# A cutting surface algorithm for semi-infinite convex programming with an application to moment robust optimization

Sanjay Mehrotra* Dávid Papp†

June 16, 2013

**Abstract**

We first present and analyze a central cutting surface algorithm for general semi-infinite convex optimization problems, and use it to develop an algorithm for distributionally robust optimization problems in which the uncertainty set consists of probability distributions with given bounds on their moments. The cutting surface algorithm is also applicable to problems with non-differentiable semi-infinite constraints indexed by an infinite-dimensional index set. Examples comparing the cutting surface algorithm to the central cutting plane algorithm of Kortanek and No demonstrate the potential of the central cutting surface algorithm even in the solution of traditional semi-infinite convex programming problems, whose constraints are differentiable, and are indexed by an index set of low dimension. Our primary motivation for the higher level of generality is to solve distributionally robust optimization problems with moment uncertainty. After the analysis of the cutting surface algorithm, we extend the authors' moment matching scenario generation algorithm to a probabilistic algorithm that finds optimal probability distributions subject to moment constraints. The combination of this distribution optimization method and the cutting surface algorithm yields a solution to a family of distributionally robust optimization problems that are considerably more general than the ones proposed to date.

Keywords: semi-infinite programming, robust optimization, stochastic programming, moment matching, column generation, cutting surface methods, cutting plane methods, moment problem

## 1 Introduction

We present a novel cutting surface algorithm for general semi-infinite convex optimization problems (SICPs) that is applicable under milder than usual assumptions on the problem formulation, extending an algorithm of Kortanek and No (1993). Our primary motivation is to solve a large class of distributionally robust optimization problems that can be posed as SICPs with convex but not necessarily differentiable constraints indexed by an uncountably infinite dimensional set of probability distributions. In the first half of the paper we introduce the SICPs considered, and describe the new central cutting surface algorithm. The connection to robust optimization is discussed in Section 4.

*Northwestern University, Department of Industrial Engineering and Management Sciences, Evanston, IL, USA. E-mail: mehrotra@iems.northwestern.edu

†Northwestern University, Department of Industrial Engineering and Management Sciences, Evanston, IL, USA. Email: dpapp@iems.northwestern.edu

We consider a general semi-infinite convex optimization problem of the following form:

$$\begin{array}{rl} \text{minimize} & x_0 \\ \text{subject to} & g(x,t) \le 0 \quad \forall\, t \in T \\ & x \in X \end{array} \tag{SICP}$$

with respect to the decision variables $x$ (whose first coordinate is denoted by $x_0$), where the sets $X$ and $T$, and the function $g\colon X \times T \mapsto \mathbb{R}$ satisfy the following conditions:

**Assumption 1.**

1. the set $X \subseteq \mathbb{R}^n$ is convex, closed and bounded;
2. there exists a *Slater point* $\bar{x}$ satisfying $\bar{x} \in X$ and $g(\bar{x}, t) \le -1$ for every $t \in T$;
3. the function $g(\cdot, t)$ is convex and subdifferentiable for every $t \in T$; moreover, these subdifferentials are uniformly bounded: there exists a $B > 0$ such that for every $x \in X$ and $t \in T$, every subgradient $d \in \partial g_x(x,t)$ satisfies $\|d\| \le B$.

Note that having one of the components of the variable vector $x$ as an objective instead of a general convex objective function is without loss of generality; we opted for this form because it simplifies both the description of our algorithm and the convergence analysis. We also remark that $T$ is not required to be either convex or finite dimensional, nor is the differentiability of $g$, or the convexity or concavity of $g$ in its second argument is required.

The minimum of (SICP) is attained, since its feasible set is closed, nonempty, and bounded, and its objective function is continuous. Our aim is to find an optimal solution to (SICP) within $\varepsilon$ accuracy, by which we mean the following.

We say that $x \in X$ is *$\varepsilon$-feasible* if $g(x,t) \le \varepsilon$ for every $t \in T$, and we say that a point $x^*_\varepsilon \in X$ is an *$\varepsilon$-optimal solution* to (SICP) if it is $\varepsilon$-feasible and

$$(x^*_\varepsilon)_0 \le x^*_0 \stackrel{\text{def}}{=} \min\{x_0 \,|\, x \in X,\ g(x,t) \le 0 \,\forall\, t \in T\}.$$

We make one final assumption, on our ability to detect the approximate infeasibility of candidate solutions to (SICP) within a prescribed error $\varepsilon \ge 0$.

**Assumption 2.** For every point $x \in X$ that is not $\varepsilon$-feasible, we can find a $t \in T$ satisfying $g(x,t) > 0$.

It is *not* required that we can find the most violated inequality $g(x,t) > 0$ or the corresponding $\arg\max_{t \in T}\{g(x,t)\}$ for any $x$.

Several algorithms have been proposed to solve semi-infinite linear and semi-infinite convex programming problems, including cutting plane methods, local reduction methods, exchange methods, and homotopy methods. See, for example, (López and Still, 2007) for a recent review on semi-infinite convex programming, including an overview on numerical methods with plenty of references.

Most existing algorithms consider only linear problems, appealing to the fact that the general convex problem (SICP) is equivalent to the semi-infinite linear programming problem

$$\begin{array}{rl} \text{minimize} & x_0 \\ \text{subject to} & u^{\mathrm{T}} x - g^*_t(u) \le 0 \quad \forall\, t \in T \text{ and } u \in \operatorname{dom} g^*_t \\ & x \in X, \end{array} \tag{SILP}$$

where $g^*_t$ denotes the conjugate function of $g(\cdot, t)$. We contend, however, that this transformation is usually very ineffective, because if $X$ is $n$-dimensional, $T$ is $d$-dimensional, and (as it is very

often the case) $d \ll n$, then the index set in the semi-infinite constraint set increases from $d$ to the considerably higher $d + n$. Also, the set $T$ and the function $g$ might have special properties that allow us to find violated inequalities $g(x, t) \leq 0$ relatively easily; a property that may not be inherited by the set $\{(t, u) \,|\, t \in T, u \in \operatorname{dom} g_t^*\}$ and the conjugate function $g^*$ in the inequality constraints of (SILP). This is also the case in our motivating application.

Another family of semi-infinite convex problems where the use of cutting surfaces is attractive consists of problems where $X$ is high-dimensional non-polyhedral set, whose polyhedral approximation to $X$ is expensive to construct. In this case, any advantage gained from the linear reformulation of the semi-infinite constraints disappears, as (SILP) still remains a nonlinear convex program.

Our algorithm is motivated by the "central cutting plane" algorithm of (Kortanek and No, 1993) for convex problems, which in turn is an extension of Gribik's algorithm (Gribik, 1979). Gribik essentially gave the same algorithm as Kortanek and No, but only for semi-infinite linear programming problems. This algorithm has been the prototype of several cutting plane algorithms in the field, and has been improved in various ways, such as in the "accelerated central cutting plane" method of (Betrò, 2004).

Our main contribution from the perspective of semi-infinite programming is that we extend the central cutting plane algorithm to a cutting surface algorithm allowing non-linear convex cuts. The possibility of dropping cuts is retained, although in our numerical examples we always found optimal solutions very quickly, before dropping cuts was necessary for efficiency. Although cutting surface algorithms have a general convex master problem to solve in each iteration instead of a linear programming problem, this difference diminishes in the presence of other convex constraints defining $X$ or a non-linear objective function.

The outline of the paper is as follows. We proceed by describing our algorithm in Section 2, and proving its correctness in Section 3. Distributionally robust optimization is reviewed in Section 4, where we also give the semi-infinite convex formulation of this problem solvable using our cutting surface algorithm, combined with an algorithm that finds optimal probability distributions satisfying moment constraints. Computational results, which include both standard semi-infinite convex benchmark problems and distributionally robust utility maximization problems, follow in Section 5; with concluding remarks in Section 6.

## 2 A central cutting surface algorithm

The pseudo-code of our cutting surface algorithm is given in Algorithm 1. A few remarks are in order before we proceed to proving its correctness.

First, correctness precisely means that the algorithm computes an $\varepsilon$-optimal solution to (SICP) as long as Assumption 2 is satisfied with the same $\varepsilon$.

Throughout the algorithm, $y^{(k-1)}$ is the best $\varepsilon$-feasible solution found so far (or the initial vector $y^{(0)}$), and its first coordinate, $y_0^{(k-1)}$ is an upper bound on the objective function value of the best $\varepsilon$-feasible point. The initial value of $y_0^{(0)}$ is an arbitrary upper bound $U$ on this optimum; the other components of $y^{(0)}$ may be initialized arbitrarily.

In Step 2 of the algorithm we attempt to improve on the current upper bound by as much as possible and identify a "central" point $x^{(k)}$ that satisfies all the added inequalities with a large slack. The algorithm stops in Step 3 when no such improvement is possible.

In each iteration $k$, either a new cut is added in Step 5 that cuts off the last, infeasible, $x^{(k)}$ (a *feasibility cut*), or it is found that $x^{(k)}$ is an $\varepsilon$-feasible solution, and the best found $\varepsilon$-feasible solution $y^{(k)}$ is updated in Step 6 (an *optimality cut*). In either case, some inactive cuts are dropped in the optional Step 7. The parameter $\beta$ adjusts how aggressively cuts are dropped; setting $\beta = \infty$ is equivalent to skipping this step altogether.

In Step 5 of every iteration $k$ a centering parameter $s^{(k)}$ needs to be chosen. There are different

strategies to select $s^{(k)}$, we mention only two of them. One possibility is to find a subgradient $d \in \partial_x g(x^{(k)}, t^{(k)})$ and set $s^{(k)} = \|d\|$, or $s^{(k)} = \alpha\|d\|$ with an arbitrary $\alpha \in (0,1)$. Another possibility, also applicable when a subgradient is difficult to find, is to set the same nonnegative constant $s^{(k)} = s < B$ in every iteration. The choice $s = 0$ results in a simple cutting surface algorithm with no centering. Below we prove that Algorithm 1 converges for both strategies.

**Algorithm 1** (Central cutting surface algorithm)**.**

Parameters: a strict upper bound $U$ on the optimal objective function value of (SICP); a tolerance $\varepsilon \geq 0$ for which Assumption 2 holds; a $B > 0$ for which Assumption 1 holds; and an arbitrary $\beta > 1$.

Step 1. (Initialization.) Set $k = 1$, $y^{(0)} = (U, 0, \ldots, 0) \in \mathbb{R}^n$, and $J^{(0)} = \emptyset$.

Step 2. (Solve master problem.) Determine the optimal solution $(x^{(k)}, \sigma^{(k)})$ of the optimization problem

$$\begin{array}{rl} \text{maximize} & \sigma \\ \text{subject to} & x_0 + \sigma \leq y_0^{(k-1)} \\ & g(x, t^{(j)}) + \sigma s^{(j)} \leq 0 \quad \forall\, j \in J^{(k-1)} \\ & x \in X. \end{array} \tag{1}$$

Step 3. (Optimal solution?) If $\sigma^{(k)} = 0$, stop and return $y^{(k-1)}$.

Step 4. (Feasible solution?) Find a $t^{(k)} \in T$ satisfying $g(x^{(k)}, t^{(k)}) > 0$ if possible.
If no such $t^{(k)}$ is found, go to Step 6.

Step 5. (Feasibility cut.) Set $J^{(k)} = J^{(k-1)} \cup \{k\}$ and $y^{(k)} = y^{(k-1)}$; choose a centering parameter $0 \leq s^{(k)} \leq B$. (See the text for different strategies.)
Go to Step 7.

Step 6. (Optimality cut; update best known $\varepsilon$-feasible solution.) Set $J^{(k)} = J^{(k-1)}$ and $y^{(k)} = x^{(k)}$.

Step 7. (Drop cuts.) Let $D = \{j \,|\, \sigma^{(j)} \geq \beta\sigma^{(k)} \text{ and } g(x^{(k)}) + \sigma^{(k)} s^{(j)} < 0\}$, and set $J^{(k)} = J^{(k)} \setminus D$.

Step 8. Increase $k$ by one, and go to Step 2.

# 3 Convergence of the algorithm

We show the correctness of the algorithm by proving the following theorems. We tacitly assume that the centering parameters $s^{(k)}$ are chosen in Step 5 according to one of the two strategies mentioned above.

**Theorem 1.** *Suppose that Algorithm 1 terminates in the $k$th iteration. Then $y^{(k-1)}$ is an $\varepsilon$-optimal solution to* (SICP).

**Theorem 2.** *Suppose that Algorithm 1 does* not *terminate. Then there exists an index $\hat{k}$ such that the sequence $(y^{(\hat{k}+i)})_{i=1,2,\ldots}$ consists entirely of $\varepsilon$-feasible solutions.*

**Theorem 3.** *Suppose that Algorithm 1 does* not *terminate. Then the sequence* $(y^{(k)})_{k=1,2,\dots}$ *has an accumulation point, and each accumulation point is an* $\varepsilon$*-optimal solution to* (SICP).

Therefore, the algorithm either finds an $\varepsilon$-optimal solution after finitely many iterations, or approaches one in the limit. Even in the second case, the $\varepsilon$-optimal solution is approached through a sequence of (eventually) $\varepsilon$-feasible solutions.

We start the proof by a series of simple observations.

**Lemma 4.** *If* $y^{(\hat{k})}$ *is* $\varepsilon$*-feasible solution to* (SICP) *for some* $\hat{k}$*, then for every* $k \geq \hat{k}$*,* $y^{(k)}$ *is also* $\varepsilon$*-feasible.*

*Proof.* If the point $x^{(k)}$ found in Step 2 is not $\varepsilon$-feasible, then a feasibility cut is found, and in Step 5 $y^{(k)}$ is set to be the last $\varepsilon$-feasible solution found. Otherwise $y^{(k)} = x^{(k)}$, set in Step 6, is $\varepsilon$-feasible. □

**Lemma 5.** *Suppose that in the beginning of the* $k$*th iteration we have* $\delta \stackrel{\text{def}}{=} y_0^{(k-1)} - x_0^* > 0$*, where* $x^*$ *is an optimal solution of* (SICP)*. Then there exists a* $\sigma_0 = \sigma_0(\delta) > 0$ *(a function of only* $\delta$*, but not of* $k$*), such that in the optimal solution of* (1) *in Step 2 we have*

$$\sigma^{(k)} \geq \sigma_0(\delta) > 0.$$

*Proof.* Let $\bar{x}$ be the Slater point whose existence is required by Assumption 1, and consider the points $x_\lambda = \lambda\bar{x} + (1-\lambda)x^*$ for $\lambda \in (0,1]$. Because of the Slater property of $\bar{x}$ and the feasibility of $x^*$, $x_\lambda$ is a feasible solution of (1) in every iteration for every $\lambda \in (0,1]$, and it satisfies the inequalities

$$\begin{aligned} g(x_\lambda, t^{(j)}) + \frac{\lambda}{B} s^{(j)} &\leq \lambda g(\bar{x}, t^{(j)}) + (1-\lambda) g(x^*, t^{(j)}) + \lambda \\ &= \lambda(g(\bar{x}, t^{(j)}) + 1) + (1-\lambda) g(x^*, t^{(j)}) \\ &\leq 0 \quad \text{for all } j \in J^{(0)} \cup J^{(1)} \cup \cdots, \end{aligned}$$

using the convexity of $g$ and $s^{(j)} \leq B$ in the first inequality and the Slater condition in the second. In the $k$th iteration, if $y_0^{(k-1)} - x_0^* = \delta > 0$, then $x_\lambda$ also satisfies the inequality

$$y_0^{(k-1)} - (x_\lambda)_0 = (x_0^* + \delta) - (\lambda\bar{x}_0 + (1-\lambda)x_0^*) = \delta - \lambda(\bar{x}_0 - x_0^*) \geq \delta/2$$

for every $\lambda > 0$ sufficiently small to satisfy $0 \leq \lambda(\bar{x}_0 - x_0^*) \leq \delta/2$.

Denoting by $\lambda_0$ such a sufficiently small value of $\lambda$, and letting

$$\sigma_0 \stackrel{\text{def}}{=} \min(\lambda_0/B, \delta/2),$$

we conclude that the pair $(x_{\lambda_0}, \sigma_0)$ is a feasible solution to (1), hence the optimal solution to (1) also satisfies $\sigma^{(k)} \geq \sigma_0 > 0$. □

Our final lemma is required only for the proof of Theorem 3.

**Lemma 6.** *Suppose that Algorithm 1 does* not *terminate. Then the sequence* $(\sigma^{(k)})_{k=1,2,\dots}$ *decreases monotonically to zero, and the sequence* $(y_0^{(k)})_{k=1,2,\dots}$ *is also monotone decreasing.*

*Proof.* For every $k$, $\sigma^{(k)} \geq 0$, because the pair $(x, \sigma) = (x^*, 0)$ is a feasible solution in each iteration. From this, and the first inequality of (1), the monotonicity of $(y_0^{(k)})_{k=1,2,\dots}$ follows.

Since $(y_0^{(k)})_{k=1,2,\dots}$ is monotone decreasing and only inactive cuts are dropped from (1) in Step 7, the sequence $(\sigma^{(k)})_{k=1,2,\dots}$ is monotone non-increasing. Therefore $(\sigma^{(k)})_{k=1,2,\dots}$ is convergent.

Let us assume (by contradiction) that $\sigma^{(k)} \searrow \sigma_0 > 0$. Then for a sufficiently large $\hat{k}$, $\sigma^{(k)} < \sigma_0\beta$ for every $k \geq \hat{k}$, implying that no cuts are dropped in Step 7 beyond the $\hat{k}$th iteration. Consider the optimal $x^{(j)}$ and $x^{(k)}$ obtained in Step 2 of the $j$th and $k$th iteration, with $k > j \geq \hat{k}$. There are two cases, based on whether a feasibility cut $g(x^{(j)}, t^{(j)}) > 0$ is found in Step 4 of the $j$th iteration or not.

If a feasibility cut is not found in the $j$th iteration, then

$$x_0^{(k)} = y_0^{(k-1)} - \sigma^{(k)} \leq y_0^{(j)} - \sigma^{(k)} = x_0^{(j)} - \sigma^{(k)}$$

follows from the first constraint of (1) in the $k$th iteration, therefore

$$\|x^{(k)} - x^{(j)}\| \geq \sigma^{(k)} \geq \sigma_0.$$

If a feasibility cut is found in the $j$th iteration, then on one hand we have

$$g(x^{(j)}, t^{(j)}) > 0,$$

and because this cut is not dropped later on, from (1) in the $k$th iteration we also have

$$g(x^{(k)}, t^{(j)}) + \sigma^{(k)} s^{(j)} \leq 0.$$

From these two inequalities we obtain

$$0 \leq \sigma^{(k)} s^{(j)} < g(x^{(j)}, t^{(j)}) - g(x^{(k)}, t^{(j)}) \leq -(d^{(j)})^{\mathrm{T}}(x^{(k)} - x^{(j)}) \leq \|d^{(j)}\| \cdot \|x^{(k)} - x^{(j)}\|$$

for every $d^{(j)} \in \partial_x g(x^{(j)}, t^{(j)})$, using the convexity of $g(\cdot, t^{(j)})$ and the Cauchy-Schwarz inequality. Note that the strict inequality implies $d^{(j)} \neq 0$. Comparing the left and right-hand sides we obtain

$$\sigma^{(k)} s^{(j)} / \|d^{(j)}\| < \|x^{(k)} - x^{(j)}\|.$$

From this inequality it follows that with either strategy mentioned above for selecting the centering parameters $s^{(j)}$ we have a $\sigma_1 > 0$ independent of $j$ and $k$ satisfying $\sigma_1 < \|x^{(k)} - x^{(j)}\|$.

In summary, regardless of whether we add a feasibility or an optimality cut in iteration $j$, we have that for every $k > j \geq \hat{k}$,

$$\|x^{(k)} - x^{(j)}\| \geq \min(\sigma_0, \sigma_1) > 0,$$

contradicting the assumption that the sequence $(x^{(k)})_{k=1,2,\ldots}$ is bounded, and therefore has an accumulation point. □

With these lemmas we are ready to prove our main theorems.

*Proof of Theorem 1.* Suppose that the algorithm terminates in the $k$th iteration. First assume by contradiction that $y^{(k-1)}$ is not an $\varepsilon$-feasible solution to (SICP). Then by Lemma 4, none of the points $y^{(0)}, \ldots, y^{(k-2)}$ are $\varepsilon$-feasible, therefore the upper bound in the first constraint of (1) is $y_0^{(k-1)} = U$ (a strict upper bound on the optimum) in every iteration. Hence, by Lemma 5, $\sigma^{(k)} > 0$, contradicting the assumption that the algorithm terminated. Therefore $y^{(k-1)}$ is $\varepsilon$-feasible.

Now suppose that $y^{(k-1)}$ is $\varepsilon$-feasible, but it is not $\varepsilon$-optimal, that is, $y^{(k-1)} > x_0^*$. Then by Lemma 5 we have $\sigma^{(k)} > 0$ for every $k$, contradicting the assumption that the algorithm terminated. □

*Proof of Theorem 2.* Using Lemma 4 it is sufficient to show that at least one $y^{(k)}$ is $\varepsilon$-feasible. Suppose otherwise, then no $x^{(k)}$ or $y^{(k)}$ obtained throughout the algorithm is $\varepsilon$-feasible. Therefore, the upper bound on the first constraint of (1) remains $y^{(k-1)} = U$ (a strict upper bound on the optimum) in every iteration. Invoking Lemma 5 we have that $\sigma^{(k)} \geq \sigma_0(U - x_0^*) > 0$, contradicting Lemma 6. □

*Proof of Theorem 3.* The compactness of the feasible set of (SICP) implies that if the algorithm does not terminate, then the sequence $(x^{(k)})_{k=1,2,\dots}$ has at least one accumulation point, and so does its subsequence $(y^{(k)})_{k=1,2,\dots}$. From Theorem 2 we also know that this sequence eventually consists entirely of $\varepsilon$-feasible points, therefore every accumulation point of the sequence $(y^{(k)})_{k=1,2,\dots}$ is also $\varepsilon$-feasible (using that the set of $\varepsilon$-feasible solutions is also compact).

Let $\hat{y}$ be one of the accumulation points, and suppose by contradiction that $\hat{y}$ is not $\varepsilon$-optimal, that is, $\hat{y}_0 > x_0^*$. Let $\delta = (\hat{y}_0 - x_0^*)/2$, where $x^*$ denotes, as before, an optimal solution to (SICP). Using Lemma 4 and the assumption $\delta > 0$, there exists a sufficiently large $\hat{k}$ such that for every $k > \hat{k}$, $y^{(k)}$ is an $\varepsilon$-feasible solution to (SICP), and $y_0^{(k-1)} \geq x_0^* + \delta$. Invoking Lemma 5 we find that in this case there exists a $\sigma_0 > 0$ such that $\sigma^{(k)} \geq \sigma_0$ for every $k > \hat{k}$, contradicting Lemma 6. □

# 4 Distributionally robust and moment robust optimization

Stochastic optimization and robust optimization are two families of optimization models introduced to tackle decision making problems under uncertain data. Broadly speaking, robust optimization handles the uncertainty by optimizing for the worst case within a prescribed set of scenarios, whereas stochastic optimization assumes that the uncertain data follows a specified distribution. *Distributionally robust optimization*, introduced in (Scarf, 1957), can be seen as a combination of these approaches, where the optimal decisions are sought for the worst case within a prescribed set of probability distributions that the data might follow. The term *robust stochastic programming* is also often used to describe optimization models of the same form.

Formally, let the uncertain data be described by a random variable supported on a set $\Xi \subseteq \mathbb{R}^d$, following an unknown distribution $P$ from a set of probability distributions $\mathfrak{P}$. Then a general distributionally robust optimization problem is an optimization model of the form

$$\min_{x \in X} \max_{P \in \mathfrak{P}} \mathbb{E}_P[H(x)], \quad \text{or (equivalently)} \quad \min_{x \in X} \max_{P \in \mathfrak{P}} \int_{\xi \in \Xi} h(x, \xi) P(d\xi), \tag{DRO}$$

where $H$ is a random cost or disutility function we seek to minimize, $h$ is its probability mass function; the argument $x$ of $H$ and $h$ is our decision vector. We assume that all expectations (integrals) exist and that the minima and maxima are well-defined.

We might also consider optimization problems with *robust stochastic constraints*, that is, constraints of the form

$$\mathbb{E}_P[G(x)] \leq 0 \quad \forall\, P \in \mathfrak{P}$$

with some convex function $G$. The algorithm presented in this section is applicable verbatim to such problems, but to keep the presentation simple, we consider only the simpler form, (DRO) in this section. However, we provide a numerical example with robust stochastic constraints in Example 4.

With the above notation, a general stochastic optimization problem is simply (DRO) with a singleton $\mathfrak{P}$, while a standard robust optimization problem is (DRO) with a set $\mathfrak{P}$ that consists only of probability distributions supported on one point within $\Xi$.

One can also view the general distributionally robust optimization problem not only as a common generalization of robust and stochastic optimization, but also as a risk-averse optimization model with an adjustable rate of risk-aversion. To see this, consider a nested sequence of sets of probability distributions $\mathfrak{P}_0 \supseteq \mathfrak{P}_1 \supseteq \cdots$, where $\mathfrak{P}_0$ is the set of all probability distributions supported on $\Xi$, and $\mathfrak{P}_\infty \stackrel{\text{def}}{=} \cap_{i=0}^{\infty} \mathfrak{P}_i$ is a singleton set. In the corresponding sequence of problems (DRO), the first one is the classic robust optimization problem, which is the most conservative (risk-averse) of all, optimizing against the worst case, and the last one is the classic stochastic optimization problem, where the optimization is against a fixed distribution. At the intermediate levels the models correspond to decreasing levels of risk-aversion.

Such a sequence of problems can be constructed in a natural way, for instance, by increasingly constraining an increasing number of moments of the underlying probability distribution. See Example 4 for a simple, abstract example. In a more realistic, data-driven setting, such bounds can be obtained by computing confidence intervals around the sample moments of the empirical distribution. In this case the (DRO) problem is also called *moment robust optimization* problem.

In most applications since Scarf's aforementioned work the set of distributions $\mathfrak{P}$ is defined by setting bounds on the moments of $P$; recent examples include (Delage and Ye, 2010), (Bertsimas et al., 2010), and (Mehrotra and Zhang, 2013). Simple lower and upper bounds (confidence intervals and ellipsoids) on moments of arbitrary order are easily obtained using standard statistical methods; (Delage and Ye, 2010) describes an alternative method to derive bounds on the first and second moments. However, to the best of our knowledge, no algorithm has been proposed until now to solve (DRO) with sets $\mathfrak{P}$ defined by constraints on moments of order higher than two.

By constraining all the moments of the distributions in $\mathfrak{P}$, (DRO) simplifies to a stochastic programming problem. In the theorem below we use the notation $m_k(P)$ to denote the moment of $P$ corresponding to the multi-index $k = (k_1, \ldots, k_n)$, that is, $m_k(P) \stackrel{\text{def}}{=} \int_\Xi \xi_1^{k_1} \cdots \xi_n^{k_n} P(d\xi)$.

**Theorem 7.** *Let $h$, $X$ and $\Xi$ as above, and assume that $\Xi$ is bounded and $h$ is continuous. Let $P$ be a probability distribution supported on $\Xi$, with moments $m_k(P)$. For each $i = 0, 1, \ldots,$ let $\mathfrak{P}_i$ denote the set of probability distributions $Q$ supported on $\Xi$ whose moments $m_k(Q)$ satisfy $m_k(Q) = m_k(P)$ for every multi-index $k$ with $0 \leq k_1 + \cdots + k_n \leq i$. Finally, for each $i = 0, 1, \ldots$ define the moment-robust optimization problem* ($\text{DRO}_i$) *as follows:*

$$\min_{x \in X} \max_{Q \in \mathfrak{P}_i} \int_{\xi \in \Xi} h(x, \xi) Q(d\xi). \tag{$\text{DRO}_i$}$$

*Then the sequence of the optimal objective function values of* ($\text{DRO}_i$) *converges to the optimal objective function value of the stochastic program*

$$\min_{x \in X} \int_{\xi \in \Xi} h(x, \xi) P(d\xi). \tag{SP}$$

*Proof.* Let $z_i$ denote the optimal objective function value of ($\text{DRO}_i$) for every $i$, and let $z_{SP}$ denote the optimal objective function value of (SP); we want to show that $\lim_{i \to \infty} z_i \to z_{SP}$.

The sequence $(z_i)_{i=0,1,\ldots}$ is convergent because it is monotone decreasing (since $\mathfrak{P}_0 \supseteq \mathfrak{P}_1 \supseteq \cdots \supseteq \cap_{i=0}^m \mathfrak{P}_i$) and it is bounded from below by $z_{SP}$:

$$\begin{aligned} z_i &= \min_{x \in X} \max_{Q \in \mathfrak{P}_i} \int_{\xi \in \Xi} h(x, \xi) Q(d\xi) \geq \max_{Q \in \mathfrak{P}_i} \min_{x \in X} \int_{\xi \in \Xi} h(x, \xi) Q(d\xi) \\ &\geq \min_{x \in X} \int_{\xi \in \Xi} h(x, \xi) P(d\xi) = z_{SP}. \end{aligned} \tag{2}$$

Consider now the stochastic programming problem (SP). Denote by $\bar{x}$ one of its optimal solutions, and let

$$\bar{z}_i \stackrel{\text{def}}{=} \max_{Q \in \mathfrak{P}_i} \int_{\xi \in \Xi} h(\bar{x}, \xi) Q(d\xi).$$

Obviously, $z_i \leq \bar{z}_i$ for every $i$. In view of (2), it suffices to show that $\bar{z}_i \to z_{SP}$.

For every $i$, choose an arbitrary $\bar{Q}_i \in \arg\max_{Q \in \mathfrak{P}_i} \int_{\xi \in \Xi} h(\bar{x}, \xi) Q(d\xi)$. Since the moments of $\bar{Q}_i$ and $P$ agree up to order $i$, we have that

$$\int_{\xi \in \Xi} p(\xi) \bar{Q}_i(d\xi) = \int_{\xi \in \Xi} p(\xi) P(d\xi) \tag{3}$$

for every polynomial $p$ of total degree at most $i$.

By assumption, the function $h(\bar{x}, \cdot)$ is continuous on the closed and bounded set $\Xi$. Let $p_j$ denote its best uniform polynomial approximation of total degree $j$; by the Weierstrass approximation theorem we have that for every $\varepsilon > 0$ there exists a degree $j(\varepsilon)$ such that $\max_{\xi \in \Xi} |h(\bar{x}, \xi) - p_{j(\varepsilon)}(\xi)| < \varepsilon$, and therefore,

$$\int_{\xi \in \Xi} |h(\bar{x}, \xi) - p_{j(\varepsilon)}(\xi)| \bar{Q}_i(d\xi) < \varepsilon \text{ and } \int_{\xi \in \Xi} |h(\bar{x}, \xi) - p_{j(\varepsilon)}(\xi)| P(d\xi) < \varepsilon. \tag{4}$$

With this $j(\varepsilon)$, every $i \geq j(\varepsilon)$ satisfies the inequalities

$$\begin{aligned}
|\bar{z}_i - z_{SP}| &= \left| \int_{\xi \in \Xi} h(\bar{x}, \xi) \bar{Q}_i(d\xi) - \int_{\xi \in \Xi} h(\bar{x}, \xi) P(d\xi) \right| \leq \\
&\leq \left| \int_{\xi \in \Xi} h(\bar{x}, \xi) \bar{Q}_i(d\xi) - \int_{\xi \in \Xi} p_{j(\varepsilon)}(\xi) \bar{Q}_i(d\xi) \right| + \left| \int_{\xi \in \Xi} p_{j(\varepsilon)}(\xi) \bar{Q}_i(d\xi) - \int_{\xi \in \Xi} h(\bar{x}, \xi) P(d\xi) \right| = \\
&= \left| \int_{\xi \in \Xi} \left( h(\bar{x}, \xi) - p_{j(\varepsilon)}(\xi) \right) \bar{Q}_i(d\xi) \right| + \left| \int_{\xi \in \Xi} p_{j(\varepsilon)}(\xi) P(d\xi) - \int_{\xi \in \Xi} h(\bar{x}, \xi) P(d\xi) \right| \leq \\
&\leq \int_{\xi \in \Xi} \left| h(\bar{x}, \xi) - p_{j(\varepsilon)}(\xi) \right| \bar{Q}_i(d\xi) + \int_{\xi \in \Xi} \left| h(\bar{x}, \xi) - p_{j(\varepsilon)}(\xi) \right| P(d\xi) < 2\varepsilon,
\end{aligned}$$

using the triangle inequality, (3), (4), and the triangle inequality again. From the inequality between the left- and the right-hand side it immediately follows that $\lim_{i \to \infty} \bar{z}_i = z_{SP}$, as claimed. □

It is interesting to note that in the above theorem the function $h(x, \cdot)$ could be replaced by any continuous function $f \colon \Xi \mapsto \mathbb{R}$ that does not depend on $x$, proving that

$$\lim_{i \to \infty} \int_{\xi \in \Xi} f(\xi) \bar{Q}_i(d\xi) = \int_{\xi \in \Xi} f(\xi) P(d\xi)$$

for *every continuous function* $f \colon \Xi \mapsto \mathbb{R}$; in other words, the sequence of measures $\bar{Q}_0, \bar{Q}_1, \ldots$ converges weakly to $P$, and so does every other sequence of measures in which the moments of the $i$th measure agree with the moments of $P$ up to order $i$. Therefore, Theorem 7 can be seen as a generalization of the well-known theorem that the moments of a probability distribution with compact support uniquely determine the distribution. For distributions with unbounded support a statement similar to Theorem 7 can only be made if the moments in question uniquely determine the probability distribution $P$. A collection of sufficient conditions under which infinite moment sequences determine a distribution can be found in the recent review article (Kleiber and Stoyanov, 2013).

Recent research has focused on conditions under which (DRO) with moment constraints can be solved in polynomial time. Delage and Ye (2010) consider an uncertainty set defined via a novel type of confidence set around the mean vector and covariance matrix, and show that (DRO) with uncertainty sets of this type can be solved in polynomial time (using the ellipsoid method) for a class of probility mass functions $h$ that are convex in $x$ but concave in $\xi$. Mehrotra and Zhang (2013) extend this result by providing polynomial time methods (using semidefinite programming) for least squares problems, which are convex in both $x$ and $\xi$. The uncertainty sets in their formulation are defined through bounds on the measure, bounds on the distance from a reference measure, and moment constraints of the same form as considered in (Delage and Ye, 2010). Bertsimas et al. (2010) considers two-stage robust stochastic models in which risk aversion is modeled in a moment robust framework using first and second order moments.

Our aim in this section is to show that Algorithm 1 is applicable to solving (DRO) for every objective $h$ that is convex in $x$ (for every $\xi \in \Xi$) as long as the set $\mathfrak{P}$ is defined through bounds on the moments of $P$, and $X$ is convex and bounded. Unlike in the papers cited above, bounds can be imposed on moments of arbitrary order, not only on the first and second moments. This allows the decision maker to shape the distributions in $\mathfrak{P}$ better. Moments up to order 4 are easily interpretable and have been used to strengthen the formulation of stochastic programming models. (Høyland et al., 2003) provides a heuristic to improve stochastic programming models using first and second order moments as well as marginal moments up to order 4. The authors in (Mehrotra and Papp, 2013) give a scenario generation algorithm for stochastic programming problems using moment bounds (of arbitrary order). In the remainder of this section we show that an extension of this scenario generation algorithm in combination with Algorithm 1 yields a solution to (DRO) with moment constraints.

## 4.1 Distributionally robust optimization as a semi-infinite convex program

Consider the second (integral) form of (DRO) with a probability mass function $h$ that is convex in $x$ for every $\xi$. If $\Xi$ and $X$ are bounded sets, the optimal objective function value can be bracketed in an interval $[z_{\min}, z_{\max}]$, and the problem can be written as a semi-infinite convex optimization problem

$$\begin{aligned} \text{minimize} \quad & z \\ \text{subject to} \quad & -z + \int_{\Xi} h(x,\xi) P(d\xi) \leq 0 \quad \forall\, P \in \mathfrak{P} \\ & (z,x) \in [z_{\min}, z_{\max}] \times X, \end{aligned} \tag{5}$$

which is a problem of the form (SICP); the set $\mathfrak{P}$ plays the role of $T$; $z$ plays the role of $x_0$. Note that the in the above problem the index set of the constraints is not a low-dimensional set, as it is common in semi-infinite convex programming, but an infinite dimensional set. Therefore, we cannot assume without further justification that violated inequalities in (SICP) can be easily found.

It can be verified, however, that this problem satisfies Assumption 1 as long as $h$ has bounded subdifferentials on the boundary of $\Xi$. Under this assumption on $h$ Algorithm 1 is applicable to (5) as long as Assumption 2 is also valid. The latter assumption translates to being able to find

$$\sup_{P \in \mathfrak{P}} \int_{\Xi} h(x,\xi) P(d\xi) \tag{6}$$

(in which $x$ is a parameter) within a prescribed $\varepsilon > 0$ error. It is this problem that we shall concentrate on.

In the moment-robust formulation of (DRO) the set $\mathfrak{P}$ is defined via bounds on some (not necessarily polynomial) moments: given continuous $\Xi \mapsto \mathbb{R}$ *basis functions* $f_1, \ldots, f_N$, and a lower and upper bound vector $\ell$ and $u$ on the corresponding moments, we set

$$\mathfrak{P} = \left\{ P \;\middle|\; \int_{\Xi} f_i(\xi) P(d\xi) \in [\ell_i, u_i],\ i = 1, \ldots, N \right\}. \tag{7}$$

In typical applications the $f_i$ form a basis of low-degree polynomials. For example, if we wish to optimize for the worst-case distribution among distributions having prescribed mean vector and covariance matrix, then the $f_i$ can be the $n$-variate monomials up to degree two (including the constant 1 function), and $\ell = u$ is the vector of prescribed moments (including the "zeroth moment", 1).

Without loss of generality we shall assume that $f_1$ is the constant one function, and $\ell_1 = u_1 = 1$. We will also use the shorthand $f$ for the vector-valued function $(f_1, \ldots, f_N)^{\mathrm{T}}$.

Our first observation is that while searching for an $\varepsilon$-optimal $P$ in (6), it is sufficient to consider finitely supported distributions.

**Theorem 8.** *For every $\varepsilon > 0$, the optimization problem (6) has an $\varepsilon$-optimal distribution supported on not more than $N + 2$ points.*

*Proof.* For every $z \in \mathbb{R}$, the set

$$L_z = \left\{ (v, w) \in \mathbb{R}^N \times \mathbb{R} \;\middle|\; \exists P : v = \int_\Xi f(\xi) P(d\xi), w = \int_\Xi h(x, \xi) P(d\xi),\ \ell \le v \le u, w \ge z \right\}$$

is an $(N + 1)$-dimensional convex set contained in the convex hull of the points

$$\{(f_1(\xi), \ldots, f_N(\xi), h(x, \xi))^{\mathrm{T}} \mid \xi \in \Xi\}.$$

Therefore by Carathéodory's theorem, as long as there exists a $(v, w) \in L_z$, there also exist $N + 2$ points $\xi_1, \ldots, \xi_{N+2}$ in $\Xi$ and nonnegative weights $w_1, \ldots, w_{N+2}$ satisfying

$$v = \sum_{k=1}^{N+2} w_k f(\xi_k) \quad \text{and} \quad w = \sum_{k=1}^{N+2} w_k h(x, \xi_k). \qquad \square$$

## 4.2 Randomized column generation

The main result of (Mehrotra and Papp, 2013) is that whenever the set $\mathfrak{P}$ of distributions is defined as in (7), a column generation algorithm using randomly sampled columns can be used to find a distribution $P \in \mathfrak{P}$ supported on at most $N$ points. In other words, a feasible solution to (6) can be found using a randomized column generation algorithm. In this section we generalize this result to show that (6) can also be solved to optimality within a prescribed $\varepsilon > 0$ accuracy using randomized column generation. The formal description of the complete algorithm is given in Algorithm 2, in the remainder of this section we provide a short informal description and the proof of correctness.

If $\Xi$ is a finite set, then the optimization problem (6) is a linear program whose decision variables are the weights $w_i$ that the distribution $P$ assigns to each point $\xi_i \in \Xi$. In an analogous fashion, (6) in the general case can be written as a semi-infinite linear programming problem with a weight function $w \colon \Xi \mapsto \mathbb{R}_0^+$ as the variable. The corresponding column generation algorithm for the solution of (6) is then the following.

We start with a finite candidate node set $\{\xi_1, \ldots, \xi_K\}$ that supports a feasible solution. Such points can be obtained (for instance) using Algorithm 1 in (Mehrotra and Papp, 2013).

At each iteration we take our current candidate node set and solve the auxiliary linear program

$$\max_{w \in \mathbb{R}^K} \left\{ \sum_{k=1}^{K} w_k h(x, \xi_k) \;\middle|\; \ell \le \sum_{k=1}^{K} w_k f(\xi_k) \le u,\ w \ge 0 \right\} \tag{8}$$

and its dual problem

$$\min_{(p_+, p_-) \in \mathbb{R}^{2N}} \left\{ p_+^{\mathrm{T}} u - p_-^{\mathrm{T}} \ell \;\middle|\; (p_+ - p_-)^{\mathrm{T}} f(\xi_k) \ge h(x, \xi_k)\ (k = 1, \ldots, K);\ p_+ \ge 0, p_- \ge 0 \right\}. \tag{9}$$

Note that by construction of the initial node set, the primal problem is always feasible, and since it is also bounded, both the primal and dual optimal solutions exist.

Let $\hat{w}$ and $(\hat{p}_+, \hat{p}_-)$ be the obtained primal and dual optimal solutions; the reduced cost of a point $\xi \in \Xi$ is then

$$\pi(\xi) \stackrel{\text{def}}{=} h(x,\xi) - (\hat{p}_+ - \hat{p}_-)^{\mathrm{T}} f(\xi). \tag{10}$$

As for every (finite or semi-infinite) linear program, if every $\xi \in \Xi$ has $\pi(\xi) \leq 0$, then the current primal-dual pair is optimal, that is, the discrete probability distribution corresponding to the points $\xi_k$ and weights $\hat{w}_k$ is an optimal solution to (6). Moreover, for problem (6) we have the following, stronger, fact.

**Theorem 9.** *Let $\xi_1, \ldots, \xi_K$, $\hat{w}$, and $\pi$ be defined as above, and let $\varepsilon \geq 0$ be given. If $\pi(\xi) \leq \varepsilon$ for every $\xi \in \Xi$, then the distribution defined by the support points $\xi_1, \ldots, \xi_K$ and weights $\hat{w}_1, \ldots, \hat{w}_K$ is an $\varepsilon$-optimal feasible solution to problem* (6).

*Proof.* The feasibility of the defined distribution follows from the definition of the auxiliary linear program (8), only the $\varepsilon$-optimality needs proof.

If the inequality $\pi(\xi) \leq \varepsilon$ holds for every $\xi \in \Xi$, then by integration we also have

$$\int_\Xi (\hat{p}_+ - \hat{p}_-)^{\mathrm{T}} f(\xi) P(d\xi) \geq \int_\Xi (h(x,\xi) - \varepsilon) P(d\xi) = \int_\Xi h(x,\xi) P(d\xi) - \varepsilon \tag{11}$$

for every probability distribution $P$. In particular, consider an optimal solution $P^*$ to (6) with $m^* \stackrel{\text{def}}{=} \int_{\xi \in \Xi} f(\xi) P^*(d\xi)$. Naturally, $\ell \leq m^* \leq u$, and so we have

$$\begin{aligned}\sum_{k=1}^{K} \hat{w}_k h(x,\xi_k) = p_+^{\mathrm{T}} u - p_-^{\mathrm{T}} \ell \geq (p_+ - p_-)^{\mathrm{T}} m^* = \\ = \int_\Xi (p_+ - p_-)^{\mathrm{T}} f(\xi) P^*(d\xi) \geq \int_\Xi h(x,\xi) P^*(d\xi) - \varepsilon,\end{aligned}$$

using strong duality for the primal-dual pair (8)-(9) in the first step, $\ell \leq m^* \leq u$ and the sign constraints on the dual variables in the second step, and inequality (11) in the last step. The inequality between the left- and right-hand sides of the above chain of inequalities is our claim. □

If we can find a $\xi$ with positive reduced cost, we can add it as $\xi_{K+1}$ to the candidate support set, and recurse. Unfortunately, finding the point $\xi$ with the highest reduced cost, or even deciding whether there exists a $\xi \in \Xi$ with positive reduced cost is NP-hard, even in the case when $\Xi = [0,1]^d$, $h$ is constant zero, and the $f_i$ are the monomials of degree at most two; this follows from the NP-hardness of quadratic optimization over the unit cube. A few special cases when a point with highest reduced cost can be found in polynomial time are shown in (Mehrotra and Papp, 2013), but these are rather restrictive for practical applications.

Our last observation is that if the functions $h(x,\cdot)$ and $f_i$ are continuously differentiable over the bounded $\Xi$, the reduced cost function (10) (as a function of $\xi$) also has bounded derivatives. Therefore, sufficiently many independent uniform random samples $\xi_j \in \Xi$ that result in $\pi(\xi_j) \leq 0$ will help us conclude that $\pi(\xi) \leq \varepsilon$ for every $\xi \in \Xi$ with high probability. In the following theorem $B(c,r)$ denotes the (Euclidean, $d$-dimensional) ball centered at $c$ with radius $r$.

**Theorem 10.** *Suppose the functions $h(x,\cdot)$ and $f_i$ are continuously differentiable over $\Xi$, and let $C$ be an upper bound on the gradient of the reduced cost function: $\max_{\xi \in \Xi} \|\nabla \pi(\xi)\| \leq C$. Furthermore, assume that a particular $\tilde{\xi} \in \Xi$ satisfies $\pi(\tilde{\xi}) > \varepsilon$. Then a uniformly randomly chosen $\xi \in \Xi$ satisfies $\pi(\xi) \leq 0$ with probability* at most $1 - p$, *where*

$$p = \min_{\xi \in \Xi} \operatorname{vol}(\Xi \cap B(\xi, \varepsilon/C)) / \operatorname{vol}(\Xi) > 0.$$

*In particular, if $\Xi \subseteq \mathbb{R}^d$ is a convex set satisfying $B(c_1, r) \subseteq \Xi \subseteq B(c_2, R)$ with some centers $c_1$ and $c_2$ and radii $r$ and $R$ we have*

$$p > (2\pi(d+2))^{-1/2} \left(\frac{r\varepsilon}{2RC}\right)^d.$$

*Proof.* If $\pi(\tilde{\xi}) > \varepsilon$, then $\pi(\xi) > 0$ for every $\xi$ in its neighborhood $\Xi \cap B(\tilde{\xi}, \varepsilon/C)$. Therefore, the assertion holds with $p(\varepsilon, C) = \min_{\xi \in \Xi} \operatorname{vol}(\Xi \cap B(\xi, \varepsilon/C))/\operatorname{vol}(\Xi)$. This minimum exists, because $\Xi$ is closed and bounded; and it is positive, because the intersection is a non-empty closed convex set for every center $\xi$.

To obtain the lower bound on $p$, we need to bound from below the volume of the intersection $\Xi \cap B(\xi, \varepsilon/C)$. Consider the right circular cone with apex $\xi$ whose base is the $(d-1)$-dimensional intersection of $B(c_1, r)$ and the hyperplane orthogonal to the line connecting $c_1$ and $\xi$. This cone is contained within $\Xi$, and all of its points are at distance $2R$ or less from $\xi$. Shrinking this cone with respect to the center $\xi$ with ratio $\varepsilon/(2RC)$ yields a cone contained in $\Xi \cap B(\xi, \varepsilon/C)$. Using the volume of this cone as a lower bound on $\operatorname{vol}(\Xi \cap B(\xi, \varepsilon/C))$ and the notation $V_d(r)$ for the volume of the $d$-dimensional ball of radius $r$, we get

$$\begin{aligned}\frac{\operatorname{vol}(\Xi \cap B(\xi, \varepsilon/C))}{\operatorname{vol}(\Xi)} &\geq \frac{(d+1)^{-1} V_{d-1}(r) r}{V_d(R)} \left(\frac{\varepsilon}{2RC}\right)^d = \frac{\pi^{(d-1)/2}\Gamma((d+2)/2)}{(d+1)\pi^{d/2}\Gamma((d+1)/2)} \left(\frac{\varepsilon r}{2RC}\right)^d \\ &= \pi^{-1/2} \frac{\Gamma((d+2)/2)}{2\Gamma((d+3)/2)} \left(\frac{\varepsilon r}{2RC}\right)^d > \pi^{-1/2} \cdot (2d+4)^{-1/2} \left(\frac{\varepsilon r}{2RC}\right)^d,\end{aligned}$$

with some lengthy (but straightforward) arithmetic in the last inequality, using the log-convexity of the gamma function. □

Theorem 10, along with Theorem 9, allows us to bound the number of uniform random samples $\xi \in \Xi$ we need to draw to be able to conclude with a fixed low error probability, that the optimal solution of (8) is an $\varepsilon$-optimal solution to (6). This is an explicit, although very conservative, bound: with $\hat{p}$ given in each iteration, and known global bounds on the gradients of $h$ and the components of $f$, an upper bound $C$ on $\|\nabla\pi(\cdot)\|$ can be easily computed in every iteration. (A global bound, valid in every iteration, can also be obtained whenever the dual variables $\hat{p}$ can be bounded a priori.) This provides the (probabilistic) stopping criterion for the column generation for Algorithm 2.

In order to use Theorem 10, we need an efficient algorithm to sample uniformly from the set $\Xi$. This is obvious if $\Xi$ has a very simple geometry, for instance, when $\Xi$ is a $d$-dimensional rectangular box, simplex, or ellipsoid. Uniform random samples can also be generated efficiently from general polyhedral sets given by their facet-defining inequalities and also from convex sets, using random walks with strongly polynomial mixing times. See, for example, the survey (Vempala, 2005) or (Kannan and Narayanan, 2012) for uniform sampling methods in polyhedra, and (Lovász and Vempala, 2006) for general convex sets; (Huang and Mehrotra, 2013) gives a more detailed and up-to-date list of references on uniform sampling on convex sets.

We can now conclude that the semi-infinite convex program formulation of (DRO) can be solved using Algorithm 1, with Algorithm 2 and an efficient uniform sampling method serving as a probabilistic version of the oracle required by Assumption 2.

# 5 Numerical results

## 5.1 Semi-infinite convex optimization problems

Most standard benchmark problems in the semi-infinite programming literature are linear. When the problem (SICP) is linear, Algorithm 1 reduces to the central cutting plane algorithm (except for

**Algorithm 2** (Randomized column generation method to solve (6)-(7))**.**

Parameters: $M$, the maximum number of random samples per iteration. (See the text for details on choosing this parameter.)

Step 1. Find a finitely supported feasible distribution to (6) using Algorithm 1 in (Mehrotra and Papp, 2013). Let $S = \{\xi_1, \ldots, x_K\}$ be its support.

Step 2. Solve the primal-dual pair (8)-(9) for the optimal $\hat{w}$, $\hat{p}_+$, and $\hat{p}_-$.

Step 3. Sample uniform random points $\xi \in \Xi$ until one with positive reduced cost $h(x, \xi) - (\hat{p}_+ - \hat{p}_-)^{\mathrm{T}} f(\xi)$ is found or the maximum number of samples $M$ is reached.

Step 4. If in the previous step a $\xi$ with positive reduced cost was found, add it to $S$, increase $K$, and return to Step 2. Otherwise stop.

our more general centering); therefore we only consider convex non-linear test problems from the literature. The results in this section are based on an implementation of the central cutting plane and central cutting surface algorithms using the AMPL modeling language and the MOSEK and CPLEX convex optimization software. The comparison between the algorithms is based solely on the number of iterations. The running times for all the examples were comparable in all instances, and were less than 5 seconds on a standard desktop computer, except for the 20- and 40-dimensional instances of Example 3, where the central cutting plane method needed considerably more time to convergege than Algorithm 1.

We start by an illustrative example comparing the central cutting plane algorithm of Kortanek and No (1993) and our central cutting surface algorithm.

**Example 1** (Tichatschke and Nebeling 1988)**.**

$$\begin{array}{ll} \text{minimize} & (x_1 - 2)^2 + (x_2 - 0.2)^2 \\ \text{subject to} & (5\sin(\pi\sqrt{t})/(1+t^2))x_1^2 - x_2 \leq 0 \quad \forall t \in [0,1] \\ & x_1 \in [-1,1], x_2 \in [0, 0.2]. \end{array} \tag{12}$$

The example is originally from (Tichatschke and Nebeling, 1988), and it is used frequently in the literature since. (In the original paper the problem appears with $t \in [0, 8]$ in place of $t \in [0, 1]$ in the infinite constraint set. We suspect that this is a typographic error: not only is that a less natural choice, but it also renders the problem non-convex.)

The optimal solution is $x = (0.20523677, 0.2)$. This problem is particularly simple, as only one cut is active at the optimal solution (it corresponds to $\hat{t} \approx 0.2134$), and this is also the most violated inequality for every $x$.

We initialized both algorithms with the trivial upper bound 5 on the minimum, corresponding to the feasible solution $(0, 0)$. Tbl. 1 shows the progress of the two algorithms (using constant centering parameter $s^{(k)} = 1$ in both algorithms), demonstrating that both algorithms have an empirical linear rate of convergence. The central cutting plane method generates more cuts (including multiple feasibility cuts at the point $\hat{t}$). On the other hand, the cutting surface algorithm generates only a single cut at $\hat{t}$ in the first iteration, and then proceeds by iterating through central feasible solutions until optimality is established.

**Example 2** (Smallest enclosing sphere)**.** The classic smallest enclosing ball and the smallest enclosing ellipsoid problems ask for the sphere or ellipsoid of minimum volume that contains a

| | cutting surface | | | cutting plane | | |
|---|---|---|---|---|---|---|
| $\sigma$ | feasibility cuts | optimality cuts | relative error | feasibility cuts | optimality cuts | relative error |
| $10^{-4}$ | 1 | 23 | $10^{-4.283}$ | 7 | 24 | $10^{-4.856}$ |
| $10^{-5}$ | 1 | 29 | $10^{-5.413}$ | 7 | 29 | $10^{-5.083}$ |
| $10^{-6}$ | 1 | 34 | $10^{-6.356}$ | 7 | 37 | $10^{-6.157}$ |
| $10^{-7}$ | 1 | 39 | $10^{-7.304}$ | 8 | 43 | $10^{-7.174}$ |

Table 1: Comparison of the central cutting surface and central cutting plane algorithms in Example 1, with centering parameters $s^{(k)} = 1$. $\sigma$ for the cutting plane algorithm is an identical measure of the distance from the optimal solutions as in Algorithm 1; both algorithms were terminated upon reaching $\sigma < 10^{-7}$. The relative error columns show the relative error from the true optimal objective function value. Both algorithms clearly exhibit linear convergence, but the cutting surface algorithm needs only a single cut and fewer iterations.

finite set of given points. Both of them admit well-known second order cone programming and semidefinite programming formulations. A natural generalization is the following: given a closed parametric surface $p(t)$, $t \in T$ (with some given $T \subseteq \mathbb{R}^n$), find the sphere or ellipsoid of minimum volume that contains all points of the surface. These problems also have a semi-infinite convex programming formulation. The smallest enclosing sphere, centered at $x$ with radius $r$, is given by the optimal solution of

$$\text{minimize } r \quad \text{subject to } \|x - p(t)\| \leq r, \ \ \forall\, t \in T,$$

whereas the smallest enclosing ellipsoid is determined by

$$\text{maximize } (\det A)^{(1/n)} \quad \text{subject to } A \succcurlyeq 0 \ \text{ and } \ \|x - Ap(t)\| \leq 1, \ \ \forall\, t \in T.$$

In the latter formulation $A \succcurlyeq 0$ denotes that the matrix $A$ is positive semidefinite. The objective function $\log(\det(A))$ could also be used in place of $\det(A)^{1/n}$; the two formulations are equivalent.

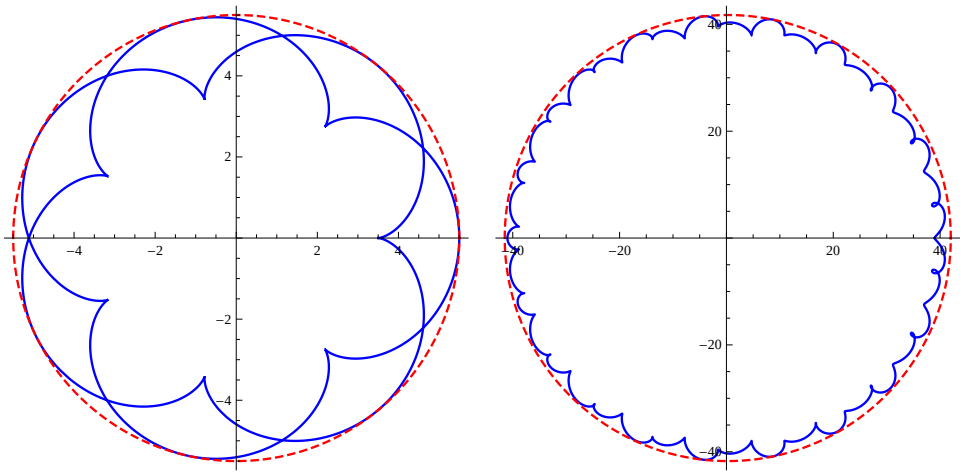
It was shown in (Papp and Alizadeh, 2011) that these problems also admit a semidefinite programming (SDP) formulation whenever every component of $p$ is a polynomial or a trigonometric polynomial of a single variable. This yields a polynomial time solution, but the formulation might suffer from ill-conditioning whenever the degrees of the polynomials (or trigonometric polynomials) involved is too large. Additionally, the sum-of-squares representations of nonnegative (trigonometric) polynomials that the SDP formulation hinges on do not generalize to multivariate polynomials. The central surface cutting algorithm does not have comparable running time guarantees to those of semidefinite programming algorithms, but it is applicable in a more general setting (including multi-dimensional index sets $T$ corresponding to multivariate polynomials), and does not suffer from ill-conditioning.

We give two examples of different complexity. First, consider the two-dimensional parametric curve

$$p(t) = (c\cos(t) - \cos(ct), c\sin(t) - \sin(ct)), \quad c = 4.5,\ t \in [0, 4\pi]. \tag{13}$$

This symmetric curve has a smallest enclosing circle centered at the origin, touching the curve at 7 points. (Fig. 1(a).)

Tbl. 2 shows the rate of convergence of the two algorithms (using constant centering parameter $s^{(k)} = 1$ in both algorithms). The initial upper bound on the minimum was set to $2(c+1)^2$, obtained by a simple term-by-term bound on the objective. In this example, the number of optimality cuts is




(a) (b)

Figure 1: The parametric curves (13) and (14), and their smallest enclosing circles.

| | cutting surface | | | cutting plane | | |
|---|---|---|---|---|---|---|
| $\sigma$ | feasibility cuts | optimality cuts | relative error | feasibility cuts | optimality cuts | relative error |
| $10^{-4}$ | 6 | 16 | $10^{-5.267}$ | 12 | 16 | $10^{-5.705}$ |
| $10^{-5}$ | 6 | 20 | $10^{-6.845}$ | 13 | 18 | $< 10^{-10}$ |
| $10^{-6}$ | 6 | 23 | $< 10^{-10}$ | 14 | 22 | $< 10^{-10}$ |
| $10^{-7}$ | 6 | 26 | $< 10^{-10}$ | 14 | 27 | $< 10^{-10}$ |
| $10^{-8}$ | 6 | 28 | $< 10^{-10}$ | 14 | 28 | $< 10^{-10}$ |

Table 2: Comparison of the central cutting surface and central cutting plane algorithms on the first curve of Example 2, with centering parameters $s^{(k)} = 1$. $\sigma$ for the cutting plane algorithm is an identical measure of the distance from the optimal solutions as in Algorithm 1; both algorithms were terminated upon reaching $\sigma < 10^{-8}$.

approximately the same for the two algorithms, but there is a difference in the number of feasibility cuts, and consequently in the total number of iterations.

Now consider an asymmetric, high-degree variant of the previous problem, depicted on Fig. 1(b):

$$p(t) = (c\cos(t) - \cos(ct), \sin(20t) + c\sin(t) - \sin(ct)), \quad c = 40,\ t \in [0, 2\pi]. \tag{14}$$

The center is no longer at the origin, and a closed form description of the circle is difficult to obtain. The semidefinite programming based solution of (Papp and Alizadeh, 2011) is theoretically possible, but practically not viable, owing to the high degree of the trigonometric polynomials involved. Tbl. 3 shows the rate of convergence of the two algorithms (using constant centering parameter $s^{(k)} = 1$ in the cutting surface algorithm).

In our final example we consider a generalization of the above problems, a problem with second order cone constraints of dimension higher than two, and investigate the hypothesis that cutting surfaces may be particularly advantageous in higher dimensions, when a polyhedral approximation of the feasible set is expensive to build.

| | cutting surface | | | cutting plane | | |
|---|---|---|---|---|---|---|
| $\sigma$ | feasibility cuts | optimality cuts | relative error | feasibility cuts | optimality cuts | relative error |
| $10^{-4}$ | 6 | 23 | $10^{-7.517}$ | 15 | 21 | $10^{-5.321}$ |
| $10^{-5}$ | 6 | 26 | $10^{-8.463}$ | 15 | 24 | $10^{-8.463}$ |
| $10^{-6}$ | 6 | 29 | $< 10^{-10}$ | 17 | 27 | $< 10^{-10}$ |
| $10^{-7}$ | 6 | 32 | $< 10^{-10}$ | 17 | 30 | $< 10^{-10}$ |
| $10^{-8}$ | 7 | 35 | $< 10^{-10}$ | 17 | 34 | $< 10^{-10}$ |

Table 3: Comparison of the central cutting surface and central cutting plane algorithms on the second curve of Example 2, with centering parameters $s^{(k)} = 1$. $\sigma$ for the cutting plane algorithm is an identical measure of the distance from the optimal solutions as in Algorithm 1; both algorithms were terminated upon reaching $\sigma < 10^{-8}$. The relative error columns show the relative error from the true optimal objective function value.

**Example 3.** Consider the SICP

$$\min_{x\in[-1,1]^n} \max_{t\in[0,1]} \sum_{i=1}^{n} (ix_i - i/n - \sin(2\pi t + i))^2.$$

It is easy to see that the optimal solution is $x = (1/n, 1/n, \ldots, 1/n)$.

The initial upper bound $U = 4n$ on the minimum can be obtained by taking a term-by-term upper bound of the objective at $x = 0$. We used this bound to initialize the central cutting surface and central cutting plane algorithms. As in the above examples, we used the centering parameter $s^{(k)} = 1$ in both algorithms.

Tbl. 4 shows the number of feasibility cuts and the the number of optimality cuts necessary until the stopping condition $\sigma < 10^{-6}$ is satisfied for different values of $n$.

| $n =$ | 5 | 10 | 20 | 40 |
|---|---|---|---|---|
| cutting surface | 13+19 | 16+17 | 15+19 | 15+22 |
| cutting plane | 93+14 | 290+15 | 1179+15 | >10000 |

Table 4: Comparison of the central cutting surface and central cutting plane algorithms on Example 3, for different values of $n$ (the number of decision variables). Each entry in the table is in the format *the number of feasibility cuts + the number of optimality cuts*, obtained with the centering parameter $s^{(k)} = 1$. Both algorithms were terminated upon reaching $\sigma < 10^{-6}$ or after 10000 cuts.

It is clear that in this example the number of feasibility cuts (and the total number of cuts) in the cutting plane algorithm grows much more rapidly with dimension than in the cutting surface algorithm. This is consistent with the fact that, unless strong centering is applied, a good polyhedral approximation (for cutting planes) or conic approximation (for cutting surfaces) of the feasible set needs to be built, which requires considerably more planar cuts han surface cuts. In the next section we consider the effect of centering further.

### 5.1.1 The effect of the centering parameter

The fact in Examples 1 and 2 most generated cuts are optimality cuts, not feasibility cuts, suggests that our default setting of the centering parameter, $s^{(k)} = 1$ in each iteration $k$, might not be

optimal. At the other extreme, $s^{(k)} = 0$ is expected to yield infeasible solutions in all iterations but the last. Another natural choice for the centering parameter, as discussed in Section 2, is the gradient of the norm of the violated inequality, which is suggested by Kortanek and No in their central cutting plane algorithm. Finally, our convergence proof shows that one can also use a constant multiple of this gradient norm. Example 3 also suggests that the centering parameter that keeps a balance between feasibility and optimality cuts might be different for the two algorithms, and that centering might be less important for cutting planes than for cutting planes (which must avoid building expensive polyhedral approximations of the feasible set around points that are far from the optimum). In this section we further examine (empirically) the effect of the centering parameter.

The smallest examples above solved by the cutting surface algorithm with no centering in only two iterations; for instance, in Example 1, the cutting surface algorithm generates one feasibility cut (at the same point $\hat{t}$ as the cutting surface algorithm with centering), and then one optimality cut, after which the optimality is proven.

For a non-trivial example, consider the second instance of the smallest enclosing sphere problems in Example 2, with the parametric curve defined in (14), and solve again the corresponding SICP problem using Algorithm 1, as well as the central cutting plane algorithm of Kortanek and No, using different constant centering parameters $s^{(k)}$. Tbls. 5 and 6 show the number of feasibility and optimality cuts for different values of this parameter.

| $s^{(k)}$ | $10^{-9}$ | $10^{-7}$ | $10^{-5}$ | $10^{-3}$ | $10^{-2}$ | $10^{-1}$ | 1. | $10^{1}$ | $10^{2}$ |
|---|---|---|---|---|---|---|---|---|---|
| cutting surfaces | | | | | | | | | |
| feasibility cuts | 9 | 8 | 7 | 7 | 7 | 7 | 7 | 9 | 10 |
| optimality cuts | 2 | 2 | 3 | 4 | 6 | 11 | 35 | 190 | 1496 |
| cutting planes | | | | | | | | | |
| feasibility cuts | 18 | 18 | 16 | 16 | 16 | 17 | 17 | 23 | 27 |
| optimality cuts | 2 | 2 | 3 | 4 | 6 | 11 | 34 | 195 | 1827 |

Table 5: The effect of centering on the number of cuts in the central cutting surface and central cutting plane algorithms using constant centering parameter.

| $s^{(k)}/\|\nabla g(s^{(k)},t^{(k)})\|$ | $10^{-9}$ | $10^{-7}$ | $10^{-5}$ | $10^{-3}$ | $10^{-2}$ | $10^{-1}$ | 1. |
|---|---|---|---|---|---|---|---|
| cutting surfaces | | | | | | | |
| feasibility cuts | 7 | 7 | 7 | 7 | 7 | 9 | 10 |
| optimality cuts | 2 | 3 | 4 | 11 | 33 | 183 | 1379 |
| cutting planes | | | | | | | |
| feasibility cuts | 18 | 18 | 16 | 16 | 16 | 26 | *22* |
| optimality cuts | 2 | 3 | 6 | 10 | 30 | 155 | *1524* |

Table 6: The effect of centering on the number of cuts in the central cutting surface and central cutting plane algorithms using a constant multiple of the gradient norm as centering parameter. The italic numbers indicate the Kortanek–No central cutting plane algorithm.

Now let us consider Example 3, and solve it again with choices for of the centering parameter. Tbl. 4 in the previous section shows the results for $s^{(k)} = 1$. Tbl. 7 shows what happens with no centering, while Tbls. 8–10 show results with centering using different multiples of the gradient norm.

| $n =$ | 5 | 10 | 20 | 40 |
|---|---|---|---|---|
| cutting surface | 14+1 | 17+1 | 22+1 | 22+1 |
| cutting plane | 94+1 | 402+1 | 4972+1 | >10000 |

Table 7: Results from Example 3 using $s^{(k)} = 0$ (no centering). Each entry in the table shows the number of feasibility cuts + the number of optimality cuts.

| $n =$ | 5 | 10 | 20 | 40 |
|---|---|---|---|---|
| cutting surface | 13+8 | 15+11 | 16+20 | 11+47 |
| cutting plane | 87+6 | 304+9 | 1139+16 | 4510+34 |

Table 8: Results from Example 3 using $s^{(k)} = 10^{-2}\|\nabla\|$. Each entry in the table shows the number of feasibility cuts + the number of optimality cuts.

| $n =$ | 5 | 10 | 20 | 40 |
|---|---|---|---|---|
| cutting surface | 15+24 | 14+48 | 13+123 | 10+369 |
| cutting plane | 99+18 | 279+36 | 922+87 | 3483+232 |

Table 9: Results from Example 3 using $s^{(k)} = 10^{-1}\|\nabla\|$. Each entry in the table shows the number of feasibility cuts + the number of optimality cuts.

| $n =$ | 5 | 10 | 20 | 40 |
|---|---|---|---|---|
| cutting surface | 12+175 | 12+383 | 11+886 | 8+3115 |
| cutting plane | 92+102 | 250+247 | 823+705 | 2990+1971 |

Table 10: Results from Example 3 using $s^{(k)} = \|\nabla\|$. Each entry in the table shows the number of feasibility cuts + the number of optimality cuts.

The results exhibit some interesting phenomena. First, the cutting surface algorithm benefits less from strong centering than cutting planes. It is also apparent that cutting planes require higher values for the centering parameter before the intermediate solutions become central (feasible). The results also indicate that the central cutting plane algorithm is more sensitive to the choice of the centering parameter than the cutting surface algorithm. Finally, it appears that in the high-dimensional instances cutting planes cannot compete with even the plain, uncentered, cutting surfaces, regardless of the type of centering used in the cutting plane method.

## 5.2 Robust, distributionally robust, and stochastic optimization

To illustrate the use of the central cutting surface algorithm in moment robust optimization (Section 4), we return to Example 1, and turn it into a problem with robust stochastic constraints:

**Example 4.**

$$\begin{array}{rll} \text{minimize} & (x_1-2)^2 + (x_2-0.2)^2 & \\ \text{subject to} & \mathbb{E}_P[(5\sin(\pi\sqrt{t})/(1+t^2))x_1^2 - x_2] \leq 0 & \forall P \in \mathfrak{P}_m \\ & x_1 \in [-1,1], x_2 \in [0,0.2], & \end{array} \tag{15}$$

where $\mathfrak{P}_m$ is a set of probability distributions supported on $\Xi = [0, 1]$ with prescribed polynomial moments up to order $m$:

$$\mathfrak{P}_m \stackrel{\text{def}}{=} \{P \,|\, \mathbb{E}_P[\xi^i] = 1/(i+1), i = 1, \ldots, m\}.$$

Setting $m = 0$ in the above formulation gives the classic robust optimization version of the problem, which is equivalent to the original Example 1.

At the other extreme, $\mathfrak{P}_\infty$ contains only the uniform distribution supported on $[0, 1]$. Therefore, solving (15) for $m = \infty$ amounts to solving a stochastic programming problem with a continuous scenario set. (Recall Theorem 7.) We solved a highly accurate deterministic approximation of this problem by replacing the continuous scenario set with a discrete one, corresponding to the 256-point Gaussian rule for numerical integration.

The solutions to problem (15) for increasing values of $m$ correspond to less and less conservative (or risk-averse) solutions. It is instructive to see how the solutions of these problems evolve as we impose more and more moment constraints, moving from the robust optimization solution to the stochastic programming solution. Interestingly, at the same time, there is no increase in the number of cuts necessary to find the optimum.

The results are summarized in Tbl. 11. Note the rather large difference between the optimal values of $x_1$ and the objective function upon the addition of the first few moment constraints.

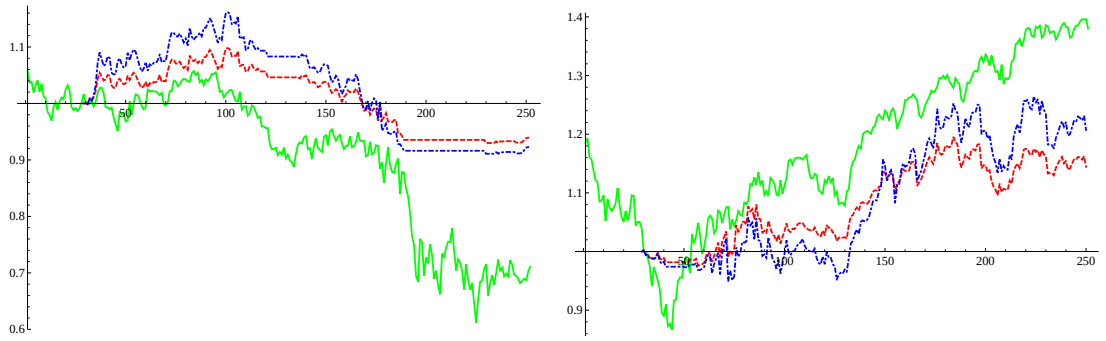
| $m$ | optimality cuts | feasibility cuts | $x_1$ | $x_2$ | $z$ |
|---|---|---|---|---|---|
| 0 | 4 | 3 | 0.20527 | 0.2 | 3.2211 |
| 1 | 5 | 3 | 0.24654 | 0.2 | 3.0746 |
| 2 | 5 | 2 | 0.24712 | 0.2 | 3.0726 |
| 3 | 5 | 2 | 0.26242 | 0.2 | 3.0192 |
| 4 | 5 | 2 | 0.26797 | 0.2 | 2.9999 |
| 5 | 5 | 2 | 0.26978 | 0.2 | 2.9937 |
| 6 | 4 | 2 | 0.27042 | 0.2 | 2.9914 |
| $\infty$ | n/a | n/a | 0.27181 | 0.2 | 2.9866 |

Table 11: Comparison of the solutions of problem (15) with different moment constraints. $m = 0$ is conventional robust optimization, $m = \infty$ corresponds to conventional stochastic programming. Intermediate values of $m$ yield solutions at different levels of risk-aversion. The solutions were obtained using Algorithm 1, with stopping condition $\sigma < 10^{-8}$, except for $m = \infty$ (see text).

### 5.2.1 A portfolio optimization example

We illustrate the use of Algorithms 1 and 2 for the solution of (DRO) using a portfolio optimization example motivated by (Delage and Ye, 2010). In our experiments we randomly chose three assets from the 30 Dow Jones assets, and tracked for a year the performance of a dynamically allocated portfolio that was rebalanced daily. Each day the 30-day history of the assets were used to estimate the moments of the return distribution, and reallocate the portfolio according following the optimal moment-robust distribution.

We split the results into two parts: we carried out the simulation using both 2008 and 2009 data to study the properties of the optimal portfolios under very different market conditions (hectic and generally downward in 2008, versus strongly increasing in 2009). In both cases we looked at portfolios optimized using different moment constraints (or, using the notation of Example 4, we used different sets $\mathfrak{P}_m$). We tracked a portfolio optimized using only first and second moment




(a) year 2008 (b) year 2009

Figure 2: The performance of two moment-robust portfolios rebalanced daily, compared to market performance. The market (solid, green) is the Dow Jones index scaled to have value 1 at the start of the experiment (day 31). The red dashed line shows the value of a portfolio optimized using the first and second moment information of the last 30 days' return. (Hence the curve starts at day 31.) The blue dot-dashed line shows the value of a portfolio optimized using the same moments and also the third and fourth marginal moments of the last 30 days' return. As expected, the first, more conservative portfolio outperforms the second one whenever the market conditions are bad, and only then. Both robust portfolios avoid the sharp drop in 2008 by not investing.

constraints, and one where the third and fourth marginal moments were also constrained. Sample plots are shown in Fig. 2, where the selected assets were AXP, HPQ, and IBM.

The results show the anticipated trends: the more conservative portfolio (optimized for the worst case among all return distributions compatible with the observed first and second moments) invests generally less, and avoids big losses better than the second portfolio (which is optimized for the worse case among a smaller set of distributions), at the price of missing out on a larger possible return.

The algorithm was implemented in Matlab R2012a (Windows 7 64-bit), using the interior-point solver IPOPT 3.10.2 for the solution of the master problems and the linear programming solver CPLEX 12.5 for the cut generation oracle subproblems, and was run on a desktop computer with an Intel Xeon 3.06GHz CPU. Tbls. 12 and 13 show the summary statistics of the algorithms performance, separately for the instances with up to second moment constraints and for the instances with moment constraints of order up to 4. The stopping criterion for the cutting surface algorithm was $\sigma < 10^{-3}$.

| | min | 25% | median | 75% | max |
|---|---|---|---|---|---|
| master problem time [sec] | 0.1708 | 0.52 | 0.77 | 1.14 | 2.05 |
| master problem iterations | 2 | 2 | 4 | 5 | 10 |
| subproblem time [sec] | 0.0140 | 1.47 | 21.28 | 43.77 | 180.77 |
| subproblem iterations | 1 | 27 | 54 | 87.25 | 186 |
| total wall-clock time [sec] | 9.4851 | 19.854 | 75.089 | 109.81 | 312 |

Table 12: Summary statistics of the moment robust optimization algorithm on the portfolio optimization example with moment constraints up to order 2. Each problem instance corresponds to one day in year 2008 or 2009; the table shows iteration count and timing results per instance.

As expected, the bottleneck of the algorithm is the randomized cut generation oracle: Algorithm

| | min | 25% | median | 75% | max |
|---|---|---|---|---|---|
| master problem time [sec] | 0.2049 | 0.412 | 0.602 | 0.775 | 1.04 |
| master problem iterations | 2 | 2 | 2 | 2 | 5 |
| subproblem time [sec] | 0.0182 | 9.551 | 15.1 | 28.1 | 88.2 |
| subproblem iterations | 1 | 3 | 43 | 87 | 986 |
| total wall-clock time [sec] | 9.6945 | 11.192 | 17.666 | 29.738 | 136.87 |

Table 13: Summary statistics of the moment robust optimization algorithm on the portfolio optimization example with moment constraints up to order 4.

2 takes considerably longer time to find a distribution whose corresponding constraint is violated than it takes to solve the master problems, which are very small convex optimization problems. Nevertheless, the cutting surface algorithm achieved very fast convergence (requiring less then 5 iterations for most instances), and therefore most problems were solvable within one minute.

# 6 Conclusion

The convergence of the central cutting surface algorithm was proven under very mild assumptions, which are essential to keep the problem at hand convex, with a non-empty interior. The possibility of using non-differentiable functions in the constraints whose subgradients may not be available, as well as using an infinite dimensional constraint set may extend the applicability of semi-infinite programming to new territories. We also found that the number of surface cuts can be considerably lower than the number of linear cuts in cutting plane algorithms, which compensates for having to solve a convex optimization problem in each iteration instead of a linear programming problem.

Our main motivation was distributionally robust optimization, but we hope that other applications involving constraints on probability distributions, and other problems involving a high-dimensional index set $T$, will be forthcoming.

Distributionally robust optimization with multivariate distributions is a relatively recent area, where not even the correct algorithmic framework to handle the arising problems can yet be agreed upon. Methods proposed in the most recent literature include interior point methods for semidefinite programming and the ellipsoid method, but these are not applicable in the presence of moment constraints of order higher than two. Our algorithm is completely novel in the sense that it is the first semi-infinite programming approach to distributionally robust optimization, and it is also the most generally applicable algorithm proposed to date.

Although it can hardly be expected that the semi-infinite programming based approach will be as efficient as the polynomial time methods proposed for the special cases, further research into moment matching scenario generation and distribution optimization algorithms may improve on the efficiency of our method. Simple heuristics might also be beneficial. For example, if several cuts (corresponding to probability distributions $P_1, \ldots, P_k$) have already been found and added to the master problem, then before searching for the next cut among distributions supported on the whole domain $\Xi$, we can first search among distributions supported on the union of the support of the distributions $P_1, \ldots, P_k$. This is a considerably cheaper step, which requires only the solution of a (finite) linear program, whose solution can be further accelerated by warmstarting.

Since without third and fourth moment information the overall shape of a distribution cannot be determined even approximately, we expect that future successful algorithms in distributionally robust optimization will also have the ability of including higher order moment information in the definition of the uncertainty sets.

## Acknowledgements

The research was partially supported by the grants DOE-SP0011568 and NSF CMMI-1100868.

## References

D. Bertsimas, X. V. Doan, K. Natarajan, and C.-P. Teo. Models for minimax stochastic linear optimization problems with risk aversion. *Mathematics of Operations Research*, 35(3):580–602, 2010.

B. Betrò. An accelerated central cutting plane algorithm for linear semi-infinite programming. *Mathematical Programming*, 101:479–495, 2004. doi: 10.1007/s10107-003-0492-5.

E. Delage and Y. Ye. Distributionally robust optimization under moment uncertainty with application to data-driven problems. *Operations Research*, 58:595–612, 2010. doi: 10.1287/opre.1090.0741.

P. R. Gribik. A central cutting plane algorithm for semi-infinite programming problems. In R. Hettich, editor, *Semi-infinite programming*, number 15 in Lecture Notes in Control and Information Systems. Springer Verlag, New York, NY, 1979.

K. Høyland, M. Kaut, and S. W. Wallace. A heuristic for moment-matching scenario generation. *Computational Optimization and Applications*, 24(2):169–185, Feb. 2003. doi: 10.1023/A:1021853807313.

K.-L. Huang and S. Mehrotra. An empirical evaluation of walk-and-round heuristics for mixed integer linear programs. *Computational Optimization and Applications*, 2013. doi: 10.1007/s10589-013-9540-0.

R. Kannan and H. Narayanan. Random walks on polytopes and an affine interior point method for linear programming. *Mathematics of Operations Research*, 37(1):1–20, Feb. 2012. doi: 10.1016/j.ejor.2006.08.045.

C. Kleiber and J. Stoyanov. Multivariate distributions and the moment problem. *Journal of Multivariate Analysis*, 113(1):7–18, 2013. ISSN 0047-259X. doi: 10.1016/j.jmva.2011.06.001. URL `http://dx.doi.org/10.1016/j.jmva.2011.06.001`.

K. O. Kortanek and H. No. A central cutting plane algorithm for convex semi-infinite programming problems. *SIAM Journal on Optimization*, 3(4):901–918, Nov. 1993.

M. López and G. Still. Semi-infinite programming. *European Journal of Operational Research*, 180:491–518, 2007. doi: 10.1016/j.ejor.2006.08.045.

L. Lovász and S. Vempala. Hit-and-run from a corner. *SIAM Journal on Computing*, 35(4):985–1005, 2006. doi: 10.1137/S009753970544727X.

S. Mehrotra and D. Papp. Generating moment matching scenarios using optimization techniques. *SIAM Journal on Optimizaton*, 23(2):963–999, 2013. URL `http://dx.doi.org/10.1137/110858082`.

S. Mehrotra and H. Zhang. Models and algorithms for distributionally robust least squares problems. *Accepted in Mathematical Programming*, 2013. URL `http://link.springer.com/article/10.1007/s10107-013-0681-9`.

D. Papp and F. Alizadeh. Semidefinite characterization of sum-of-squares cones in algebras. *Accepted in SIAM Journal on Optimization*, 2011.

H. E. Scarf. A min-max solution of an inventory problem. Technical Report P-910, The RAND Corporation, 1957.

R. Tichatschke and V. Nebeling. A cutting-plane method for quadratic semi infinite programming problems. *Optimization*, 19(6):803–817, 1988. doi: 10.1080/02331938808843393.

S. Vempala. Geometric random walks: a survey. *Combinatorial and Computational Geometry*, 52: 573–612, 2005.