arXiv:1312.4489v3 [math.OC] 11 Jan 2017

A UTILITY THEORY BASED INTERACTIVE APPROACH TO ROBUSTNESS IN LINEAR OPTIMIZATION

MEHDI KARIMI, SOMAYEH MOAZENI, AND LEVENT TUNÇEL

Abstract. We treat uncertain linear programming problems by utilizing the notion of weighted analytic centers and notions from the area of multi-criteria decision making. After introducing our approach, we develop interactive cutting-plane algorithms for robust optimization, based on concave and quasi-concave utility functions. In addition to practical advantages, due to the flexibility of our approach, we are able to prove that under a theoretical framework due to Bertsimas and Sim [14], which establishes the existence of certain convex formulation of robust optimization problems, the robust optimal solutions generated by our algorithms are at least as desirable to the decision maker as any solution generated by many other robust optimization algorithms in the theoretical framework. We present some probabilistic bounds for feasibility of robust solutions and evaluate our approach by means of computational experiments.

1. Introduction

Optimization problems are widespread in real life decision making situations. However, data perturbations as well as uncertainty in at least part of the data are very difficult to avoid in practice. Therefore, in most cases we have to deal with the reality that some aspects of the data of the optimization problem at hand are uncertain. This uncertainty is caused by many sources such as forecasting, or approximations in the design of mathematical models, or data approximation, or noise in measurements. In order to handle optimization problems under uncertainty, several techniques have been proposed. The most common, widely-known approaches are

- **Sensitivity analysis:** typically, the influence of data uncertainty is initially ignored, and then the obtained solution is justified/analyzed based on the data perturbations [16].
- **Chance constrained programming:** we use some stochastic models of uncertain data to replace the deterministic constraints by their probabilistic counterparts [44, 51, 22, 57]. It is a natural way of converting the uncertain optimization problem into a deterministic one. However, most of the time the result is a computationally intractable problem [6].

Date: January 12, 2017.

Mehdi Karimi: (m7karimi@uwaterloo.ca) Department of Combinatorics and Optimization, Graduate Student, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada. Research of this author was supported in part by a Discovery Grant from NSERC and by ONR Research Grant N00014-12-10049.

Somayeh Moazeni: (smoazeni@stevens.edu) School of Systems and Enterprises, Stevens Institute of Technology, Babbio Center, 1 Castle Point Terrace on Hudson Hoboken, NJ 07030, U.S.A. Research of this author was supported in part by Discovery Grants from NSERC.

Levent Tunçel: (ltuncel@uwaterloo.ca) Department of Combinatorics and Optimization, Faculty of Mathematics, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada. Research of this author was supported in part by Discovery Grants from NSERC and by ONR Research Grant N00014-12-10049.

- **Stochastic programming:** the goal is to find a solution that is feasible for all (or almost all) possible instances of the data and to optimize the expectation of some function of the decisions and the random variables [59].
- **Robust optimization:** robust optimization is the method that is most closely related to our approach. Generally speaking, robust optimization can be applied to any optimization problem where the uncertain data can be separated from the problem's structure. Having been heavily studied for convex optimization problems [6, 63, 9, 8, 10, 7, 14, 15, 13, 11, 50], robust optimization is also applicable to discrete [67, 43, 21] and more general nonconvex optimization problems [61]. Robustness can be achieved by solving the robust counterpart or utilizing other unconventional methods such as simulated annealing algorithm [55]. Our focus in this paper is on uncertain linear programming problems. Uncertainty in the data means that the exact values of the data are not known, at the time when the solution has to be determined. In robust optimization framework, uncertainty in the data is described through *uncertainty sets*, which contain all possible values that may be realized for the uncertain parameters. Generally speaking, the distinction between robust optimization and stochastic programming is that robust optimization does not require the specification of the exact distribution. Stochastic programming performs well when the distributions of the uncertainties are exactly known, and robust optimization can be very useful when there is little information about those distributions.

Since the interest in robust formulations was revived in the 1990s, many researchers have introduced new formulations for robust optimization framework in linear programming and general convex programming [63, 9, 8, 10, 7, 14, 15, 13, 11, 50]. Ben-Tal and Nemirovski [9, 8] provided some of the first formulations for robust LP with detailed mathematical analysis. Bertsimas and Sim [14] proposed an approach that offers control on the degree of conservatism for every constraint as well as the objective function. Bertsimas et al. [11] characterize the robust counterpart of an LP problem with uncertainty set described by an arbitrary norm. By choosing appropriate norms, they recover the formulations proposed in the above papers [9, 8, 11].

The goal of classical robust optimization is to find a solution that is capable to cope best of all with *all* realizations of the data from a given (usually bounded) uncertainty set [6, 5]. By the classical definition of robustness [6, 10, 13, 24], a *robust optimal solution* is the solution of the following problem:

$$\max_{x \in \mathbb{R}^n} \left\{ \inf_{\tilde{c} \in \mathcal{C}} \langle \tilde{c}, x \rangle : \tilde{A}x \leq \tilde{b} \ , \forall \tilde{b} \in \mathcal{B}, \forall \tilde{A} \in \mathcal{A} \right\}, \tag{1}$$

where $\mathcal{C}$, $\mathcal{A}$, and $\mathcal{B}$ are given uncertainty sets for $\tilde{c}$, $\tilde{A}$, and $\tilde{b}$, respectively. Throughout this paper, we refer to the formulation of (1) as *classical robust formulation.*

1.1. Some drawbacks of robust optimization.

Classical robust optimization is a powerful method to deal with optimization problems with uncertain data, however, we can raise some criticisms. One of the assumptions for robust optimization is that the uncertainty set must be precisely specified before solving the problem. Even if the uncertainty is only in the right-hand-side, expecting the Decision Maker (DM) to construct accurately an ellipsoid or even a hypercube for the uncertainty set may not always be reasonable. Recently, a new approach has been proposed, called *distributionally robust* optimization, that tries to cover the gap between robust optimization and stochastic programming [27, 30, 60]. In this approach, one seeks a solution that is feasible for the worst-case probability distribution in a set of possible distributions. In a recent paper, Shapiro [60] studied distributionally robust stochastic programming in a scenario that the uncertainty set of probability measures is "close" to a reference measure. It is mentioned in [30] and also emphasized in a plenary lecture by Kuhn in ISMP2015 that, in real life applications, determining uncertainty sets precisely or determining safe operation probabilities accurately is at least very challenging.

Another main criticism of classical robust optimization is that satisfying all of the constraints, if not make the problem infeasible, may lead to an objective value very far from the optimal value of the nominal problem. This issue is more critical for large deviations. As an example, [8, 46] considered some of the problems in the NETLIB library (under reasonable assumptions on uncertainty of certain entries) and showed that classical robust counterparts of most of the problems in NETLIB become infeasible for a small perturbation. Moreover, in many other problems, objective value of the classical robust optimal solution is very low and may be unsatisfactory for the decision maker.

Several modifications of classical robust optimization have been introduced to deal with this issue. One, for example, is *globalized robust conterparts* introduced in Section 3 of [6]. The idea is to consider some constraints as "soft" whose violation can be tolerated to some degree. In this method, we take care of what happens when the data leaves the nominal uncertainty set. In other words, we have "controlled deterioration" of the constraint. These modified approaches have more flexibility than the classical robust methodology, but we have the problem that the modified robust counterpart of uncertain problems may become computationally intractable. Although the modified robust optimization framework rectifies this drawback to some extent, it intensifies the first criticism by putting more pressure on the DM to specify deterministic uncertainty sets before solving the problem.

Another criticism of classical robust optimization is that it gives the same "weight" to all the constraints. In practice, this is not the case as some constraints may be more important for the DM. There are some options in classical robust optimization like changing the uncertainty set which again intensifies the first criticism. We will see that our approach can alleviate these difficulties.

1.2. **Contributions and overview of this paper.** We present a framework which allows a fine-tuning of the classical tradeoff between robustness and conservativeness by the DM and engages DM continuously and in a more effective way throughout the optimization process. Under a suitable theoretical modeling setup, we prove that the classical robust optimization approach is a special case of our framework. We demonstrate that it is possible to efficiently perform optimization under this framework and finally, we illustrate some of our methods in our computational experiments.

One of the main contributions of this paper is the development of cutting-plane algorithms for robust optimization using the notion of weighted analytic centers in a small dimensional weight-space. We also design algorithms in the slack variable space as a theoretical stepping stone towards the more applicable and more efficient weight-space cutting-plane algorithms. Ultimately, we are proposing that our approach be used in practice with a small number (say somewhere in the order of 1 to 20) of *driving factors* that really matter to the DM. These driving factors are independent of the number of variables and constraints, and determine the dimension of the weight space (for interaction with the DM). Working in a low dimensional weight-space not only simplifies the interaction for the DM, but also makes our cutting-plane algorithms more efficient.

The notion of moving across a weight space has been widely used in the area of multi-criteria decision making: when we have several competing objective functions to optimize, a natural approach is to optimize a weighted sum of them [35], [32]. Authors in [35] presented an algorithm for evaluating and ranking items with multiple attributes. [35] is related to our work as the proposed algorithm is a cutting-plane one. However, our algorithm uses the concept of weighted analytic center which is completely different. Authors in [32] proposed a family of models (denoted by McRow) for multi-expert multi-criteria decision making. Their work is close to ours as they derived compact formulations of the McRow model by assuming some structure for the weight region, such as polyhedral or conic descriptions. Our work also has fundamental differences with [32]: cutting-plane algorithms in the weight-space find a weight vector w in a fixed weight region (the unit simplex) such that the weighted analytic center of w, say $x(w)$, is the desired solution for the DM. The algorithms we design in this paper make it possible to implement the ideas we mentioned above to help overcome some of the difficulties for robust optimization to reach a broader, practicing user base. For some further details and related discussion, also see Moazeni [46] and Karimi [37].

We introduce our formulation and the notations we use in the paper in Section 2. In Section 3, we explain our approach and prove that, under a theoretical framework due to Bertsimas and Sim [14], our approach is as least as strong as classical robust optimization. In this section, we also introduce the notion of weighted analytic centers. In Section 4, we design the cutting-plane algorithms, and explain some practical uses of our approach. Some preliminary computational results are presented in Section 5. In Section 6, we briefly talk about the extension of the approach to semidefinite programming and quasi-concave utility functions, and then conclude the paper.

2. Formulation, notations, and assumptions

Before introducing our approach in the next section, let us first explain some of the assumptions and notations we are going to use. Much of the prior work on robust linear programming addresses the uncertainty through the coefficient matrix. Bertsimas and Sim [15] considered linear programming problems in which all data except the right-hand-side (RHS) vector is uncertain. In [10, 9, 13], it is assumed that the uncertainty affects the coefficient matrix and the RHS vector. Some papers deal with uncertainty only in the coefficient matrix [8, 14, 11]. Optimization problems in which all of the data in the objective function, RHS vector and the coefficient matrix are subject to uncertainty, have been considered in [7]. As we explain in Section 3, the nominal data and a rough outer approximation of the uncertainty set are enough for our approach. However, the structure of uncertainty region is useful for the probability analysis. In this paper, we deal with the general setup that any part of the data (A, b, c) may be subject to uncertainty; however, we handle the uncertainty in A and c by first pushing the objective function into the constraints, then in the new formulation (without an objective function), by pushing all uncertainty into the RHS. Moreover, in at least some applications, the amount of uncertainty in A is limited whereas the uncertainty in the RHS and the objective function vectors may be very significant. Some of the supporting arguments for this viewpoint are:

(1) Instead of specifying uncertainty for each local variable, we can handle the uncertainties by lumping them into some global variables. These global variables can be, for example, the whole budget, human resources, availability of certain critical raw materials, government quotas, etc. It may be easier for the DM to specify the uncertainty set for these global variables. Then, we can approximate the uncertainty in the coefficient matrix with the uncertainty in the RHS and the objective function. In other words, we may fix the coefficient matrix on one of the

samples from the uncertainty set and then handle the uncertainty by introducing uncertainty to the RHS vector as in [12].

(2) A certain coefficient matrix is typical for many real world problems. In many applications of planning and network design problems such as scheduling, manufacturing, electric utilities, telecommunications, inventory management and transportation, uncertainty might only affect costs (coefficients of the objective function) and demands (the RHS vector)[49, 56, 45]. **Transportation systems:** in some problems, the nodes and the arcs are fixed. However, the cost associated to each arc is not known precisely. **Traffic assignment problems:** in most models, we may assume that the drivers have perfect information about the arcs and nodes. However, their route choice behavior makes the travelling time uncertain. **Distribution systems:** in some applications, the locations of warehouses and their capacities (in inventory planning and distribution problems) are well-known and fixed for the DM. However, the size of orders and the demand rate for an item could translate to an uncertain RHS vector. Holding costs, set up costs and shortage costs, which affect the optimal inventory cost, are also typically uncertain. These affect at least the objective function. **Medical/health applications:** in these applications (see for instance, [20, 17, 62, 19]) the DM may be a group of people (including medical doctors and a patient) who are more comfortable with a few, say 4-20, driving factors which may be more easily handled by the mathematical model, if these factors could be represented as uncertain RHS values.

    In the aforementioned applications, well-understood existing resources, reliable structures (well-established street and road networks, warehouses, and machines which are not going to change), and logical components of the formulation are translated into a certain coefficient matrix. The data in the objective function and the RHS vector are usually estimated by statistical techniques by the DM, or affected by uncertain elements such as institutional, social, or economic market conditions. Therefore, determining these coefficients with precision is often difficult or practically impossible. Hence, considering uncertainty in the objective function and the RHS vector seems to be very applicable, and motivates us to consider such formulation in LP problems separately.

(3) Uncertainty when restricted to the RHS and the objective function is easier to handle mathematically, in probabilistic analyses as well as in sensitivity analyses.

As explained above, we represent the objective function as a constraint $\langle c, x\rangle \geq v$, where v is a lower bound specified by the information from the DM. For example, if the DM decides that the objective value must not be below a certain amount, we can put v equal to that value. Therefore, as the input for our approach, we need the nominal values of A and c that we denote by $A^{(0)}$ and $c^{(0)}$, largest realizable values of b_is collected in $b^{(0)}$, and a lower bound v on the objective value. In this paper, we prefer to work only with the slack variables. For any feasible point x, we define the slack variables $\bar{s} = b^{(0)} - A^{(0)}x$ and $s_0 = -v + \langle c^{(0)}, x\rangle$. The following LP program extracted from (1) is the framework of our algorithms:

$$
\begin{array}{lll}
(2) & \max & s_0 \\
& \text{s.t.} & \langle -c^{(0)}, x\rangle + s_0 = -v, \\
& & A^{(0)}x + \bar{s} = b^{(0)}, \\
& & s := (s_0, \bar{s})^\top \geq 0.
\end{array}
$$

Our algorithms are designed for the feasible region of (2) and we do not need the information about the uncertainty sets. However, in Appendix A where we show how the uncertainty sets can affect our solutions, we assume that all the uncertainty has been pushed into the RHS. In view of (2), let us define

$$A := \begin{bmatrix} -c^{(0)} \\ A^{(0)} \end{bmatrix}, \quad b := \begin{bmatrix} -v \\ b^{(0)} \end{bmatrix}.$$

From now on, we may assume that $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. Here, without loss of generality, we impose the following restrictions on the problem (for details, see [46]): The matrix A has full column rank, i.e., $\text{rank}(A) = n \leq m$. The set $\{x \in \mathbb{R}^n : Ax \leq b\}$ is bounded and has nonempty interior. In this paper, vectors and matrices are denoted, respectively, by lower and uppercase letters. The matrices Y and S represent diagonal matrices, having the components of vectors y and s on their main diagonals, respectively. The letters e and e_i denote a vector of all ones and the ith unit basis vector with the appropriate dimension, respectively. The rows of a matrix are shown by superscripts of the row, i.e., $a^{(i)}$ is the i-th row of the matrix A. The inner product of two vectors $a, b \in \mathbb{R}^n$ is shown both by $\langle a, b\rangle$ and $a^\top b$. For a matrix A, we show the *range* of A with $\mathcal{R}(A)$ and the *null space* of A with $\mathcal{N}(A)$.

In the next section, we introduce our utility theory based approach and compare it to classical robust optimization. In order to use robust optimization efficiently, a tractable robust counterpart is needed for a problem with uncertainty. We introduce a general framework that covers many interesting robust counterparts in the literature, and then prove two theorems that show our approach is at least as general as this framework for classical robust optimization.

3. A utility theory based interactive approach and weighted analytic centers

3.1. **A utility theory based interactive approach.** Consider A and b defined in Section 2. Let us define $B_s := \{b - Ax : \; Ax \leq b\}$ as the set of all feasible slack vectors. Then we can write (2) as

$$\begin{array}{ll} \max & U(s) \\ s.t. & s \in B_s, \end{array} \tag{3}$$

where $U(s) := s_0$. This $U(s)$, which we denote as utility function, is the simplest one that takes into account only maximizing the objective function. Intuitively, we can cover a huge class of problems by using more complicated utility functions in problem (3). In this paper, we try to solve (3) for a general utility function $U : \mathbb{R}^m \to \mathbb{R}$ that models all the preferences of the DM. We do *not* have access to this utility function, however assume that, for a slack vector s, we can ask the DM questions to extract some information about the function. In many applications, robustness of a solution may be a monotone function of the slack variables (this typically corresponds to quasi-concave utility function in our theoretical development); however, this kind of property of the utility function is not as restrictive in our approach as it may seem since we can also handle quasi-concave utility functions. We can also use modeling techniques from goal programming (see [34]). Assuming that $U(s)$ is concave or quasi-concave, we retrieve the supergradient of $U(s)$ at some points through a sequence of simple questions such as pairwise comparison questions (see for instance [39, 38, 42]).

Table 1 compares our utility theory based interactive approach with classical robust optimization on the input to the algorithm, interaction with the DM, and handling large scale problems. Note that our approach is different from the heavily studied *Reinforcement Learning* [36, 64]. Reinforcement

Learning is a method of using statistical techniques and dynamic programming to estimate an explicit utility function, whereas in our approach, we do not need an explicit formulation or even an estimate. Our interactive approach has the additional benefit that in case the DM is inconsistent in his/her answers, since our approach is interactive and operates with very local information, we can provide the DM with a better chance of correcting mistakes as well as *learning* throughout the interactive process, what is possible within the given constraints and preferences.

TABLE 1. Classical robust optimization versus our utility theory based approach.

	Classical Robust Opt.	**Utility Theory Based Interactive Approach**
Input	Nominal values of A, b, and c. Uncertainty regions for A, b, and c, e.g., high-dimensional ellipsoids and/or intervals [1].	Nominal values of A and c, a lower bound on the objective function v, and a suitable [2] vector $b^{(0)}$.
Communication with the DM	Once at the modeling phase in receiving uncertainty regions. Once at the end, delivering the robust optimal solution.	Interactive throughout the whole optimization process.
Handling large scale problems	Large scale optimization techniques can be used for the robust counterpart.	In addition to large scale optimization techniques, a driving factor idea can be used to drastically reduce the dimension of search space and communication space for the DM.
	Specifying the uncertainty region at one shot becomes even harder as size grows.	Connection between this small problem and the original problem requires an expert.

In the rest of this subsection, we prove that, under a general theoretical framework, the solutions generated by our algorithms are at least as desirable to the DM as the solutions generated by many other robust optimization algorithms. The solution that a robust optimization technique returns is an optimal solution of a tractable robust counterpart for the LP problem with uncertainty. In the first theorem, we prove that given any optimal solution x^* of a classical robust optimization problem, there exists a concave utility function U such that the problem

$$\begin{array}{ll} \max & g(x) := U(b - Ax) \\ \text{s.t.} & a_i^\top x \leq b_i, \;\; i \in \{1, \ldots, m\}. \end{array} \tag{4}$$

[1]We quote an axiom for robust optimization from the book [6]: "The decision maker is fully responsible for consequences of the decisions to be made when, and only when, the actual data is within the prespecified uncertainty set".

[2]see problem (5) and the explanation before equation (2).

has a unique optimal solution x^*. We emphasize again that problem (3) is not explicitly available. Next, we prove that a proper utility function always exists.

Many classical robust optimization models and their approximations can be written as follows

$$\begin{array}{ll} \max & c^\top x \\ \text{s.t.} & a_i^\top x + f_i(x) \leq b_i, \ \ i \in \{1,\dots,m\}, \end{array} \tag{5}$$

where $f_i(x)$, $i \in \{1,\dots,m\}$, is a convex function such that $f_i(x) \geq 0$ for all feasible x. If the convex uncertainty set $\mathcal{A}_i$ is known for each $i \in \{1,\dots,m\}$ and $a_i \in \mathcal{A}_i$, then we have $f_i(x) := \sup_{\tilde{a}\in\mathcal{A}_i} \tilde{a}^\top x - a_i^\top x$. By changing $f_i(x)$, different formulations can be derived. In the following we mention some examples. Assume that for each entry A_{ij} of matrix A we have $A_{ij} \in [a_{ij} - \hat{a}_{ij}, a_{ij} + \hat{a}_{ij}]$. It can easily be seen [14] that the classical robust optimization problem is equivalent to (5) for $f_i(x) = \hat{a}_i^\top |x|$. For the second example, assume that $A \in \{A \ : \ \left\|M(\text{vec}(A) - \text{vec}(\bar{A}))\right\| \leq \Delta\}$ for a given $\bar{A}$ where $\|.\|$ is a general norm and M is an invertible matrix. $\text{vec}(A)$ is a vector in $\mathbb{R}^{mn\times 1}$ created by stacking the columns of A on top of one another. It is proved in [11] that many approximate robust optimization models can be formulated by changing the norm. It is also proved in [11] that this robust optimization model can be formulated as (5) by $f_i(x) = \Delta \left\|M^{-T}x_i\right\|_*$, where $\|.\|_*$ is the dual norm and $x_i \in \mathbb{R}^{mn\times 1}$ is a vector that contains x in entries $(i-1)n+1$ through in, and zero everywhere else.

Now, utilizing Karush-Kuhn-Tucker (KKT) theorem, we establish the following theorem.

Theorem 3.1. *Assume that* (5) *has Slater points. Then, for every optimal solution x^* of* (5)*, there exists a concave function $g(x)$ (or equivalently $U(s)$) such that x^* is the unique solution of* (4)*.*

Proof. For the optimality condition of (5) we have: There exists $\lambda \in \mathbb{R}^m_+$ such that

$$\begin{aligned} & c - \sum_{i=1}^{m} \lambda_i (a_i + \nabla f_i(x)) = 0 \\ & \lambda_i (a_i^\top x + f_i(x) - b_i) = 0, \quad i \in \{1,\dots,m\}. \end{aligned} \tag{6}$$

Since the Slater condition holds for (5), optimality conditions (6) are necessary and sufficient. Let x^* be an optimal solution of (5), and let $J \subseteq \{1,\dots,m\}$ denote the set of indices for which $\lambda_i \neq 0, i \in J$. Let us define $g(x)$ as follows:

$$g(x) := c^\top x + \sum_{i\in J} \mu_i \ln(b_i + t_i - a_i^\top x - f_i(x)), \tag{7}$$

where $t_i > 0, i \in J$, are arbitrary numbers. We claim that $g(x)$ is concave. $b_i + t_i - a_i^\top x - f_i(x)$ is a concave function and $\ln(x)$ is increasing concave, hence $\ln(b_i + t_i - a_i^\top x - f_i(x))$ is a concave function for $i \in \{1,\dots,m\}$. $g(x)$ is the summation of an affine function and some concave functions and so is concave. The gradient of $g(x)$ is

$$\nabla g(x) = c - \sum_{i\in J} \frac{\mu_i}{b_i + t_i - a_i^\top x - f_i(x)} (a_i + \nabla f_i(x)). \tag{8}$$

Now define $\mu_i, i \in J$, as

$$\mu_i := \lambda_i \left[b_i + t_i - a_i^\top x^* - f_i(x^*) \right]. \tag{9}$$

Using (9) and comparing (8) and (6), we conclude that x^* is a solution of (4), as we wanted. Uniqueness is because $g(x)$ is strictly convex. □

The above argument proves the existence of a suitable utility function. A related, purely theoretical question is that can we construct such a utility function without having a solution of (6)? In the following, we construct a function with objective value arbitrarily close to the objective value of (5). Assume that strong duality holds for (5). Let us define $g(x) := c^\top x + \mu \sum_{i=1}^m \ln(b_i - a_i^\top x - f_i(x))$ and assume that $\hat{x}$ is the maximizer of $g(x)$. We have

$$\nabla g(\hat{x}) = c - \sum_{i=1}^{m} \frac{\mu}{b_i - a_i^\top \hat{x} - f_i(\hat{x})}(a_i + \nabla f_i(\hat{x})) = 0. \tag{10}$$

This means that $\hat{x}$ is the maximizer of the Lagrangian of the problem in (5), $L(\lambda, x)$, for $\hat{\lambda}_i := \mu/(b_i - a_i^\top \hat{x} - f_i(\hat{x}))$, $i \in \{1, \dots, m\}$. So by strong duality, we have

$$\begin{aligned} c^\top x^* \le L(\hat{\lambda}, \hat{x}) &= c^\top \hat{x} + \sum_{i=1}^{m} \frac{\mu}{b_i - a_i^\top \hat{x} - f_i(\hat{x})}(b_i - a_i^\top \hat{x} - f_i(\hat{x})) \\ &= c^\top \hat{x} + m\mu. \end{aligned} \tag{11}$$

(11) shows that by choosing μ small enough, we can construct $g(x)$ such that the optimal objective value of (4) is arbitrarily close to the optimal objective value of (5).

Note that many other approaches to robust optimization and decision making under uncertainty (including the generalized robust counterpart introduced by Ben-Tal and Nemirovski [6], and the approach of Iancu and Trichakis [33] using the notion of pareto robust optimization) can be included as a special case of our framework. A good starting point to prove the existence of a utility function is to start with indicator functions of sets encoding feasibility conditions. This approach first leads to utility functions that are not continuous; however, as we showed above, these functions can be smoothed by use of barriers which then lead to differentiable utility functions with desired properties.

To illustrate the above points, we can prove the stronger version (from the viewpoint of optimal solution sets) of Theorem 3.1.

Theorem 3.2. *Assume that* (5) *has Slater points. Then, there exists a concave function $g(x)$ (or equivalently $U(s)$) such that the sets of optimal solutions of* (4) *and* (5) *are the same.*

Proof. Note that because $f_i(x) \ge 0$, $i \in \{1, \dots, m\}$, for all feasible points x, the feasible region of (5) is a subset of the feasible region of (4). As the objective function of (5) is linear, the set of optimal solutions of (5), denoted by X_{opt}, is a convex set. For an obvious choice, if we define the concave function

$$g(x) := \begin{cases} 0 & x \in X_{opt} \\ -\infty & o.w. \end{cases},$$

then (4) has the same set of optimal solutions as (5). To show that there exists a continuous concave function, let us assume X_{opt} is represented by the following system

$$X_{opt} = \left\{ x \in \mathbb{R}^n : \bar{A}x = \bar{b},\ h_i(x) \le 0, i \in \{1, \dots, q\} \right\},$$

where $\bar{A}x = \bar{b}$ defines an affine subspace, and $h_i(x)$, $i \in \{1, \ldots, q\}$, are continuous convex functions. Consider the following function:

$$g(x) := \min \left\{0, -h_1(x), \cdots, -h_q(x), -\|\bar{A}x - \bar{b}\|^2\right\}.$$

This function is concave because it is the minimum of concave functions. The maximum of the function is 0 and is achieved only on X_{opt}. Therefore, the sets of optimal solutions of (4) and (5) are the same. □

We can make a connection between the feasible slack vectors of an LP and the notion of weighted-analytic-centers. There are strong justifications for using weight space (w-space) instead of s-space that we will see when we design the algorithms. Besides, by using the notion of weighted center, we benefit from differentiability and smoothness of our functions in our formulations. Weight-space and weighted-analytic-centers approach embeds a "highly differentiable" structure into the algorithms. Such tools are extremely useful in both the theory and applications of optimization. In contrast, classical robust optimization and other competing techniques usually end up delivering a final solution where differentiability cannot be expected; this happens because their potential optimal solutions located on the boundary of the domain of some of the structures defining the problem.

3.2. **Definition of weighted center.** For every $i \in \{1, 2, \ldots, m\}$, let $\mathcal{F}_i$ be a closed convex subset of $\mathbb{R}^n$ such that $\mathcal{F} := \bigcap_{i=1}^m \mathcal{F}_i$ is bounded and has nonempty interior.
Let $F_i : \operatorname{int}(\mathcal{F}_i) \to \mathbb{R}$ be a self-concordant barrier for $\mathcal{F}_i$, $i \in \{1, 2, \ldots, m\}$ (For a definition of self-concordant barrier functions see [53]). For every $w \in \mathbb{R}^m_{++}$, we define the w-center of $\mathcal{F}$ as

$$\arg\min \left\{ \sum_{i=1}^m w_i F_i(x) : x \in \mathcal{F} \right\}.$$

Consider the special case when each $\mathcal{F}_i$ is a closed half-space in $\mathbb{R}^n$. Then the following result is well-known (see for example [41, 46, 47, 1]).

Theorem 3.3. *Suppose for every $i \in \{1, 2, \ldots, m\}$, $a^{(i)} \in \mathbb{R}^n \setminus \{0\}$ and $b_i \in \mathbb{R}$ are given such that:*

$$\mathcal{F} := \left\{ x \in \mathbb{R}^n : \langle a^{(i)}, x\rangle \le b_i, \forall i \in \{1, 2, \ldots, m\} \right\},$$

is bounded and $\operatorname{int}(\mathcal{F})$ is nonempty. Also, for every $i \in \{1, 2, \ldots, m\}$ define $F_i(x) := -\ln(b_i - \langle a^{(i)}, x\rangle)$. Then for every $w \in \mathbb{R}^m_{++}$, there exists a unique w-center in the interior of $\mathcal{F}$, $x(w)$. Conversely, for every $x \in \operatorname{int}(\mathcal{F})$, there exists some weight vector $w(x) \in \mathbb{R}^m_{++}$ such that x is the unique $w(x)$-center of $\mathcal{F}$.

Define the following family of convex optimization problems:

$$\begin{array}{llll} \min & -\sum_{i=1}^m w_i \ln(s_i) & \min & \langle b, y\rangle - \sum_{i=1}^m w_i \ln(y_i) \\ \text{s.t.} & Ax + s = b & \text{s.t.} & A^\top y = 0 \end{array} \tag{12}$$

For every weight vector $w > 0$, the objective functions of the above problems are strictly convex on their domains. Moreover, the objective function values tend to $+\infty$ along any sequence of their interior points (strictly feasible points), converging to a point on their respective boundary. So, the above problems have minimizers in the interior of their respective feasible regions. Since the objective functions are strictly convex, the minimizers are unique. Therefore, for every given $w > 0$, the above problems have unique solutions $(x(w), s(w))$ and $y(w)$. If we write the optimality conditions for both problems, there exist $\bar{y} \in \mathbb{R}^m$ for the first problem and $\bar{s} \in \mathbb{R}^m, \bar{x} \in \mathbb{R}^n$ for the second problem such that

$$\begin{aligned} Ax(w) + s(w) &= b \qquad & A\bar{x} + \bar{s} &= b \\ A^\top \bar{y} &= 0 \qquad & A^\top y(w) &= 0 \\ S(w)\bar{y} &= w, \qquad & \bar{S}y(w) &= w. \end{aligned} \tag{13}$$

By the above uniqueness discussion, we must have $\bar{y} = y(w)$, $\bar{s} = s(w)$, and $\bar{x} = x(w)$, and these two systems are actually the same. These solutions can be used to define many *primal-dual weighted-central-paths* as the solution set $\{(x(w), y(w), s(w)) : w > 0\}$ of the following system of equations and strict inequalities:

$$\begin{aligned} &Ax + s = b, \ s > 0, \\ &A^\top y = 0, \\ &Sy = w, \end{aligned} \tag{14}$$

When we set $w := te$, $t > 0$, we obtain the usual primal-dual central-path. Figure 1 illustrates some weighted central paths.

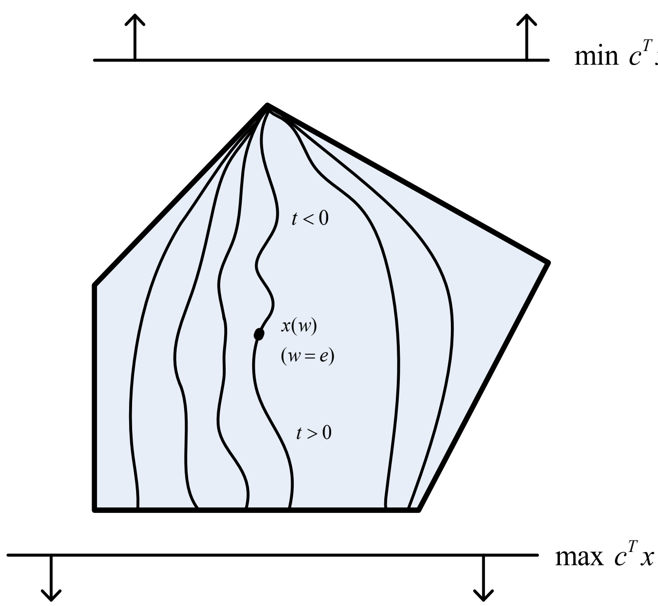

FIGURE 1. Primal-dual central paths.

For every given weight vector w, $(x(w), y(w), s(w))$ is obtained uniquely from (14) and $x(w)$ is called the *weighted center* of w. We may also refer to $(x(w), y(w), s(w))$ as the weighted center of w. For every given $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$, $y > 0$, that satisfy the above system, w and $s(w)$ are obtained uniquely. However, for a given $x \in \mathbb{R}^n$, there are many weight vectors w that may give x as the w-center of the corresponding polytope. Without loss of generality, we restrict ourselves to the weights on the unit simplex, i.e., we consider weighted center (x, y, s) corresponding to weight vectors w such that $\sum_{i=1}^m w_i = 1$. We call this simplex of weight vectors W:

$$W := \{w \in \mathbb{R}^m : w > 0,\ e^\top w = 1\}.$$

We will show that weight vectors in W are enough to represent the feasible region (a special case can be $w = \frac{1}{m}e$). We define the following notion for future reference:

Definition 3.1. *A vector $s \in \mathbb{R}^m$ or $y \in \mathbb{R}^m$ is called* centric *if there exists x such that (x, y, s) satisfies* (14) *for a weight vector $w > 0$ where $e^\top w = 1$.*

Let s and y be centric. First, we note that the simplex of the weight vectors can be divided into regions of constant y-vector (W_y) and constant s-vector (W_s).

$$W_y := \{w \in W :\ y(w) = y\}, \quad W_s := \{w \in W :\ s(w) = s\}. \tag{15}$$

By using Lemma B.2, if $(\hat{x}, \hat{y}, \hat{s})$ is the solution of system (14) corresponding to the weight vector $\hat{w} \in W$, and $\bar{y} > 0$ is any centric y-vector, then $(\hat{x}, \bar{y}, \hat{s})$ is the solution of system (14) corresponding to the weight vector $\bar{Y}(\hat{Y})^{-1}\hat{w}$. This means that for every centric vector $\hat{s}$ and any centric vector y, $\hat{S}y$ is a weight vector in the simplex.

For every pair of centric vectors s and y, W_s and W_y are convex. To see this, let $(x, \bar{y}, s)$ and (x, y, s) be the weighted centers of $\hat{w}$ and w. Then, it is easy to see that for every $\beta \in [0, 1]$, $(x, \beta\bar{y} + (1-\beta)y, s)$ is the weighted center of $\beta\hat{w} + (1-\beta)w$. Various properties of W_s and W_y are studied in Appendix B, but the following simple examples make the geometry of W_s and W_y clearer. We present two examples with $m = 3,\ n = 1$.

Example 3.1. *For the first example, let $b := [1\ 0\ 0]^\top$ and $A := [1\ \ -1\ \ -1]^\top$. By using* (14)*, the set of centric s-vectors is $B_s = \{[(1-x),\ x,\ x]^\top : x \in (0, 1)\}$. The set of centric y-vectors is specified by solving $A^\top y = 0$ and $b^\top y = 1$, while $y > 0$. We can see that in this example, as shown in Figure 2, W_s regions are parallel line segments while W_y regions are line segments which all intersect at $[1\ 0\ 0]^\top$. For the second example, let $A := [1\ \ -1\ \ 0]^\top$ and $b := [1\ 0\ 1]^\top$. W_s and W_y regions are shown in Figure 3 derived by solving* (14)*. As can be seen, this time W_y regions are parallel line segments and W_s regions are line segments which intersect at the point $[0\ 0\ 1]^\top$.*

These examples show that the affine hulls of W_{y^1} and W_{y^2} might not intersect for two centric y-vectors y^1 and y^2. This is also true for the affine hulls of W_{s^1} and W_{s^2} for two centric s-vectors s^1 and s^2.

Example 3.2. *For the second example, let $A := [3\ \ -3\ \ -2]^\top$ and $b := [1\ 1\ 0]^\top$. W_s and W_y regions are shown in Figure 4, derived by solving* (14)*. In this example, none of W_y regions, W_s regions, or their affine hulls intersect in a single point.*

4. Algorithms

In this section, we develop some cutting-plane algorithms which find an optimal solution for the DM, using the facts we established in the previous sections. As we mentioned in Section 3, we assume

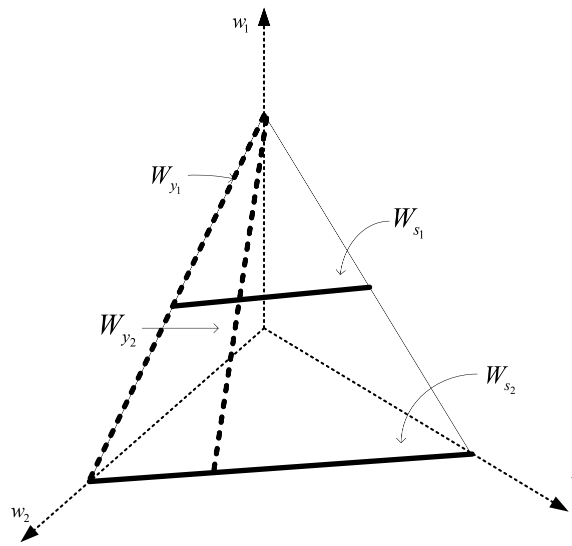

FIGURE 2. W_s and W_y regions for the first example in Example 3.1.

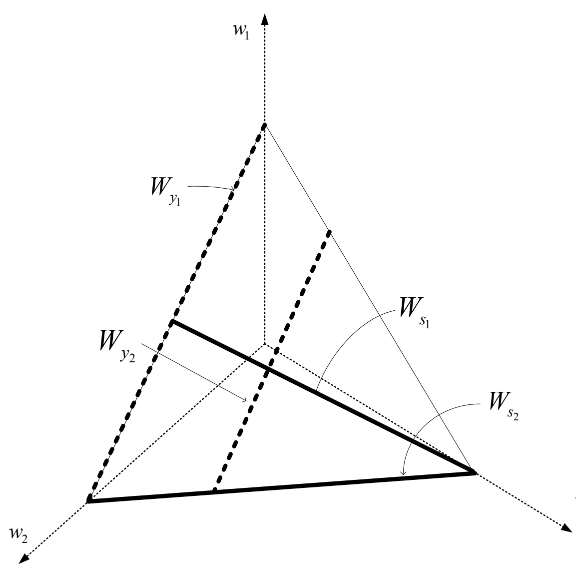

FIGURE 3. W_s and W_y regions for the second example in Example 3.1.

that the DM's preferences, knowledge, wisdom, expertise, etc. can be modeled in principle by a utility function (as a function of the slack variables s), i.e., $U(s)$, and our problem is to maximize this utility function over the set of centric (Definition 3.1) s-vectors B_s. (Of course, we do not assume to have access to this function U, except through our limited interactions with the DM.) Therefore, our

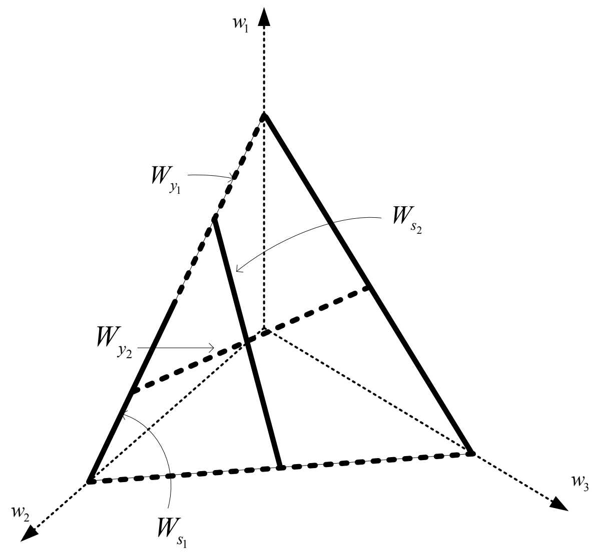

FIGURE 4. W_s and W_y regions for Example 3.2.

problem becomes

$$\begin{array}{ll} \max & U(s) \\ s.t. & s \in B_s. \end{array} \tag{16}$$

In the following, we denote an optimal solution of (16) with s^{opt}. In many applications, it is possible to capture choices with concave, quasi-concave, or nondecreasing utility functions. We are going to start with the assumption of concave $U(s)$. We will see in Subsection 6.2 that the algorithm can easily be refined to be used for quasi-concave functions.

Suppose we start the algorithm from a point $w^0 \in \mathbb{R}^m$ with the corresponding s-vector $s^0 \in \mathbb{R}^m$. Using the idea of supergradient, we can introduce cuts in the s-space or w-space to shrink the set of s-vectors or w-vectors, such that the shrunken space contains an optimal point. In the following subsections, we discuss these algorithms in the s-space and in the w-space. Our main algorithm is the one in the w-space, however, the s-space algorithm helps us understand the other better.

Our algorithms are based on the notion of supergradient of a concave function. Therefore, before stating the algorithms, we express a summary of the results we want to use. These properties are typically proven for convex functions in the literature [58, 18], however we can translate all of them to concave functions. It is a well-known fact that for a concave function $f : \mathbb{R}^n \to \mathbb{R}$, any local maximizer is also a global maximizer. If a strictly concave function attains its global maximizer, it is unique. The following theorem is fundamental for developing our cutting-plane algorithms.

Theorem 4.1. *Assume that $f : \mathbb{R}^n \to \mathbb{R}$ is a concave function and let $x^0 \in \operatorname{relint}(\operatorname{dom} f)$. Then there exists $g \in \mathbb{R}^n$ such that*

$$f(x) \le f(x^0) + g^\top (x - x^0), \quad \forall x \in \mathbb{R}^n. \tag{17}$$

If f is differentiable at x^0, then g is unique, and $g = \nabla f(x^0)$.

The vector g that satisfies (17) is called the *supergradient* of f at x^0. The set of all supergradients of f at x_0 is called the *superdifferential* of f at x^0, and is denoted $\partial f(x^0)$. By Theorem 4.1, if f is differentiable at x^0, then $\partial f(x^0) = \{\nabla f(x^0)\}$. The following lemma about supergradient, which is a simple application of the chain rule, is also useful.

Lemma 4.1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a concave function, and $D \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$ be arbitrary matrices. Then, $g(x) := f(Dx + b)$ is a concave function and we have:*

$$\partial g(x) = D^\top \partial f(Dx + b).$$

4.1. **Cutting-plane algorithm in the s-space.** Assume that we have a starting point s^0 and we can obtain a supergradient g^0 of U at s^0 from the DM. By using (17), for every s,

$$U(s) - U(s^0) \geq 0 \quad \Rightarrow \quad (g^0)^\top (s - s^0) \geq 0. \tag{18}$$

This means that all optimal points are in the half-space $(g^0)^\top (s - s^0) \geq 0$. So, by adding this cut, we can shrink the s-space and guarantee that there exists an optimal solution in the shrunken part. We can translate this cut to a cut in the x-space by using (14):

$$(g^0)^\top (s - s^0) = (g^0)^\top (b - Ax - b + Ax^0) = (g^0)^\top A(x^0 - x).$$

Using this equation, we consider the cut as a new constraint of the original problem; $(g^0)^\top Ax \leq (g^0)^\top Ax^0$. Let us define $a^{(m+1)} = (g^0)^\top A$ and $b_{m+1} = (g^0)^\top Ax^0$. We redefine $\mathcal{F}$ by adding this new constraint and find the weighted center for a chosen weight vector w^1. The step-by-step algorithm is as follows:

***S*-space Algorithm:**

- Step 1: Set $w^0 = \frac{1}{m}e$ and find the w^0-centers (x^0, y^0, s^0) with respect to $\mathcal{F}$.
- Step 2: Set $k = 0$, $A_0 = A$, $b^0 = b$, and $\mathcal{F}_0 = \mathcal{F}$.
- Step 3: If s^k satisfies the DM, return (x^k, y^k, s^k) and **stop**.
- Step 4: Set $k = k + 1$. Find g^{k-1}, the supergradient of $U(s)$ at s^{k-1}. Set

$$A_k = \begin{bmatrix} A_{k-1} \\ (g^{k-1})^\top A_{k-1} \end{bmatrix}, \qquad b^k = \begin{bmatrix} b^{k-1} \\ (g^{k-1})^\top A_{k-1}x^{k-1} \end{bmatrix},$$
$$\mathcal{F}_k := \left\{ x \in \mathbb{R}^n : \langle a_k^{(i)}, x\rangle \leq b_i^k, \forall i \in \{1, 2, \ldots, m + k\} \right\}. \tag{19}$$

- Step 5: Set $w_i^k = \frac{1}{m^2}$ for $i \in \{m + 1, \ldots, m + k\}$ and $w_i^k = \frac{1}{m} - \frac{k}{m^2}$ for $i \in \{1, \ldots, m\}$. Find the w^k-center (x^k, y^k, s^k) with respect to $\mathcal{F}_k$. Return to Step 3.

The logic behind Step 5 is that we want to give smaller weights to the new constraints than the original ones (however, our choices above are just examples; implementers should make suitable, practical choices that are tailored to their specific application). A main problem with this algorithm is that the dimension of the weight-space is very large and is increased by one every time we add a constraint. We show that this problem is solved by our w-space algorithm and the notion of driving factors in the following subsections.

4.2. **Cutting-plane algorithm in the w-space.** In this subsection, we consider the cuts in the w-space. To do that, we first try a natural way of extending the algorithm in the s-space to the one in the w-space. We show that this extension only works for a limited subset of utility functions. Then, we develop an algorithm applicable to all concave utility functions.

Like the s-space, we try to use the supergradients of $U(s)$. Let U_w denote the utility function as a function of w. From (14) we have $Ys = w$; so, $U_w(w) = U(s) = U(Y^{-1}w)$. If Y were constant for all weight vectors, $U_w(w)$ would be a concave function, and we could use Lemma 4.1 to find the supergradient at each point. The problem here is that Y is not necessarily the same for different weight vectors. Assume that we confine ourselves to weight vectors in the simplex W with the same y-vector (W_y). $U_w(w)$ is a concave function on W_y, so, we can define its supergradient. By Lemma 4.1, we conclude that $\partial U_w(w) = Y^{-1}\partial U(s)$ for all $w \in W_y$.

Suppose we start at w^0 with the weighted center (x^0, y^0, s^0). Let us define $g^{0w} := (Y^0)^{-1}g^0$, where g^0 is a supergradient of $U(s)$ at s^0. Then, from (17) we have,

$$U_w(w) \leq U_w(w^0) + (g^{0w})^\top (w - w^0), \quad \forall w \in W_{y^0}. \tag{20}$$

If we confine the weight-space to W_y, by the same procedure used for s-space, we can introduce cuts in the w-space using (20). The problem is that we do not have a proper characterization of W_y. On the other hand, U_w may not be a concave function on the whole simplex. Assume that s^{opt} is an optimal solution of (16), and $W_{s^{opt}}$ is the set of weight vectors in the simplex with s-vector s^{opt}. It is easy to see that $W_{s^{opt}}$ is convex. We also have the following lemma:

Lemma 4.2. *Let (x', y', s') be the weighted center corresponding to w', s^{opt} be an optimal solution of (16), and g' be the supergradient of $U(s)$ at s'. Then, $S^{opt}y'$ is in the half-space ${g'}_w^\top(w - w') \geq 0$, where $g'_w = {Y'}^{-1}g'$.*

Proof. We have ${g'}_w^\top(S^{opt}y' - w') = {g'}^\top {Y'}^{-1}(S^{opt}y' - S'y') = {g'}^\top(s^{opt} - s') \geq 0$. The last inequality follows from the fact that s^{opt} is a maximizer and g' is a supergradient of $U(s)$ at s'. □

The above lemma shows that using hyperplanes of the form ${g'}^\top {Y'}^{-1}(w - w')$, we can always keep a point from $W_{s^{opt}}$. Now, using the fact that $W_{s^{opt}}$ is convex and the above lemma, the question is: if we use a sequence of these hyperplanes, can we always keep a point from $W_{s^{opt}}$? A simpler question is: We start with w^0 and shrink the simplex W into the intersection of the half-space $(g^{0w})^\top(w - w^0) \geq 0$ and the simplex, say W_0. Then we choose an arbitrary weight vector w^1 with weighted center (x^1, y^1, s^1) from the shrunken space W_0. If g^1 is a supergradient of $U(s)$ at s^1, then we shrink W_0 into the intersection of W_0 and the half-space $(g^{1w})^\top(w - w^1) \geq 0$, where $g^{1w} = (Y^1)^{-1}g^1$, and call the last shrunken space W_1. Is it always true that a weight vector with s-vector s^{opt} exists in W_1? In the following, we show that this is true for some utility functions, but not true in general. We define a special set of functions that have good properties for cuts in the w-space, and the above algorithm works for them.

Definition 4.1. *A function $f : \mathbb{R}^m_{++} \to \mathbb{R}$ is called* Non-Decreasing under Affine Scaling (NDAS) *if for every $d \in \mathbb{R}^m_{++}$ we have:*

(1) $f(s) \leq \max\{f(Ds), f(D^{-1}s)\}, \quad \forall s \in \mathbb{R}^m_{++}$.

(2) If for a single $s^0 \in \mathbb{R}^m_{++}$ we have $f(s^0) \leq f(Ds^0)$, then $f(s) \leq f(Ds)$ for all $s \in \mathbb{R}^m_{++}$.

For every $t \in \mathbb{R}^m$ the function $f_1(s) := \sum_{i=1}^m t_i \log s_i$ is NDAS. Indeed, for every $s, d \in \mathbb{R}^m_{++}$ we have:

$$\begin{aligned} f_1(s) - f_1(Ds) &= -\sum_{i=1}^m t_i \log d_i, \\ f_1(s) - f_1(D^{-1}s) &= -\sum_{i=1}^m t_i \log \frac{1}{d_i} = \sum_{i=1}^m t_i \log d_i, \end{aligned}$$

and so we have $2f_1(s) = f_1(Ds) + f_1(D^{-1}s)$. The second property is also easy to verify and the function is NDAS. $f_1(s)$ is also important due to its relation to a family of classical utility functions in mathematical economics: Cobb-Douglas production function which is defined as $U_{cd}(s) = \prod_{i=1}^m s_i^{t_i}$, where $t \in \mathbb{R}^m_{++}$. Usage of this function to simulate problems in economics goes back to at least the 1920's. Maximization of $U_{cd}(s)$ is equivalent to the maximization of its logarithm which is equal to $f_1(s) = \ln(U_{cd}(s)) = \sum_{i=1}^m t_i \log s_i$. Authors in [35] considered the Cobb-Douglas utility function to present an algorithm for evaluating and ranking items with multiple attributes. [35] is related to our work as the proposed algorithm is a cutting-plane one. [35] also used the idea of weight-space as the utility function is the weighted sum of the attributes. However, our algorithm uses the concept of weighted analytic center which is different. Now, we have the following proposition.

Proposition 4.1. *Assume that $U(s)$ is a NDAS concave function. Let (x^0, y^0, s^0) and (x^1, y^1, s^1) be the weighted centers of w^0 and w^1, and g^0 and g^1 be the supergradients of $U(s)$ at s^0 and s^1, respectively. Then we have*

$$\Big\{w: \ (g^{0w})^\top (w - w^0) \geq 0, \ (g^{1w})^\top (w - w^1) \geq 0\Big\} \cap W_{s^{opt}} \neq \ \phi,$$

where $g^{0w} = (Y^0)^{-1} g^0$ and $g^{1w} = (Y^1)^{-1} g^1$.

Proof. Consider the weight vectors $Y^0 s^{opt}$ and $Y^1 s^{opt}$. Our two hyperplanes are

$$\begin{aligned} P_0 &:= \{w: \ \ (g^0)^\top (Y^0)^{-1}(w - Y^0 s^0) = 0\}, \\ P_1 &:= \{w: \ \ (g^1)^\top (Y^1)^{-1}(w - Y^1 s^1) = 0\}. \end{aligned}$$

By Lemma 4.2, $Y^0 s^{opt}$ is in the half-space $(g^0)^\top (Y^0)^{-1}(w - Y^0 s^0) \geq 0$ and $Y^1 s^{opt}$ is in the half-space $(g^1)^\top (Y^1)^{-1}(w - Y^1 s_1) \geq 0$. If one of these two points is also in the other half-space, then we are done. So, assume that

$$(g^0)^\top (Y^0)^{-1}(Y^1 s^{opt} - Y^0 s^0) < 0 \ \text{ and } \ (g^1)^\top (Y^1)^{-1}(Y^0 s^{opt} - Y^1 s^1) < 0$$

(we are seeking contradiction), which is equivalent to

$$(g^0)^\top ((Y^0)^{-1} Y^1 s^{opt} - s^0) < 0 \ \text{ and } \ (g^1)^\top ((Y^1)^{-1} Y^0 s^{opt} - s^1) < 0. \tag{21}$$

Using (20) and (21) we conclude that

$$\begin{aligned} &U((Y^0)^{-1} Y^1 s^{opt}) < U(s^0) \leq U(s^{opt}) \ \text{ and} \\ &U((Y^1)^{-1} (Y^0) s^{opt}) < U(s^1) \leq U(s^{opt}). \end{aligned}$$

However, note that $(Y^0)^{-1}Y^1 = ((Y^1)^{-1}Y^0)^{-1}$ and this is a contradiction to Definition 4.1. So (21) is not true and at least one of $Y^0 s^{opt}$ and $Y^1 s^{opt}$ is in

$$\{w : \ (g^{0w})^\top (w - w^0) \geq 0, \ (g^{1w})^\top (w - w^1) \geq 0\}.$$

$\square$

By Proposition 4.1, using the first two hyperplanes, the intersection of the shrunken space and $W_{s^{opt}}$ is not empty. Now, we want to show that we can continue shrinking the space and have nonempty intersection with $W_{s^{opt}}$.

Proposition 4.2. *Assume that $U(s)$ is a NDAS concave function. Let (x^i, y^i, s^i) be the weighted centers of w^i, $i \in \{0, \dots, k\}$, and g^i be the supergradients of $U(s)$ at s^i. Let us define*

$$W^i := \left\{ w : \ (g^{iw})^\top (w - w^i) \geq 0 \right\} \cap W,$$

where $g^{iw} = (Y^i)^{-1} g^i$. Assume we picked the points such that

$$w^i \in \operatorname{relint} \left(\bigcap_{j=0}^{i-1} W^j \right), \qquad i \in \{1, \dots, k\}. \tag{22}$$

Then we have

$$\left(\bigcap_{j=0}^{k} W^j \right) \ \cap \ W_{s^{opt}} \neq \ \phi, \tag{23}$$

where s^{opt} is an optimal solution of (16).

Proof. Among the three representations of W_s were given in (52), we use the second one in the following. If (23) is not true, then the following system is infeasible:

$$\begin{aligned} &A^\top (S^{opt})^{-1} w = 0, \quad e^\top w = 1, \quad w \geq 0, \\ &(g^{iw})^\top (w - w^i) \geq 0, \quad i \in \{0, \dots, k\}. \end{aligned} \tag{24}$$

By Farkas' Lemma, there exist $v \in \mathbb{R}^n$, $p \in \mathbb{R}$, and $q \in \mathbb{R}^k_+$ such that:

$$\begin{aligned} &(S^{opt})^{-1} A v + pe - \sum_{i=0}^{k} q_i g^{iw} \geq 0 \quad \Leftrightarrow \quad Av + ps^{opt} - \sum_{i=0}^{k} q_i S^{opt} (Y^i)^{-1} g^i \geq 0, \\ &p - \sum_{i=0}^{k} q_i (g^{iw})^\top w^i < 0 \quad \Leftrightarrow \quad p - \sum_{i=0}^{k} q_i (g^i)^\top s^i < 0. \end{aligned} \tag{25}$$

Now for each $j \in \{0, \dots, k\}$, we multiply both sides of the first inequality in (25) with $e^\top Y^j$, then we have:

$$\begin{aligned} &p - \sum_{i=0}^{k} q_i (s^{opt})^\top Y^j (Y^i)^{-1} g^i \geq 0, \quad \forall j \in \{0, \dots, k\}, \\ &p - \sum_{i=0}^{k} q_i (g^i)^\top s^i < 0, \end{aligned} \tag{26}$$

where we used the facts that $e^\top Y^j Av = (A^\top y^j)^\top v = 0$ and $e^\top Y^j s^{opt} = 1$. If we multiply the first set of inequalities in (26) with -1 and add it to the second one we have

$$q_j(g^j)^\top(s^{opt} - s^j) + \sum_{i\neq j} q_i(g^i)^\top(Y^j(Y^i)^{-1}s^{opt} - s^i) < 0, \tag{27}$$

for all $j \in \{0,\ldots,k\}$. $q \in \mathbb{R}^k_+$ and $(g^j)^\top(s^{opt} - s^j) \geq 0$ by supergradient inequality. Hence, from (27), for each $j \in \{0,\ldots,k\}$, there exists $\phi_j \in \{0,\ldots,k\}\backslash\{j\}$ such that $(g^{\phi_j})^\top(Y^j(Y^{\phi_j})^{-1}s^{opt} - s^{\phi_j}) < 0$ which, using (18), means $U(Y^j(Y^{\phi_j})^{-1}s^{opt}) < U(s^{\phi_j}) \leq U(s^{opt})$. Therefore, by the first property of NDAS functions, we must have

$$U(Y^{\phi_j}(Y^j)^{-1}s^{opt}) \geq U(s^{opt}). \tag{28}$$

Now, it is easy to see that there exists a sequence $j_1,\ldots,j_t \in \{0,\ldots,k\}$ such that $\phi_{j_i} = j_{i+1}$ and $\phi_{j_t} = j_1$. By using (28) and the second property of NDAS functions $t-1$ times we can write:

$$\begin{aligned} U(s^{opt}) &\leq U(Y^{j_2}(Y^{j_1})^{-1}s^{opt}) \\ &\leq U(Y^{j_3}(Y^{j_2})^{-1}Y^{j_2}(Y^{j_1})^{-1}s^{opt}) \\ &\leq \cdots \leq U(Y^{j_t}(Y^{j_{t-1}})^{-1}\cdots Y^{j_2}(Y^{j_1})^{-1}s^{opt}) \\ &= U(Y^{j_t}(Y^{j_1})^{-1}s^{opt}). \end{aligned} \tag{29}$$

However, we had $U(Y^{j_t}(Y^{j_1})^{-1}s^{opt}) = U(Y^{j_t}(Y^{\phi_{j_t}})^{-1}s^{opt}) < U(s^{opt})$ which is a contradiction to (29). This means the system (24) is feasible and we are done. □

Proposition 4.2 shows that the above-mentioned cutting-plane algorithm works for the NDAS functions. However, the conclusion of the proposition is not true for a general concave function. For a counter example, see Example C.1 in Appendix C. To be able to perform a cutting-plane algorithm in the w-space, we modify the definition of cutting hyperplanes. In the next two propositions, we introduce a new set of cutting-planes.

Proposition 4.3. *For every point $Y^0s^0 \in W$, there exists a hyperplane P passing through it such that:*
1- P contains all the points in W_{s^0}, and
2- P cuts W_{y^0} the same way as $(g^0)^\top(Y^0)^{-1}(w - Y^0s^0) = 0$ cuts it; the intersections of P and $(g^0)^\top(Y^0)^{-1}(w - Y^0s^0) = 0$ with W_{y^0} is the same, and the projections of their normals onto W_{y^0} have the same direction.

Proof. Assume that $w^0 = Y^0s^0$ is the point that is chosen and let u^0 be the normal vector to the desired hyperplane P. First, we want the hyperplane to contain W_{s^0}. This means that for all centric $\hat{y}$, the vector $S^0y^0 - S^0\hat{y}$ is on P, i.e., we have $(u^0)^\top S^0(y^0 - \hat{y}) = 0$. Since $A^\top(y^0 - \hat{y}) = 0$, we can put $u^0 = (S^0)^{-1}Ah^0$ with an arbitrary h^0 and we have:

$$(u^0)^\top S^0(y^0 - \hat{y}) = (h^0)^\top A^\top(S^0)^{-1}S^0(y^0 - \hat{y}) = 0.$$

Now, we want to find h^0 such that $(u^0)^\top(w - Y^0s^0)$ cuts W_{y^0} the same way as $(g^0)^\top(Y^0)^{-1}(w - Y^0s^0)$ cuts it. We actually want to find h^0 which satisfies the stronger property that $(u^0)^\top(w - Y^0s^0) = (g^0)^\top(Y_0)^{-1}(w - Y^0s^0)$ for all $w \in W_{y^0}$. All the points in W_{y^0} are of the form $Y^0\hat{s}$, so we must have $(u^0)^\top Y^0(\hat{s}-s^0) = (g^0)^\top(\hat{s}-s^0)$. Since $(\hat{s}-s^0)$ is in the range of A, this equation is true for every $\hat{s}$ if and only if:

$$(u^0)^\top Y^0 Ax = (g^0)^\top Ax \;\Rightarrow\; ((u^0)^\top Y^0 - (g^0)^\top)Ax = 0, \quad \forall x \in \mathbb{R}^n.$$

This means that $Y^0u^0 - g^0$ must be in $\mathcal{R}(A)^\perp = \mathcal{N}(A^\top)$, which means $A^\top(Y^0u^0 - g^0) = 0$. However, we had from above that $u^0 = (S^0)^{-1}Ah^0$ and hence:

$$A^\top Y^0 u^0 = A^\top g^0 \;\Rightarrow\; A^\top Y^0 (S^0)^{-1}Ah^0 = A^\top g^0 \;\Rightarrow\; h^0 = (A^\top Y^0(S^0)^{-1}A)^{-1}A^\top g^0. \tag{30}$$

So, the hyperplane with normal vector $u^0 = (S^0)^{-1}Ah^0$, where $h^0 = (A^\top Y^0(S^0)^{-1}A)^{-1}A^\top g^0$ has the required properties. Since this hyperplane cuts W_{y^0} the same way as $(g^0)^\top(Y^0)^{-1}(w - Y^0s^0)$ does, we conclude that $(u^0)^\top(Y^0s^{opt} - Y^0s^0) \geq 0$. Therefore, Y^0s^{opt} is in the half-space $(u^0)^\top(w - Y^0s^0) \geq 0$. $\square$

The normal of the hyperplane derived in Proposition 4.3 has a nice interpretation with respect to orthogonal projection and the primal-dual scaling $Y^{-1}S$. We have:

$$\begin{aligned}
u^0 &= (S^0)^{-1}A(A^\top Y^0(S^0)^{-1}A)^{-1}A^\top g^0 \\
&= (Y^0)^{-1/2}(S^0)^{-1/2} \\
&\quad \underbrace{[((Y^0)^{1/2}(S^0)^{-1/2}A)(A^\top Y^0(S^0)^{-1}A)^{-1}(A^\top(S^0)^{-1/2}(Y^0)^{1/2})]}_{\Pi}(Y^0)^{-1/2}(S^0)^{1/2}g_0 \\
&= (Y^0)^{-1/2}(S^0)^{-1/2}P(Y^0)^{-1/2}(S^0)^{1/2}g_0, \qquad (31)
\end{aligned}$$

where Π is the orthogonal projection onto the range of $(Y^0)^{1/2}(S^0)^{-1/2}A$. Note that a main benefit of the hyperplane in Proposition 4.3 is that when we choose a point Y^0s^0, we can cut away W_{s^0}. Now, we prove the following proposition which shows we can cut the simplex with a sequence of hyperplanes such that the intersection of their corresponding half-spaces contain a point from $W_{s^{opt}}$.

Proposition 4.4. *Assume that we choose the points $Y^0s^0, Y^1s^1 \in W$. The hyperplane P passing through Y^1s^1, with the normal vector $u^1 := (S^1)^{-1}Ah^1$, $h^1 = (A^\top Y^0(S^1)^{-1}A)^{-1}A^\top g^1$ satisfies the following properties:*
1- P contains all the points in W_{s^1}, and
2- $(u^1)^\top(Y^0s^{opt} - Y^1s^1) \geq 0$ for every feasible maximizer of $U(s)$.

Proof. As in the proof of Proposition 4.3, if we set $u^1 = (S^1)^{-1}Ah^1$, then the hyperplane contains all the points in W_{s^1}. To satisfy the second property, we want to find h^1 with the stronger property that

$$(u^1)^\top(Y^0\hat{s} - Y^1s^1) = (g^1)^\top(\hat{s} - s^1), \tag{32}$$

for all the centric $\hat{s}$. The reason is that we already have $(g^1)^\top(s_{opt} - s^1) \geq 0$. By the choice of $u^1 = (S^1)^{-1}Ah^1$, for every centric y we have

$$(u^1)^\top S^1 y = (h^1)^\top A^\top (S^1)^{-1}S^1 y = (h^1)^\top A^\top y = 0.$$

So, we have $(u^1)^\top Y^1 s^1 = (u^1)^\top Y^0 s^1 = 0$ and we can continue the above equation as follows:

$$\begin{aligned}
(g^1)^\top(\hat{s} - s^1) &= (u^1)^\top(Y^0\hat{s} - Y^1s^1) = (u^1)^\top(Y^0\hat{s}) \\
&= (u^1)^\top(Y^0\hat{s} - Y^0s^1) \\
&= (u^1)^\top Y^0(\hat{s} - s^1).
\end{aligned}$$

Now we can continue in a similar way as in the proof of Proposition 4.3. Since $(\hat{s} - s^0)$ is in the range of A, we must have:

$$((u^1)^\top Y^0 - (g^1)^\top)Ax = 0, \quad \forall x \in \mathbb{R}^n.$$

By the same reasoning, we have:

$$A^\top Y^0 u^1 = A^\top g^1 \ \Rightarrow\ A^\top Y^0 (S^1)^{-1} A h^1 = A^\top g^1 \ \Rightarrow\ h^1 = (A^\top Y^0 (S^1)^{-1} A)^{-1} A^\top g^1. \tag{33}$$

So, the hyperplane with normal vector $u^1 = (S^1)^{-1} A h^1$, where $h^1 = (A^\top Y^0 (S^1)^{-1} A)^{-1} A^\top g^1$ has the required properties. $\square$

By Proposition 4.4, we can create a sequence of points and hyperplanes such that the corresponding half-spaces contain $Y^0 s^{opt}$. The algorithm is as follows:

W-space Algorithm:

- Step 1: Set $w^0 = \frac{1}{m} e$ and find the w^0-centers (x^0, y^0, s^0) with respect to $\mathcal{F}$.
- Step 2: Set $k = 0$, and $W_0 = W$.
- Step 3: If s^k satisfies the optimality condition, return (x^k, y^k, s^k) and **stop**.
- Step 4: Find g^k, the supergradient of $U(s)$ at s^k. Find h^k by solving the following equation

$$A^\top Y^0 (S^k)^{-1} A h^k = A^\top g^k. \tag{34}$$

- Step 5: Set $u^k = (S^k)^{-1} A h^k$ and $W_{k+1} = W_k \cap \{w :\ (u^k)^\top (w - w^k) \geq 0\}$. Pick a point w^{k+1} from W_{k+1} (see subsection 4.4) and find the w^{k+1}-center $(x^{k+1}, y^{k+1}, s^{k+1})$ with respect to $\mathcal{F}$. Set $k = k + 1$ and return to Step 3.

A clear advantage of this algorithm over the one in the s-space is that we do not have to increase the dimension of the w-space at each pass and subsequently we do not have to assign weights to the newly added constraints. So, the above algorithm is straightforward to implement. The convergence of the algorithm depends on the choice of w^{k+1} in Step 5, which we discuss in Subsection 4.4. We can also use the properties of the weighted center we derived in Appendix B to improve the performance of the algorithm.

4.3. **Some implementation ideas.**

In the previous subsections, we introduced an algorithm that is highly cooperative with the DM and proved many interesting features about it. In this subsection, we set forth some implementation ideas.

4.3.1. *Driving factors.*

As we mentioned, one of our main criticisms of classical robust optimization is that it is not practical to ask the DM to specify an m-dimensional ellipsoid for the uncertainty set. Our approach improves this situation by asking easier questions. The idea is similar to those used in the area of multi-criteria optimization. Consider the system of inequalities $Ax \leq b$ and the corresponding slack vector $s = b - Ax$ representing the problem. The DM might prefer to directly consider only a few factors that really matter, we call them *Driving Factors*. For example, the driving factors for a DM might be budget amount, profit, allocated human resources, etc. We can represent k driving factors by $(c^i)^\top x$, $i \in \{1, \ldots, k\}$, and the problem for the DM is to maximize the utility function $U((c^1)^\top x, \dots, (c^k)^\top x)$. Similar to the way we added the objective of the linear program to the constraints, we can add k constraints to problem and write (16) as:

$$\begin{aligned} \max \quad & U(\xi_1, \dots, \xi_k) \\ s.t. \quad & \xi_i = \hat{b}_i - (c^i)^\top x, \;\; \xi \geq 0, \;\; i \in \{1, \dots, k\} \\ & s = b - Ax, \;\; s \geq 0. \end{aligned} \tag{35}$$

As can be seen, the supergradient vector has only k nonzero elements which makes it much easier for the DM to specify it for $k \ll m$. k is usually very small and we can figure out approximate gradients by asking pair-wise comparison questions among the driving factors. However, it still may have the problem that the cutting plane algorithm is in a high-dimensional space and it might be slow. We can take one step further to resolve this difficulty.

Consider the following setup:

- A very large system of equalities and inequalities $Ax + s = b$, $s \geq 0$.
- A very small driving factor system in the space of ξ 'variables. Our goal is to solve problem

$$\begin{aligned} \max \quad & U(\xi_1, \dots, \xi_k) \\ s.t. \quad & \xi \in B_\xi, \end{aligned} \tag{36}$$

  where $B_\xi := \left\{\xi \in \mathbb{R}^k_+ : \xi_i = \hat{b}_i - (c^i)^\top x, i \in \{1, \dots, k\}, Ax \leq b\right\}$ and, without loss of generality, we may assume that $U(\xi_1, \dots, \xi_k)$ is a monotone non-decreasing function of $\xi_1, \dots, \xi_k$.
- A matrix C and a vector d such that $\xi = Cs + d$.
- A matrix $\bar{C}$ that translates a displacement in the driving factor space d_ξ to a displacement in the s-space d_s, i.e., $d_s = \bar{C} d_\xi$. Note that considering $d_\xi = C d_s$, there are infinite number of choices for $\bar{C}$ and finding the most effective one can be done by an optimizer/expert. The system $d_s = \bar{C} d_\xi$ actually showing how to change big space variables when there is a change in the driving factors. For example, assume that d_ξ requires decreasing workforce in a retail corporation with several branches. The change in the workforce for each individual branch, embedded in d_s, should be done by an optimizer/expert. However, to mention one possibility, we may consider the pseudoinverse of C as $\bar{C} := C^\top (CC^\top)^{-1}$.

(36) is a problem in a k-dimensional space (say, $k \in \{1, 2, \dots, 20\}$) and can be solved efficiently with our cutting-plane algorithms. Assume that at the k'th iteration we have a feasible slack vector s^k in the big space and a feasible slack vector ξ^k in the very small driving factor space, and by applying our algorithm in the driving factor space, we get a search direction d_ξ. Using $\bar{C}$ we get $d_s := \bar{C} d_\xi$ to update $s^{k+1} = s^k + \alpha d_s$ for an appropriate $\alpha > 0$. Algorithm in the w_ξ-space stops quickly, and we have a good estimate of the optimal weights in w-space.

The DM deals only with problem (36) directly, however, an optimizer/expert needs to translate the cuts (and the information extracted from the DM) in w_ξ-space into changes in the current assignment of weights in the big w-space and coordinate the search between the w_ξ-space and w-space (see Figure 5).

4.3.2. *Approximate gradients.* In the previous subsection, we derived a cutting-plane algorithm in the w-space. As can be seen from Propositions 4.3 and 4.4, for the implementation we need the

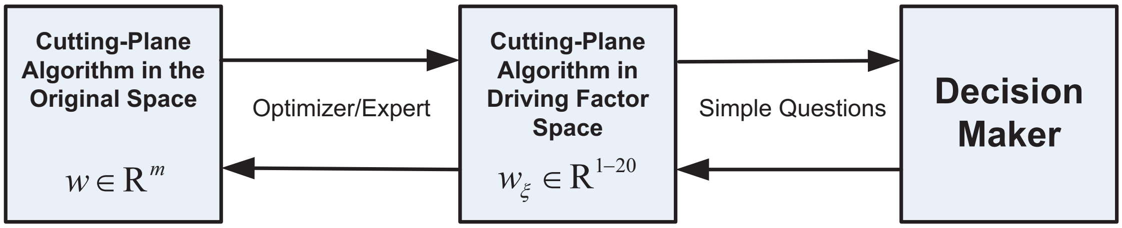

FIGURE 5. Diagram for the driving factor approach.

supergradients of the utility function $U(s)$. However, we usually do not have an explicit formula for $U(s)$ and our knowledge about it comes from the interaction with the DM. Supplying supergradient information on preferences (i.e., the utility function) might still be a difficult task for the DM. So, we have to simplify our questions for the DM and try to adapt our algorithm accordingly.

We try to derive approximate supergradients based on simple questions from the DM. The idea is similar to the one used by Arbel and Oren in [3]. Assume that $U(s)$ is differentiable which means the supergradient at each point is unique and equal to the gradient of the function at that point. Assume that the algorithm is at the point s. By Taylor's Theorem (first order expansion) for arbitrarily small scalars $\epsilon_i > 0$ we have:

$$\begin{aligned} u_i := U(s + \epsilon_i e_i) &\approx U(s) + \frac{\partial U(s)}{\partial s_i}\epsilon_i \\ \Rightarrow \frac{\partial U(s)}{\partial s_i} &\approx \frac{u_i - u_0}{\epsilon_i}, \qquad u_0 := U(s). \end{aligned} \tag{37}$$

Assume that we have $m+1$ points s and $s + \epsilon_i e_i$, $i \in \{1, \dots, m\}$. By the above equations, if we have the value of $U(s)$ at these points, we can find the approximate gradient. But in the absence of true utility function, we have to find these values through proper questions from the DM. Here, we assume that we can ask the DM about the relative preference for the value of the function at these $m+1$ points. For example, DM can use a method called Analytic Hierarchy Process (AHP) to assess relative preference. We use these relative preferences to find the approximate gradient.

Assume that the DM provides us with the priority vector p, then we have the following relationship between p and u_i's

$$\begin{aligned} & \frac{u_i}{u_j} = \frac{p_i}{p_j}, \qquad i, j \in \{0, \dots, m\}, \\ \Rightarrow \quad & \frac{u_i - u_0}{u_0} = \frac{p_i - p_0}{p_0}, \\ \Rightarrow \quad & u_i - u_0 = \beta_0 (p_i - p_0), \qquad \beta_0 := \frac{u_0}{p_0}. \end{aligned} \tag{38}$$

Now, we can substitute the values of $u_i - u_0$ from (38) into (37) and we have

$$\nabla U(s) = \beta_0 \begin{bmatrix} \frac{p_1 - p_0}{\epsilon_1} & \dots & \frac{p_m - p_0}{\epsilon_m} \end{bmatrix}^\top . \tag{39}$$

The problem here is that we do not have the parameter β_0. However, this parameter is not important in our algorithm because we are looking for normals to our proper hyperplanes and, as it can be seen in Propositions 4.3 and 4.4, a scaled gradient vector can also be used to calculate h^0 and h^1. Therefore, we can simply ignore β_0 in our algorithm.

Note that supergradients may be approximate due to the imperfect nature of the interaction with the DM. However, this issue gives us an opportunity to highlight another advantage of our approach when compared to classical robust optimization. Small errors in the determination of uncertainty regions in classical robust optimization may change the solution set rather dramatically or even make the underlying problem infeasible. In our approach, however flawed the supergradient information, as long as the halfspace defined by it contains an optimal solution or an approximately optimal solution, our algorithms are guaranteed to perform well. Therefore, this approximation in data is more serious in classical robust optimization that the DM needs to specify the whole uncertainty region, versus our approach wherein the supergradient is basically the normal to the halfspace used in reducing the set of weights under consideration.

4.4. **Convergence of the algorithm.** In this subsection, we focus on the convergence results for the w-space algorithm as our main algorithm. Note that no matter what the problem is, our cutting plane algorithm is applied to the unit simplex in the w-space. This makes the analysis straightforward and lets us use many results from the literature. On the other hand, if we use the driving factor scheme introduced above, our weight space has always dimension $k \leq 20$ and cutting plane algorithms become really fast. We define W_O as the set of all weights w such that the weighted center of w is acceptable to the DM. In all cutting plane algorithms, a "center" of the shrunken space must be chosen as the test point, which is crucial in the speed and convergence results of cutting plane algorithms.

Introduction of cutting-plane algorithms goes back at least to the 1960's and one of the first appealing ones is the center of gravity version [54]. The center of gravity algorithm has not been used in practice because computing the center of gravity, in general, is difficult. However, it is noteworthy due to its theoretical properties. For example, Grünbaum [29] proved that by using any cutting-plane through the center, more than $1/\exp(1) \approx 0.3678$ of the feasible set is cut out. Such results guarantee a geometric convergence rate with a sizeable constant [40, 28]. Many different types of centers have been proposed in the literature. A group of algorithms use the center of a specific localization set, which is updated at each step. One of them is the ellipsoid method [68] where the localization set is represented by an ellipsoid containing an optimal solution. Ellipsoid method can be related to our algorithm as we can use it to find the new weight vectors at each iteration. Another family of cutting-plane algorithms are based on *volumetric barriers* or *volumetric centers* [65, 66, 2]. Vaidya used the volumetric center to design a new algorithm for minimizing a convex function over a convex set [65]. More sophisticated algorithms have been developed based on Vaidya's volumetric cutting plane method [66, 2]. Let us summarize the above discussions about the three centering methods in a theorem:

Theorem 4.2. *Assume that at Step 5 of the w-space algorithm, we set w^{k+1} as one of the three following centers of W^{k+1}:*

- *the center of gravity,*
- *the center of the minimum volume ellipsoid containing W^{k+1},*

- *the volumetric center.*

Also assume that W_O contains a ball of radius ϵ. Then, using the driving factor approach with $k = O(1)$, the algorithm stops in $O\left(\ln\left(\frac{1}{\epsilon}\right)\right)$ iterations, with a solution acceptable to the DM.

The cutting-plane method which is most relevant to our algorithm and is more efficient in practice is the analytic-center cutting plane method (ACCPM), see [26] for a survey. In this method, the new point at each iteration is an approximate analytic center of the remaining polytope. The complexity of such algorithms has been widely studied in the literature [52, 25]. Let us prove the following theorem:

Theorem 4.3. *Assume that at Step 5 of the w-space algorithm, we calculate w^{k+1} as the analytic center of W^{k+1}. Also assume that W_O contains a ball of radius ϵ. Then, using the driving factor approach with $k = O(1)$, the algorithm stops in $O^*\left(\frac{1}{\epsilon^2}\right)$ iterations with a solution acceptable to the DM, where O^* means we ignore some logarithmic terms.*

Proof. We use existing proved results in [25] and [26]. Consider the proof in Section 4 of [26] for feasibility version of the analytic center cutting plane algorithm. The considered problem is finding $w \in C \cap [0,1]^n$, where C is a closed convex set and contains a ball of radius ϵ. C is also equipped with an oracle that returns a cutting plane $\langle a, w - \bar{w}\rangle \geq 0$ whenever $\bar{w} \notin C$. Note that we designed our approach so that our weight vectors are from the unit simplex, so $0 \leq w \leq e$. If we let $C = W_O$, it has all the required properties we mentioned. Therefore, all the discussions are carried forward and we have the $O^*\left(\frac{n^2}{\epsilon^2}\right)$ iterations bound. However, as we use the driving factor approach, we further have $n = k = O(1)$. Hence, for our approach, the complexity bound is $O^*\left(\frac{1}{\epsilon^2}\right)$. □

An alternative way to interpret the convergence properties above is after at most $O\left(\ln\left(\frac{1}{\epsilon}\right)\right)$ iterations (ellipsoidal center algorithm) or $O^*\left(\frac{1}{\epsilon^2}\right)$ iterations (analytic center algorithm) our current iterate is within an ϵ-neighborhood of a weight vector in W_O. [26] also has a discussion on how to modify the complexity if we use the approximate analytic center. Note that by some cut elimination and a complicated analysis, the authors in [4] proved a stronger convergence result for ACCPM that we mention for our approach in the following remark:

Remark 4.1. *If we use the cut elimination approach of* [4]*, we can improve the convergence result in Theorem 4.3 to $O\left(\ln^2\left(\frac{1}{\epsilon}\right)\right)$ iterations.*

5. Illustrative preliminary computational experiments

In this section, we present some numerical results to illustrate the performance of the algorithms in the w-space designed in Section 4. As we mentioned in previous sections, the utility function is not assumed to be explicitly available in our approach. So, for computational experiments with our algorithms, we maintain the same assumption. We choose a utility function; however, the algorithm does not "see" the utility function we chose. The algorithm interacts with the utility function only through the supergradient oracle. LP problems we use are chosen from the NETLIB library of LPs. Most of these LP problems are not in the format we have used throughout the paper which is the standard inequality form. Hence, we convert each problem to the standard equality form and then use the dual problem. In this section, the problem $\max\{(c^{(0)})^\top x : \ Ax \leq b^{(0)}\}$ is the converted one. In the following, we consider several numerical examples.

Example 1: In this example, we consider a simple problem of maximizing a quadratic function. Consider the ADLITTLE problem (in the converted form) with 139 constraints and 56 variables. We apply the algorithm to function $U_{ij}(s) = -(s_i - s_j)^2$ which makes two slack variables as close as possible. This function may not have any practical application, however, shows a simple example difficult to solve by classical robust optimization.

The stopping criteria is $\|g\| \leq 10^{-6}$. For U_{23} the algorithm takes 36 iterations and returns $U_{23} = -5 \times 10^{-11}$. For U_{34} the algorithm takes 35 iterations and returns $U_{34} = -2.4 \times 10^{-12}$.

Example 2: Consider the ADLITTLE problem and assume that three constraints $\{68, 71, 74\}$ are important for the DM. Assume that the DM estimates that there is 20 percent uncertainty in the RHS of these inequalities. We have $(b_{68}, b_{71}, b_{74}) = (500, 493, 506)$ and so the desired slack variables are around $(s_{68}, s_{71}, s_{74}) = (100, 98, 101)$. By using the classical robust optimization method that satisfies the worst case scenario, the optimal objective value is $obj_c = 1.6894 \times 10^5$.

Now assume that the following utility function represents DM's preferences:

$$U_1(s) = t_{68} \ln(s_{68}) + t_{71} \ln(s_{71}) + t_{74} \ln(s_{74}) + t_m \ln(s_m).$$

This function is a NDAS function that we defined in Definition 4.1. Assume that the DM set $t_m = 10$ and $t_{68} = t_{71} = t_{74} = 1$. By using our algorithm, we get the objective value of $obj_1 = 1.7137 \times 10^5$ with the slack variables $(s_{68}, s_{71}, s_{74}) = (82, 83, 132)$. As we observe, the objective value is higher than the classical robust optimization method while two of the slack conditions are not satisfied. However, the slack variables are close to the desired ones. If the DM sets $t_m = 20$, we get the objective value of $obj_2 = 1.9694 \times 10^5$ with the slack variables $(s_{68}, s_{71}, s_{74}) = (40, 41, 79)$. However, all the iterates might be interesting for the DM. The following results are also returned by the algorithm before the optimal one:

$$\begin{aligned} obj_3 &= 1.8847 \times 10^5, \quad (s_{68}, s_{71}, s_{74}) = (56, 58, 83), \\ obj_4 &= 1.7 \times 10^5, \quad (s_{68}, s_{71}, s_{74}) = (82, 84, 125). \end{aligned}$$

Now assume that the DM wants to put more weight on constraints 68 and 71 and so set $t_{68} = t_{71} = 2$, $t_{74} = 1$ and $t_m = 20$. In this case, the algorithm returns $obj_5 = 1.8026 \times 10^5$ with the slack variables $(s_{68}, s_{71}, s_{74}) = (82, 84, 64)$.

Example 3: In this example, we consider the DEGEN2 problem (in the converted form) with 757 constraints and 442 variables. The optimal solution of this LP is $obj_1 = -1.4352 \times 10^3$. Assume that constraints 245, 246, and 247 are important for the DM who wants them as large as possible, however, at the optimal solution we have $s(245) = s(246) = s(247) = 0$. The DM also wants the optimal objective value to be at least -1.5×10^3. As we stated before, we add the objective function as a constraint to the system. To have the objective value at least -1.5×10^3, we can add this constraint as $c^\top x = -1500 + s_{m+1}$. For the utility function, the DM can use the NDAS function

$$U(s) = \ln(s_{245}) + \ln(s_{246}) + \ln(s_{247}).$$

By running the algorithm for the above utility function, we get
$(s_{245}, s_{246}, s_{247}) = (7.75, 17.31, 17.8)$ with objective value $obj_2 \approx -1500$ after 50 iterations and
$(s_{245}, s_{246}, s_{247}) = (15.6, 27.58, 27.58)$ with $obj_3 \approx -1500$ after 100 iterations.

Example 4: In this example, we consider utility functions introduced at the end of Appendix A. Consider problem SCORPION with optimal objective value of $obj_1 = 1.8781 \times 10^3$. Assume that the uncertainty in constraints 211 to 215 are important for the DM and we have $\|\Delta b_i\|_1 = 0.7 b_i^{(0)}$, $i \in \{211, \dots, 215\}$, where Δb_i was defined in (45). Let $\hat{x}$ be the solution of MATLAB's LP solver, then we have $s_{211} = \cdots = s_{215} = 0$ which is not satisfactory for the DM. Besides, assume that the DM wants the objective value to be at least 1800. To satisfy that, we add the $(m+1)$th constraint as $s_{m+1} = -1800 + (c^{(0)})^\top x$ which guarantees $(c^{(0)})^\top x \geq 1800$. For the utility function, first we define $u_i(s_i)$, $i \in \{211, \dots, 215\}$ similar to Figure 7 with $\epsilon_i^1 = \|\Delta b_i\|_1 = 0.7 b_i^{(0)}$ and $\epsilon_i^2 = \infty$. So we have for $i \in \{211, \dots, 215\}$:

$$u_i(s_i) = \begin{cases} s_i & s_i < \|\Delta b_i\|_1 \\ \|\Delta b_i\|_1 & s_i \geq \|\Delta b_i\|_1. \end{cases} \tag{40}$$

Now, we can define $U(s) := \sum_{i=211}^{215} \ln u_i(s_i)$. By running the algorithm, the supergradient goes to zero after 65 iterations and the algorithm stops. Denote the solution by x^*, then the results are as follows:

$$\begin{aligned} &(c^{(0)})^\top x^* = 1800.3, \\ &b_{211}^{(0)} = 3.86, \quad b_{212}^{(0)} = 48.26, \quad b_{211}^{(0)} = 21.81, \quad b_{211}^{(0)} = 48.26, \quad b_{211}^{(0)} = 3.86, \\ &s_{211}^* = 3.29, \quad s_{212}^* = 19.47, \quad s_{211}^* = 7.39, \quad s_{211}^* = 16.97, \quad s_{211}^* = 3.24. \end{aligned} \tag{41}$$

Now, assume that the DM wants the objective value to be at least 1850 and the $(m+1)$th constraint becomes $s_{m+1} = -1850 + (c^{(0)})^\top x$. In this case, the norm of the supergradient reaches zero, after 104 iterations. The norm of supergradients versus the number of iterations are shown in Figure 6 for these two cases. Denote the solution after 100 iterations by $\bar{x}^*$, then we have:

$$\begin{aligned} &(c^{(0)})^\top \bar{x}^* = 1850, \\ &\bar{s}_{211}^* = 1.22, \quad \bar{s}_{212}^* = 16.74, \quad \bar{s}_{211}^* = 6.80, \quad \bar{s}_{211}^* = 14.54, \quad \bar{s}_{211}^* = 1.25. \end{aligned} \tag{42}$$

Let $\bar{x}$ be the returned value in the second case after 65 iterations. It is clearly not robust feasible; however, we can use bound (49) to find an upper bound on the probability of infeasibility. Assume that $N = 10$ and all the entries of Δb_i are equal. Then, bound (49) reduces to $B(N, \delta_i N)$, where $\delta_i = \frac{s_i}{\|\Delta b_i\|_1}$. The probabilities of infeasibility of $\bar{x}$ for constraints 211 to 215 are given in Table 5 (using bound (49)).

6. Extensions and conclusion

6.1. **Extension to Semidefinite Optimization (SDP).** Semidefinite Programming is a special case of Conic Programming where the cone is a direct product of semidefinite cones. Many convex optimization problems can be modeled by SDP. Since our method is based on a barrier function for a polytope in $\mathbb{R}^n$, it can be generalized and used as an approximation method for robust semidefinite

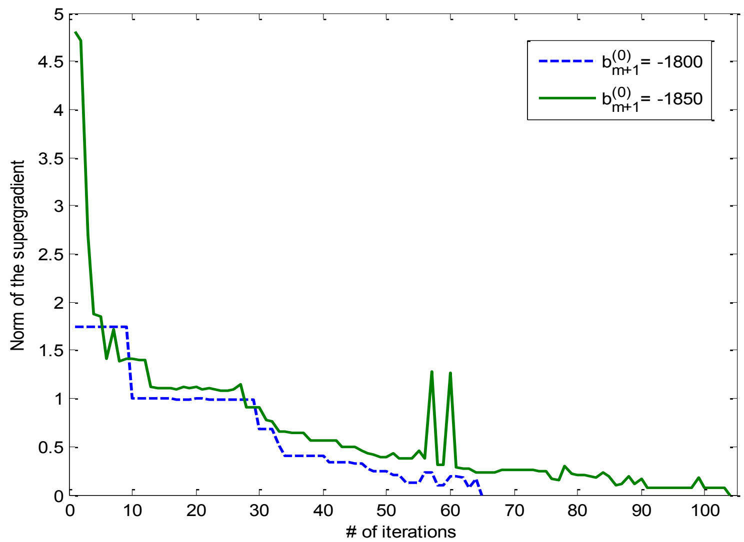

FIGURE 6. Norm of the supergradient versus the number of iterations for Example 5.

i	$\Pr(\langle a_j, \bar{x}\rangle > \tilde{b}_j)$
211	0
212	0.0827
213	0.0018
214	0.0866
215	0

TABLE 2. The probability of infeasibility of $\bar{x}$ for constraints 211 to 215.

programming that is NP-hard for ellipsoidal uncertainty sets. An SDP problem can be formulated as follows

$$\begin{aligned} \sup \quad & \langle \tilde{c}, x\rangle, \\ \text{s.t.} \quad & \sum_{j=1}^{t_i} A_i^{(j)} x_j + S_i = \tilde{B}_i, \quad \forall i \in \{1,2,...,m\}, \\ & S_i \succeq 0, \quad \forall i \in \{1,2,\dots,m\}, \end{aligned}$$

where $A_i^{(j)}$ and $\tilde{B}_i$ are symmetric matrices of appropriate size, and $\succeq$ is the Löwner order; for two square, symmetric matrices C_1 and C_2 with the same size, we have $C_1 \succeq C_2$ iff $C_1 - C_2$ is a semidefinite matrix. For every $i \in \{1,\dots,m\}$, define

$$\mathcal{F}_i := \{x \in \mathbb{R}^n : \sum_{j=1}^{t_i} A_i^{(j)} x_j \preceq \tilde{B}_i\}.$$

Assume that $\text{int}(\mathcal{F}_i) \neq \emptyset$ and let $F_i : \text{int}(\mathcal{F}_i) \to \mathcal{R}$ be a self-concordant barrier for $\mathcal{F}_i$. The typical self-concordant barrier for SDP is $F_i(x) = -\ln\left(\det\left(\tilde{B}_i - \sum_{j=1}^{t_i} A_i^{(j)} x_j\right)\right)$. Assume

$$\mathcal{F} := \bigcap_{i=1}^{m} \mathcal{F}_i$$

is bounded and its interior is nonempty. Now, as in the definition of the weighted center for LP, we can define a weighted center for SDP. For every $w \in \mathbb{R}^m_{++}$, we can define the weighted center as follows:

$$\arg\min\left\{\sum_{i=1}^{m} w_i F_i(x) : x \in \mathcal{F}\right\} \tag{43}$$

The problem with this definition is that we do not have many of the interesting properties we proved for LP. The main one is that the weighted centers do not cover the relative interior of the whole feasible region and we cannot sweep the whole feasible region by moving in the w-space. There are other notions of weighted centers that address this problem; however, they are more difficult to work with algorithmically. Extending the results we derived for LP to SDP can be a good future research direction to follow.

6.2. **Quasi-concave utility functions.** The definition of the quasi-concave function is as follows:

Definition 6.1. *A function $f : \mathbb{R}^n \to \mathbb{R}$ is* quasi-concave *if its domain is convex, and for every $\alpha \in \mathbb{R}$, the set*

$$\{x \in \text{dom} f \;:\; f(x) \geq \alpha\}$$

is also convex.

All concave functions are quasi-concave, however, the converse is not true. Quasi-concave functions are important in many fields such as game theory and economics. In microeconomics, many utility functions are modeled as quasi-concave functions. For differentiable functions, we have the following useful proposition:

Proposition 6.1. *A differentiable function f is quasi-concave if and only if the domain of f is convex and for every x and y in* dom$f$ *we have:*

$$f(y) \geq f(x) \quad \Rightarrow \quad (\nabla f(x))^\top (y - x) \geq 0 \tag{44}$$

(44) is similar to (18), which is the property of the supergradient we used to design our algorithms. The whole point is that for a differentiable quasi-concave function $U(s)$ and any arbitrary point s^0, the maximizers of $U(s)$ are in the half-space
$(\nabla U(s^0))^\top (s - s^0) \geq 0$. This means that we can extend our algorithms to differentiable quasi-concave utility functions simply by replacing supergradient with gradient, and all the results for s-space and w-space stay valid.

6.3. **Conclusion.** In this paper, we presented new algorithms in a framework for robust optimization designed to mitigate some of the major drawbacks of robust optimization in practice. Our algorithms have the potential of increasing the applicability of robust optimization. Some of the advantages of our new algorithms are:

(1) Instead of a single, isolated, and very demanding interaction with the DM, our algorithms interact continuously with the DM throughout the optimization process with more reasonable demands from the DM in each iteration. One of the benefits of our approach is that the DM "learns" what is feasible to achieve throughout the process. Another benefit is that the DM is more likely to be satisfied (or at least be content) with the final solution. Moreover, being personally involved in the production of the final solution, the DM bears some responsibility for it and is more likely to adapt it in practice.
(2) Our algorithms operate in the weight-space using only driving factors with the DM. This helps reduce the dimension of the problem, simplify the demands on the DM while computing the most important aspect of the problem at hand.
(3) Weight-space and weighted-analytic-centers approach embeds a "highly differentiable" structure into the algorithms. Such tools are extremely useful in both the theory and applications of optimization. In contrast, classical robust optimization and other competing techniques usually end up delivering a final solution where differentiability cannot be expected.

Note that many elements of our approach can be partly utilized in other approaches to robust optimization and decision making situations under uncertainty. Moreover, our work creates natural connections between robust optimization and multi-attribute utility theory, elicitation methods used in multi-criteria decision making problems and goal programming theory (see [39, 48, 34]).

Developing similar algorithms for semidefinite programming is left as a future research topic. As we explained in Subsection 6.1, we can define a similar notion of weighted center for SDP. However, these weighted centers do not have some of the properties we used for LP, and we may have to switch to other notions of weighted centers that are more difficult to work with algorithmically, and have fewer desired properties compared to the LP setting.

References

[1] J. Anderson, and S. Jibrin, An Interior Point Method for Linear Programming Using Weighted Analytic Centers, *Journal of the Arizona-Nevada Academy of Science*, 41.1 (2009): 1–7.
[2] K. M. Anstreicher, On Vaidya's Volumetric Cutting Plane Method for Convex Programming, *Mathematics of Operations Research*, 22 (1997), 63–89.
[3] A. Ardel, and S. Oren, Using Approximate Gradients in Developing an Interactive Interior Primal-dDal Multiobjective Linear Programming Algorithm, *European Journal of Operational Research*, 89 (1996), 202–211.
[4] D. S. Atkinson, and P. M. Vaidya, A cutting plane algorithm for convex programming that uses analytic centers, *Mathematical Programming* 69.1-3, (1995): 1–43.
[5] A. Ben-Tal, S. Boyd and A. Nemirovski, Extending Scope Of Robust Optimization: Comprehensive Robust Counterparts of Uncertain Problems, *Mathematical Programming*, 107 (2006) 63–89.
[6] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust Optimization*, Princeton Series in Applied Mathematics, (2009).
[7] A. Ben-Tal and A. Goryashko and E. Guslitzer and A. Nemirovski, Adjustable Robust Solutions Of Uncertain Linear Programs, *Mathematical Programming*, 99 (2004) 351–376.

[8] A. Ben-Tal and A. Nemirovski, Robust Solutions Of Linear Programming Problems Contaminated With Uncertain Data, *Math. Prog.* 88 (2000) 411–424

[9] A. Ben-Tal and A. Nemirovski, Robust Solutions Of Uncertain Linear Programs, *Operation Research Letters*, 25 (1999), 1–13.

[10] A. Ben-Tal and A. Nemirovski, Robust Convex Optimization, *Mathematics of Operations Research*, 23 (1998), 769–805.

[11] D. Bertsimas and D. Pachamanova and M. Sim, Robust Linear Optimization Under General Norms, *Operations Research Letters*, 32 (2004), 510–516.

[12] D. Bertsimas, and I. Popescu, Optimal Inequalities in Probability Theory- a Convex Optimization Approach, *SIAM J. Optim.*, 15 (2005), 780–804.

[13] D. Bertsimas and M. Sim, Tractable approximations to robust conic optimization problems, *Mathematical Programming*, June 2005.

[14] D. Bertsimas and M. Sim, The price of robustness, *Operations Research*, 52 (2004), 35–53.

[15] D. Bertsimas and M. Sim, Robust discrete optimization and network flows, *Math. Program.*, 98 (2003), 49–71.

[16] J.F. Bonnans, and A. Shapiro, *Perturbation analysis of optimization problems*, Springer, 2000.

[17] T. Bortfeld, T. C. Y. Chan, A. Trofimov, and J. N. Tsitsiklis, Robust management of motion uncertainty in intensity-modulated radiation therapy, *Oper. Res.* 56 (2008), 1461–1473.

[18] S. Boyd, and L. Vanderberghe, *Convex optimization*, Cambridge University Press, 2004.

[19] T.C. Chan and V. V. Mišić, Adaptive and robust radiation therapy optimization for lung cancer, *European J. Oper. Res.* 231 (2013), 745–756.

[20] M. Chu, Y. Zinchenko, S. G. Henderson, and M. B. Sharpe, Robust optimization for intensity modulated radiation therapy treatment planning under uncertainty, *Physics in Medicine and Biology* 50 (2006), 5463–5477.

[21] A. A. Coco, J. C. A. Júnior, T. F. Noronha, and A. C. Santos, An integer linear programming formulation and heuristics for the minmax relative regret robust shortest path problem, *Journal of Global Optimization*, 60.2 (2014), 265–287.

[22] E. Erdoğan and G. Iyengar, Ambiguous chance constrained problems and robust optimization, *Mathematical Programming*, 107 (2006), 37–90.

[23] J. H. Gallier, *Geometric methods and applications: for computer science and engineering*, Springer, 2001.

[24] L. El. Ghaoui and F. Oustry and H. Lebret, Robust solutions to uncertain semidefinite programs, *SIAM Journal on Optimization*, 9 (1998), 33–52.

[25] J. L. Goffin, Z. Q. Luo, and Y. Ye, On the complexity of a column generation algorithm for convex and quasiconvex feasibility problems, *Large Scale Optimization: State of the Art, Kluwer Academic Publishers*, (1993), 187–196.

[26] J. L. Goffin and J. P. Vial, Convex non-differentiable optimization: a survey focused on the analytic center cutting-plane method, *Optimization Methods & Software*, 17 (2002), 805–867.

[27] J. Goh, and M. Sim. Distributionally robust optimization and its tractable approximations. *Operations Research*, 58.4-part-1 (2010), 902–917.

[28] M. Grötschel, L. Lovász, and A. Schrijver, The ellipsoid method and its consequences in combinatorial optimization, *Combinatorica*,1.2 (1981), 169–197.

[29] B. Grünbaum, Partitions of mass-distributions and convex bodies by hyperplanes, *Pacific J. Math.*, 10 (1960), 1257–1261.

[30] G. A. Hanasusanto, V. Roitch, D. Kuhn, and W. Wiesemann, A distributionally robust perspective on uncertainty quantification and chance constrained programming, *Mathematical Programming* 151, no. 1 (2015): 35–62.

[31] W. Hoeffding, Probability Inequalities For Sums Of Bounded Random Variables, *Journal of the American Statistical Association* 58 (1963), 13–30.

[32] J. Hu, and S. Mehrotra, Robust and Stochastically Weighted Multiobjective Optimization Models and Reformulations, *Operations Research*, 60 (2012), 936–953.

[33] D. A. Iancu, and N. Trichakis. Pareto efficiency in robust optimization, *Management Science*, 60.1 (2013), 130–147.

[34] J.P. Ignizio, *Goal programming and extensions*, Lexington Books, Lexington, MA, (1976).

[35] V. S. Iyengar, J. Lee, and M. Campbell, Q-Eval: Evaluating Multiple Attribute Items Using Queries, *Proceedings of the 3rd ACM conference on Electronic Commerce*, (2001), 144-153.

[36] L. P. Kaelbling, M. L. Littman and A. W. Moore, Reinforcement Learning: A Survey, *Journal of Artificial Intelligence Research*, 4 (1996), 237–285.

[37] M. Karimi, *A Quick-and-Dirty Approach to Robustness in Linear Optimization*, Master's Thesis, University of Waterloo, 2012.

[38] R. Keeney, *Value-focused Thinking*, Harvard University Press, London, (1992).
[39] R. Keeney, and H. Raiffa, *Decision with Multiple Objectives*, Wiley, New York, (1976).
[40] L. G. Khachiyan, Polynomial algorithms in linear programming, *USSR Computational Mathematics and Mathematical Physics* 20.1 (1980), 53–72.
[41] M. Kojima, N. Megiddo, T. Noma, and A. Yoshise, *A unified approach to interior point algorithms for linear complementarity problems.* Vol. 538. Springer Science & Business Media; (1991).
[42] M. Köksalan, J. Wallenius, and S. Zionts, *Multiple Criteria Decision Making: From Early History to the 21st Century*, Singapore, World Scientific, (2011).
[43] D. Lu, and F. Gzara, The robust crew pairing problem: model and solution methodology, *Journal of Global Optimization*, 62.1 (2015), 29–54.
[44] L. B. Miller, and H. Wagner, Chance-constrained programming with joint constraints, *Operations Research*, 13 (1965), 930–945.
[45] M. Minoux, On 2-stage robust LP with RHS uncertainty: complexity results and applications, *Journal of Global Optimization*, 49.3 (2011), 521–537.
[46] S. Moazeni, *Flexible Robustness in Linear Optimization*, Master's Thesis, University of Waterloo, 2006.
[47] R. DC. Monteiro, and P. R. Zanjácomo. General interior-point maps and existence of weighted paths for nonlinear semidefinite complementarity problems, *Mathematics of Operations Research*, 25.3 (2000), 381–399.
[48] M. G. Morgan, and M. Henrion, *Uncertainty - A Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis*, Cambridge University Press, New York, NY, USA, (1990).
[49] S. Mudchanatongsuk, F. Ordonez, and and J. Liu, Robust Solutions For Network Design Under Transportation Cost And Demand Uncertainty, USC ISE Working paper 2005–05.
[50] J. M. Mulvey, R. J. Vanderbei, and S. A. Zenios, Robust Optimization of Large-Scale Systems, *Operations Research*, 43 (1995), 264–281.
[51] A. Nemirovski, and A. Shapiro, Convex approximations of chance constrained programs, *SIAM Journal of Optimization*, 4 (2006) 969–996.
[52] Yu. Nesterov. Complexity estimates of some cutting-plane methods based on the analytic barrier. *Mathematical Programming, Series B*, 69 (1995), 149–176.
[53] Yu. Nesterov, and A. Nemirovskii, *Interior Point Polynomial Algorithm in Convex Programming*, SIAM; Studies in Applied and Numerical Mathematics, 1994.
[54] D. J. Newman, Location of the maximum on unimodal surfaces, *Journal of the ACM*, 12 (1965), 395–398.
[55] D. Bertsimas, and O. Nohadani, Robust optimization with simulated annealing, *Journal of Global Optimization*, 48.2 (2010), 323–334.
[56] F. Ordonez and J. Zhao, Robust Capacity Expansion Of Network Flows, USC-ISE Working paper 2004–01.
[57] P. Parpas, B. Rustem, and E. N. Pistikopoulos, Global optimization of robust chance constrained problems, *Journal of Global Optimization*, 43.2-3 (2009), 231–247.
[58] R. T. Rockafellar, *Convex Analysis*, Princeton University Press, 1997.
[59] T. Santoso, S. Ahmed, M. Goetschalckx, and A. Shapiro, A stochastic programming approach for supply chain network design under uncertainty, *European Journal of Operational Research*, 167 (2005) 96–115.
[60] A. Shapiro, Distributionally robust stochastic programming, *Optimization Online*, (2016).
[61] H. D. Sherali, and V. Ganesan, An Inverse Reliability-based approach for designing under uncertainty with application to robust piston design, *Journal of Global Optimization*, 37.1 (2007), 47–62.
[62] M. Y. Sir, M. A. Epelman, and S. M. Pollock, Stochastic programming for off-line adaptive radiotherapy, *Ann. Oper. Res.* 196 (2012) 767–797.
[63] A. L. Soyster, Convex programming with set-inclusive constraints and applications to inexact linear programming, *Operations Research*, 21 (1973), 1154–1157.
[64] R. S. Sutton, and A. G. Barto, *Reinforcement learning: An introduction.* MIT press, 1998.
[65] P. M. Vaidya, A new algorithm for minimizing convex functions over convex sets, *Symposium on Foundations of Computer Science*, (1989), 338–343.
[66] P. M. Vaidya, and D. S. Atkinson, A Technique for Bounding the Number of Iterations in Path Following Algorithms, *Complexity in Numerical Optimization*, World Scientific, Singapore, (1993), 462–489.
[67] F. Wang, D. Xu, and C. Wu, Combinatorial approximation algorithms for the robust facility location problem with penalties, *Journal of Global Optimization*, (2014), 1–14.
[68] D. B. Yudin and A.S. Nemirovski, Informational complexity and efficient methods for solving complex extremal problems, *Matekon*, 13 (1977), 25–45.

Appendix A. Probabilistic Analysis

Probabilistic analysis is tied to robust optimization. One of the recent trends in robust optimization research is the attempt to try reducing conservatism to get better results, and at the same time keeping a good level of robustness. In other words, we have to show that our proposed answer has a low probability of infeasibility. In this section, we derive some probability bounds for our algorithms based on weight and slack vectors. These bounds can be given to the DM with each answer and the DM can use them to improve the next feedback.

A.1. **Representing the robust feasible region with weight vectors.** Before starting the probabilistic analysis, want to relate the notion of weights to the parameters of the uncertainty set. As we explained in Subsection 2, we consider our uncertainty sets as follows:

$$B_i := \left\{ \tilde{b}_i \ : \ \exists \tilde{z} = (\tilde{z}_i^1, \dots, \tilde{z}_i^{N_i}) \in [-1,1]^{N_i} \ s.t. \ \tilde{b}_i = b_i^{(0)} + \sum_{l=1}^{N_i} \Delta b_i^l \tilde{z}_i^l \right\}, \tag{45}$$

where $\{\tilde{z}_i^l\}_{l=1}^{N_i}$, $i \in \{1,\dots,m\}$ are independent random variables, and Δb_i^l is the scaling factor of $\tilde{z}_i^l$. We assume that the support of $\tilde{z}_i^l$ contains $\tilde{z}_i^l = -1$, i.e., $Pr\{\tilde{z}_i^l = -1\} \neq 0$. Let us define another set which is related to the weight vectors:

$$\mathcal{W} := \left\{ (w_1, \dots, w_m) : w_i \in [y_i(w)\|\Delta b_i\|_1, 1), \ \sum_{i=1}^m w_i = 1 \right\}, \tag{46}$$

where $y(w)$ is the y-vector of w. Our goal is to explicitly specify a set of weights whose corresponding w-center makes the feasible solution of the robust counterpart.

Proposition A.1. *Let x satisfy $Ax \le \tilde{b}$ for every $\tilde{b} \in B_1 \times B_2 \times \cdots \times B_m$. Then there exists some $w \in \mathcal{W}$, so that x is the weighted analytic center with respect to the weight vector w, i.e., $x = x(w)$. In other words,*

$$\left\{ x \ : \ Ax \le \tilde{b}, \ \forall \tilde{b} \in B_1 \times B_2 \times \cdots \times B_m \right\} \subseteq \{x(w) \ : \ w \in \mathcal{W}\}.$$

Proof. Let $\hat{w} > 0$ be an arbitrary vector such that $\sum_{i=1}^m \hat{w}_i = 1$, and let $(\hat{x}, \hat{y}, \hat{s})$ be the weighted center corresponding to it. Assume that x is in the robust feasible region; we must have $\langle a_i, x \rangle \le b_i^{(0)} + \langle \Delta b_i, \tilde{z}_i \rangle$ for every $\tilde{z}_i$ with nonzero probability, particularly for $\tilde{z}_i = -e$ where e is all ones vector. So

$$\langle a_i, x \rangle - b_i^{(0)} \le \langle \Delta b_i, \tilde{z}_i \rangle = \langle \Delta b_i, -e \rangle = -\|\Delta b_i\|.$$

Define $s_i := b_i^{(0)} - \langle a_i, x \rangle$. Thus, from the above equation, for every $i \in \{1,\dots,m\}$ we have

$$0 < \|\Delta b_i\|_1 \le s_i,$$

and consequently $\hat{y}_i \|\Delta b_i\|_1 \le \hat{y}_i s_i$ using the fact that $\hat{y}_i > 0$. For every $i \in \{1,\dots,m\}$, we set

$$w_i := \hat{y}_i s_i.$$

Since $(x, \hat{y}, s)$ satisfies the optimality conditions, we have $x = x(w)$. It remains to show that $w \in \mathcal{W}$. First note that:

$$\sum_{i=1}^m w_i = \sum_{i=1}^m s_i \hat{y}_i = \sum_{i=1}^m \hat{s}_i \hat{y}_i = \sum_{i=1}^m \hat{w}_i = 1,$$

where for the second equality we used Lemma B.1. Now, using the fact that $w_i \geq 0$ for every $i \in \{1, \dots, m\}$, we have $w_i < \sum_{j=1}^m w_j = 1$. We already proved that $\hat{y}_i \|\Delta b_i\|_1 \leq \hat{y}_i s_i = w_i$. These two inequalities prove that $w_i \in [\hat{y}_i \|\Delta b_i\|_1, 1)$. □

The above proposition shows that when the robust counterpart problem with respect to the uncertainty set $B_1 \times B_2 \times \cdots \times B_m$ is feasible, the set $\mathcal{W}$ is nonempty. In the next proposition we prove that the equality holds in the above inclusion.

Proposition A.2. *(a)We have*

$$\{x \ : \ Ax \leq \tilde{b}, \ \forall \tilde{b} \in B_1 \times B_2 \times \cdots \times B_m\} = \{x(w) \ : \ w \in \mathcal{W}\}.$$

(b) Assume that $w > 0$ satisfies $\sum_{i=1}^m w_i = 1$, and y is its corresponding y-vector. For every $i \in \{1, \dots, m\}$, we have

$$w_i \geq y_i \|\Delta b_i\|_1 \Rightarrow \langle a_i, x(w) \rangle \leq \tilde{b}_i, \quad \forall \tilde{b}_i \in B_i.$$

Proof. (a) $\subseteq$ part was proved in Proposition A.1. For $\supseteq$, let $w \in \mathcal{W}$ and (x, y, s) be its corresponding weighted center. By $w \in \mathcal{W}$ we have

$$y_i \|\Delta b_i\|_1 \leq w_i = s_i y_i = (b_i^{(0)} - \langle a_i, x \rangle) y_i \Longrightarrow \|\Delta b_i\|_1 \leq (b_i^{(0)} - \langle a_i, x \rangle).$$

Therefore, for all $\tilde{z}_i \in \times_{i=1}^m [-1, 1]$,

$$\langle a_i, x \rangle \leq b_i^{(0)} - \|\Delta b_i\|_1 \leq b_i^{(0)} - \sum_{l=1}^N \Delta b_i^l \ \tilde{z}_i^l = b_i^{(0)} + \langle \tilde{z}_i, \Delta b_i \rangle,$$

which proves x is a robust feasible solution with respect to the uncertainty set $B_1 \times B_2 \times \cdots \times B_m$.
(b) Assume that $w > 0$ satisfies $\sum_{i=1}^m w_i = 1$, y is its corresponding y-vector, and there exists $i \in \{1, \dots, m\}$ such that $w_i \geq y_i \|\Delta b_i\|_1$. If there exists $\tilde{b}_i \in B_i$ such that $\langle a_i, x(w) \rangle > \tilde{b}_i$ where $\tilde{b}_i = b_i^{(0)} + \sum_{l=1}^{N_i} \Delta b_i^l \tilde{z}_i^l$, by using $\tilde{z}_i^l \geq -1$ we have

$$\begin{aligned}
\langle a_i, x(w) \rangle > \tilde{b}_i \quad &\Rightarrow \quad \langle a_i, x(w) \rangle > b_i^{(0)} + \sum_{l=1}^{N_i} \Delta b_i^l \ \tilde{z}_i^l \ \geq \ b_i^{(0)} - \sum_{l=1}^{N_i} \Delta b_i^l \\
&\Rightarrow \quad \sum_{l=1}^{N_i} \Delta b_i^l > b_i^{(0)} - \langle a_i, x(w) \rangle = s_i(w) \\
&\Rightarrow \quad y_i \sum_{l=1}^{N_i} \Delta b_i^l > y_i s_i(w) = w_i \geq y_i \sum_{l=1}^{N_i} \Delta b_i^l \\
&\Rightarrow \quad \sum_{l=1}^{N_i} \Delta b_i^l \ > \ \sum_{l=1}^{N_i} \Delta b_i^l,
\end{aligned}$$

which is a contradiction. We conclude that $\langle a_i, x(w) \rangle \leq \tilde{b}_i$ for all $\tilde{b}_i \in B_i$. □

A.2. **Probability bounds.** Without loss of generality, we make the following assumptions on $\tilde{b}$ and $\tilde{c}$:

- For every $i \in \{1, 2, \ldots, m\}$, $\tilde{b}_i$ can be written as $\tilde{b}_i = b_i^{(0)} + \sum_{l=1}^{N_i} \Delta b_i^l \tilde{z}_i^l$ where $\{\tilde{z}_i^l\}_{l=1}^{N_i}$ are independent random variables for every $i \in \{1, \ldots, m\}$.
- For each $\tilde{c}_i$, $i \in \{1, \ldots, n\}$, we have $\tilde{c}_i = c_i^{(0)} + \sum_{l=1}^{N_{ic}} \Delta c_i^l \tilde{z}_{ic}^l$ where $\{\tilde{z}_{ic}^l\}_{l=1}^{N_{ic}}$ are independent random variables.

As can be seen above, each variable $\tilde{b}_i$ is the summation of a nominal value $b_i^{(0)}$ with scaled random variables $\{\tilde{z}_i^l\}_{l=1}^{N_i}$. In practice, the number of these random variables N_i is small compared to the dimension of A as we explained above: each random variable $\tilde{z}_i^l$ represents a major source of uncertainty in the system.

Suppose we wish to find a robust feasible solution with respect to the uncertainty set $B_1 \times B_2 \times \cdots \times B_m$, where B_i was defined in (45). By Proposition A.2, it is equivalent to finding the weighted center for a $w \in \mathcal{W}$, where $\mathcal{W}$ is defined in (46). However, finding such a weight vector is not straight forward as we do not have an explicit formula for $\mathcal{W}$. Assume that we pick an arbitrary weight vector $w > 0$ such that $\sum_{i=1}^m w_i = 1$, with the weighted center (x, y, s). Let us define the vector δ for w as

$$\delta_i = \frac{w_i}{y_i \|\Delta b_i\|_1}, \quad i \in \{1, 2, \ldots, m\},$$

where Δb_i was defined in (45). For each $i \in \{1, \ldots, m\}$, if $1 \leq \delta_i$, by Proposition A.2-(b) we have $\langle a_i, x(w) \rangle \leq \tilde{b}_i$ for all $\tilde{b}_i \in B_i$. So, the problem is with the constraints that $1 > \delta_i$. For every such constraint, we can find a bound on the probability that $\langle a_j, x(w) \rangle > \tilde{b}_j$. As in the proof of Proposition A.2-(b), in general we can write:

$$\begin{aligned} \Pr\{\langle a_j, x \rangle > \tilde{b}_j\} &= \Pr\left\{ -y_i \sum_{l=1}^{N_i} \Delta b_i^l \ \tilde{z}_i^l > w_i = y_i \delta_i \|\Delta b_i\|_1 \right\} \\ &= \Pr\left\{ -\sum_{l=1}^{N_i} \Delta b_i^l \ \tilde{z}_i^l > \delta_i \|\Delta b_i\|_1 \right\} \leq \exp\left(-\frac{\delta_i^2 (\|\Delta b_i\|_1)^2}{2 \sum_{l=1}^{N_i} (\Delta b_i^l)^2} \right), \end{aligned} \tag{47}$$

where the last inequality is derived by using Hoeffding's inequality:

Lemma A.1. *(*Hoeffding's inequality[31]*) Let $v_1, v_2, \ldots, v_n$ be independent random variables with finite first and second moments, and for every $i \in \{1, 2, \ldots, n\}$, $\tau_i \leq v_i \leq \rho_i$. Then for every $\varphi > 0$*

$$\Pr\left\{ \sum_{i=1}^n v_i - E\left(\sum_{i=1}^n v_i \right) \geq n\varphi \right\} \leq \exp\left[\frac{-2n^2 \varphi^2}{\sum_{i=1}^n (\rho_i - \tau_i)^2} \right].$$

Bertsimas and Sim [14] derived the best possible bound, i.e., a bound that is achievable. The corresponding lemma proved in [14] is as follows:

Lemma A.2. *(a) If $\tilde{z}_i^l$, $l \in \{1, \ldots, N_i\}$, are independent and symmetrically distributed random variables in $[-1, 1]$, p is a positive constant, and $\gamma_{il} \leq 1$, $l \in \{1, \ldots, N_i\}$, then*

$$\Pr\left\{ \sum_{l=1}^{N_i} \gamma_{il} \ \tilde{z}_i^l \geq p \right\} \leq B(N_i, p), \tag{48}$$

where

$$B(N_i,p)=\frac{1}{2^{N_i}}\left[(1-\mu)\binom{N_i}{\lfloor\nu\rfloor}+\sum_{i=\lfloor\nu\rfloor+1}^{N_i}\binom{N_i}{i}\right],$$

where $\nu:=(N_i+p)/2$, *and* $\mu:=\nu-\lfloor\nu\rfloor$.
(b) The bound in (48) *is tight for* $\tilde{z}_i^l$ *having a discrete probability distribution:* $\Pr\{\tilde{z}_i^l=1\}=\Pr\{\tilde{z}_i^l=-1\}=1/2$, $\gamma_{il}=1$, $l\in\{1,\dots,N_i\}$, *an integral value of* $p\geq 1$, *and* $p+N_i$ *being even.*

We can use the bound for our relation (47) as follows. Assume that $\tilde{z}_i^l$, $l\in\{1,\dots,N_i\}$, are independent and symmetrically distributed random variables in $[-1,1]$. Also denote by $\max(\Delta b_i)$, the maximum entry of Δb_i. Using (47), We can write

$$\begin{aligned}\Pr\{\langle a_j,x\rangle>\tilde{b}_j\} &= \Pr\left\{\sum_{l=1}^{N_i}\Delta b_i^l\ \tilde{z}_i^l>\delta_i\|\Delta b_i\|_1\right\}\\ &\leq \Pr\left\{\sum_{l=1}^{N_i}\frac{\Delta b_i^l}{\max(\Delta b_i)}\ \tilde{z}_i^l\geq\delta_i\frac{\|\Delta b_i\|_1}{\max(\Delta b_i)}\right\}\\ &\leq B\left(N_i,\delta_i\frac{\|\Delta b_i\|_1}{\max(\Delta b_i)}\right). \qquad (49)\end{aligned}$$

To compare these two bounds, assume that all the entries of Δb_i are equal. Bound (47) reduces to $\exp(-\delta_i^2N_i/2)$, and bound (49) reduces to $B(N_i,\delta_iN_i)$. We can prove that bound (49) dominates bound (47). Moreover, bound (49) is somehow the best possible bound as it can be achieved by a special probability distribution as in Lemma A.2. The above probability bounds do not take part in our algorithm explicitly. However, for each solution, we can present these bounds to the DM and s/he can use them to improve the feedback to the algorithm. As an example of how these bounds may be used for the DM, we show how to construct a concave utility function $U(s)$ based on these probability bounds. Bounds (47) and (49) are functions of $\delta_i=\frac{w_i}{y_i\|\Delta b_i\|_1}=\frac{s_i}{\|\Delta b_i\|_1}$ and as a result, functions of s. Now, assume that based on the probability bounds, the DM defines a function $u_i(s_i)$ for each slack variable s_i as shown in Figure 7. $u_i(s_i)$ increases as s_i increases, and then at the point ϵ_i^1 becomes flat. At $s_i=\epsilon_i^2$ it starts to decrease to reach zero. Parameters ϵ_i^1 and ϵ_i^2 are specified by the DM's desired bounds. Now, we can define the utility function as $U(s):=\prod_{j=1}^m u_i(s_i)$. This function is not concave, but maximization of it is equivalent to the maximization of $\ln(U(s))$ which is concave.

Appendix B. Properties of w-space

In this appendix, we study the properties of weight space as well as W_s and W_y regions. Let us start from the following well-known lemma:

Lemma B.1. *Let* (x,y,s) *and* $(\hat{x},\hat{y},\hat{s})$ *be the solutions of system* (14) *corresponding to the weight vectors* w, $\hat{w}\in\mathbb{R}^m_{++}$, *respectively. For every* $\bar{y}$ *in the null space of* $A^\top$ *we have:*

$$\langle\hat{s},\bar{y}\rangle=\langle s,\bar{y}\rangle.$$

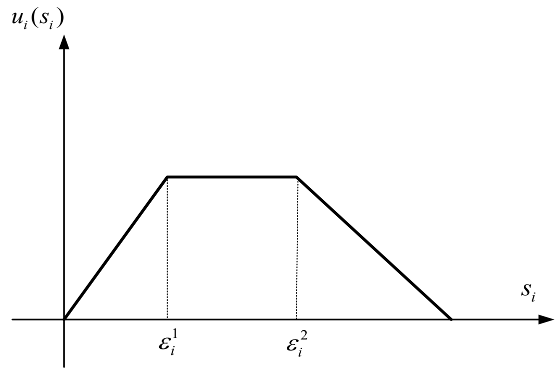

FIGURE 7. The function $u_i(s_i)$ defined for the slack variable s_i

Proof. From (14), we have $s = b - Ax$ and $\hat{s} = b - A\hat{x}$, which results in $s - \hat{s} = A(x - \hat{x})$. Hence we have $s - \hat{s} \in \mathcal{R}(A)$. As the null space of $A^\top$ and the range of A are orthogonal, for every $\bar{y} \in \mathcal{N}(A^\top)$ we can write:

$$\langle s - \hat{s}, \bar{y} \rangle = 0 \quad \Rightarrow \quad \langle \hat{s}, \bar{y} \rangle = \langle s, \bar{y} \rangle.$$

□

Let $(\hat{x}, \hat{y}, \hat{s})$ be the solution of system (14) corresponding to the weight vector $\hat{w}$. Moreover, assume that $\bar{y} > 0$ is such that $A^\top \bar{y} = 0$. Then, by using Lemma B.1, we can show that $(\hat{x}, \bar{y}, \hat{s})$ is the solution of system (14) corresponding to the weight vector $\bar{Y}(\hat{Y})^{-1}\hat{w}$. Hence, there may be many weight vectors that give the same w-center. A stronger result is the following lemma which shows that in some cases, we can find the weighted center for a combination of weight vectors by using the combination of their weighted centers.

Lemma B.2. *Let $(x^{(i)}, y^{(i)}, s^{(i)})$, $i \in \{1, \dots, \ell\}$, be solutions of system (14), corresponding to the weights $w^{(i)}$. Then, for every set of $\beta_i \in [0, 1]$, $i \in \{1, \dots, \ell\}$, such that $\sum_{i=1}^{\ell} \beta_i = 1$, and for every $j \in \{1, \dots, \ell\}$, we have $(\sum_{i=1}^{\ell} \beta_i x^{(i)}, y^{(j)}, \sum_{i=1}^{\ell} \beta_i s^{(i)})$ is the w-center of $\mathcal{F}$, where*

$$w := \sum_{i=1}^{\ell} \beta_i Y^{(j)} (Y^{(i)})^{-1} w^{(i)}.$$

Moreover,

$$\sum_{i=1}^{m} w_i = \sum_{i=1}^{m} w_i^{(j)}.$$

Proof. According to the assumptions, for every $i \in \{1, \dots, \ell\}$, we have

$$\begin{aligned}
&Ax^{(i)} + s^{(i)} = b^{(0)}, \quad s > 0, \\
&A^\top y^{(i)} = 0, \\
&S^{(i)} y^{(i)} = w^{(i)}.
\end{aligned}$$

Now, it can be seen that $(\sum_{i=1}^{\ell}\beta_i x^{(i)}, y^{(j)}, \sum_{i=1}^{\ell}\beta_i s^{(i)})$ satisfies the system:

$$\begin{aligned} & A(\sum_{i=1}^{\ell}\beta_i x^{(i)}) + (\sum_{i=1}^{\ell}\beta_i s^{(i)}) = b^{(0)}, \quad (\sum_{i=1}^{\ell}\beta_i s^{(i)}) > 0, \\ & A^\top y^{(j)} = 0, \\ & (\sum_{i=1}^{\ell}\beta_i S^{(i)})y^{(j)} = \sum_{i=1}^{\ell}\beta_i Y^{(j)}(Y^{(i)})^{-1}w^{(i)}. \end{aligned} \tag{50}$$

Since the w-center of $\mathcal{F}$ is unique, the proof for the first part is done.
For the second part, from (50) we can write

$$\sum_{i=1}^{m} w_i = \sum_{i=1}^{m}(\sum_{p=1}^{\ell}\beta_p s_i^{(p)})y_i^{(j)} = \sum_{p=1}^{\ell}\beta_p(\sum_{i=1}^{m} s_i^{(p)}y_i^{(j)}) = \sum_{p=1}^{\ell}\beta_p\langle s^{(p)}, y^{(j)}\rangle.$$

By Lemma B.1, we have $\langle s^{(p)}, y^{(j)}\rangle = \langle s^{(i)}, y^{(j)}\rangle$. Therefore, we can continue the above series of equations as follows:

$$\sum_{i=1}^{m} w_i = \sum_{p=1}^{\ell}\beta_p\langle s^{(j)}, y^{(j)}\rangle = \sum_{p=1}^{\ell}\beta_p(\sum_{i=1}^{m} s_i^{(j)}y_i^{(j)}) = (\sum_{i=1}^{m} w_i^{(j)})\sum_{p=1}^{\ell}\beta_p = \sum_{i=1}^{m} w_i^{(j)}.$$

□

B.1. **Properties of w-space.** In this subsection, we study the structure of the w-space, which is important for the design of the algorithms in Section 4. Let s and y be centric. First, we note that the simplex of the weight vectors can be divided into regions of constant y-vector (W_y) and constant s-vector (W_s). By using Lemma B.2, if $(\hat{x}, \hat{y}, \hat{s})$ is the solution of system (14) corresponding to the weight vector $\hat{w} \in W$, and $\bar{y} > 0$ is any centric y-vector, then $(\hat{x}, \bar{y}, \hat{s})$ is the solution of system (14) corresponding to the weight vector $\bar{Y}(\hat{Y})^{-1}\hat{w}$. This means that for every centric vector $\hat{s}$ and any centric vector y, $\hat{S}y$ is a weight vector in the simplex.

For every pair of centric vectors s and y, W_s and W_y are convex. To see this, let $(x, \bar{y}, s)$ and (x, y, s) be the weighted centers of $\hat{w}$ and w. Then, it is easy to see that for every $\beta \in [0, 1]$, $(x, \beta\bar{y} + (1-\beta)y, s)$ is the weighted center of $\beta\hat{w} + (1-\beta)w$. With a similar reasoning, W_y is convex for every centric y.

Using (14), we can express W_s and W_y as follows:

$$W_y = Y[(\mathcal{R}(A) + b) \cap \mathbb{R}^m_{++}] \cap B_1(0, 1), \tag{51}$$

$$W_s = S[\mathcal{N}(A^\top) \cap \mathbb{R}^m_{++}] \cap B_1(0, 1), \tag{52}$$

where $B_1(0, 1)$ is the unit ball in 1-norm centered at zero vector. Here, we want to find another formulation for W_y that might work better in some cases. We use the following lemma.

Lemma B.3. *Assume that the rows of $B_y \in \mathbb{R}^{(m-n)\times m}$ make a basis for the null space of $A^\top Y$. Then there exists $x \in \mathbb{R}^n$ such that $YAx + w = Yb$ if and only if $B_y w = B_y Yb$. I.e., $(Yb - w) \in \mathcal{R}(YA)$ iff $(Yb - w) \in \mathcal{N}(B_y)$.*

Proof. Assume that there exists x such that $YAx + w = Yb$. By multiplying both sides with B_y from the left and using the fact that $B_y YA = 0$ we have the result. For the other direction, assume that $B_y w = B_y Yb$. Then $B_y(w - Yb) = 0$ which means $w - Yb$ is in the null space of B_y. Then, using the orthogonal decomposition theorem, we have $\mathcal{N}(B_y) = \mathcal{R}(B_y^\top)^\perp = \mathcal{N}(A^\top Y)^\perp = \mathcal{R}(YA)$. Thus, there exists x such that $YAx + w = Yb$. □

Assume that $B \in \mathbb{R}^{(m-n)\times m}$ is such that its rows make a basis for the null space of $A^\top$. For every vector y, we have $A^\top y = A^\top Y(Y^{-1}y)$, so if y is in the null space of $A^\top$, $Y^{-1}y$ is in the null space of $A^\top Y$. Hence, if the rows of B make a basis for the null space of $A^\top$, the rows of BY^{-1} make a basis for the null space of $A^\top Y$ and we can write $B_y = BY^{-1}$. Using Lemma B.3, there exists x such that $YAx + w = Yb$ if and only if $BY^{-1}w = BY^{-1}Yb = Bb$, and we can write (51) as:

$$W_y = \left\{ w > 0 \;:\; BY^{-1}w = Bb,\; e^\top w = 1 \right\}. \tag{53}$$

Let us denote the affine hull with aff(.). We can prove the following lemma about W_s and W_y.

Lemma B.4. *Assume that s and y are centric, we have*

$$W_s = \mathrm{aff}(W_s) \cap W \quad \text{and} \quad W_y = \mathrm{aff}(W_y) \cap W.$$

Proof. We prove the first one and our proof for the second one is the same. Clearly we have $W_s \subseteq \mathrm{aff}(W_s) \cap W$. To prove the other side, assume by contradiction that there exist $w \in \mathrm{aff}(W_s) \cap W$ such that $w \notin W_s$. Pick an arbitrary $\hat{w} \in \mathrm{relint}(W_s)$ and consider all the points $w(\beta) = \beta w + (1-\beta)\hat{w}$ for $\beta \in [0,1]$. Both w and $\hat{w}$ are in $\mathrm{aff}(W_s)$, so all the points $w(\beta)$ are also in $\mathrm{aff}(W_s)$. $w(0) \in W_s$ and $w(1) \notin W_s$, so let $\hat{\beta}$ be $\sup\{\beta : w(\beta) \in W_s\}$.

Note that all the points in W_s has the same s-vector, so we have $w(\beta) = Sy(\beta)$ for $\beta \in [0, \hat{\beta})$. By using (14) we must also have $w(\hat{\beta}) \in W_s$. We want to prove that $\hat{\beta} = 1$. Assume that $\hat{\beta} < 1$. All the points on the line segment between $w(0)$ and $w(\hat{\beta})$ have the same s-vector and we can write them as $S(\gamma y(0) + (1-\gamma)y(\hat{\beta}))$ for $\gamma \in [0,1]$. But note that $y(\hat{\beta}) > 0$, so there is a small enough $\epsilon > 0$ such that $y_\epsilon = (-\epsilon y(0) + (1+\epsilon)y(\hat{\beta})) > 0$ and hence Sy_ϵ is a weight vector in W_s. However, it is also a vector on the line segment between $w(\hat{\beta})$ and w which is a contradiction to $\hat{\beta} = \sup\{\beta : w(\beta) \in W_s\}$. So $\hat{\beta} = 1$ and $w = w(1) \in W_s$ which is a contradiction. Hence $W_s \supseteq \mathrm{aff}(W_s) \cap W$ and we are done. □

We conclude that W is sliced in two ways by W_y's and W_s's for centric s and y vectors. For each centric s and each centric y, W_y and W_s intersect at a single point Sy on the simplex. We want to prove that the smallest affine subspace containing W_s and W_y is $\mathrm{aff}(W) = \{w : e^\top w = 1\}$. To that end, we prove some results on the intersection of affine subspaces. We start with the following definition:

Definition B.1. *The* recession cone *of a convex set $C \in \mathbb{R}^n$ is denoted by* rec(C) *and defined as:*

$$\mathrm{rec}(C) := \{y \in \mathbb{R}^n \;:\; (x+y) \in C, \quad \forall x \in C\}.$$

The lineality space *of a convex set C is denoted by* lin(C) *and defined as:*

$$\mathrm{lin}(C) := (\mathrm{rec}(C)) \cap (-\mathrm{rec}(C)).$$

Let U be an affine subspace of $\mathbb{R}^m$. If $y \in \mathrm{rec}(U)$, then $-y \in \mathrm{rec}(U)$, which means $(\mathrm{rec}(U)) = (-\mathrm{rec}(U))$. Therefore, by Definition B.1, we have $\mathrm{lin}(U) = \mathrm{rec}(U)$. Then, by using the definition of the affine space we have:

$$\mathrm{lin}(U) := \{u_1 - u_2 : \forall u_1, u_2 \in U\}. \tag{54}$$

In other words, $\mathrm{lin}(U)$ is a linear subspace such that $U = u + \mathrm{lin}(U)$ for all $u \in U$ where $'+'$ is the Minkowski sum. The following two lemmas are standard, see, for instance, [23].

Lemma B.5. *Given a pair of nonempty affine subspaces U and V in $\mathbb{R}^n$, the following facts hold:*
(1) $U \cap V \neq \emptyset$ iff for every $u \in U$ and $v \in V$, we have $(v - u) \in \mathrm{lin}(U) + \mathrm{lin}(V)$.
(2) $U \cap V$ consists of a single point iff for every $u \in U$ and $v \in V$, we have

$$(v - u) \in \mathrm{lin}(U) + \mathrm{lin}(V) \quad \text{and} \quad \mathrm{lin}(U) \cap \mathrm{lin}(V) = \{0\}.$$

(3) For every $u \in U$ and $v \in V$, we have

$$\mathrm{lin}(\mathrm{aff}(U \cup V)) = \mathrm{lin}(U) + \mathrm{lin}(V) + \{\alpha(v - u) : \ \alpha \in \mathbb{R}\}.$$

Lemma B.6. *Let U and V be nonempty affine subspaces in $\mathbb{R}^n$. Then we have the following properties:*

(1) if $U \cap V = \emptyset$, then

$$\dim(\mathrm{aff}(U \cup V)) = \dim(U) + \dim(V) + 1 - \dim(\mathrm{lin}(U) \cap \mathrm{lin}(V)),$$

(2) if $U \cap V \neq \emptyset$, then

$$\dim(\mathrm{aff}(U \cup V)) = \dim(U) + \dim(V) - \dim(U \cap V).$$

Using the above lemmas, we deduce the following proposition.

Proposition B.1. *Assume that s and y are centric s-vector and y-vector, respectively. Then the smallest affine subspace containing W_s and W_y is* $\mathrm{aff}(W) = \{w : e^\top w = 1\}$.

Proof. We assumed that $A \in \mathbb{R}^{m \times n}$ has full column rank, i.e., $\mathrm{rank}(A) = n \leq m$ and the interior of $\{x : Ax \leq b\}$ is not empty. Let B_s denote the set of all centric s-vectors, i.e., the set of s-vectors for which there exist (x, y, s) satisfies all the equations in (14). We claim that $B_s = \{s > 0 : s = b - Ax\}$. For every $s \in \{s > 0 : s = b - Ax\}$, pick an arbitrary $y > 0$ such that $A^\top y = 0$. For every scalar α we have $A^\top(\alpha y) = 0$, so we can choose α such that $\alpha y^\top s = 1$. Hence $(x, \alpha y, s)$ satisfies (14) and we conclude that $B_s = \{s > 0 : s = b - Ax\}$. The range of A has dimension n and since B_s is not empty; it is easy to see that the dimension of B_s is also n. Moreover, we have $W_y = Y B_s$ and since Y is non-singular, we have $\dim(W_y) = n$.

Now denote by B_y the set of centric y-vectors. By (14), we have $A^\top y = 0$. The dimension of the null space of $A^\top$ is $(n - m)$. In addition, we have to consider the restriction $e^\top w = 1$; we have

$$1 = e^\top w = e^\top(Ys) = s^\top y = (b - Ax)^\top y = b^\top y - x^\top A^\top y = b^\top y.$$

So, we have $b^\top y = 1$ for centric y-vectors which reduces the dimension by one (since $b \notin \mathcal{R}(A)$), and $\dim(B_y) = m - n - 1$. We have $W_s = S B_y$ and so by the same explanation $\dim(W_s) = m - n - 1$.

We proved that W_s and W_y intersect at only a single point $w = Sy$, so $\dim(W_s \cap W_y) = 0$. By using Lemma B.6-(2) the dimension of the smallest affine subspace containing W_s and W_y is

$$\dim(W_s) + \dim(W_y) - \dim(W_s \cap W_y) = n + m - n - 1 = m - 1.$$

The dimension of aff(W) is also $m-1$, so by Lemma B.4 aff(W) is the least affine subspace containing W_s and W_y. □

Appendix C.

Example C.1. *The statement of Proposition 4.1 is not true for a general concave function.*

Proof. Consider the first example of Example 3.1. We have $m = 3$, $n = 1$, $A = [1,\ -1,\ -1]^\top$, and $b = [1,\ 0,\ 0]^\top$. Using (14), the set of centric s-vectors is

$$B_s = \{[1-z, \quad z, \quad z]^\top : z \in (0,1)\}.$$

The set of centric y-vectors, B_y, is specified by solving $A^\top y = 0$ and $y^\top b = 1$ while $y > 0$ and we can see that $B_y = \{[1, \quad z, \quad 1-z]^\top : z \in (0,1)\}$. As shown in Figure 2, W_s regions are parallel line segments while W_y regions are line segments that all intersect at $[1,\ 0,\ 0]^\top$.

Now, assume that the function $U(s)$ is as follows (does not depend on s_3)

$$U(s) = \begin{cases} 3s_1 - s_2, & \text{if} \quad s_1 \le s_2; \\ -s_1 + 3s_2, & \text{if} \quad s_1 > s_2. \end{cases} \tag{55}$$

This function is piecewise linear and it is easy to see that it is concave. $U(s)$ is also differentiable at all the points except the points $s_1 = s_2$. At any point that the function is differentiable, the supergradient is equal to the gradient of the function at that point. Hence, we have $\partial U(s) = \{[3,\ -1,\ 0]^\top\}$ for $s_1 < s_2$ and $\partial U(s) = \{[-1,\ 3,\ \ 0]^\top\}$ for $s_1 > s_2$.

If we consider $U(s)$ on B_s, we can see that the maximum of the function is attained at the point that $s_1 = s_2$, so $s_{opt} = [1/2,\ 1/2,\ 1/2]^\top$. Now assume that we start at $w^0 = S^0 y^0 = [0.4,\ 0.1,\ 0.5]^\top$. Because we have $y_1 = 1$ for all centric y-vectors, $w_1^0 = s_1^0$, and we can easily find s^0 and y^0 as $s^0 = [0.4,\ 0.6,\ 0.6]^\top$ and $y^0 = [1,\ 1/6,\ 5/6]^\top$. The hyperplane passing through w^0 is $(g^0)^\top (Y^0)^{-1}(w - w^0) = 0$ and since $s_1^0 < s_2^0$ we have

$$(g^0)^\top (Y^0)^{-1} = [3,\ -1,\ \ 0](Y^0)^{-1} = [3,\ -6,\ \ 0], \tag{56}$$

and we can write the hyperplane as $3(w_1 - 0.4) - 6(w_2 - 0.1) = 0$. In the next step, we have to choose a point w^1 such that $(g^0)^\top (Y^0)^{-1}(w^1 - w^0) \ge 0$. Let us pick $w^1 = [0.6,\ 0.19,\ 0.21]^\top$ for which we can easily find $s^1 = [0.6,\ 0.4,\ 0.4]^\top$ and $y^1 = [1,\ 0.475,\ 0.525]^\top$. For this point we have $s_1^1 > s_2^2$, so $(g^1)^\top (Y^1)^{-1} = [-1,\ 6.32,\ 0]^\top$ and the hyperplane passing through w^1 is $-(w_1 - 0.6) + 6.32(w_2 - 0.19) = 0$. The intersection of two hyperplanes on the simplex can be found by solving the following system of equations:

$$\left\{\begin{array}{l} 3w_1 - 6w_2 = 0.6 \\ -w_1 - 6w_2 = 0.6 \\ w_1 + w_2 + w_3 = 1 \end{array}\right. \Rightarrow w^* = \begin{bmatrix} 0.57 \\ 0.185 \\ 0.245 \end{bmatrix}. \tag{57}$$

The intersection of simplex and the hyperplanes $(g^0)^\top (Y^0)^{-1}(w - w^0) = 0$ and $(g^1)^\top (Y^1)^{-1}(w - w^1) = 0$ are shown in Figure 8. The intersection of simplex with $\{w : (g^0)^\top (Y^0)^{-1}(w - w^0) \geq 0, \ (g^1)^\top (Y^1)^{-1}(w - w^1) \geq 0\}$ is shown by hatching lines. As can be seen, we have:

$$\left\{w : \ (g^{0w})^\top (w - w^0) \geq 0, \ (g^{1w})^\top (w - w^1) \geq 0\right\} \cap W_{s_{op}} = \ \phi.$$

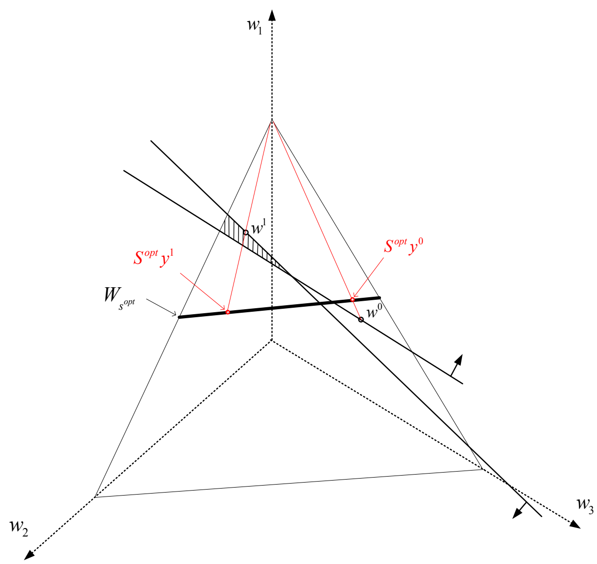

FIGURE 8. Intersection of simplex and the hyperplanes $(g^0)^\top (Y^0)^{-1}(w - w^0) = 0$ and $(g^1)^\top (Y^1)^{-1}(w - w^1) = 0$ in Example C.1.

□