Bounding-Focused Discretization Methods for the Global Optimization of Nonconvex Semi-Infinite Programs

Evren M. Turan[1], Johannes Jäschke[1], and Rohit Kannan[2]

[1]Department of Chemical Engineering, Norwegian University of Science and Technology (NTNU), Trondheim, Norway. E-mail: evren.m.turan@ntnu.no, johannes.jaschke@ntnu.no
[2]Grado Department of Industrial and Systems Engineering, Virginia Tech, Blacksburg, VA, USA, E-mail: rohitkannan@vt.edu

June 12, 2025

Abstract

We use sensitivity analysis to design *bounding-focused* discretization (cutting-surface) methods for the global optimization of nonconvex semi-infinite programs (SIPs). We begin by formulating the optimal bounding-focused discretization of SIPs as a max-min problem and propose variants that are more computationally tractable. We then use parametric sensitivity theory to design an effective heuristic approach for solving these max-min problems. We also show how our new iterative discretization methods may be modified to ensure that the solutions of their discretizations converge to an optimal solution of the SIP. We then formulate optimal bounding-focused *generalized* discretization of SIPs as max-min problems and design heuristic algorithms for their solution. Numerical experiments on standard nonconvex SIP test instances from the literature demonstrate that our new bounding-focused discretization methods can significantly reduce the number of iterations for convergence relative to a state-of-the-art feasibility-focused discretization method.

Key words: Semi-infinite programming, Robust optimization, Discretization, Global optimization, Cutting-surface, Sensitivity analysis

1 Introduction

Semi-infinite programs (SIPs) are mathematical optimization problems with a finite number of decision variables and an infinite number of constraints. They can be used to model several problems in science and engineering, such as robust optimization, controller design, Chebyshev approximation, and design centering [17, 23, 28, 52]. Our focus is on the *global* optimization of SIPs of the form:

$$v^* := \min_{x \in X} \; f(x) \tag{SIP}$$
$$\text{s.t. } g(x,y) \leq 0, \quad \forall y \in Y, \tag{1}$$

where $X \subset \mathbb{R}^{d_x}$ and $Y \subset \mathbb{R}^{d_y}$ are nonempty compact sets, $|Y| = \infty$, and functions $f : \mathbb{R}^{d_x} \to \mathbb{R}$ and $g : \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \to \mathbb{R}$ are continuous. We assume (SIP) is feasible, but do not assume that X, Y, f, or g is convex. We detail extensions to SIPs with multiple semi-infinite constraints in Section 4.

A key challenge in solving (SIP) is that evaluating feasibility of a candidate solution $x \in X$ requires the *global* solution of the following lower-level problem:

$$G(x) := \max_{y \in Y} \; g(x,y). \tag{LLP(x)}$$

Clearly, $x \in X$ is feasible for (SIP) if and only if $G(x) \leq 0$.

Algorithm 1 The Blankenship and Falk algorithm [5]

Input: feasibility tolerance $\varepsilon_f \geq 0$, initial discretization $Y_d = \emptyset$.
for $k = 1, 2, \ldots$ **do**
  Solve problem (LBP) globally to get solution x^k, lower bound LBD^k.
  Solve problem (LLP(x)) with $x = x^k$ globally to get solution $y^{BF,k} \in Y$.
  if $G(x^k) \leq \varepsilon_f$ **then**
    Terminate with ε_f-feasible solution x^k to (SIP).
  else
    Set $Y_d \leftarrow Y_d \cup \{y^{BF,k}\}$.
  end if
end for

Iteration No.	1	2	3	4	5	10	15	20	25	28
Lower Bound	4	4.19	4.38	4.56	4.74	5.62	6.41	7.12	7.73	8

Table 1: Lower bounds generated by the BF algorithm 1 on Example 1. This algorithm needs more than 20 iterations to approximate $v^* = 8$ to within 10%.

Several papers propose algorithms for the global minimization of nonconvex SIPs [4, 15–17, 22, 30, 31, 34, 46, 51]. They primarily construct lower bounds for (SIP) by replacing the semi-infinite constraint (1) with a *finite* discretization (cf. [8, 27]):

$$\begin{aligned} \min_{x \in X} \ & f(x) \\ \text{s.t.} \ & g(x, y) \leq 0, \quad \forall y \in Y_d, \end{aligned} \tag{LBP}$$

where $Y_d \subsetneq Y$ with $|Y_d| < \infty$. A lower bound for the optimal value v^* of (SIP) can be obtained by solving this (nonconvex) problem to *global* optimality.

The choice of discretization Y_d can greatly impact the tightness of the lower bound obtained by solving (LBP). Because naïve discretization approaches may necessitate a large discretization for (LBP) to effectively approximate (SIP) [48], techniques for adaptively populating Y_d are of interest. Global optimization methods for (SIP) predominantly rely on the *feasibility-focused* discretization method of Blankenship and Falk (BF; see Algorithm 1) [5], which is *the* state-of-the-art discretization method for nonconvex SIPs. This method populates Y_d with points in Y corresponding to the largest violation of constraint (1) at incumbent solutions of (LBP).

The sequence of non-decreasing lower bounds $\{LBD^k\}_k$ generated by the BF algorithm converges to v^* under our assumptions on (SIP) (see, e.g., [5, Theorem 2.1]). However, as Example 1 below illustrates, the BF algorithm may require an excessively large discretization Y_d before the sequence $\{LBD^k\}_k$ converges to within a specified tolerance of the optimal value v^*.

Example 1. [32, Example (DP)] Consider (SIP) with $d_x = 1$, $d_y = 1$, $X = [0, 6]$, $Y = [2, 6]$, $f(x) = 10 - x_1$, and $g(x, y) = \frac{y_1^2}{1+\exp(-40(x_1 - y_1))} + x_1 - y_1 - 2$. The global solution is $x^* = 2$ with objective $v^* = 8$.

Solving (LBP) with $Y_d = \{2\}$ (prescribed by our new discretization methods) yields a lower bound of v^*. However, as shown in Table 1, the state-of-the-art BF algorithm needs more than 20 iterations to even yield a lower bound that is within 10% of v^*. Figure 1 contrasts the above discretizations of the semi-infinite constraint.

Example 1 motivates our study of *new bounding-focused* discretization methods for (SIP) that can mitigate the slow convergence of the feasibility-focused BF algorithm. The main idea of our bounding-focused discretization methods is to populate the discretization Y_d with points in Y such that the lower bound from (LBP) is maximized. Since determining optimal bounding-focused discretizations may be challenging,

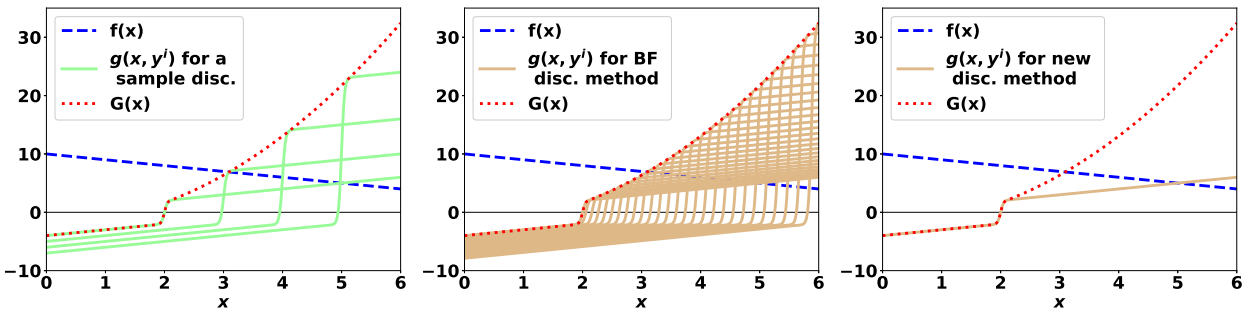


Figure 1: Benefits of bounding-focused discretization: Left: objective $f(x)$ and constraint $G(x)$ for Example 1 along with discretization $Y_d = \{2, 3, 4, 5\}$ (cf. [15, Figure 3.1]). Middle: BF method needs 27 discretization points for its lower bounds to converge to v^*. Right: our proposed bounding-focused discretization method in Section 2 only requires a single discretization point ($|Y_d| = 1$).

we consider more tractable variants and design efficient heuristic approaches to determine such discretizations. We also study how to construct bounding-focused *generalized* discretizations.

While we only investigate discretization methods that yield tighter lower bounds, our ideas may be adapted to design discretization methods for finding feasible solutions faster (cf. [31, 51]). We assume for our theoretical results that all subproblems solved to global optimality are solved exactly in finite time; our approaches may also be adapted to the setting where such subproblems are only solved to ε-global optimality for some $\varepsilon > 0$ [15, 16, 24].

This paper is organized as follows. Section 2 proposes new bounding-focused discretization methods, designs effective heuristic solution approaches, and presents theoretical guarantees. Section 3 proposes bounding-focused generalized discretization methods for (SIP) and designs effective heuristic solution strategies. Section 4 outlines extensions. Section 5 presents detailed computational results that show our bounding-focused discretization methods significantly reduce the number of iterations for convergence relative to the BF algorithm. Section 6 concludes with avenues for future work. The appendix contains a review of parametric sensitivity theory as well as omitted proofs.

Notation Let $[n] := \{1, \ldots, n\}$ and $\text{mid}(v^1, v^2, v^3)$ denote the componentwise median of vectors v^1, v^2, v^3. Given $\delta > 0$, $v \in \mathbb{R}^n$, $S \subset \mathbb{R}^n$, let $\|v\|$ denote the Euclidean norm of v, $B_\delta(v)$ denote the open Euclidean ball of radius δ centered at v, $\text{proj}_S(v)$ denote (an element of) the Euclidean projection of v onto S, and $\text{diam}(S)$ denote the diameter of S with respect to the Euclidean norm. We say (SIP) is convex if X is convex and f and $g(\cdot, y)$ are convex on X for each $y \in Y$ (note that Y and $-g(x, \cdot)$ need not be convex in this case).

2 Bounding-focused discretization methods

We propose new bounding-focused discretization methods for (SIP) that can achieve faster rate of convergence of lower bounds than the BF algorithm 1. The key idea of these discretization methods is to populate Y_d with points in Y that yield the highest lower bound. In the first iteration, instead of updating Y_d with a solution $y^{BF,1}$ of (LLP(x)) at $x = x^1$ as in the BF algorithm 1, we propose to solve the following max-min problem to determine a discretization $Y_d = \{\bar{y}^1\}$ that results in the highest lower bound:

$$\bar{y}^1 \in \arg\max_{y^1 \in Y} \min_{x \in X} \; f(x) \quad \text{s.t. } g(x, y^1) \leq 0. \tag{2}$$

We assume for simplicity that the maxima in all of our subproblems is attained (otherwise, we may pick any ε-optimal solution for some small $\varepsilon > 0$). Techniques for solving problem (2) are discussed in Section 2.3. We consider two approaches for updating the discretization Y_d at iteration $k > 1$.

The first approach discards the discretization Y_d^{k-1} from iteration $k-1$ and determines a new discretization at iteration k by solving the max-min problem:

$$(\bar{y}^1, \ldots, \bar{y}^k) \in \underset{(y^1,\ldots,y^k)\in Y^k}{\arg\max} \phi_k(y^1, \ldots, y^k) := \min_{x\in X} \; f(x) \quad \text{s.t. } g(x, y^i) \le 0, \quad \forall i \in [k], \tag{3}$$

where $\phi_k : Y^k \to \mathbb{R}$ denotes the value function of the inner-minimization in problem (3) at iteration k. Formulation (3) is inspired by the idea of strong partitioning proposed by Kannan et al. [26]. The resulting discretization $Y_d^k := \{\bar{y}^1, \ldots, \bar{y}^k\}$ at iteration k yields the highest lower bound among all possible relaxations (LBP) with at most k discretization points. However, the outer-maximization in problem (3) involves $k \times d_y$ variables compared to only d_y variables in problem (2).

To mitigate this increased computational burden, our second approach updates the discretization $Y_d^{k-1} := \{\bar{y}^1, \ldots, \bar{y}^{k-1}\}$ at iteration $k-1$ by adding a single point $\bar{y}^k \in Y$ that maximizes the lower bound improvement. This can be formulated as the max-min problem:

$$\bar{y}^k \in \underset{y^k\in Y}{\arg\max} \; \psi_k\big(y^k; Y_d^{k-1}\big) := \min_{x\in X} \; f(x) \quad \text{s.t. } g(x, y) \le 0, \quad \forall y \in Y_d^{k-1}, \; g(x, y^k) \le 0, \tag{4}$$

where $\psi_k : Y \to \mathbb{R}$ denotes the value function of the inner minimization, and $Y_d^k = Y_d^{k-1} \cup \{\bar{y}^k\}$ is the discretization specified at iteration k. Problem (4) is a greedy approximation of problem (3). We introduce two variants of these discretization methods in Section 2.1 and establish theoretical guarantees for our bounding-focused discretization methods in Section 2.2.

In contrast with the approach of Tsoukalas and Rustem [51], which treats the violation of the semi-infinite constraint (1) and the objective of (SIP) as two competing objectives, problems (3) and (4) directly optimize the discretization for the best lower bound. Baltean-Lugojan et al. [2] propose bounding-focused cuts with a similar flavor but tailored for outer-approximating semidefinite programs, which are a family of *convex* SIPs. Similar to problem (3), Coniglio and Tieves [10] formulate bounding-focused cut selection for integer linear programs as a bilevel problem and reformulate it as a single-level bilinear program using linear programming duality (cf. Section 2.3). Paulus et al. [38] consider bounding-focused cut selection for mixed-integer linear programs (MILPs), where they use explicit enumeration to choose the best bounding-focused cut from a *finite* list of candidate cuts. Finally, Das et al. [11] use the well-known result that problem (3) with $k = d_x$ discretization points yields an exact reformulation of *convex* SIPs under mild assumptions (see Proposition 2 in Section 2.2). They use simulated annealing to solve the resulting max-min problem and report encouraging results on small-scale convex SIPs.

2.1 Outline of bounding-focused discretization algorithms

Algorithm 2 outlines a prototype bounding-focused discretization method for (SIP). Similar to the BF algorithm 1, at each iteration, it solves (LBP) to *global* optimality to determine a candidate solution x^k and a corresponding lower bound LBD^k. It then solves the lower-level problem (LLP(x)) with $x = x^k$ to *global* optimality to determine a point $\hat{y}^k$, which is used to check if x^k is ε_f-feasible for (SIP). The key difference between Algorithm 2 and the BF algorithm 1 is on lines 8–11 of Algorithm 2. If x^k is not ε_f-feasible, Algorithm 2 solves a max-min problem (heuristically) to identify new points that may be used to update the discretization Y_d when a sufficient bound increase condition holds. In contrast, the BF algorithm 1 always adds $\hat{y}^k$ to the discretization.

We consider four realizations of Algorithm 2 that only vary on lines 8–11: `OPT`, `GREEDY`, `2GREEDY`, and `HYBRID`.

- `OPT` seeks to discard the discretization Y_d^{k-1} at iteration $k-1$ and replace it with a fresh discretization obtained by solving problem (3). It initializes the solution of this max-min problem with $Y_d^{k-1} \cup \{\hat{y}^k\}$.

Algorithm 2 Prototype bounding-focused discretization algorithm

1: **Input**: feasibility tolerance $\varepsilon_f \geq 0$, minimum bound improvement $\delta \geq 0$, and initial discretization $Y_d = \emptyset$.
2: **for** $k = 1, 2, \ldots$ **do**
3: Solve problem (LBP) globally to get solution x^k, lower bound LBD^k.
4: Solve problem (LLP(x)) with $x = x^k$ globally to get solution $\hat{y}^k \in Y$.
5: **if** $G(x^k) \leq \varepsilon_f$ **then**
6: **Terminate** with ε_f-feasible solution x^k to (SIP).
7: **else**
8: Solve a max-min problem (heuristically) to get $\{\bar{y}^{k,1}, \ldots, \bar{y}^{k,n_k}\}$.
9: Solve the inner-min problem to *global* optimality at the max-min solution $\{\bar{y}^{k,1}, \ldots, \bar{y}^{k,n_k}\}$. Let η_k^* denote its global optimal value.
10: **if** $\eta_k^* \geq LBD^k + \delta$ **then**
11: Update the discretization Y_d using $\{\bar{y}^{k,1}, \ldots, \bar{y}^{k,n_k}\}$.
12: **else**
13: Set $Y_d \leftarrow Y_d \cup \{\hat{y}^k\}$.
14: **end if**
15: **end if**
16: **end for**

- `GREEDY` adds a single point $\bar{y}^k$ to the discretization $Y_d^{k-1} := \{\bar{y}^1, \ldots, \bar{y}^{k-1}\}$ at iteration $k-1$ by solving problem (4) with the initialization $\hat{y}^k$.
- `2GREEDY` first updates the discretization at iteration $k-1$ with $\hat{y}^k$, i.e., $Y_d^{k-1} \leftarrow Y_d^{k-1} \cup \{\hat{y}^k\}$. It then solves problem (4) to find another point to add to the discretization using a perturbation of $\hat{y}^k$ as the initialization.
- `HYBRID` mitigates the increasing computational burden of Algorithm `OPT` as the iteration count k increases. For the first K iterations, it solves problem (3) to try and determine a fresh discretization with sufficient lower bound improvement (similar to `OPT`). From iteration $K+1$, it then switches to the `GREEDY` strategy and solves problem (4) to try and find a single best point to add to the previous discretization Y_d^{k-1}.

All four realizations of Algorithm 2 use the point $\hat{y}^k$ to either construct an initial guess, or to add to the discretization. Algorithm `2GREEDY` looks to add two discretization points per iteration (including $\hat{y}^k$) with the goal of reducing the number of global solves of (LBP) and (LLP(x)) and the overall time required for the sequence of lower bounds $\{LBD^k\}$ to converge to v^*.

We emphasize that except on line 9 of Algorithm 2, we do *not* require the inner-minimizations and outer-maximizations of our max-min problems to be solved to global optimality. Although the additional global solve at each iteration of Algorithm 2 may seem expensive, it is worth noting that if the condition on line 10 is satisfied, then the global solve on line 3 can be skipped in the subsequent iteration. Furthermore, this additional global solve need not be performed at every iteration; it can be performed infrequently (e.g., only every tenth iteration k) without compromising convergence guarantees or the validity of the lower bound LBD^k. Finally, this global solve becomes redundant for convex SIPs under mild assumptions, since the inner-minimization problems are convex programs in this case. In practice, we skip this extra global solve and instead rely heuristically on the local solution of the inner-minimization problem to verify the sufficient bound increase condition.

2.2 Convergence guarantees

We show the sequence $\{LBD^k\}_k$ of lower bounds determined by Algorithms `OPT`, `GREEDY`, `2GREEDY`, and `HYBRID` converge to v^* under different assumptions.

Our first result is the most useful one in practice. It allows our max-min formulations to be solved using *any* heuristic under the following conditions: (i) the inner-minimization problem is solved to global optimality *once* at the candidate max-min solution (see line 9 of Algorithm 2), and (ii) the minimum bound improvement required at each iteration $\delta > 0$.

Theorem 1. Consider Algorithm 2 with $\varepsilon_f = 0$, $\delta > 0$. Suppose the discretization Y_d is updated using Algorithm `OPT`, `GREEDY`, `2GREEDY`, or `HYBRID`. Then $\lim_{k\to\infty} LBD^k = v^*$.

Proof. Since f and g are continuous, X and Y are compact, and (SIP) is assumed to be feasible, the optimal value v^* is finite and bounded below by $\min_{x\in X} f(x) > -\infty$. Line 11 of Algorithm 2 updates the discretization Y_d using the points $\bar{y}^{k,1}, \ldots, \bar{y}^{k,n_k}$ only if this candidate discretization increases the lower bound in iteration $k+1$ by at least δ. Since $v^* - LBD^0 = v^* - \min_{x\in X} f(x) < \infty$, line 11 of Algorithm 2 can be executed only finitely many times before the lower bound converges to v^*. Therefore, if Algorithm 2 does not converge in a finite number of iterations, then line 13 is executed for all k large enough and the asymptotic behavior of Algorithm 2 is the same as that of the BF algorithm 1. The result that $LBD^k \to v^*$ then follows from Lemma 2.2 of Mitsos [31] (cf. Theorem 3.1 of Harwood et al. [24]). □

The remaining results in this section are mainly of theoretical interest since they assume our max-min formulations are solved to *global* optimality (which is in general impractical because this may be as hard as solving (SIP) itself).

The following result identifies favorable properties of Algorithm `OPT` when the max-min problem (3) is solved to global optimality at each iteration.

Proposition 2. Consider Algorithm `OPT` with $\varepsilon_f = \delta = 0$, and suppose the max-min problem (3) is solved to *global* optimality. Then $\lim_{k\to\infty} LBD^k = v^*$. Moreover, suppose (SIP) is convex and $\exists \bar{x} \in X$ such that $G(\bar{x}) < 0$. Then $LBD^k = v^*$ for each iteration $k \geq d_x$, i.e., Algorithm `OPT` converges in at most d_x iterations.

Our next results establish convergence rates of the sequence of lower bounds generated using Algorithms BF and `OPT`. We begin with the following result on the number of iterations required for the BF algorithm to converge.

Proposition 3. Consider the BF algorithm 1 with $\varepsilon_f > 0$. Suppose $\{g(\cdot, y)\}_{y\in Y}$ is uniformly Lipschitz continuous on X with Lipschitz constant $L_{g,x} > 0$, i.e.,

$$|g(x,y) - g(\bar{x},y)| \leq L_{g,x}\|x - \bar{x}\|, \quad \forall x, \bar{x} \in X,\ y \in Y.$$

Additionally, suppose $\{g(x,\cdot)\}_{x\in X}$ is uniformly Lipschitz continuous on Y with Lipschitz constant $L_{g,y} > 0$, i.e.,

$$|g(x,y) - g(x,\bar{y})| \leq L_{g,y}\|y - \bar{y}\|, \quad \forall y, \bar{y} \in Y,\ x \in X.$$

Then Algorithm 1 terminates in at most N iterations, where

$$N := \min\left\{ \left\lceil \left(\frac{\operatorname{diam}(Y) L_{g,y}}{\varepsilon_f} + 1 \right)^{d_y} \right\rceil, \left\lceil \left(\frac{\operatorname{diam}(X) L_{g,x}}{\varepsilon_f} + 1 \right)^{d_x} \right\rceil \right\}.$$

The estimate of the number of iterations needed for the BF algorithm to converge depends exponentially on the dimensions of (SIP). When $d_y \ll d_x$, as in many applications, the term $\left(\frac{\operatorname{diam}(Y) L_{g,y}}{\varepsilon_f} + 1\right)^{d_y}$ may be significantly smaller than $\left(\frac{\operatorname{diam}(X) L_{g,x}}{\varepsilon_f} + 1\right)^{d_x}$. Proposition 3 may be sharpened by estimating the number of balls needed to cover an enlargement of $\{y^*(x) : x \in X\}$ instead of an enlargement of Y. We now link Proposition 3 to the convergence rate of the sequence of lower bounds furnished by Algorithms BF and `OPT`.

Theorem 4. Suppose the value function $V(z) := \min\{f(x) : x \in X,\ G(x) \leq z\}$ is Lipschitz continuous on $[0, \bar{\varepsilon}]$ with Lipschitz constant $L_V > 0$, for some $\bar{\varepsilon} > 0$. Additionally, suppose $\varepsilon_f = 0$, the family $\{g(\cdot, y)\}_{y \in Y}$ is uniformly Lipschitz continuous on X with Lipschitz constant $L_{g,x} > 0$, and the family $\{g(x, \cdot)\}_{x \in X}$ is uniformly Lipschitz continuous on Y with Lipschitz constant $L_{g,y} > 0$. Consider Algorithm OPT with $\delta = 0$, and assume that the max-min problem (3) is solved to global optimality. Then, for any $\varepsilon \in (0, L_V \bar{\varepsilon})$, the lower bounds produced by Algorithms BF and OPT exceed $v^* - \varepsilon$ whenever $k \geq \min\left\{\left\lceil\left(\frac{\operatorname{diam}(Y) L_{g,y} L_V}{\varepsilon} + 1\right)^{d_y}\right\rceil, \left\lceil\left(\frac{\operatorname{diam}(X) L_{g,x} L_V}{\varepsilon} + 1\right)^{d_x}\right\rceil\right\}$.

The Lipschitz assumption in Theorem 4 is satisfied by convex SIPs when the objective f is Lipschitz and Slater's condition holds (see Corollary 2 to Theorem 6.3.2 in Clarke [9]). Chapter 6 of Clarke [9] also details other constraint qualifications under which this Lipschitz assumption holds.

The following example illustrates that the above rate of convergence cannot be improved in general. Specifically, this example shows that discretization-based lower bounding methods involving (LBP) (such as the BF algorithm and *any* realization of Algorithm 2, including OPT) may require an exponential number of discretization points in the problem dimensions to converge. Although this behavior is expected, we are not aware of such an example in the SIP literature (similar examples are known for the classical cutting-plane method in convex optimization, see [37, Ex. 3.3.1], [25, Ex. 1]).

Example 2. (Based on [25, Example 1]) Consider (SIP) with $X = [-1, 1]^{d_x}$, $Y = \{y \in \mathbb{R}^{d_x} : \|y\|^2 = d_x - 1\}$, $f(x) = -\|x\|^2$, and $g(x, y) = \sum_{i=1}^{d_x}(x_i - y_i)y_i$. Note that this semi-infinite constraint (1) may be reformulated as the convex constraint $\|x\| \leq \sqrt{d_x - 1}$. Any $x \in X$ with $\|x\| = \sqrt{d_x - 1}$ solves (SIP) with $v^* = 1 - d_x$.

Lemma 2.1 of Hijazi et al. [25] implies any discretization point $y \in Y$ can exclude at most one vertex of the cube X. Because every vertex of X is a solution to (LBP) with the discretization $Y_d = \emptyset$, *any* discretization-based lower bounding algorithm that solves (LBP) requires exponentially many discretization points in the dimension d_x for its sequence of lower bounds $\{LBD^k\}_k$ to exceed $v^* - 0.5$.

We end this section by demonstrating the sequence $\{LBD^k\}_k$ of lower bounds determined by Algorithm GREEDY *may not* converge to v^* when $\delta = 0$ (i.e., if the fallback to the BF algorithm on line 13 of Algorithm 2 is not used), even if the max-min problems (4) are always solved to global optimality.

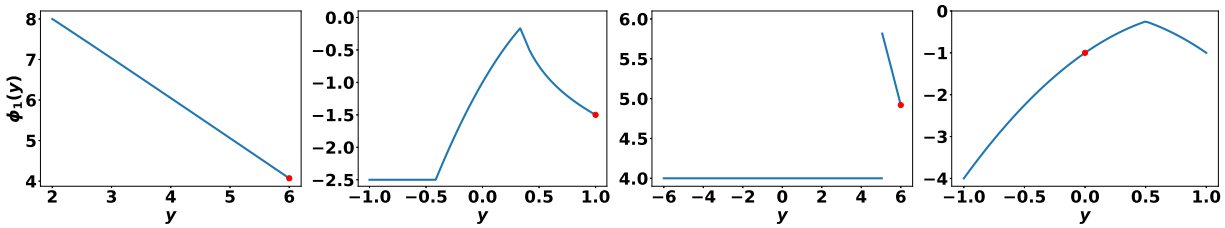
Example 2: Consider Algorithm GREEDY with $\varepsilon_f = \delta = 0$, and suppose the max-min problems (4) are solved to global optimality at each iteration. Assume without loss of generality that the incumbent solution $x^1 = \mathbf{1}$ at the first iteration, where $\mathbf{1}$ denotes a vector of ones. Then $\hat{y}^1 = \sqrt{1 - \frac{1}{d_x}}\mathbf{1}$. The sequence of vectors $\{\bar{y}^k\}$ with $\bar{y}^j = -\hat{y}^1$ for each $j \geq 1$ is *one* sequence of optimal solutions to the max-min problems (4) solved by Algorithm GREEDY. However, $LBD^k = -d_x < v^*$ for each $k \geq 1$ since x^1 remains the incumbent solution at each iteration k. Therefore, Algorithm GREEDY *may not* converge when $\delta = 0$.

2.3 Solving the max-min problems

While solving the max-min problems (3) or (4) to global optimality is clearly desirable, this may be as difficult as solving (SIP) itself. Hence, we exploit the fact that solving these problems heuristically is sufficient to obtain a discretization with desirable convergence properties (see Theorem 1 and the discussion in Section 2.1). Numerical experiments in Section 5 empirically demonstrate that our heuristic approaches for solving problems (3) or (4) almost always yield better discretizations with faster rate of convergence than the BF algorithm.

Properties of the value functions ϕ_k and ψ_k in problems (3) and (4) Before we outline our approach for solving problems (3) and (4), we plot the value functions ϕ_k and ψ_k of these problems for some examples from the literature. We consider the following three examples in addition to Example 1.

Example 3. [41, Example 2.1] Consider (SIP) with $d_x = 2$, $d_y = 1$, $X = [-1, 1]^2$, $Y = [-1, 1]$, $f(x) = -x_1 + 1.5x_2$, and $g(x, y) = -y_1^2 + 2y_1x_1 - x_2$. The global solution is $x^* = \left(\frac{1}{3}, \frac{1}{9}\right)$ with $v^* = -\frac{1}{6}$.



(a) Example 1 (b) Example 3 (c) Example 4 (d) Example 5

Figure 2: Value functions ϕ_1 (note the discontinuity for Example 4). The red dot indicates ϕ_1 at the point $y^{BF,1}$ determined by the BF algorithm at iteration 1.

Example 4. [51, Example 2.1] Consider (SIP) with $d_x = 1$, $d_y = 1$, $X = [-6, 6]$, $Y = [-6, 6]$, $f(x) = 10 - x_1$, and $g(x, y) = -x_1^4 + x_1^2 - x_1^2 y_1^2 + 2x_1^3 y_1 - 4$. The global solution is $x^* = 2$ with $v^* = 8$.

Example 5. [32, Example (H)] Consider (SIP) with $d_x = 2$, $d_y = 1$, $X = [0, 1] \times [-10^3, 10^3]$, $Y = [-1, 1]$, $f(x) = x_2$, and $g(x, y) = -(x_1 - y_1)^2 - x_2$. Any $x^* = (\bar{x}_1, 0)$ with $\bar{x}_1 \in [0, 1]$ is a global solution with optimal value $v^* = 0$.

Figure 2 plots the (global) value function ϕ_1 for Examples 1, 3, 4, and 5. It illustrates that ϕ_1 may be nonconcave, nondifferentiable, or even discontinuous with large flat regions. Additionally, the supremum in problem (2) is not attained for Example 4. Figure 2 also empirically illustrates that the BF point $y^{BF,1}$ provides a good initial guess for solving problem (2).

Figure 3 shows that the (global) value functions ψ_k in Algorithm `GREEDY` may become increasingly challenging to optimize over as k increases[1]; however, solving the lower-level problem (LLP(x)) at incumbent (lower bounding) solutions x^k empirically continues to yield a good initial guess $\hat{y}^k$.

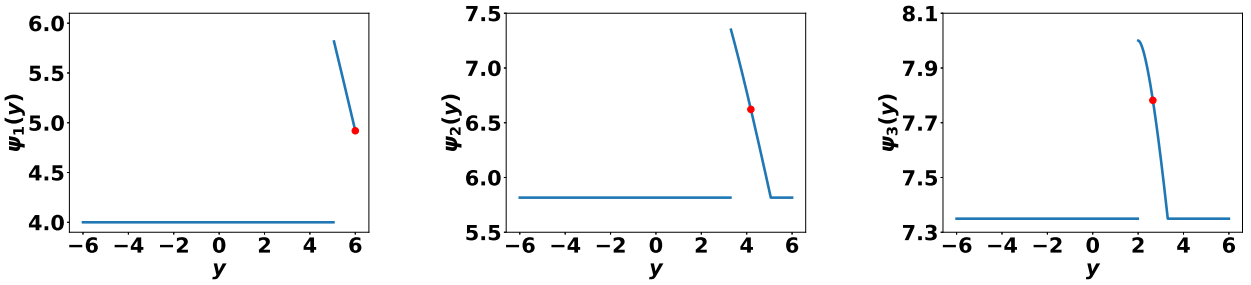
Overall, Figures 2 and 3 highlight that exploiting (generalized) gradient information can yield effective heuristics for solving problems (3) and (4).

An effective heuristic solution approach Due to the potential nonsmooth and discontinuous nature of the functions ϕ_k and ψ_k, we propose to solve the max-min problem of Algorithm 2 using gradients (whenever they exist) of ϕ_k and ψ_k within a bundle solver for nonsmooth nonconvex optimization [29]. Each iteration of the bundle method requires function and generalized gradient evaluations. We estimate the values of the functions ϕ_k and ψ_k by solving the inner-minimization problems in problems (3) and (4) to *local* optimality. We then try and use Theorem 8 in Appendix A to compute gradients of the local minimum value function ϕ_k or ψ_k (this involves the solution of a linear system of equations) when its assumptions hold. If some of the assumptions of Theorem 8 do not hold during the solution of problem (3) or (4), we try and use the heuristics detailed below to estimate a generalized gradient of ϕ_k or ψ_k. Note that we terminate the solution of the max-min problem and return its best found solution if the local solver fails to successfully solve the inner-minimization problem at any step.

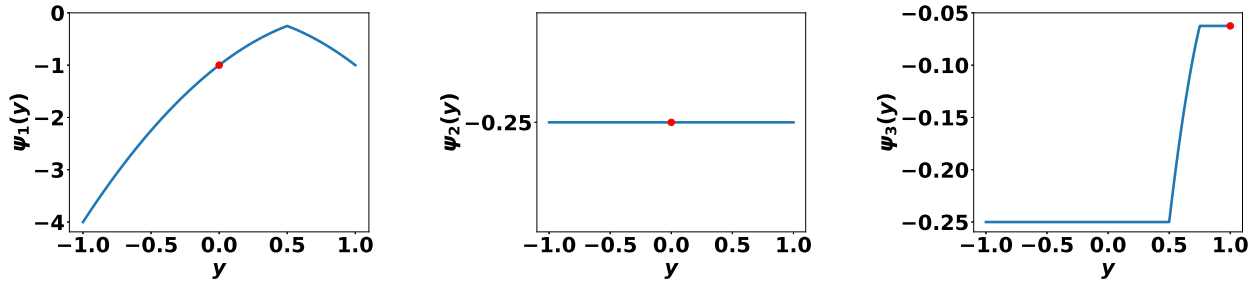
Heuristics for estimating a generalized gradient of ϕ_k or ψ_k Whenever its assumptions hold, we use Theorem 8 to compute a gradient of the functions ϕ_k and ψ_k in problems (3) and (4). In practice, we find that all of these assumptions may not hold at *every* iterate of the max-min solution.

When only the SC condition fails, the Lagrange multipliers are not unique. If SSOSC holds for all dual variables, then the value functions ϕ_k and ψ_k are piecewise-differentiable, with a kink at the evaluation point due to changes in the active constraint set. In this case, a directional derivative can be computed by appropriately selecting a subset of weakly active constraints [39, Theorem 1]. Rather than explicitly checking whether only SC fails, we always handle weakly active constraints using the following heuristic procedure.

[1] We do not plot ψ_2 and ψ_3 for Examples 1 and 3 as $\max_{y \in Y} \phi_1(y) = v^*$ for these instances.



(a) Example 4



(b) Example 5

Figure 3: Value functions ψ_1, ψ_2, ψ_3 in the first three iterations of problem (4) for Examples 4 and 5 (all three functions are discontinuous for Example 4). The red dot indicates ψ_k at the initial guess $\hat{y}^k$ (see line 4 of Algorithm 2) for problem (4) when the discretization is set using Algorithm `GREEDY` in Section 2.1.

We exclude weakly active bound constraints by default since including them causes the derivative of the variable at the bound to be zero. Based on numerical experiments, we heuristically choose to include all other weakly active constraints. An alternative to the above heuristic is to use the results of Stechlinski et al. [44] to rigorously compute generalized gradients when SC does not hold; we do not adopt this approach because of its higher computational cost.

If either $J_z(p)$ or $H_{z,\lambda}$ in Theorem 8 is singular (e.g., if LICQ does not hold), we add a small regularization term before solving the corresponding linear system to determine sensitivities. If $H_{z,\lambda}$ is non-square, we follow the heuristic in Section 4.3 of Agrawal et al. [1] and compute the minimum-norm least squares solution to the linear system. If an error occurs during the solution of the max-min problem (e.g., the local optimizer fails to converge), then we terminate the solution of the max-min problem and return the best found solution.

Enhancements to the heuristic method for solving max-min problems We list ways in which Algorithms `OPT`, `GREEDY`, `2GREEDY`, and `HYBRID` can be enhanced in practice. First, the solution of the sequence of inner-minimization problems in (3) or (4) can be effectively warm-started using active-set methods. Second, solving these inner-minimization problems using multi-start techniques can increase the likelihood that we compute the global value functions ϕ_k and ψ_k. Finally, as we saw in Examples 3 to 5, the assumptions of Theorem 8 may not hold throughout the domain of ϕ_k and ψ_k. In such situations, we may either return the best found solution to problems (3) and (4), or randomly perturb the current iterate in an attempt to avoid points of nondifferentiability.

Alternative approaches for solving the max-min problems (3) and (4) If the inner-minimization problems in (3) and (4) are convex and satisfy a constraint qualification (e.g., Slater's condition), then we might be able to use strong duality to reformulate problems (3) and (4) into single-level maximization problems. The resulting problems can then be solved to local optimality to update the discretization Y_d. Alternatively, suppose the functions in (SIP) are continuously differentiable and the inner-minimization

problems in (3) and (4) satisfy a constraint qualification for each feasible point of the outer-maximization problems. Then we can relax problems (3) and (4) into maximization problems with complementarity constraints using the KKT conditions for the inner-minimization problems, which can then be solved locally to update Y_d (cf. [47]). We compare our heuristic approach for solving problems (3) and (4) with this KKT-based relaxation approach in Section 5.

When the assumptions of Theorem 8 fail to hold, we may also be able to either use generalized gradients [13, 35] of ϕ_k and ψ_k, or directional derivatives [39] of ϕ_k and ψ_k and their generalizations [43, 44] to solve problems (3) and (4) heuristically [6]. Techniques for minimizing discontinuous functions (see, e.g., Ermoliev et al. [20]) may also be used to maximize ϕ_k and ψ_k over their domains when none of the aforementioned approaches are applicable.

3 Generalized bounding-focused discretizations

We propose new *generalized* bounding-focused discretization methods for (SIP) that can achieve faster rate of convergence of the sequence of lower bounds than *any* traditional discretization method relying on (LBP).

To motivate these generalized discretization methods, note that (SIP) can be equivalently reformulated as the bilevel problem [12, 45]:

$$\begin{aligned} \min_{x \in X} \ & f(x) \\ \text{s.t.} \ & g(x, y^*(x)) \leq 0, \end{aligned} \tag{5}$$

where $y^* : X \to Y$ maps $x \in X$ to *an* optimal solution to (LLP(x)).

The BF algorithm 1 may be viewed as approximating $y^*(x)$ with solutions $\{y^*(x^k)\}_k$ of the lower-level problem (LLP(x)) at incumbent solutions $\{x^k\}_k$ to (LBP). This zeroth-order approximation of y^* may be crude. Assuming y^* is differentiable (cf. Theorem 8), Seidel and Küfer [41] and Djelassi [15] instead propose to use the first-order approximation $y^*(x) \approx y^*(x^k) + J_y^*(x^k)(x - x^k)$ at incumbent solutions $\{x^k\}_k$, where $J_y^*(x^k)$ is the $d_y \times d_x$ Jacobian matrix with rows $(\nabla_x y_1^*(x^k))^{\mathrm{T}}, \ldots, (\nabla_x y_{d_y}^*(x^k))^{\mathrm{T}}$. In particular, Section 3.4 of Djelassi [15] suggests the following generalization of the lower bounding problem (LBP):

$$\begin{aligned} \min_{x \in X} \ & f(x) \\ \text{s.t.} \ & g\big(x, \operatorname{proj}_Y(Ax + b)\big) \leq 0, \quad \forall (A, b) \in Y_d^G, \end{aligned} \tag{G-LBP}$$

where tuples (A, b) in the generalized discretization Y_d^G satisfy $A \in \mathbb{R}^{d_y \times d_x}$ and $b \in \mathbb{R}^{d_y}$. Setting $Y_d^G = \{(0, y^{BF,1}), \ldots, (0, y^{BF,k})\}$ recovers the BF lower bounding problem at iteration k. Djelassi [15] proposes to improve the BF lower bound using the generalized discretization $Y_d^G = \{(J_y^*(x^1), y^*(x^1) - J_y^*(x^1)x^1), \ldots, (J_y^*(x^k), y^*(x^k) - J_y^*(x^k)x^k)\}$ at iteration k given a sequence of candidate solutions $\{x^k\}_k \subset X$ to (SIP). The projection step ensures (G-LBP) is a relaxation of (SIP), which implies solving (G-LBP) to *global* optimality yields a lower bound on the optimal value v^*. However, it also makes (G-LBP) nonsmooth and more challenging to solve than (LBP).

We propose bounding-focused generalized discretizations of (SIP) in the form of (G-LBP) that can achieve faster rate of convergence of lower bounds than the discretization methods in Section 2. Our key idea is to populate the generalized discretization Y_d^G with tuples (A, b) that yield the highest lower bound. In the first iteration, we solve the following max-min problem to determine a generalized discretization $Y_d^G = \{(\bar{A}^1, \bar{b}^1)\}$ that results in the highest lower bound:

$$\begin{aligned} (\bar{A}^1, \bar{b}^1) \in \operatorname*{arg\,max}_{A^1 \in \mathbb{R}^{d_y \times d_x}, b^1 \in \mathbb{R}^{d_y}} \ & \min_{x \in X} \ f(x) \\ & \text{s.t.} \ g\big(x, \operatorname{proj}_Y(A^1 x + b^1)\big) \leq 0. \end{aligned} \tag{6}$$

We again assume for simplicity that the maxima in all of our subproblems is attained. We propose the following extensions of problems (3) and (4) for updating the bounding-focused generalized discretization at iteration $k > 1$.

Algorithm 3 Prototype bounding-focused generalized discretization method

Input: feasibility tolerance $\varepsilon_f \geq 0$, minimum bound improvement $\delta \geq 0$, and initial generalized discretization $Y_d^G = \emptyset$.
for $k = 1, 2, \ldots$ **do**
  Solve problem (G-LBP) globally to get solution x^k, lower bound LBD^k.
  Solve problem (LLP(x)) with $x = x^k$ globally to get solution $y^*(x^k)$.
  If assumptions of Theorem 8 hold for (LLP(x)) with $x = x^k$, compute the Jacobian matrix $J_y^*(x^k)$.
  if $G(x^k) \leq \varepsilon_f$ **then**
    Terminate with ε_f-feasible solution x^k to (SIP).
  else
    Solve a max-min problem (heuristically) to get candidate generalized discretization tuples $\{(\bar{A}^{k,1}, \bar{b}^{k,1}), \ldots, (\bar{A}^{k,n_k}, \bar{b}^{k,n_k})\}$.
    Solve the inner-min problem to *global* optimality at the above candidate solution. Let η_k^* denote its global optimal value.
    if $\eta_k^* \geq LBD^k + \delta$ **then**
      Update Y_d^G using $\{(\bar{A}^{k,1}, \bar{b}^{k,1}), \ldots, (\bar{A}^{k,n_k}, \bar{b}^{k,n_k})\}$.
    else
      Set $Y_d^G \leftarrow Y_d^G \cup \{(J_y^*(x^k), y^*(x^k) - J_y^*(x^k)x^k)\}$ if assumptions of Theorem 8 hold, and $Y_d^G \leftarrow Y_d^G \cup \{(0, y^*(x^k))\}$ otherwise
    end if
  end if
end for

The first approach discards the generalized discretization $Y_d^{G,k-1}$ at iteration $k-1$ and determines a fresh generalized discretization at iteration k by solving the max-min problem:

$$
\begin{aligned}
(\bar{A}^1, \bar{b}^2, \ldots, \bar{A}^k, \bar{b}^k) \in & \underset{\substack{A^1,\ldots,A^k \in \mathbb{R}^{d_y \times d_x} \\ b^1,\ldots,b^k \in \mathbb{R}^{d_y}}}{\arg\max} \phi_k^G(A^1, b^1, \ldots, A^k, b^k), \qquad (7) \\
\phi_k^G(A^1, b^1, \ldots, A^k, b^k) := & \min_{x \in X} \ f(x) \\
& \text{s.t. } g\big(x, \mathrm{proj}_Y(A^i x + b^i)\big) \leq 0, \quad \forall i \in [k].
\end{aligned}
$$

The resulting generalized discretization $Y_d^{G,k} := \{(\bar{A}^1, \bar{b}^1), \ldots, (\bar{A}^k, \bar{b}^k)\}$ at iteration k yields the highest lower bound among all possible relaxations (G-LBP) with $|Y_d^G| = k$. To mitigate the computational cost of solving problem (7), our second approach updates the generalized discretization $Y_d^{G,k-1} := \{(\bar{A}^1, \bar{b}^1), \ldots, (\bar{A}^{k-1}, \bar{b}^{k-1})\}$ at iteration $k-1$ by adding a single tuple $(\bar{A}^k, \bar{b}^k)$ that maximizes lower bound improvement. This can be formulated as:

$$
\begin{aligned}
(\bar{A}^k, \bar{b}^k) \in & \underset{A^k \in \mathbb{R}^{d_y \times d_x}, b^k \in \mathbb{R}^{d_y}}{\arg\max} \psi_k^G\big(A^k, b^k; Y_d^{G,k-1}\big), \qquad (8) \\
\psi_k^G\big(A^k, b^k; Y_d^{G,k-1}\big) := & \min_{x \in X} \ f(x) \\
& \text{s.t. } g(x, \mathrm{proj}_Y(Ax + b)) \leq 0, \quad \forall (A, b) \in Y_d^{G,k-1}, \\
& \quad g(x, \mathrm{proj}_Y(A^k x + b^k)) \leq 0.
\end{aligned}
$$

Problem (8) can be viewed as a greedy approximation of problem (7). We propose two variants of these bounding-focused generalized discretization methods in Section 3.1 and establish theoretical guarantees in Section 3.2.

3.1 Outline of bounding-focused generalized discretization methods

Algorithm 3 outlines a prototype bounding-focused generalized discretization method for (SIP) (cf. Algorithm 2). The key difference with the approach outlined in Djelassi [15] is on lines 9–12 of Algorithm 3. If x^k is not ε_f-feasible, Algorithm 3 solves a max-min problem (heuristically) to identify new tuples that could be used to update the generalized discretization Y_d^G, whereas Djelassi [15] always looks to update Y_d^G with the tuple $(J_y^*(x^k), y^*(x^k) - J_y^*(x^k)x^k)$ corresponding to a linear approximation of $y^*(x)$ at $x = x^k$.

We consider four realizations of Algorithm 3 that only vary on lines 9–12: `G-OPT`, `G-GREEDY`, `G-2GREEDY`, and `G-HYBRID`. Algorithms `G-OPT`, `G-GREEDY`, and `G-HYBRID` are direct analogs of `OPT`, `GREEDY`, and `HYBRID` that rely on problems (7) and (8) instead of problems (3) and (4). Algorithm `G-2GREEDY` first adds either $(J_y^*(x^k), y^*(x^k) - J_y^*(x^k)x^k)$ or $(0, y^*(x^k))$ to Y_d^G (depending on whether the assumptions of Theorem 8 hold). It then solves problem (8) to try and find another tuple to add to the generalized discretization.

3.2 Convergence guarantees

We begin by establishing convergence of the sequence of lower bounds $\{LBD^k\}_k$ generated by Algorithms `G-OPT`, `G-GREEDY`, `G-2GREEDY`, and `G-HYBRID` to v^*. Like Theorem 1, this result also allows the max-min problems (7) and (8) to be solved using *any* heuristic so long as $\delta > 0$ and the inner-minimization problem is solved to global optimality *once* per iteration at the candidate max-min solution (see line 10 of Algorithm 3). Following the discussion in Section 2.1, we note that the additional global solve on line 10 of Algorithm 3 becomes redundant when the condition on line 11 is satisfied. In practice, we skip this extra global solve and instead rely heuristically on the local solution of the inner-minimization problem to verify the sufficient bound increase condition on line 11.

Theorem 5. Consider Algorithm 3 with $\varepsilon_f = 0$ and $\delta > 0$. Suppose the set Y is convex and the generalized discretization Y_d^G is updated using Algorithm `G-OPT`, `G-GREEDY`, `G-2GREEDY`, or `G-HYBRID`. Then $\lim_{k\to\infty} LBD^k = v^*$.

Proof. The proof follows a similar outline as the proof of Theorem 1 and Lemma 2.2 of Mitsos [31] (cf. Theorem 3.1 of Harwood et al. [24]).

If the solution x^l to the lower bounding problem (G-LBP) at iteration $l \in \mathbb{N}$ is feasible to (SIP), then we have $LBD^k = v^*$ for all $k \geq l$. This holds because problem (G-LBP) is a relaxation of (SIP), and the discretization methods `G-OPT`, `G-GREEDY`, `G-2GREEDY`, and `G-HYBRID` ensure that the lower bound LBD^k is monotonically non-decreasing with respect to the iteration number k. Hence, the stated result follows directly in this case.

Suppose that the lower bounding solution $x^k \in X$ is infeasible to (SIP) at each iteration $k \in \mathbb{N}$. Since X is compact, we can assume (by moving to a subsequence) that $x^k \to x^* \in X$. We show that x^* is feasible to (SIP), which implies $LBD^k \to v^*$.

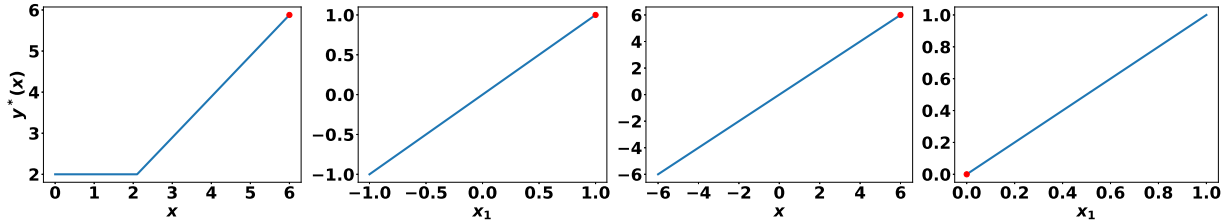
By mirroring the arguments in Theorem 1, note that line 14 of Algorithm 3 must be run infinitely often with $Y_d^G \leftarrow Y_d^G \cup \{(J_y^*(x^k), y^*(x^k) - J_y^*(x^k)x^k)\}$ or $Y_d^G \leftarrow Y_d^G \cup \{(0, y^*(x^k))\}$. Therefore, the asymptotic behavior of Algorithm 3 is the same as the algorithm that adds at each iteration either $(J_y^*(x^k), y^*(x^k) - J_y^*(x^k)x^k)$ to Y_d^G if Theorem 8 holds, or $(0, y^*(x^k))$ to Y_d^G otherwise. We show that $LBD^k \to v^*$ for the above algorithm.

Define the index sets $\mathcal{J}_k := \{j \in [k] : \text{Theorem 8 holds}\}$ and $\mathcal{L}_k := \{1, \ldots, k\} \backslash \mathcal{J}_k$ corresponding to the above algorithm. By construction, we have $\forall l, k$ such that $l > k$:

$$g(x^l, \text{proj}_Y(\tilde{A}^k x^l + \tilde{b}^k)) \leq 0, \text{ if } k \in \mathcal{J}_k, \quad \text{and} \quad g(x^l, y^*(x^k)) \leq 0, \text{ if } k \in \mathcal{L}_k,$$

where $\tilde{A}^k := J_y^*(x^k)$, $\tilde{b}^k := y^*(x^k) - J_y^*(x^k)x^k$ if $k \in \mathcal{J}_k$. Continuity of g, $\text{proj}_Y(\cdot)$ and compactness of X, Y ensure uniform continuity, which implies that for any $\varepsilon > 0$, there exists $\kappa > 0$ such that for all $x \in X$ with $\|x - x^l\| < \kappa$ and $\forall l, k$ with $l > k$:

$$g(x, \text{proj}_Y(\tilde{A}^k x + \tilde{b}^k)) < \varepsilon, \text{ if } k \in \mathcal{J}_k, \quad \text{and} \quad g(x, y^*(x^k)) < \varepsilon, \text{ if } k \in \mathcal{L}_k. \tag{9}$$



(a) Example 1 (b) Example 3 (c) Example 4 (d) Example 5

Figure 4: Optimal solution mapping y^* (note that it only depends on x_1 for Examples 3 and 5). The red dot indicates $y^*(x^1)$ at $x^1 \in \arg\min_{x \in X} f(x)$.

Since $x^k \to x^*$, we have $\|x^l - x^k\| < \kappa$, $\forall l, k$ with $l > k \geq \bar{K}$. Setting $x = x^k$ in (9) and noting $\tilde{A}^k x^k + \tilde{b}^k = y^*(x^k)$ if $k \in \mathcal{J}_k$ yields $0 < g(x^k, y^*(x^k)) < \varepsilon$, $\forall k \geq \bar{K}$. Therefore, $g(x^k, y^*(x^k)) = G(x^k) \to 0$ and continuity of G implies $G(x^*) = 0$. □

Our next theorem establishes the rate of convergence of the lower bounds generated using Algorithm G-OPT when the max-min problem (7) is solved to *global* optimality at each iteration (this result is mainly of theoretical interest).

We require the following lemma, which is sharp for affine functions.

Lemma 6. Suppose $Z \subset \mathbb{R}^N$ is compact and convex and $F : Z \to \mathbb{R}^M$ is continuously differentiable with a Lipschitz continuous gradient. Let $L_{\nabla F}$ denote the Lipschitz constant of ∇F on Z. Then $\forall \varepsilon > 0$, there exist $J = \left\lceil \left(1 + \operatorname{diam}(Z)\sqrt{\frac{L_{\nabla F}}{2\varepsilon}}\right)^N \right\rceil$ affine functions $\{(\alpha^j)^{\mathrm{T}} z + \beta^j\}_{j=1}^J$ with $\sup_{z \in Z} \min_{j \in [J]} \left\| F(z) - \left((\alpha^j)^{\mathrm{T}} z + \beta^j\right) \right\| \leq \varepsilon$.

Theorem 7. Consider Algorithm G-OPT with $\varepsilon_f > 0$ and $\delta = 0$. Suppose X and Y are convex sets and $\{g(x, \cdot)\}_{x \in X}$ is uniformly Lipschitz continuous on Y with Lipschitz constant $L_{g,y} > 0$. Additionally, suppose y^* is continuously differentiable on X with a Lipschitz continuous gradient. Let $L_{\nabla y}$ denote the Lipschitz constant of ∇y^* on X. If the max-min problem (7) is solved to *global* optimality at each iteration, then Algorithm G-OPT terminates with an ε_f-feasible point in at most $\left\lceil \left(1 + \operatorname{diam}(X)\sqrt{\frac{L_{\nabla y} L_{g,y}}{2\varepsilon_f}}\right)^{d_x} \right\rceil$ iterations.

Chapter 3 of Fiacco [21] presents conditions when the assumption on y^* holds. The convexity assumption on the set X may be relaxed by considering any convex superset of X. Note that the bound on the number of iterations in Theorem 7 scales like $\varepsilon_f^{-0.5 d_x}$ compared to the $\varepsilon_f^{-d_x}$ scaling in Proposition 3. The rate at which $LBD^k \to v^*$ for G-OPT can be derived similar to Theorem 4. Since we may set $\bar{A}^k = 0$, $\forall k$, Algorithm G-OPT generates tighter lower bounds than Algorithm OPT if we solve the max-min problem (7) to global optimality.

3.3 Solving the max-min problems

In addition to the challenges outlined in Section 2, solving problems (7) and (8) globally may also be challenging due to the nonsmooth projection operator. Therefore, we design effective heuristic methods for solving these problems and empirically show in Section 5 that they often yield good generalized discretizations with fast convergence of lower bounds.

Properties of the optimal solution mapping y^* Before we outline our heuristic approach for solving problems (7) and (8), we provide empirical motivation for (bounding-focused) generalized discretization methods. Figure 4 plots the optimal solution mapping y^* for Examples 1, 3, 4, and 5. Interestingly, this mapping is well-behaved for all four examples (it is piecewise-linear for Example 1 and linear for Examples 3 to 5). Moreover, using $Y_d^G = \{(J_y^*(x^1), y^*(x^1) - J_y^*(x^1)x^1)\}$ in (G-LBP) at the point $(x^1, y^*(x^1))$ highlighted in these plots yields a lower bound equal to v^* for all four examples. However, this favorable situation

may not always be the case, and the mapping y^* may be nonconvex, nonsmooth, and even discontinuous in general. For example, any (SIP) with $X = Y = [0, 1]$ and $g(x, y) = (x - 0.5)y$ results in the optimal solution mapping $y^*(x) = \mathbb{1}(x - 0.5)$, where $\mathbb{1}(z) = 1$ if $z \geq 0$ and zero otherwise, which is discontinuous at $x = 0.5$. Example 2 from Section 2.2 provides another instance where y^* is discontinuous.

Example 2: Pick any $\bar{y} \in Y$, and consider the optimal solution mapping y^*:

$$y^*(x) := \begin{cases} \frac{x}{\|x\|}\sqrt{d_x - 1}, & \text{if } x \neq 0 \\ \bar{y}, & \text{if } x = 0 \end{cases}.$$

This mapping is discontinuous at $x = 0$ irrespective of the choice of $\bar{y} \in Y$. However, setting $Y_d^G = \{(I, 0)\}$ in (G-LBP), where I is the identity matrix, yields a lower bound equal to v^* since $\text{proj}_Y(x) = y^*(x)$, $\forall x \in X$. Therefore, a generalized discretization with $|Y_d^G| = 1$ is sufficient for convergence, which is in stark contrast with the discretization methods in Section 2.2 that require exponentially many iterations in the dimension d_x to converge.

A heuristic solution approach We only consider the setting where $Y = [y^L, y^U]$ for some $y^L, y^U \in \mathbb{R}^{d_y}$. The function $\text{proj}_Y(\cdot)$ can then be reformulated as the MILP-representable function $\text{mid}(y^L, \cdot, y^U)$ using the "lambda formulation" (see [53, Formulation (2.5)]), and (G-LBP) can be written as a mixed-integer nonlinear program (MINLP). Section 3.4 of Djelassi [15] also considers more general settings for Y.

Similar to the functions ϕ_k and ψ_k in Section 2, the functions ϕ_k^G and ψ_k^G are potentially nonsmooth and discontinuous. Therefore, we propose to solve the max-min problems in Algorithm 3 approximately by using gradients (whenever they exist) of a smooth approximation of ϕ_k^G and ψ_k^G within a bundle solver for nonsmooth, nonconvex optimization [29].

We begin by replacing the function $\text{mid}\{y^L, \cdot, y^U\}$ in problems (7) and (8) with the approximation $-t^{-1}\log((\exp(ty^L) + \exp(t\,\cdot))^{-1} + \exp(-ty^U))$, where the smoothing parameter t is set to 100. We estimate values of ϕ_k^G and ψ_k^G by solving these smooth approximations of the inner-minimization problems to *local* optimality. We then attempt to apply Theorem 8 to compute gradients of the smooth approximation of the local minimum value function of ϕ_k^G and ψ_k^G. Whenever the assumptions of Theorem 8 fail to hold, we estimate a generalized gradient of ϕ_k^G and ψ_k^G using similar heuristics as in Section 2.3 (however, based on numerical experiments, we exclude all weakly active constraints when SC does not hold). If the bundle solver terminates after a single iteration, we restart its solution using a random initialization for (A^k, b^k).

4 Generalizations

Multiple semi-infinite constraints Suppose (SIP) includes $|\mathcal{I}|$ semi-infinite constraints $g_i(x, y) \leq 0$, $\forall y \in Y^i$, $i \in \mathcal{I}$. The formulation below extends the max-min problem (2) for constructing a bounding-focused discretization at the first iteration of Algorithm 2.

$$(\bar{y}^1, \ldots, \bar{y}^{|\mathcal{I}|}) \in \underset{(y^1,\ldots,y^{|\mathcal{I}|}) \in Y^1 \times \cdots \times Y^{|\mathcal{I}|}}{\arg\max} \; \min_{x \in X} \; f(x)$$
$$\text{s.t. } g_i(x, y^i) \leq 0, \quad \forall i \in \mathcal{I}.$$

Extensions of the max-min problems (3) and (4) and the bounding-focused generalized discretization methods in Section 3 readily follow.

Generalized discretizations based on nonlinear approximations of y^* Instead of restricting ourselves to bounding-focused linear approximations of $y^*(x)$ as in Section 3, we can construct bounding-focused *nonlinear* approximations of y^* for potentially faster convergence. Let $\gamma : X \times \Theta \to \mathbb{R}^{d_y}$ be any family of

functions. We propose the following extension of (G-LBP):

$$\begin{aligned} \min_{x \in X} \ & f(x) \\ \text{s.t.} \ & g\big(x, \text{proj}_Y(\gamma(x,\theta))\big) \leq 0, \quad \forall \theta \in \Theta_d. \end{aligned} \tag{10}$$

Clearly, (G-LBP) is a special case of problem (10) where $\{\gamma(\cdot,\theta)\}_\theta$ is the set of all parametric affine (in x) functions. Extensions of the max-min problems (7) and (8) to determine an optimal sequence of parameters $\{\theta^k\}_k$ readily follow.

Mixed-integer SIPs The presence of integer variables in (SIP) precludes the use of the sensitivity theory in Appendix A for solving the max-min problems (3), (4), (7), and (8). Because we can heuristically solve these max-min problems *without* sacrificing convergence of our (generalized) discretization methods, one heuristic is to use sensitivities of the value functions of the inner-minimization problems with the integer variables fixed to an optimal solution. An alternative is to use smoothing-based approaches [20] for approximating sensitivity information.

5 Numerical results

We compare Algorithms `GREEDY`, `2GREEDY`, `HYBRID`, and `OPT` in Section 2.1 and Algorithms `G-GREEDY`, `G-2GREEDY`, `G-HYBRID`, and `G-OPT` in Section 3.1 with the BF algorithm on *standard* nonconvex SIP instances from the literature [32, 41, 51, 54]. These include a larger, parameterized version of Problem 5 from Watson [54], which we label "Watson 5_10" (cf. [50]). We only consider instances with scalar semi-infinite constraints and sets Y of the form $Y = [y^L, y^U]$. We omit instances with trigonometric functions as they are not supported by BARON [40]. Although these instances are small-scale nonconvex SIPs ($d_x \leq 10$ and $d_y \leq 2$), solving them to *global* optimality is not trivial.

We also test our algorithms on larger-scale convex SIPs ($d_x \in [21, 105]$ and $d_y \in [5, 13]$) from Cerulli et al. [7], where (LLP(x)) is a (potentially nonconvex) quadratic program. Our computational times are lower than those reported in Cerulli et al. [7], which we attribute to three differences in our implementation: (*i*) we use BARON instead of Gurobi, (*ii*) we use a different termination criterion (detailed in the next section), and (*iii*) we enforce the symmetry constraint in the formulation symbolically using JuMP.

Because our focus is on designing discretization methods with tight lower bounds, we only compare our approaches with the BF algorithm (the state-of-the-art discretization method for nonconvex SIPs) and do not consider algorithms for constructing feasible points (see, e.g., [16, 31, 51]).

5.1 Implementational details

Our codes are compiled using Julia 1.7.3, JuMP 1.24.0 [19], BARON 24.1.3 [40] or Gurobi 12.0.1 as the global solver, Knitro 14.2.0 as the local NLP solver, Gurobi 12.0.1 as the LP solver, and the bundle solver MPBNGC 2.0 [29] for the max-min problems. Our codes will be made available at `https://github.com/Process-Optimization-and-Control/Bounding-Focused-SIP-Discretizations`. All codes were run on a computer with an AMD Ryzen 7 PRO 8840U CPU (3.30 GHz, 8 cores) and 32 GB of RAM.

General algorithmic parameters Since (LBP) and (G-LBP) may not yield a feasible point finitely, we terminate our algorithms when the lower bound converges to within an absolute or relative tolerance of 10^{-3} of v^*. When available, v^* is specified as the analytical solution or the best known feasible solution reported in the literature [7, 15, 41, 51]; otherwise, it is computed offline by running the BF algorithm with $\varepsilon_f = 10^{-8}$ until an approximate feasible point is found. This termination criterion is chosen to directly assess the impact of our discretization methods on accelerating the convergence of the lower bound.

We use $\varepsilon_f = \delta = 10^{-8}$ in Algorithms 1, 2, and 3. The feasibility and optimality tolerances in Gurobi are set to 10^{-8}, with a maximum time limit of five minutes. All of BARON's parameters are set to default, except for `MaxTime` $= 300$ seconds. Knitro is used with the following parameters: `algorithm` $= 0$, `ftol` $= 10^{-8}$,

Instance	d_x	d_y	BF	GREEDY	2GREEDY	HYBRID	OPT
			number of iterations for convergence				
Watson 2	2	1	**2**	**2**	**2**	**2**	**2**
Watson 5	3	1	5	4	**2**	3	3
Watson 5_10	10	1	3	4	**2**	3	3
Watson 6	2	1	3	**2**	**2**	**2**	**2**
Watson 7	3	2	**2**	**2**	**2**	**2**	**2**
Watson 8	6	2	15	27	**6**	11	21
Watson 9	6	2	9	8	**5**	10	12
Watson h	2	1	18	21	**13**	23	25
Watson n	2	1	3	**2**	**2**	**2**	**2**
Seidel & Küfer 2.1	2	1	8	**3**	**3**	**3**	**3**
Tsoukalas & Rustem 2.1	1	1	8	**4**	5	5	8
Mitsos 4_3	3	1	5	4	**3**	4	4
Mitsos 4_6	6	1	7	7	**6**	8	7
Mitsos DP	1	1	28	**2**	**2**	**2**	**2**
Cerulli et al. PSD 1	21	5	**2**	**2**	**2**	**2**	**2**
Cerulli et al. PSD 2	21	5	**2**	**2**	**2**	**2**	**2**
Cerulli et al. PSD 3	21	5	**2**	**2**	**2**	**2**	**2**
Cerulli et al. PSD 4	21	5	**2**	**2**	**2**	**2**	**2**
Cerulli et al. PSD 5	66	10	5	**2**	5	5	5
Cerulli et al. PSD 6	66	10	6	**2**	6	**2**	**2**
Cerulli et al. PSD 7	105	13	7	**3**	7	4	4
Cerulli et al. PSD 8	105	13	5	**2**	5	**2**	**2**

Table 2: Comparison of BF, GREEDY, 2GREEDY, HYBRID, and OPT algorithms. Bold entries correspond to the minimum number of iterations for each instance.

maxtime $= 10$, and feastol $= 10^{-8}$. We enable Knitro's multistart heuristic with a maximum of five starts. Due to a limitation in Knitro's Julia interface, the multistart procedure is executed serially rather than in parallel, which inflates the reported solve times for the max-min problems.

Algorithm-specific parameters The starting point for the max-min problem solved by the 2GREEDY method is specified as $y^L + \zeta(y^U - y^L)$, where ζ is a diagonal matrix with random diagonal elements $\zeta_{ii} \sim U([0,1])$. Similarly, for the G-2GREEDY method, the elements of A^k are initialized randomly from $U([0,1])$ and b^k is set to $y^L + \zeta(y^U - y^L) - A^k x^k$ with $\zeta_{ii} \sim U([0,1])$. We set the parameter $K = 3$ for Algorithms HYBRID and G-HYBRID.

5.2 Bounding-focused discretization methods

Table 2 details the instance, dimensions d_x and d_y, and the number of iterations k required by each method for the lower bound to converge to v^* (note that the number of discretization points at termination is $2k-2$ for 2GREEDY and $k-1$ for the other methods). Computational times are reported in Table 3, with performance profiles [18] shown in Figure 5. Most problems are solved within a few seconds, with the notable exception of the larger Cerulli instances, where both the max-min solve time and the lower bounding time can be significant. Although our new discretization methods reduce the number of global solves, their computational advantage is often offset by the added cost of solving max-min problems, especially since the BF algorithm exhibits relatively low runtime on all but two instances.

On almost all instances, our proposed bounding-focused discretization methods require fewer iterations for convergence than the BF algorithm (the main exception is "Watson h", where numerical issues affect our methods). Notably, for most instances, our new discretization methods converge within only two iterations

Instance	d_x	d_y	BF	GREEDY	2GREEDY	HYBRID	OPT
			Total time to solve LBP/LLP/max-min (seconds)				
Watson 2	2	1	0.07 / 0.06 / 0.00	0.06 / 0.05 / 0.27	0.06 / 0.06 / 0.01	0.07 / 0.04 / 0.29	0.09 / 0.04 / 0.25
Watson 5	3	1	0.16 / 0.43 / 0.00	0.08 / 0.28 / 0.36	0.05 / 0.18 / 0.07	0.05 / 0.07 / 0.36	0.04 / 0.06 / 0.35
Watson 5_10	10	1	0.12 / 0.08 / 0.00	0.17 / 0.13 / 0.58	0.04 / 0.07 / 0.07	0.08 / 0.08 / 0.64	0.07 / 0.08 / 0.58
Watson 6	2	1	0.15 / 0.05 / 0.00	0.09 / 0.03 / 0.02	0.08 / 0.03 / 0.01	0.07 / 0.04 / 0.02	0.06 / 0.02 / 0.03
Watson 7	3	2	0.06 / 0.03 / 0.00	0.08 / 0.02 / 0.01	0.04 / 0.02 / 0.01	0.05 / 0.02 / 0.00	0.05 / 0.02 / 0.00
Watson 8	6	2	0.00 / 1.29 / 0.00	0.01 / 2.73 / 4.31	0.00 / 0.37 / 0.68	0.00 / 0.93 / 1.25	0.00 / 1.89 / 5.13
Watson 9	6	2	0.00 / 0.29 / 0.00	0.00 / 0.31 / 0.37	0.00 / 0.22 / 0.29	0.00 / 0.46 / 1.02	0.00 / 0.56 / 2.26
Watson h	2	1	1.83 / 0.65 / 0.00	1.67 / 0.84 / 2.52	1.13 / 0.46 / 1.43	1.82 / 0.82 / 1.58	2.11 / 0.99 / 3.13
Watson n	2	1	0.05 / 0.04 / 0.00	0.03 / 0.03 / 0.01	0.05 / 0.06 / 0.02	0.03 / 0.03 / 0.01	0.02 / 0.02 / 0.02
Seidel & Küfer 2.1	2	1	0.24 / 0.25 / 0.00	0.10 / 0.04 / 0.19	0.09 / 0.09 / 0.07	0.05 / 0.05 / 0.22	0.10 / 0.04 / 0.23
Tsoukalas & Rustem 2.1	1	1	0.47 / 0.29 / 0.00	0.21 / 0.13 / 0.37	0.31 / 0.17 / 0.74	0.23 / 0.19 / 0.57	0.46 / 0.31 / 1.66
Mitsos 4_3	3	1	0.00 / 0.21 / 0.00	0.00 / 0.14 / 0.08	0.00 / 0.11 / 0.02	0.00 / 0.12 / 0.20	0.00 / 0.11 / 0.19
Mitsos 4_6	6	1	0.00 / 0.74 / 0.00	0.00 / 0.60 / 0.20	0.00 / 0.49 / 0.39	0.00 / 0.78 / 0.68	0.00 / 0.57 / 0.73
Mitsos DP	1	1	1.51 / 1.30 / 0.00	0.06 / 0.08 / 0.03	0.04 / 0.08 / 0.03	0.06 / 0.06 / 0.03	0.04 / 0.06 / 0.02
Cerulli et al. PSD 1	21	5	0.29 / 0.09 / 0.00	0.21 / 0.07 / 2.11	0.28 / 0.09 / 2.12	0.25 / 0.06 / 2.23	0.15 / 0.04 / 2.24
Cerulli et al. PSD 2	21	5	0.24 / 0.09 / 0.00	0.18 / 0.08 / 2.11	0.26 / 0.09 / 1.81	0.24 / 0.06 / 2.24	0.15 / 0.08 / 2.26
Cerulli et al. PSD 3	21	5	0.25 / 0.09 / 0.00	0.27 / 0.09 / 3.35	0.17 / 0.06 / 1.88	0.26 / 0.03 / 3.57	0.19 / 0.04 / 3.58
Cerulli et al. PSD 4	21	5	0.28 / 0.04 / 0.00	0.21 / 0.04 / 2.00	0.28 / 0.04 / 2.27	0.26 / 0.03 / 2.12	0.15 / 0.03 / 2.11
Cerulli et al. PSD 5	66	10	1.53 / 0.13 / 0.00	0.61 / 0.04 / 31.50	1.69 / 0.11 / 805.54	1.73 / 0.11 / 570.17	1.72 / 0.09 / 571.48
Cerulli et al. PSD 6	66	10	1.75 / 0.14 / 0.00	0.68 / 0.06 / 72.45	2.03 / 0.13 / 559.22	0.58 / 0.04 / 73.35	0.61 / 0.03 / 74.03
Cerulli et al. PSD 7	105	13	5.55 / 0.17 / 0.00	2.44 / 0.07 / 313.27	5.76 / 0.16 / 1171.40	3.16 / 0.08 / 295.84	4.04 / 0.12 / 296.55
Cerulli et al. PSD 8	105	13	4.15 / 317.20 / 0.00	1.66 / 87.14 / 65.57	4.13 / 323.67 / 342.43	1.69 / 87.38 / 65.61	1.62 / 87.67 / 65.52

Table 3: Comparison of the total time taken to solve subproblems using the BF, GREEDY, 2GREEDY, HYBRID, and OPT algorithms.

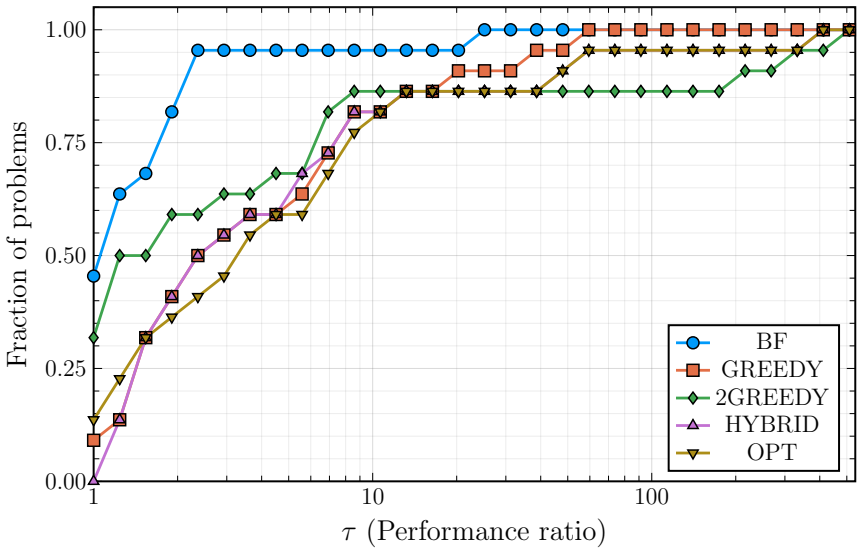


Figure 5: Performance profiles of the bounding-focused discretization methods.

(corresponding to a single discretization point for `GREEDY`, `HYBRID`, and `OPT`, and two points for `2GREEDY`). Although `OPT` is theoretically expected to perform at least as well as the other discretization methods in terms of iteration count, Table 2 reveals that this is not always the case in practice. This discrepancy arises because the solution of the max-min problem (3) can sometimes get stuck at poor local maxima. Overall, among the proposed methods, Algorithms `GREEDY` and `2GREEDY` deliver the best performance on small-scale instances in terms of both iteration count and computational time.

Table 4, Table 5, and Figure 6 present the performance of our discretization methods when the max-min problems (3) and (4) are relaxed and solved as mathematical programs with complementarity constraints (MPCCs), see Section 2.3. These MPCCs are solved using Knitro's dedicated algorithm for such problems. Comparing Table 4 with Table 2, we observe that the MPCC-based discretizations sometimes converge in fewer iterations, but in other cases require more iterations. However, the total time spent solving the max-min problems is generally lower with the MPCC approach (see Table 5), which results in improved performance profiles for the proposed methods compared to the BF algorithm (with the exception of `OPT`, which suffers from significant numerical issues on the "Tsoukalas & Rustem 2.1" and Cerulli et al. [7] instances).

5.3 Bounding-focused generalized discretization methods

We test our generalized discretization methods on small-scale SIP instances, as solving (G-LBP) to global optimality can be challenging for larger-scale SIPs.

Table 6, Table 7, and Figure 7 summarize the performance of our bounding-focused generalized discretization methods relative to the BF algorithm. While these methods perform well on some instances, they exceed the time limit for solving (G-LBP) on several others. BARON appears to stall while solving these MINLPs, potentially due to weak relaxations of (G-LBP). In most cases, the bundle method also terminates after one iteration during the solution of problems (3) and (4), which suggests that our initial guess is either already locally optimal or fails to provide a clear direction for improvement (possibly due to the smooth approximation of the projection operator). With the exception of "Watson h" and "Tsoukalas & Rustem 2.1", the generalized discretization methods do not offer a significant advantage over the bounding-focused discretization methods in Section 2.

Instance	d_x	d_y	BF	GREEDY	2GREEDY	HYBRID	OPT
			number of iterations for convergence				
Watson 2	2	1	**2**	**2**	**2**	**2**	**2**
Watson 5	3	1	5	6	**2**	7	5
Watson 5_10	10	1	3	4	**2**	3	3
Watson 6	2	1	3	**3**	**3**	**3**	**3**
Watson 7	3	2	**2**	**2**	**2**	**2**	**2**
Watson 8	6	2	15	7	**5**	15	8
Watson 9	6	2	9	5	**4**	13	5
Watson h	2	1	**18**	**18**	**18**	**18**	**18**
Watson n	2	1	3	**2**	**2**	**2**	**2**
Seidel & Küfer 2.1	2	1	8	**2**	3	**2**	**2**
Tsoukalas & Rustem 2.1	1	1	8	9	**6**	9	> 100
Mitsos 4_3	3	1	5	3	**2**	3	3
Mitsos 4_6	6	1	7	7	6	**4**	**4**
Mitsos DP	1	1	28	**2**	**2**	**2**	**2**
Cerulli et al. PSD 1	21	5	**2**	3	**2**	5	> 100
Cerulli et al. PSD 2	21	5	**2**	3	**2**	5	> 100
Cerulli et al. PSD 3	21	5	**2**	3	**2**	5	> 100
Cerulli et al. PSD 4	21	5	**2**	3	**2**	5	> 100
Cerulli et al. PSD 5	66	10	**5**	**5**	**5**	8	92
Cerulli et al. PSD 6	66	10	**6**	**6**	**6**	9	32
Cerulli et al. PSD 7	105	13	**7**	8	**7**	10	32
Cerulli et al. PSD 8	105	13	**5**	6	**5**	8	34

Table 4: Results with the MPCC relaxation of the max-min problems. Bold entries correspond to the minimum number of iterations for each instance.

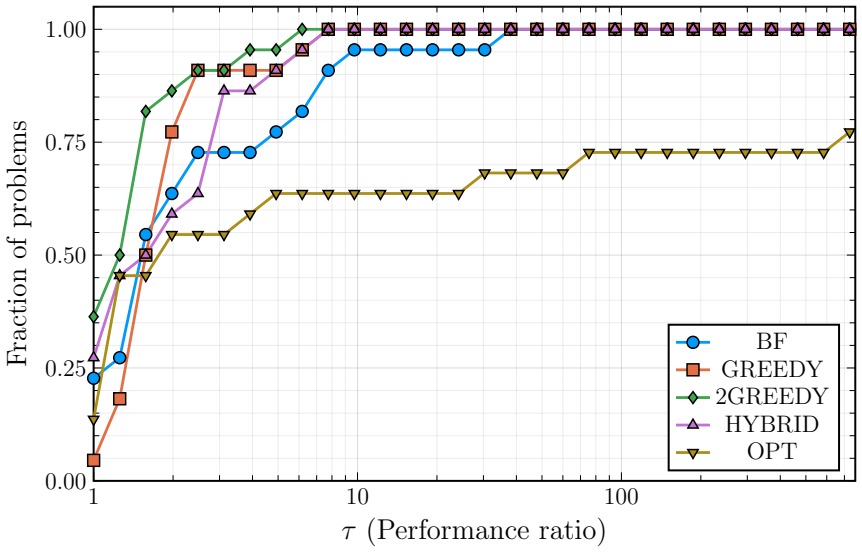


Figure 6: Performance profiles of the bounding-focused discretization methods when the max-min problem is relaxed and solved as an MPCC.

Instance	d_x	d_y	BF	GREEDY	2GREEDY	HYBRID	OPT
			Total time to solve LBP/LLP/max-min (seconds)				
Watson 2	2	1	0.07 / 0.06 / 0.00	0.08 / 0.08 / 0.00	0.06 / 0.06 / 0.00	0.05 / 0.04 / 0.00	0.05 / 0.05 / 0.00
Watson 5	3	1	0.16 / 0.43 / 0.00	0.10 / 0.44 / 0.02	0.02 / 0.06 / 0.00	0.19 / 0.16 / 0.02	0.09 / 0.19 / 0.00
Watson 5_10	10	1	0.12 / 0.08 / 0.00	0.12 / 0.13 / 0.02	0.05 / 0.06 / 0.00	0.07 / 0.07 / 0.00	0.06 / 0.05 / 0.02
Watson 6	2	1	0.15 / 0.05 / 0.00	0.14 / 0.04 / 0.02	0.13 / 0.03 / 0.02	0.11 / 0.02 / 0.00	0.11 / 0.03 / 0.02
Watson 7	3	2	0.06 / 0.03 / 0.00	0.04 / 0.02 / 0.00	0.04 / 0.02 / 0.00	0.04 / 0.01 / 0.00	0.02 / 0.02 / 0.00
Watson 8	6	2	0.00 / 1.29 / 0.00	0.00 / 0.33 / 0.03	0.00 / 0.15 / 0.02	0.00 / 1.03 / 0.08	0.00 / 0.27 / 0.03
Watson 9	6	2	0.00 / 0.29 / 0.00	0.00 / 0.22 / 0.02	0.00 / 0.10 / 0.03	0.00 / 0.69 / 0.05	0.00 / 0.14 / 0.08
Watson h	2	1	1.83 / 0.65 / 0.00	1.69 / 0.52 / 0.13	1.63 / 0.46 / 0.16	1.54 / 0.38 / 0.16	1.50 / 0.41 / 0.17
Watson n	2	1	0.05 / 0.04 / 0.00	0.02 / 0.05 / 0.00	0.04 / 0.07 / 0.00	0.02 / 0.03 / 0.00	0.04 / 0.02 / 0.00
Seidel & Küfer 2.1	2	1	0.24 / 0.25 / 0.00	0.06 / 0.03 / 0.00	0.10 / 0.22 / 0.00	0.07 / 0.03 / 0.00	0.04 / 0.02 / 0.00
Tsoukalas & Rustem 2.1	1	1	0.47 / 0.29 / 0.00	0.50 / 0.22 / 0.09	0.31 / 0.19 / 0.06	0.35 / 0.20 / 0.08	Did not converge
Mitsos 4_3	3	1	0.00 / 0.21 / 0.00	0.00 / 0.06 / 0.00	0.00 / 0.07 / 0.00	0.00 / 0.05 / 0.00	0.00 / 0.06 / 0.00
Mitsos 4_6	6	1	0.00 / 0.74 / 0.00	0.00 / 0.68 / 0.03	0.00 / 0.54 / 0.00	0.00 / 0.14 / 0.02	0.00 / 0.12 / 0.02
Mitsos DP	1	1	1.51 / 1.30 / 0.00	0.02 / 0.06 / 0.00	0.05 / 0.07 / 0.00	0.04 / 0.04 / 0.00	0.04 / 0.05 / 0.00
Cerulli et al. PSD 1	21	5	0.29 / 0.09 / 0.00	0.38 / 0.11 / 0.02	0.18 / 0.07 / 0.00	0.56 / 0.15 / 0.02	Did not converge
Cerulli et al. PSD 2	21	5	0.24 / 0.09 / 0.00	0.33 / 0.12 / 0.02	0.16 / 0.09 / 0.00	0.49 / 0.20 / 0.02	Did not converge
Cerulli et al. PSD 3	21	5	0.25 / 0.09 / 0.00	0.34 / 0.11 / 0.00	0.17 / 0.08 / 0.00	0.54 / 0.15 / 0.00	Did not converge
Cerulli et al. PSD 4	21	5	0.28 / 0.04 / 0.00	0.40 / 0.06 / 0.02	0.32 / 0.03 / 0.00	0.49 / 0.09 / 0.00	Did not converge
Cerulli et al. PSD 5	66	10	1.53 / 0.13 / 0.00	1.90 / 0.13 / 0.50	1.76 / 0.11 / 0.27	2.91 / 0.18 / 0.67	36.50 / 1.70 / 1120.73
Cerulli et al. PSD 6	66	10	1.75 / 0.14 / 0.00	2.28 / 0.10 / 1.08	2.03 / 0.09 / 0.38	3.11 / 0.15 / 1.47	12.08 / 0.69 / 119.33
Cerulli et al. PSD 7	105	13	5.55 / 0.17 / 0.00	6.92 / 0.18 / 3.88	5.93 / 0.15 / 4.53	8.44 / 0.21 / 7.56	27.74 / 0.66 / 112.41
Cerulli et al. PSD 8	105	13	4.15 / 317.20 / 0.00	5.07 / 322.16 / 4.69	4.09 / 318.09 / 3.86	6.67 / 332.77 / 2.56	28.58 / 918.75 / 404.45

Table 5: Comparison of the total time taken to solve subproblems using the BF, GREEDY, 2GREEDY, HYBRID, and OPT algorithms when the max-min problem is relaxed and solved as an MPCC. Instances marked as "Did not converge" either exceeded the maximum time limit while solving (LBP) or (LLP(x)), or reached the maximum number of iterations.

Instance	d_x	d_y	BF	G-GREEDY	G-2GREEDY	G-HYBRID	G-OPT
			number of iterations for convergence				
Watson 2	2	1	**2**	**2**	**2**	**2**	**2**
Watson 5	3	1	5	5	**3**	4	4
Watson 5_10	10	1	**3**	**3**	**3**	**3**	**3**
Watson 6	2	1	3	**2**	**2**	**2**	**2**
Watson 7	3	2	**2**	TLE	TLE	39	TLE
Watson 8	6	2	15	TLE	TLE	TLE	TLE
Watson 9	6	2	9	TLE	TLE	TLE	TLE
Watson h	2	1	18	**2**	**2**	**2**	**2**
Watson n	2	1	3	**3**	**3**	**3**	**3**
Seidel & Küfer 2.1	2	1	8	**2**	**2**	**2**	**2**
Tsoukalas & Rustem 2.1	1	1	8	4	**3**	4	4
Mitsos 4_3	3	1	**5**	9	**5**	**5**	6
Mitsos 4_6	6	1	**7**	TLE	TLE	**7**	**7**
Mitsos DP	1	1	28	**2**	**2**	**2**	**2**

Table 6: Comparison of the BF, `G-GREEDY`, `G-2GREEDY`, `G-HYBRID`, and `G-OPT` algorithms. Bold entries correspond to the minimum number of iterations for each instance, and TLE denotes the time limit was exceeded while solving (G-LBP) or (LLP(x)).

5.4 Discussion of results

Tables 2 to 7 and Figures 5 to 7 show that our bounding-focused (generalized) discretization methods have the potential to significantly reduce the number of iterations for convergence relative to the BF algorithm. While Algorithms `GREEDY` and `2GREEDY` may not result in *optimal* bounding-focused discretizations, we find that they offer *viable* alternatives to the BF algorithm. These algorithms require fewer iterations to converge and only necessitate the heuristic solution of the max-min problem (4) with d_y variables at each iteration. Numerical experiments suggest that these new bounding-focused discretization methods are competitive with the BF algorithm on small and medium-scale SIPs, especially when the max-min problem is relaxed and solved as an MPCC.

While our preliminary numerical results are encouraging, there is a need for the design of more efficient and reliable algorithms to solve our max-min formulations, particularly addressing the generalized discretization problems (7) and (8). We anticipate that our new discretization methods will offer significant advantages when the BF algorithm requires numerous iterations to converge, and the solution of (LBP) to global optimality is relatively time-consuming. Our new bounding-focused discretization methods can also be used as an expert strategy for machine learning approaches that seek to learn an optimal sequence of discretizations for solving families of nonconvex SIPs (cf. [14, 26, 38]). Our future work will delve into exploring the effectiveness of these new discretization methods specifically on large-scale nonconvex SIPs.

6 Future work

There are many interesting avenues for future work. First, we wish to explore the effectiveness of our new discretization methods for large-scale nonconvex SIPs. Second, we would like to extend our bounding-focused (generalized) discretization methods to generalized semi-infinite programs [17, 33]. Third, our bounding-focused discretization methods could be modified (cf. [31]) to generate feasible points to (SIP). Fourth, extensions of our max-min formulations can enable the design of more efficient cutting-plane methods for a broader class of optimization problems (cf. [38]). Finally, using machine learning to learn a sequence of optimal discretizations (cf. [14, 26, 38]) can mitigate the computational burden of solving our max-min problems for larger dimensions.

Instance	d_x	d_y	BF	G-GREEDY	G-2GREEDY	G-HYBRID	G-OPT
			Total time to solve (G-)LBP/LLP/max-min (seconds)				
Watson 2	2	1	0.07 / 0.06 / 0.00	0.234 / 0.04 / 1.32	0.151 / 0.05 / 0.47	0.129 / 0.04 / 0.02	0.098 / 0.04 / 0.03
Watson 5	3	1	0.16 / 0.43 / 0.00	0.853 / 0.17 / 1.00	0.463 / 0.14 / 0.08	0.765 / 0.13 / 0.28	0.586 / 0.09 / 1.86
Watson 5_10	10	1	0.12 / 0.08 / 0.00	0.23 / 0.05 / 0.12	0.473 / 0.08 / 0.03	0.19 / 0.08 / 0.05	0.18 / 0.07 / 0.05
Watson 6	2	1	0.15 / 0.05 / 0.00	0.174 / 0.03 / 0.14	0.132 / 0.02 / 0.02	0.133 / 0.02 / 0.13	0.226 / 0.02 / 0.15
Watson 7	3	2	0.06 / 0.03 / 0.00	Did not converge	Did not converge	263.52 / 0.50 / 327.58	Did not converge
Watson 8	6	2	0.002 / 1.29 / 0.00	Did not converge	Did not converge	Did not converge	Did not converge
Watson 9	6	2	0.002 / 0.29 / 0.00	Did not converge	Did not converge	Did not converge	Did not converge
Watson h	2	1	1.83 / 0.65 / 0.00	0.666 / 0.04 / 0.02	0.732 / 0.08 / 0.02	0.626 / 0.03 / 0.01	0.642 / 0.03 / 0.02
Watson n	2	1	0.05 / 0.04 / 0.00	0.246 / 0.04 / 1.74	0.302 / 0.04 / 4.58	0.166 / 0.03 / 1.77	0.159 / 0.04 / 1.77
Seidel & Küfer 2.1	2	1	0.24 / 0.25 / 0.00	0.212 / 0.06 / 0.16	0.563 / 0.03 / 0.19	0.186 / 0.04 / 0.36	0.306 / 0.06 / 0.38
Tsoukalas & Rustem 2.1	1	1	0.47 / 0.29 / 0.00	1.099 / 0.14 / 0.40	0.457 / 0.11 / 0.18	0.781 / 0.12 / 0.49	0.887 / 0.11 / 0.52
Mitsos 4_3	3	1	0.001 / 0.21 / 0.00	1.728 / 0.26 / 5.10	1.024 / 0.18 / 0.80	0.995 / 0.17 / 1.76	1.677 / 0.20 / 3.05
Mitsos 4_6	6	1	0.001 / 0.74 / 0.00	Did not converge	Did not converge	46.659 / 0.63 / 14.83	46.144 / 0.65 / 14.76
Mitsos DP	1	1	1.51 / 1.3 / 0.00	0.10 / 0.04 / 0.02	0.45 / 0.04 / 0.01	0.10 / 0.05 / 0.01	0.11 / 0.04 / 0.01

Table 7: Comparison of the total time taken to solve subproblems using the BF, G-GREEDY, G-2GREEDY, G-HYBRID, and G-OPT algorithms. Instances marked as "Did not converge" either exceeded the maximum time limit while solving (G-LBP) or (LLP(x)), or reached the maximum number of iterations.

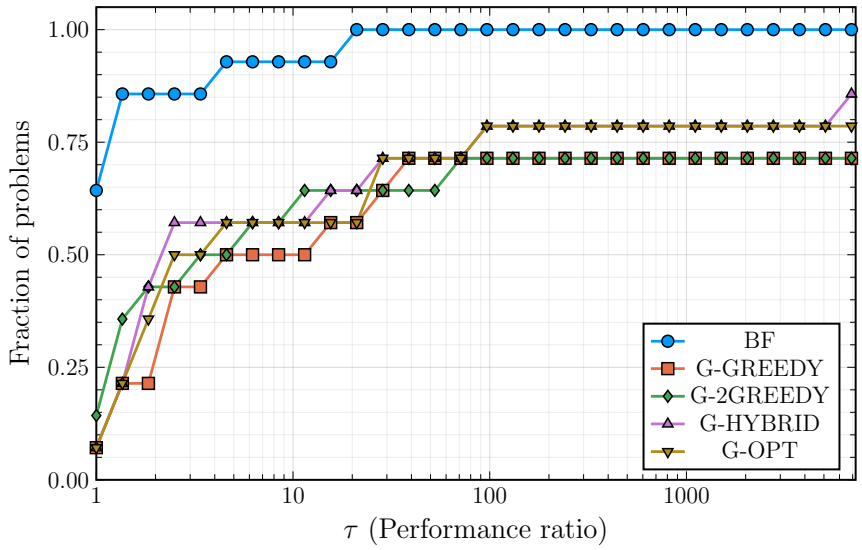


Figure 7: Performance profiles of the bounding-focused generalized discretization methods.

Acknowledgments

E.M.T. and J.J. acknowledge the support of the Norwegian Research Council through the AutoPRO project (RN: 309628). R.K. acknowledges funding from the Center for Nonlinear Studies at Los Alamos National Laboratory and projects 20210078DR and 20230091ER of the U.S. Department of Energy's LANL LDRD program. We thank Dr. Harsha Nagarajan (LANL) and Dr. Qi Zhang (UMN) for helpful discussions.

References

[1] A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, and J. Z. Kolter. Differentiable convex optimization layers. *Advances in Neural Information Processing Systems*, 32, 2019.

[2] R. Baltean-Lugojan, P. Bonami, R. Misener, and A. Tramontani. Scoring positive semidefinite cutting planes for quadratic optimization via trained neural networks. *Optimization Online. URL: http://www.optimization-online.org/DB_HTML/2018/11/6943.html*, 2019.

[3] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, Massachusetts, 2nd edition, 1999.

[4] B. Bhattacharjee, P. Lemonidis, W. H. Green Jr, and P. I. Barton. Global solution of semi-infinite programs. *Mathematical Programming*, 103(2):283–307, 2005.

[5] J. W. Blankenship and J. E. Falk. Infinitely constrained optimization problems. *Journal of Optimization Theory and Applications*, 19(2):261–281, 1976.

[6] J. V. Burke, F. E. Curtis, A. S. Lewis, M. L. Overton, and L. E. Simões. *Gradient sampling methods for nonsmooth optimization*, pages 201–225. Springer, Cham, 2020. doi: 10.1007/978-3-030-34910-3_6.

[7] M. Cerulli, A. Oustry, C. d'Ambrosio, and L. Liberti. Convergent algorithms for a class of convex semi-infinite programs. *SIAM Journal on Optimization*, 32(4):2493–2526, 2022.

[8] E. W. Cheney and A. A. Goldstein. Newton's method for convex programming and Tchebycheff approximation. *Numerische Mathematik*, 1(1):253–268, 1959.

[9] F. H. Clarke. *Optimization and nonsmooth analysis*. SIAM, Philadelphia, 1990.

[10] S. Coniglio and M. Tieves. On the generation of cutting planes which maximize the bound improvement. In *International Symposium on Experimental Algorithms*, pages 97–109. Springer, 2015.

[11] S. Das, A. Aravind, A. Cherukuri, and D. Chatterjee. Near-optimal solutions of convex semi-infinite programs via targeted sampling. *Annals of Operations Research*, 318(1):129–146, 2022.

[12] S. Dempe. *Foundations of bilevel programming*. Springer Science & Business Media, New York, 2002.

[13] S. Dempe. *Bilevel Optimization: Reformulation and First Optimality Conditions*, pages 1–20. Springer, Singapore, 2017. doi: 10.1007/978-981-10-4774-9_1.

[14] A. Deza and E. B. Khalil. Machine learning for cutting planes in integer programming: A survey. *arXiv preprint arXiv:2302.09166*, 2023.

[15] H. Djelassi. *Discretization-Based Algorithms for the Global Solution of Hierarchical Programs*. PhD thesis, RWTH Aachen, 2020.

[16] H. Djelassi and A. Mitsos. A hybrid discretization algorithm with guaranteed feasibility for the global solution of semi-infinite programs. *Journal of Global Optimization*, 68(2):227–253, 2017.

[17] H. Djelassi, A. Mitsos, and O. Stein. Recent advances in nonconvex semi-infinite programming: Applications and algorithms. *EURO Journal on Computational Optimization*, 9:100006, 2021.

[18] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91:201–213, 2002.

[19] I. Dunning, J. Huchette, and M. Lubin. JuMP: A modeling language for mathematical optimization. *SIAM Review*, 59 (2):295–320, 2017.

[20] Y. M. Ermoliev, V. I. Norkin, and R. J. Wets. The minimization of semicontinuous functions: mollifier subgradients. *SIAM Journal on Control and Optimization*, 33(1):149–167, 1995.

[21] A. V. Fiacco. *Introduction to sensitivity and stability analysis in nonlinear programming*, volume 165. Academic Press, New York, 1983.

[22] C. A. Floudas and O. Stein. The adaptive convexification algorithm: a feasible point method for semi-infinite programming. *SIAM Journal on Optimization*, 18(4):1187–1208, 2008.

[23] I. E. Grossmann, K. P. Halemane, and R. E. Swaney. Optimization strategies for flexible chemical processes. *Computers & Chemical Engineering*, 7(4):439–462, 1983.

[24] S. M. Harwood, D. J. Papageorgiou, and F. Trespalacios. A note on semi-infinite program bounding methods. *Optimization Letters*, 15(4):1485–1490, 2021.

[25] H. Hijazi, P. Bonami, and A. Ouorou. An outer-inner approximation for separable mixed-integer nonlinear programs. *INFORMS Journal on Computing*, 26(1):31–44, 2014.

[26] R. Kannan, H. Nagarajan, and D. Deka. Strong partitioning and a machine learning approximation for accelerating the global optimization of nonconvex QCQPs. *arXiv preprint arXiv:2301.00306*, 2022.

[27] J. E. Kelley, Jr. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

[28] M. López and G. Still. Semi-infinite programming. *European Journal of Operational Research*, 180(2):491–518, 2007.

[29] M. M. Mäkelä. Multiobjective proximal bundle method for nonconvex nonsmooth optimization: Fortran subroutine MPBNGC 2.0. URL: http://napsu.karmitsa.fi/publications/pbncgc_report.pdf. *Reports of the Department of Mathematical Information Technology, Series B. Scientific Computing, B*, 13, 2003.

[30] A. Marendet, A. Goldsztejn, G. Chabert, and C. Jermann. A standard branch-and-bound approach for nonlinear semi-infinite problems. *European Journal of Operational Research*, 282(2):438–452, 2020.

[31] A. Mitsos. Global optimization of semi-infinite programs via restriction of the right-hand side. *Optimization*, 60(10-11): 1291–1308, 2011.

[32] A. Mitsos. A test set of semi-infinite programs (revised by hatim djelassi). Last Accessed on January 11, 2023. https://www.avt.rwth-aachen.de/cms/AVT/Forschung/Systemverfahrenstechnik/~kpdo/A-Test-Set-of-Semi-Infinite-Programs/, 2016.

[33] A. Mitsos and A. Tsoukalas. Global optimization of generalized semi-infinite programs via restriction of the right hand side. *Journal of Global Optimization*, 61(1):1–17, 2015.

[34] A. Mitsos, P. Lemonidis, C. K. Lee, and P. I. Barton. Relaxation-based bounds for semi-infinite programs. *SIAM Journal on Optimization*, 19(1):77–113, 2008.

[35] B. S. Mordukhovich, N. M. Nam, and N. D. Yen. Subgradients of marginal functions in parametric mathematical programming. *Mathematical Programming*, 116(1-2):369–396, 2009.

[36] A. Mutapcic and S. Boyd. Cutting-set methods for robust convex optimization with pessimizing oracles. *Optimization Methods & Software*, 24(3):381–406, 2009.

[37] Y. Nesterov. *Lectures on convex optimization*, volume 137. Springer, Switzerland, 2018.

[38] M. B. Paulus, G. Zarpellon, A. Krause, L. Charlin, and C. Maddison. Learning to cut by looking ahead: Cutting plane selection via imitation learning. In *International Conference on Machine Learning*, pages 17584–17600, 2022.

[39] D. Ralph and S. Dempe. Directional derivatives of the solution of a parametric nonlinear program. *Mathematical Programming*, 70(1-3):159–172, 1995.

[40] N. V. Sahinidis. BARON: A general purpose global optimization software package. *Journal of Global Optimization*, 8(2): 201–205, 1996.

[41] T. Seidel and K.-H. Küfer. An adaptive discretization method solving semi-infinite optimization problems with quadratic rate of convergence. *Optimization*, 71(8):2211–2239, 2022.

[42] A. Shapiro. Semi-infinite programming, duality, discretization and optimality conditions. *Optimization*, 58(2):133–161, 2009.

[43] P. Stechlinski, K. A. Khan, and P. I. Barton. Generalized sensitivity analysis of nonlinear programs. *SIAM Journal on Optimization*, 28(1):272–301, 2018.

[44] P. Stechlinski, J. Jäschke, and P. I. Barton. Generalized sensitivity analysis of nonlinear programs using a sequence of quadratic programs. *Optimization*, 68(2-3):485–508, 2019.

[45] O. Stein. *Bi-level strategies in semi-infinite programming*, volume 71. Springer Science & Business Media, New York, 2003.

[46] O. Stein and P. Steuermann. The adaptive convexification algorithm for semi-infinite programming with arbitrary index sets. *Mathematical Programming*, 136(1):183–207, 2012.

[47] O. Stein and G. Still. Solving semi-infinite optimization problems with interior point techniques. *SIAM Journal on Control and Optimization*, 42(3):769–788, 2003.

[48] G. Still. Discretization in semi-infinite programming: the rate of convergence. *Mathematical Programming*, 91(1):53–69, 2001.

[49] G. Still. Lectures on parametric optimization: An introduction. *Optimization Online. URL: https://optimization-online.org/2018/04/6587/*, 2018.

[50] Y. Tanaka, M. Fukushima, and T. Ibaraki. A globally convergent SQP method for semi-infinite nonlinear optimization. *Journal of Computational and Applied Mathematics*, 23(2):141–153, 1988.

[51] A. Tsoukalas and B. Rustem. A feasible point adaptation of the Blankenship and Falk algorithm for semi-infinite programming. *Optimization Letters*, 5(4):705–716, 2011.

[52] E. M. Turan, R. Kannan, and J. Jäschke. Design of PID controllers using semi-infinite programming. In Y. Yamashita and M. Kano, editors, *14th International Symposium on Process Systems Engineering*, volume 49 of *Computer Aided Chemical Engineering*, pages 439–444. Elsevier, 2022. doi: https://doi.org/10.1016/B978-0-323-85159-6.50073-7.

[53] J. P. Vielma. Mixed integer linear programming formulation techniques. *SIAM Review*, 57(1):3–57, 2015.

[54] G. A. Watson. Numerical experiments with globally convergent methods for semi-infinite programming problems. In *Semi-infinite programming and applications*, pages 193–205. Springer, Berlin, Heidelberg, 1983.

A Review of sensitivity theory

We briefly review standard results from parametric sensitivity theory [21, 49]. Consider the parametric nonlinear program (NLP):

$$\begin{aligned} \min_{z\in\mathbb{R}^n} \; & F(z,p) \\ \text{s.t. } & c_i(z,p) \le 0, \quad \forall i \in \mathcal{I}, \\ & c_i(z,p) = 0, \quad \forall i \in \mathcal{E}, \end{aligned} \tag{11}$$

where $z \in \mathbb{R}^n$ are decision variables, $p \in \mathbb{R}^d$ are parameters, $F : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}$ is the objective function, $c : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^{|\mathcal{I}|+|\mathcal{E}|}$ are the constraint functions, and $\mathcal{I}$ and $\mathcal{E}$ are finite index sets. We write $z^*(p)$ and $\nu^*(p)$ to denote a local minimum of problem (11) and its optimal value $\nu^*(p) := F(z^*(p), p)$.

The Lagrangian for problem (11) is $L(z, \lambda, p) := F(z,p) + \lambda^{\mathrm{T}} c(z,p)$, for Lagrange multipliers $\lambda \in \mathbb{R}^{|\mathcal{I}|+|\mathcal{E}|}$. Let $\lambda^*(p)$ denote Lagrange multipliers satisfying the KKT conditions at $z^*(p)$, and $\mathcal{A}(z,p) := \{i \in \mathcal{I} : c_i(z,p) = 0\} \cup \mathcal{E}$ denote the indices of active constraints at a feasible point z.

We now present sufficient conditions under which $\nabla_p \nu^*(p)$ and $\nabla_p z^*(p)$ may be computed (see Fiacco [21] or Still [49] for details).

Theorem 8. [Parametric sensitivities] Let $z^*(p)$ be a KKT point for problem (11) with associated Lagrange multipliers $\lambda^*(p)$. Suppose for some $\bar{p} \in \mathbb{R}^d$, functions F and c are twice continuously differentiable in a neighborhood of $(z^*(\bar{p}), \bar{p})$. Assume that the following conditions hold at $(z^*(\bar{p}), \lambda^*(\bar{p}))$:

- Linear independence constraint qualification (LICQ): the vectors $\nabla_z c_i(z^*(\bar{p}), \bar{p})$, $i \in \mathcal{A}(z^*(\bar{p}), \bar{p})$, are linearly independent.

- Strict complementarity (SC): $\lambda_i^*(\bar{p}) - c_i(z^*(\bar{p}), \bar{p})) > 0$, $\forall i \in \mathcal{I}$.

Additionally, suppose either

(a) $|\mathcal{A}(z^*(\bar{p}), \bar{p})| = n$, or

(b) the strong second order sufficient condition (SSOSC) holds at $(z^*(\bar{p}), \lambda^*(\bar{p}))$:

$$w^{\mathrm{T}} \nabla_z^2 L(z^*(\bar{p}), \lambda^*(\bar{p}), \bar{p}) w > 0, \quad \forall w \in W \backslash \{0\},$$

$$\begin{aligned} W := \big\{ w \in \mathbb{R}^n : (\nabla_z c_i(z^*(\bar{p}), \bar{p}))^{\mathrm{T}} w = 0, \ \ \forall i \in \{ i \in \mathcal{A}(z^*(\bar{p}), \bar{p}) \cap \mathcal{I} : \lambda_i^*(\bar{p}) > 0 \}, \\ (\nabla_z c_i(z^*(\bar{p}), \bar{p}))^{\mathrm{T}} w = 0, \ \ \forall i \in \mathcal{E} \big\}. \end{aligned}$$

Then $\exists \delta > 0$ such that $\forall p \in B_\delta(\bar{p})$, we can choose the mappings $z^*(p)$ and $\lambda^*(p)$ to be continuously differentiable on $B_\delta(\bar{p})$ and $z^*(p)$ to be a strict local minimizer of (11). Additionally, for all $p \in B_\delta(\bar{p})$, the gradient of the value function ν^* is given by

$$\nabla_p \nu^*(p) = \nabla_p L(z^*(p), \lambda^*(p), p),$$

and the gradient of the solution mapping $z^*(p)$ may be computed for each $p \in B_\delta(\bar{p})$ as follows depending on whether condition (a) or (b) above holds:

(a) Let $J_z(p) \in \mathbb{R}^{n \times n}$ and $J_p(p) \in \mathbb{R}^{n \times p}$ be matrices with rows $(\nabla_z c_i(z^*(p), p))^{\mathrm{T}}$, $i \in \mathcal{A}(z^*(p), p)$, and $(\nabla_p c_i(z^*(p), p))^{\mathrm{T}}$, $i \in \mathcal{A}(z^*(p), p)$, respectively. Then

$$\nabla_p z^*(p) = -[J_z(p)]^{-1} J_p(p).$$

(b) Let $H_{z,\lambda}(p) := \begin{bmatrix} \nabla_z^2 L(z^*(p), \lambda^*(p), p) & J_z(p) \\ (J_z(p))^{\mathrm{T}} & 0 \end{bmatrix}$, where $J_z(p)$ is a $|\mathcal{A}(z^*(p), p)| \times n$ matrix with rows $(\nabla_z c_i(z^*(p), p))^{\mathrm{T}}$, $i \in \mathcal{A}(z^*(p), p)$. Then

$$\begin{bmatrix} \nabla_p z^*(p) \\ \nabla_p \lambda_{\mathcal{A}}^*(p) \end{bmatrix} = -[H_{z,\lambda}(p)]^{-1} \begin{bmatrix} \nabla_{pz} L(z^*(p), \lambda^*(p), p) \\ \big(\nabla_p c_i(z^*(p), p)\big)_{i \in \mathcal{A}(z^*(p), p)} \end{bmatrix},$$

where $\lambda_{\mathcal{A}}^*(p)$ denotes the Lagrange multipliers of the active constraints at $z^*(p)$.

Proof. See Chapter 3 of Fiacco [21], or the unified Theorem 4.4 in Still [49]. □

Lemma 6.2 of Still [49] presents weaker assumptions under which the (local) value function ν^* is locally Lipschitz continuous. Theorem 1.12 of Dempe [13] and its surrounding discussion provides estimates of generalized gradients of ν^* in the above setting. Weaker assumptions for the solution mapping z^* to be Hölder continuous or locally Lipschitz continuous are presented in Theorems 6.2 to 6.5 of Still [49].

B Proofs

B.1 Proof of Proposition 2

The first part follows directly from the definition of the max-min problem (3), since the sequence of lower bounds obtained using Algorithm `OPT` dominates the sequence of lower bounds obtained using the BF algorithm. The second part follows, e.g., from Theorem 3.2 of Shapiro [42].

B.2 Proof of Proposition 3

The fact that $N \leq \left\lceil \left(\frac{\text{diam}(X) L_{g,x}}{\varepsilon_f} + 1 \right)^{d_x} \right\rceil$ follows, e.g., from Section 5.2 of Mutapcic and Boyd [36]. We now argue that $N \leq \left\lceil \left(\frac{\text{diam}(Y) L_{g,y}}{\varepsilon_f} + 1 \right)^{d_y} \right\rceil$.

Suppose the BF algorithm has not converged by iteration $k > 1$. Then for each $1 \leq j < k$, the candidate BF solution x^k at iteration k satisfies:

$$\begin{aligned} g(x^k, y^{BF,k}) > \varepsilon_f \text{ and } g(x^k, y^{BF,j}) \leq 0 &\implies g(x^k, y^{BF,k}) - g(x^k, y^{BF,j}) > \varepsilon_f \\ &\implies \|y^{BF,k} - y^{BF,j}\| > \frac{\varepsilon_f}{L_{gy}}. \end{aligned}$$

Therefore, the given upper bound on the number of iterations required for the BF algorithm to converge can be obtained by calculating the number of Euclidean balls of radius $\frac{\varepsilon_f}{L_{gy}}$ needed to cover $\left(Y + \frac{\varepsilon_f}{2L_{gy}} B\right)$, where B denotes the unit ball in $\mathbb{R}^{d_y}$ and $+$ denotes the Minkowski sum (cf. [36]).

B.3 Proof of Theorem 4

Lipschitz continuity of the value function V implies

$$V(\tilde{\varepsilon}) \geq V(0) - L_V \tilde{\varepsilon} = v^* - L_V \tilde{\varepsilon}, \quad \forall \tilde{\varepsilon} \in (0, \bar{\varepsilon}).$$

By mirroring the proof of Proposition 3, we conclude that whenever

$$k \geq \min \left\{ \left\lceil \left(\frac{\text{diam}(Y) L_{g,y}}{\tilde{\varepsilon}} + 1 \right)^{d_y} \right\rceil, \left\lceil \left(\frac{\text{diam}(X) L_{g,x}}{\tilde{\varepsilon}} + 1 \right)^{d_x} \right\rceil \right\},$$

the iterate x^k produced by the BF algorithm satisfies $G(x^k) \leq \tilde{\varepsilon}$. Consequently, for any such k, we have $LBD^k \geq V(\tilde{\varepsilon}) \geq v^* - L_V \tilde{\varepsilon}$ for the BF algorithm. The desired result for the BF algorithm follows by setting $\tilde{\varepsilon} = \frac{\varepsilon}{L_V}$.

The result for Algorithm `OPT` then readily follows since the sequence of lower bounds obtained using Algorithm `OPT` dominate the sequence of lower bounds obtained using the BF algorithm.

B.4 Proof of Lemma 6

The integral form of Taylor's theorem implies for any $z, \bar{z} \in Z$:

$$\|F(z) - F(\bar{z}) - \nabla F(\bar{z})^{\mathrm{T}} (z - \bar{z})\| \leq \frac{L_{\nabla F}}{2} \|z - \bar{z}\|^2.$$

The inequality $\|F(z) - F(\bar{z}) - \nabla F(\bar{z})^{\mathrm{T}} (z - \bar{z})\| \leq \varepsilon$ holds whenever we have $z \in \left\{ v \in Z : \|v - \bar{z}\| \leq \sqrt{\frac{2\varepsilon}{L_{\nabla F}}} \right\}$. The stated result follows by covering Z using balls of radius $\sqrt{\frac{2\varepsilon}{L_{\nabla F}}}$ (cf. proof of Proposition 3), setting z^j to be the center of the jth ball, and setting $\alpha^j = \nabla F(z^j)$, $\beta^j = F(z^j) - \nabla F(z^j)^{\mathrm{T}} z^j$.

B.5 Proof of Theorem 7

Suppose Algorithm `G-OPT` has not converged by iteration $k > 1$. The candidate solution x^k at iteration k of Algorithm `G-OPT` satisfies for each $1 \leq j < k$:

$$
\begin{aligned}
& g(x^k, y^*(x^k)) > \varepsilon_f \text{ and } g(x^k, \text{proj}_Y(\bar{A}^j x^k + \bar{b}^j)) \leq 0 \\
\implies & g(x^k, y^*(x^k)) - g(x^k, \text{proj}_Y(\bar{A}^j x^k + \bar{b}^j)) > \varepsilon_f \\
\implies & L_{g,y} \| y^*(x^k) - \text{proj}_Y(\bar{A}^j x^k + \bar{b}^j) \| > \varepsilon_f, \\
\implies & L_{g,y} \| \text{proj}_Y(y^*(x^k)) - \text{proj}_Y(\bar{A}^j x^k + \bar{b}^j) \| > \varepsilon_f, \\
\implies & \| y^*(x^k) - (\bar{A}^j x^k + \bar{b}^j) \| > \frac{\varepsilon_f}{L_{g,y}},
\end{aligned}
$$

where $Y_d^G = \{(\bar{A}^1, \bar{b}^1), \ldots, (\bar{A}^{k-1}, \bar{b}^{k-1})\}$ denotes the generalized discretization at the start of iteration k, and the final step follows by the projection theorem (see Proposition 2.1.3 of Bertsekas [3]). Therefore, an upper bound on the number of iterations for Algorithm `G-OPT` to converge can be obtained by estimating the minimal number k of generalized discretization cuts required for $\sup_{x \in X} \min_{j \in [k]} \| y^*(x) - (\bar{A}^j x + \bar{b}^j) \| \leq \frac{\varepsilon_f}{L_{g,y}}$. The stated result then follows from Lemma 6.