaller at $\boldsymbol{y}$, which is a convex combination of $\lambda \boldsymbol{y} + (1 - \lambda)\boldsymbol{d}^k_*$ and $\boldsymbol{d}^k_*$, also a contradiction.

Part 2. of the statement holds by the geometry of the region, and part 1. Suppose for a contradiction that for some agent i, $(\boldsymbol{d}_*^k)_i < (\boldsymbol{d}^k)_i$. Then setting the i-th coordinate of $\boldsymbol{d}_*^k$ to $(\boldsymbol{d}^k)_i$ will move it strictly closer to $\boldsymbol{d}^k$. Furthermore, this move is in the $\mathbb{R}_{\geq 0}^n$ direction, and thus the new point will lie in the set, contradicting the minimality of $\boldsymbol{d}_*^k$. Therefore, coordinate-wise, $\boldsymbol{d}_*^k \geq \boldsymbol{d}^k$, ensuring that the former has strictly positive entries. This also proves that the entries of $\boldsymbol{a}^k$ are non-negative. It remains to show they are positive.

Suppose $\boldsymbol{d}_*^k$ agrees with $\boldsymbol{d}^k$ in the i-th component, *i.e.* $(\boldsymbol{a}^k)_i = 0$. This cannot happen in every entry, as otherwise $\boldsymbol{d}^k$ would have been feasible. By part 1., we have that $\boldsymbol{a}^k$ is normal to a supporting hyperplane for $\mathcal{D} + \mathbb{R}_{\geq 0}^n$ at $\boldsymbol{d}_*^k$. Furthermore, $\langle \boldsymbol{a}^k, \boldsymbol{d}_*^k \rangle > 0$, since $\boldsymbol{a}^k$ has non-negative entries with at least one positive entry, and the entries of $\boldsymbol{d}_*^k$ are positive. However, letting $\boldsymbol{e}_i$ be the i-th standard basis vector, we have that $D_i(\boldsymbol{1})\boldsymbol{e}_i$ lies in $\mathcal{D}$, but $\langle \boldsymbol{a}^k, D_i(\boldsymbol{1})\boldsymbol{e}_i \rangle = 0 < \langle \boldsymbol{a}^k, \boldsymbol{d}_*^k \rangle$, a contradiction. Thus the entries of $\boldsymbol{a}^k$ must all be positive as well.

This allows us to show that $\boldsymbol{d}_*^k \in \mathcal{D}$. Suppose not, then $\boldsymbol{d}_*^k \in \mathcal{D} + (\mathbb{R}_{\geq 0}^n \setminus \{\boldsymbol{0}\})$. We have established that coordinate-wise, $\boldsymbol{d}_*^k > \boldsymbol{d}^k$. Therefore, there must exist some coordinate i such that we can reduce $(\boldsymbol{d}_*^k)_i$ and remain feasible, contradicting the minimality.

Finally, since we simply define $\boldsymbol{d}^{k+1}$ as being the coordinate-wise inverse of $\boldsymbol{a}^k$, it must also have positive entries. $\qquad\square$

Claim 44. *Program 8 is a convex program, i.e. it minimizes a convex function over a convex domain.*

Proof. Since $\sum_{i \in [n]} \beta_i^2$ is a convex function, our program indeed minimizes a convex function. We now argue that the domain is convex. To this end, first note that the constraints $\sum_{i \in [n]} (\boldsymbol{z})_{ij} \geq 1$ for all $j \in [m]$, and $(\boldsymbol{z})_{ij} \geq 0$ for all $i \in [n]$, $j \in [m]$ are linear constraints and the intersections of all these half-spaces is convex. Now, we show that each constraint $D_i(\boldsymbol{z}_i) - (\boldsymbol{d}^k)_i - (\boldsymbol{\beta})_i \leq 0$ is also convex. Let $(\boldsymbol{z}^1, \boldsymbol{\beta}^1)$ and $(\boldsymbol{z}^2, \boldsymbol{\beta}^2)$ be two points such that $D_i(\boldsymbol{z}_i^1) - (\boldsymbol{d}^k)_i - (\boldsymbol{\beta}^1)_i \leq 0$ and $D_i(\boldsymbol{z}_i^2) - (\boldsymbol{d}^k)_i - (\boldsymbol{\beta}^2)_i \leq 0$. Fix any $\lambda \in (0,1)$ and let $(\boldsymbol{z}', \boldsymbol{\beta}') = \lambda \cdot (\boldsymbol{z}^1, \boldsymbol{\beta}^1) + (1-\lambda) \cdot (\boldsymbol{z}^2, \boldsymbol{\beta}^2)$. To prove convexity of the domain, it suffices to show that $D_i(\boldsymbol{z}_i') - (\boldsymbol{d}^k)_i - (\boldsymbol{\beta}')_i \leq 0$. To this end, note that

$$
\begin{aligned}
D_i(\boldsymbol{z}_i') & - (\boldsymbol{d}^k)_i - (\boldsymbol{\beta}')_i \\
&= D_i(\lambda \cdot \boldsymbol{z}_i^1 + (1-\lambda) \cdot \boldsymbol{z}_i^2) - (\boldsymbol{d}^k)_i - \lambda \cdot (\boldsymbol{\beta}^1)_i - (1-\lambda) \cdot (\boldsymbol{\beta}^2)_i \\
&\leq \lambda \cdot D_i(\boldsymbol{z}_i^1) + (1-\lambda) \cdot D_i(\boldsymbol{z}_i^2) - (\boldsymbol{d}^k)_i - \lambda \cdot (\boldsymbol{\beta}^1)_i - (1-\lambda) \cdot (\boldsymbol{\beta}^2)_i \quad (D_i(\cdot) \text{ convex}) \\
&= \lambda \cdot (D_i(\boldsymbol{z}_i^1) - (\boldsymbol{d}^k)_i - (\boldsymbol{\beta}^1)_i) + (1-\lambda) \cdot (D_i(\boldsymbol{z}_i^2) - (\boldsymbol{d}^k)_i - (\boldsymbol{\beta}^2)_i) \\
&\leq 0 + 0 = 0. \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \square
\end{aligned}
$$

B   Approximate CEEI to Approximate EF and PO

CEEI allocations are known ensure envy-freeness and Pareto-optimality. In this section we show that we can determine an allocation that is approximately envy-free and approximately Pareto-optimal from an ε-CEEI. To this end, let $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ and $\boldsymbol{p}$ be the allocation and the price vector at a $(1-\varepsilon)$-CEEI. Let $\alpha_j = \sum_{i \in [n]} x_{ij}$. Note that if $\alpha_j > 1$, then j is over allocated and if $\alpha_j < 1$, then j is under allocated. Let $\alpha = max_j \alpha_j$ and $\alpha' = min_j \alpha_j$. Note that $1 - \varepsilon \leq \alpha' \leq \alpha \leq 1 + \varepsilon$. We define a new allocation $\boldsymbol{y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_n)$ such that $y_{ij} = x_{ij}/(\alpha_j)$. Note that for all chores $j \in [m]$, we have $\sum_{i \in [n]} y_{ij} = 1$, *i.e.* no chore is over-allocated or under-allocated. Observe that the total earning for each agent has not changed significantly: for each $i \in [n]$, we have $(1/\alpha) \cdot \langle \boldsymbol{x}_i, \boldsymbol{p} \rangle \leq$

$\langle \boldsymbol{y}_i, \boldsymbol{p} \rangle \leq (1/\alpha') \cdot \langle \boldsymbol{x}_i, \boldsymbol{p} \rangle$. Similarly, by 1-homogeneity, the disutilities for the agents have also not changed significantly: for each $i \in [n]$, we have $(1/\alpha) \cdot D_i(\boldsymbol{x}_i) \leq D_i(\boldsymbol{y}_i) \leq (1/\alpha') \cdot D_i(\boldsymbol{x}_i)$.

Claim 45. *Allocation* $\boldsymbol{y}$ *is* $(1-4\varepsilon)$-*envy-free, i.e. for all pairs of agents* i *and* i', *we have* $(1-4\varepsilon) \cdot D_i(\boldsymbol{y}_i) \leq D_i(\boldsymbol{y}_{i'})$.

Proof. We first show that allocation $\boldsymbol{x}$ is approximately-envy-free. Thereafter, since the disutilities of the agents in allocation $\boldsymbol{y}$ are not significantly different from their disutilities in $\boldsymbol{x}$, the approximate-envy-freeness for $\boldsymbol{y}$ will follow easily. Consider two agents i and i'. Let $\boldsymbol{x}'_{i'} = (1 - \varepsilon)^{-1}\boldsymbol{x}_{i'}$. Since $\boldsymbol{x}$ is at a $(1-\varepsilon)$-CEEI, by condition (1) in Definition 1, we have $(1-\varepsilon)\langle \boldsymbol{x}_i, \boldsymbol{p} \rangle \leq \langle \boldsymbol{x}_{i'}, \boldsymbol{p} \rangle$, implying that $\langle \boldsymbol{x}_i, \boldsymbol{p} \rangle \leq \langle \boldsymbol{x}'_{i'}, \boldsymbol{p} \rangle$. Therefore, $(1 - \varepsilon) \cdot D_i(\boldsymbol{x}_i) \leq D_i(\boldsymbol{x}'_{i'}) = (1 - \varepsilon)^{-1} \cdot D_i(\boldsymbol{x}_{i'})$ (by 1-homogeneity), further implying that $(1 - \varepsilon)^2 \cdot D_i(\boldsymbol{x}_i) \leq D_i(\boldsymbol{x}_{i'})$.

Now we show approximate-envy-freeness for $\boldsymbol{y}$: Since $(1/\alpha) \cdot D_i(\boldsymbol{x}_i) \leq D_i(\boldsymbol{y}_i) \leq (1/\alpha') \cdot D_i(\boldsymbol{x}_i)$, we have that

$$D_i(\boldsymbol{y}_{i'}) \ \geq \ \frac{1}{\alpha} \cdot D_i(\boldsymbol{x}_{i'}) \ \geq \ \frac{(1-\varepsilon)^2}{\alpha} \cdot D_i(\boldsymbol{x}_i) \ \geq \ \frac{(1-\varepsilon)^2 \cdot \alpha'}{\alpha} \cdot D_i(\boldsymbol{y}_i).$$

Using the fact that $\alpha \leq 1 + \varepsilon$ and $\alpha' \geq 1 - \varepsilon$ we have that

$$D_i(\boldsymbol{y}_{i'}) \ \geq \ \frac{(1-\varepsilon)^3}{1+\varepsilon} \cdot D_i(\boldsymbol{y}_i) \ \geq \ (1 - 4\varepsilon) \cdot D_i(\boldsymbol{y}_i). \qquad \square$$

Claim 46. *Allocation* $\boldsymbol{y}$ *is* $(1 - 2\varepsilon)$-*Pareto-optimal, i.e., there exists no allocation* $\boldsymbol{y}'$ *such that* $D_i(\boldsymbol{y}'_i) \leq (1 - 2\varepsilon) \cdot D_i(\boldsymbol{y}_i)$ *for all* $i \in [n]$ *with at least one strict inequality, and such that* $\boldsymbol{y}'$ *does not under- or over-allocate any item.*

Proof. Assume otherwise. Let $\boldsymbol{y}'$ be an allocation such that $D_i(\boldsymbol{y}'_i) \leq (1 - \varepsilon) \cdot D_i(\boldsymbol{y}_i)$ for all $i \in [n]$ and let i' be an agent such that $D_{i'}(\boldsymbol{y}'_{i'}) < (1 - 2\varepsilon) \cdot D_{i'}(\boldsymbol{y}_{i'}) \leq (1 - \varepsilon) \cdot D_{i'}(\boldsymbol{y}_{i'})$. Since $\boldsymbol{y}$ is at a $(1 - \varepsilon)$-CEEI, by condition (2), in Definition 1, we can conclude that $\langle \boldsymbol{y}'_{i'}, \boldsymbol{p} \rangle < \langle \boldsymbol{y}_{i'}.\boldsymbol{p} \rangle$. Note that since both $\boldsymbol{y}$ and $\boldsymbol{y}'$ are feasible allocations we have

$$\sum_{i \in [n]} \langle \boldsymbol{y}'_i, \boldsymbol{p} \rangle \ = \ \sum_{j \in [m]} p_j \cdot \sum_{i \in [n]} y'_{ij} \ = \ \sum_{j \in [m]} p_j \ = \ \sum_{j \in [m]} p_j \cdot \sum_{i \in [n]} y_{ij} \ = \ \sum_{i \in [n]} \langle \boldsymbol{y}_i, \boldsymbol{p} \rangle.$$

The last two equalities require that $\boldsymbol{y}'$ does not under- or over-allocate any item, as we have assumed. The original allocation $\boldsymbol{y}$ satisfies these conditions by construction. These equalities imply $\sum_{i \in [n]} \langle \boldsymbol{y}'_i - \boldsymbol{y}_i, \boldsymbol{p} \rangle = 0$. Since we have $\langle \boldsymbol{y}'_{i'}, \boldsymbol{p} \rangle < \langle \boldsymbol{y}_{i'}, \boldsymbol{p} \rangle$, there must be an agent ℓ such that $\langle \boldsymbol{y}'_\ell, \boldsymbol{p} \rangle > \langle \boldsymbol{y}_\ell, \boldsymbol{p} \rangle$. Again, by condition (2) in Definition 1, we can conclude that $D_\ell(\boldsymbol{y}'_\ell) \geq (1 - \varepsilon) \cdot D_\ell(\boldsymbol{y}_\ell) > (1 - 2\varepsilon) \cdot D_\ell(\boldsymbol{y}_\ell)$, which is a contradiction. $\qquad \square$