How to solve a design centering problem

How to solve a design centering problem *

Stuart M. Harwood † Paul I. Barton ‡

May 4, 2017

Abstract

This work considers the problem of design centering. Geometrically, this can be thought of as inscribing one shape in another. Theoretical approaches and reformulations from the literature are reviewed; many of these are inspired by the literature on generalized semi-infinite programming, a generalization of design centering. However, the motivation for this work relates more to engineering applications of robust design. Consequently, the focus is on specific forms of design spaces (inscribed shapes) and the case when the constraints of the problem may be implicitly defined, such as by the solution of a system of differential equations. This causes issues for many existing approaches, and so this work proposes two restriction-based approaches for solving robust design problems that are applicable to engineering problems. Another feasible-point method from the literature is investigated as well. The details of the numerical implementations of all these methods are discussed. The discussion of these implementations in the particular setting of robust design in engineering problems is new.

keywords: gsip, design centering, lower level duality, global optimization

1 Introduction

This work discusses the theoretical and practical issues involved with solving design centering problems. The problems considered will be in the general form

$$\begin{aligned} &\sup_{\mathbf{x}} \ \mathrm{vol}(D(\mathbf{x})) \\ &\text{s.t. } D(\mathbf{x}) \subset G, \\ &\mathbf{x} \in X, \end{aligned} \tag{DC}$$

where $Y \subset \mathbb{R}^{n_y}$, $\mathbf{g} : Y \to \mathbb{R}^m$, $G = \{\mathbf{y} \in Y : \mathbf{g}(\mathbf{y}) \leq \mathbf{0}\}$, $X \subset \mathbb{R}^{n_x}$, D is a set-valued mapping from X to $\mathbb{R}^{n_y}$ (denoted $D : X \rightrightarrows \mathbb{R}^{n_y}$), and $\mathrm{vol}(\cdot)$ denotes the "volume" of a set (or some suitable proxy- in this work D will either be ball- or interval-valued, and the choice of volume/proxy will be clear). $D(\mathbf{x})$ is called a "candidate" design space, which is feasible if $D(\mathbf{x})$ is a subset of G, and optimal if it is the "largest" such feasible design space.

Ensuring feasibility of the solution is typically of paramount importance in any method for the solution of (DC). An application of (DC) is to *robust design* problems. In this case, $\mathbf{g}$ represents constraints on a system or process. Given some input parameters $\mathbf{y}$, one desires, for instance, on-specification product, or perhaps more importantly, safe system behavior, indicated by $\mathbf{g}(\mathbf{y}) \leq \mathbf{0}$.

*Current version as of: May 4, 2017. This research was funded by Novartis Pharmaceuticals as part of the Novartis-MIT Center for Continuous Manufacturing.

†Current affiliation: ExxonMobil Research and Engineering, Annandale, NJ 08801

‡Process Systems Engineering Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139 (pib@mit.edu)

In robust design, one seeks a nominal set point $\mathbf{y}_c$ at which to operate the system, and further determine the amount one can deviate from this set point (with respect to some norm) and still have safe process behavior. Then the result is that one seeks a set $D(\mathbf{y}_c, \delta) = \{\mathbf{y} : \|\mathbf{y} - \mathbf{y}_c\| \leq \delta\} \subset G$. One goal might be to maximize operational flexibility, in which case the largest $D(\mathbf{y}_c, \delta)$ is sought, i.e. $(\mathbf{y}_c, \delta)$ with the largest δ. This example provides some basic motivation for the focus of this work: The focus on the case that D is ball- or interval-valued comes from the fact that a solution should yield an explicit bound on the maximum acceptable deviation from some nominal set point. The focus on solution methods that are feasible point methods comes from the fact that a solution which violates $D(\mathbf{x}) \subset G$ is not acceptable, especially when safety is concerned.

Under some subtle assumptions (discussed in §2), problem (DC) is equivalent to a generalized semi-infinite program (GSIP) expressed as

$$\begin{aligned} &\sup_{\mathbf{x}} \operatorname{vol}(D(\mathbf{x})) \\ &\text{s.t. } g_i(\mathbf{y}) \leq 0, \quad \forall \mathbf{y} \in D(\mathbf{x}), \quad \forall i \in \{1, \ldots, m\}, \\ &\quad \mathbf{x} \in X. \end{aligned} \tag{GSIP}$$

We note that problem (GSIP) is a specific instance of the broader class of generalized semi-infinite programs, in which $\mathbf{g}$ is allowed dependence on the variables $\mathbf{x}$. Because design centering problems are a particular instance of GSIP, this work approaches design centering problems from the perspective of and with tools from the GSIP literature (see [59] for a recent review). This approach is hardly original [58, 61, 72], however, bringing together these ideas in one work is useful. Further, this work compares different numerical approaches from the GSIP literature. In particular, global optimization methods are considered; as a consequence, challenges and advantages appear that are not present when applying local optimization methods.

The end goal of this work is the case when the constraints of the system $\mathbf{g}$ are implicitly defined by the solution of systems of algebraic or differential equations, as is often the case for robust design in engineering applications. In this case, explicit expressions for $\mathbf{g}$ and its derivatives are, in general, difficult to obtain, and many methods for GSIP require this information in a numerical implementation. Consequently, the focus turns toward approximate solution methods inspired by global, feasible point methods. This discussion, and in particular the challenges in implementing the numerical methods, is original. Some of these approximations come from restrictions of (DC) which are apparent when considering the GSIP reformulation. Other approximations come from terminating a feasible point method early.

Connections to previous work in the literature are pointed out throughout this work. We mention a few references which do not quite fit into the rest of the organization of this work. Approaches to robust design and design centering specific to various application domains abound. In mechanical engineering, robust design of structures is considered in [46]. A design centering method is applied to a complex system such as a life-support system in [53]. A design centering applied to circuit design is considered in [4]. An approach in the specific case that $n_y = 3$ is considered in [45], with the classic application of gemstone cutting.

The rest of this work is organized as follows. Section 2 discusses some important concepts and the relationship between (DC) and (GSIP), and assumptions that will hold for the rest of this work. Some interesting cases of (DC) and connections to other problems including "flexibility indices" are also discussed. Section 3 discusses the case when $\mathbf{g}$ is an affine function. Reformulations as smooth, convex programs with polyhedral feasible sets are possible in this case. The main purpose of this section is to point out this special and tractable case. These reformulations are not necessarily apparent when approaching (DC) from the more general perspective taken in the GSIP literature, which is to use duality results for the lower level programs (see §2) to reformulate the infinite

constraints. This approach is the subject of Section 4; thus in this section the lower level programs are convex programs, or more generally, strong duality holds for the lower-level programs. A number of reformulations of (DC) to simpler problems (finite nonlinear programs (NLPs) or standard semi-infinite programs (SIPs)) are possible. The application of global optimization methods to these reformulations reveals some interesting behavior that is not apparent when applying local methods, as in previous work; see §4.2.2. A numerical example demonstrates that infeasible points can be found when applying a global method to previously published reformulations. Section 5 discusses the most general case, when the lower-level programs are not necessarily convex, and the subsequent need for the aforementioned approximate solution methods inspired by global, feasible point methods. The numerical developments in this section are new, and in particular the result that a significant amount of the implementation can rely on commercially available optimization software is interesting. Examples of robust design from engineering applications are considered. Section 6 concludes with some final thoughts.

2 Preliminaries

2.1 Notation

As already seen, vectors are denoted by lowercase bold letters, while matrices are denoted by uppercase bold letters. A vector of ones whose size is inferred from context is denoted $\mathbf{1}$. Similarly, a vector or matrix of zeros whose size is inferred from context is denoted $\mathbf{0}$. Sets are typically denoted by uppercase italic letters. A vector-valued function $\mathbf{f}$ is called convex if each component f_i is convex, and similarly for concavity. Denote the set of symmetric matrices in $\mathbb{R}^{n\times n}$ by $\mathbb{S}^{n\times n}$. For a symmetric matrix $\mathbf{M}$, the notation $\mathbf{M} \succeq \mathbf{0}$ ($\mathbf{M} \preceq \mathbf{0}$) means that $\mathbf{M}$ is positive (negative) semidefinite. Similarly, $\mathbf{M} \succ \mathbf{0}$ ($\mathbf{M} \prec \mathbf{0}$) means $\mathbf{M}$ is positive (negative) definite. For vectors, inequalities hold componentwise. Denote the determinant of a matrix $\mathbf{M}$ by $\det(\mathbf{M})$ and a square diagonal matrix with diagonal given by the vector $\mathbf{m}$ by $\mathrm{diag}(\mathbf{m})$. Denote the dual norm of a norm $\|\cdot\|$ by $\|\cdot\|_*$.

2.2 Equivalence of (DC) and (GSIP)

Throughout this work, the terms "equivalent" and "equivalence" are used to relate two mathematical programs in the standard way.

Definition 1. *(Equivalence) Two mathematical programs*

$$\begin{array}{ll} \sup\limits_{\mathbf{x}} & f(\mathbf{x}) \\ \text{s.t.} & \mathbf{x} \in X, \end{array} \qquad \begin{array}{ll} \sup\limits_{\mathbf{x},\mathbf{z}} & f(\mathbf{x}) \\ \text{s.t.} & (\mathbf{x},\mathbf{z}) \in S, \end{array}$$

are said to be equivalent if for each $\mathbf{x} \in X$, there exists $\mathbf{z}$ such that $(\mathbf{x},\mathbf{z}) \in S$, and for each $(\mathbf{x},\mathbf{z}) \in S$, $\mathbf{x} \in X$.

It is clear that if two programs are equivalent, then the solution sets (if nonempty) have the same "$\mathbf{x}$" components since the objective functions are the same.

Next, we state an assumption under which it is shown that (DC) and (GSIP) are equivalent. Nearly all of the results in this work include hypotheses which imply this assumption.

Assumption 1. *Assume that in* (DC), *$D(\mathbf{x})$ is a subset of Y for all $\mathbf{x} \in X$.*

Consider the *lower level programs* (LLPs) of (GSIP), for $\mathbf{x} \in X$ and $i \in \{1, \ldots, m\}$:

$$g_i^*(\mathbf{x}) = \sup \{g_i(\mathbf{y}) : \mathbf{y} \in D(\mathbf{x})\}. \qquad \text{(LLP } i\text{)}$$

In the context of robust design, $\mathbf{y}$ represents parameters or inputs to a system. In the context of GSIP, $\mathbf{y}$ are called the lower (level) variables, while $\mathbf{x}$ are called the upper variables. For $\mathbf{x} \in X$, it is clear that if $D(\mathbf{x})$ is nonempty, then $\mathbf{x}$ is feasible in (GSIP) iff $g_i^*(\mathbf{x}) \leq 0$ for each i. On the other hand, if $D(\mathbf{x})$ is empty, then no constraints are required to hold in (GSIP), and so $\mathbf{x}$ is feasible (alternatively one could define the supremum of a real function on the empty set as $-\infty$).

Consider the fact that G is defined as a subset of Y. If Assumption 1 did not hold, the constraints in (GSIP) would need to be modified to read

$$g_i(\mathbf{y}) \leq 0, \quad \forall \mathbf{y} \in D(\mathbf{x}) \cap Y.$$

However, this leads to complications. If, for instance, $D(\mathbf{x})$ is nonempty, but $D(\mathbf{x}) \cap Y$ is empty, then neither $D(\mathbf{x})$ nor $D(\mathbf{x}) \cap Y$ are acceptable solutions to (DC).

Understanding this, the equivalence of (DC) and (GSIP), established in the following result, is intuitive.

Proposition 1. *Under Assumption 1, problems* (DC) *and* (GSIP) *are equivalent.*

Proof. Since the objective functions in (DC) and (GSIP) are the same, we just need to establish that their feasible sets are the same. So consider $\mathbf{x}$ feasible in (DC). Then $\mathbf{x} \in X$ and $D(\mathbf{x}) \subset G$. This means that for all $\mathbf{y} \in D(\mathbf{x})$, $\mathbf{y} \in Y$ and $\mathbf{g}(\mathbf{y}) \leq \mathbf{0}$. We immediately have that $\mathbf{x}$ is feasible in (GSIP).

Conversely, choose $\mathbf{x}$ feasible in (GSIP). Again, $\mathbf{x} \in X$ and for all $\mathbf{y} \in D(\mathbf{x})$, $\mathbf{g}(\mathbf{y}) \leq \mathbf{0}$. Under Assumption 1, $D(\mathbf{x})$ is a subset of Y, and so for all $\mathbf{y} \in D(\mathbf{x})$, we have $\mathbf{y} \in G$. Thus $D(\mathbf{x}) \subset G$ and $\mathbf{x}$ is feasible in (DC). □

2.3 Related problems

A related problem is that of calculating a "feasibility index" for process design under uncertainty. This idea goes back to [27, 66, 67], and has been addressed more recently in [20, 63, 64]. This problem can be written equivalently as an SIP in the forms

$$\begin{array}{lcl} \inf\limits_{\mathbf{x}} f(\mathbf{x}) & \iff & \inf\limits_{\mathbf{x}} f(\mathbf{x}) \\ \text{s.t. } \widetilde{g}(\mathbf{x}, \mathbf{y}) \leq 0, \quad \forall \mathbf{y} \in \widetilde{Y}, & & \text{s.t. } 0 \geq \sup\{\widetilde{g}(\mathbf{x}, \mathbf{y}) : \mathbf{y} \in \widetilde{Y}\}. \end{array} \qquad (1)$$

One interpretation of this problem is that $\mathbf{x}$ represents some process design decisions, while $\mathbf{y}$ is a vector of uncertain model parameters. The goal is to minimize some economic objective f of the design variables which guarantees safe process design for any realization of the uncertain parameter $\mathbf{y}$ (indicated by $\widetilde{g}(\mathbf{x}, \mathbf{y}) \leq 0$, for any $\mathbf{y} \in \widetilde{Y}$).

The definition of the "flexibility index" in Equation 8 of [66] is a kind of GSIP. The rest of that work focuses on conditions that allow this definition to be reformulated as an SIP of the form (1). The results in [66] focus on the case when D is interval-valued. A similar argument is repeated below, which depends on having a design space which is the image of the unit ball under an affine mapping. In this case the proxy for volume is taken to be the determinant of the matrix in the affine transformation.

Proposition 2. *Suppose* $X \subset Y \times \mathbb{R}^{n_y \times n_y}$, $D : (\mathbf{y}_c, \mathbf{P}) \mapsto \{\mathbf{y}_c + \mathbf{P}\mathbf{y}_d : \|\mathbf{y}_d\| \leq 1\}$ *for some norm* $\|\cdot\|$ *on* $\mathbb{R}^{n_y}$, $D(\mathbf{y}_c, \mathbf{P}) \subset Y$ *for all* $(\mathbf{y}_c, \mathbf{P}) \in X$, *and* $\mathrm{vol}(D(\mathbf{y}_c, \mathbf{P})) = \det(\mathbf{P})$. *Then* (GSIP) *is equivalent to the SIP*

$$\begin{aligned} &\sup_{\mathbf{y}_c, \mathbf{P}} \ \det(\mathbf{P}) \\ &\text{s.t. } \mathbf{g}(\mathbf{y}_c + \mathbf{P}\mathbf{y}_d) \leq \mathbf{0}, \quad \forall \mathbf{y}_d \in B_1 \equiv \{\mathbf{y} : \|\mathbf{y}\| \leq 1\}, \\ &\quad (\mathbf{y}_c, \mathbf{P}) \in X. \end{aligned} \tag{2}$$

Proof. The reformulation is immediate given the form of D. But to be explicit, consider the problem for given $(\mathbf{y}_c, \mathbf{P}) \in X$

$$g_i^{**}(\mathbf{y}_c, \mathbf{P}) = \sup\{g_i(\mathbf{y}_c + \mathbf{P}\mathbf{y}_d) : \mathbf{y}_d \in B_1\}.$$

For $\mathbf{y}$ feasible in (LLP i), by definition there exists $\mathbf{y}_d \in B_1$ such that $\mathbf{y} = \mathbf{y}_c + \mathbf{P}\mathbf{y}_d$. Thus $g_i^*(\mathbf{y}_c, \mathbf{P}) \leq g_i^{**}(\mathbf{y}_c, \mathbf{P})$. Conversely, for $\mathbf{y}_d \in B_1$, there exists $\mathbf{y} \in D(\mathbf{y}_c, \mathbf{P})$ with $\mathbf{y} = \mathbf{y}_c + \mathbf{P}\mathbf{y}_d$. Thus $g_i^{**}(\mathbf{y}_c, \mathbf{P}) \leq g_i^*(\mathbf{y}_c, \mathbf{P})$, and together the inequalities imply that the feasible sets of (GSIP) and (2) are the same, and so equivalence follows. □

In robust design applications, an extra step is required to make use of this form of D. To check that an operating condition or process parameters $\mathbf{y}$ are in the calculated design space requires checking that the norm of the solution $\mathbf{y}_d$ of $\mathbf{y} - \mathbf{y}_c = \mathbf{P}\mathbf{y}_d$ is less than one. Consequently, the LU factorization of the optimal $\mathbf{P}$ should be computed to minimize this computation, especially if it is to be performed online. Further, SIP (2) is still a somewhat abstract problem and assuming more structure leads to more tractable restrictions such as

$$\begin{aligned} &\sup_{\mathbf{y}_c, \mathbf{d}} \ \sum_{i=1}^{n_y} \ln(d_i) \\ &\text{s.t. } \mathbf{g}(\mathbf{y}_c + \mathrm{diag}(\mathbf{d})\mathbf{y}_d) \leq \mathbf{0}, \quad \forall \mathbf{y}_d : \|\mathbf{y}_d\| \leq 1, \\ &\quad (\mathbf{y}_c, \mathbf{d}) \in X \subset \{(\mathbf{y}_c, \mathbf{d}) \in Y \times \mathbb{R}^{n_y} : \mathbf{d} > \mathbf{0}\}, \end{aligned} \tag{3}$$

where in effect the variable $\mathbf{P}$ in SIP (2) has been restricted to (a subset of) the space of diagonal positive-definite matrices (see also the proof of Corollary 5 for justification of the use of the logarithm of the objective). Of course (3) is still a semi-infinite problem, but under further assumptions on $\mathbf{g}$ and the norm used, finite convex reformulations are possible (see §3.2).

3 Affine constraints

This section deals with the case that $\mathbf{g}$ is an affine function and $Y = \mathbb{R}^{n_y}$. Specifically, assume that for each $i \in \{1, \ldots, m\}$,

$$g_i(\mathbf{y}) = \mathbf{c}_i^{\mathrm{T}}\mathbf{y} - b_i,$$

for some $\mathbf{c}_i \in \mathbb{R}^{n_y}$ and $b_i \in \mathbb{R}$. Consequently, G is a (convex) polyhedron. Reformulations for different forms of D are given; in each case the reformulation is a convex program.

In §4, reformulations of (GSIP) are presented which rely on strong duality holding for each (LLP i). Consequently, the reformulations in §4 will be applicable to the current situation with $\mathbf{g}$ affine and D convex-valued. However, in the best case those reformulations involve nonconvex NLPs, which do not reduce to convex programs under the assumptions of the present section. Thus, it is worthwhile to be aware of the special reformulations in the present section.

3.1 Ball-valued design space

Let the upper variables $\mathbf{x}$ of (DC) be $(\mathbf{y}_c, \delta) \in \mathbb{R}^{n_y} \times \mathbb{R}$, and let $D(\mathbf{x})$ be the closed δ-ball around $\mathbf{y}_c$ for some norm $\|\cdot\|$: $D : (\mathbf{y}_c, \delta) \mapsto \{\mathbf{y} : \|\mathbf{y} - \mathbf{y}_c\| \leq \delta\}$. Then problem (DC) becomes

$$\begin{aligned} &\sup_{\mathbf{y}_c, \delta} \ \delta \\ &\text{s.t. } \mathbf{c}_i^{\mathrm{T}} \mathbf{y} - b_i \leq 0, \quad \forall \mathbf{y} : \|\mathbf{y} - \mathbf{y}_c\| \leq \delta, \quad \forall i \in \{1, \ldots, m\}, \\ &\quad \mathbf{y}_c \in \mathbb{R}^{n_y}, \delta \geq 0. \end{aligned} \tag{4}$$

Assumption 1 holds, since $D(\mathbf{y}_c, \delta)$ must trivially be a subset of $Y = \mathbb{R}^{n_y}$.

Problem (4) can be reformulated as a linear program (LP), following the ideas in §8.5 of [16]. Problem (4) is related to the problem of Chebyshev centering. For specific norms, this reformulation also appears in [29].

Theorem 3. *Problem* (4) *is equivalent to the LP*

$$\begin{aligned} &\sup_{\mathbf{y}_c, \delta} \ \delta \\ &\text{s.t. } \mathbf{c}_i^{\mathrm{T}} \mathbf{y}_c + (\|\mathbf{c}_i\|_*)\delta \leq b_i \quad \forall i \in \{1, \ldots, m\}, \\ &\quad \mathbf{y}_c \in \mathbb{R}^{n_y}, \delta \geq 0. \end{aligned} \tag{5}$$

Proof. See [16, §8.5.1]. □

3.2 General ellipsoidal design space

A convex reformulation is also possible when the design space is an ellipsoid and its "shape" is a decision variable. This is a special case of the reformulation in Proposition 2. In this case, let $X \subset \{(\mathbf{y}_c, \mathbf{P}) \in \mathbb{R}^{n_y} \times \mathbb{S}^{n_y \times n_y} : \mathbf{P} \succ \mathbf{0}\}$ and $D : (\mathbf{y}_c, \mathbf{P}) \mapsto \{\mathbf{y}_c + \mathbf{P}\mathbf{y}_d : \|\mathbf{y}_d\|_2 \leq 1\}$, i.e., the design space is the image of the unit two-norm ball under an affine transformation, and thus an ellipsoid. Let $\mathrm{vol}(D(\cdot)) : (\mathbf{y}_c, \mathbf{P}) \mapsto \det(\mathbf{P})$ and $Y = \mathbb{R}^{n_y}$. Problem (DC) becomes

$$\begin{aligned} &\sup_{\mathbf{y}_c, \mathbf{P}} \ \det(\mathbf{P}) \\ &\text{s.t. } \mathbf{c}_i^{\mathrm{T}}(\mathbf{y}_c + \mathbf{P}\mathbf{y}_d) - b_i \leq 0, \quad \forall \mathbf{y}_d : \|\mathbf{y}_d\|_2 \leq 1, \quad \forall i \in \{1, \ldots, m\}, \\ &\quad (\mathbf{y}_c, \mathbf{P}) \in X. \end{aligned}$$

Similarly to the result in Theorem 3 (and taking the logarithm of the objective), this becomes

$$\begin{aligned} &\sup_{\mathbf{y}_c, \mathbf{P}} \ \ln(\det(\mathbf{P})) \\ &\text{s.t. } \mathbf{c}_i^{\mathrm{T}} \mathbf{y}_c + \left\|\mathbf{c}_i^{\mathrm{T}} \mathbf{P}\right\|_2 - b_i \leq 0, \quad \forall i \in \{1, \ldots, m\}, \\ &\quad (\mathbf{y}_c, \mathbf{P}) \in X. \end{aligned} \tag{6}$$

Problem (6) is in fact a convex program and enjoys a rich history of analysis; see [16, §8.4.2], [44, Sections 6.4.4 and 6.5], [33]. However, further reformulation to a "standard" form (such as a semidefinite program) is necessary in order to apply general-purpose software for cone programs such as YALMIP [3, 36] and CVX [24, 23] (both of which provide front-ends for the solvers SeDuMi [65, 2] and MOSEK [1]). This is possible by the arguments in [44, §6.4.4] or [10, §4.2].

3.3 Interval-valued design space

In the case of the infinity-norm, one can generalize the form of the design space a little more, and still obtain a fairly tractable formulation. In this case, let the upper variables $\mathbf{x}$ of (DC) be $(\mathbf{y}^L, \mathbf{y}^U) \in \mathbb{R}^{n_y} \times \mathbb{R}^{n_y}$, and let $D(\mathbf{y}^L, \mathbf{y}^U)$ be a nonempty compact interval: $D : (\mathbf{y}^L, \mathbf{y}^U) \mapsto [\mathbf{y}^L, \mathbf{y}^U]$. Again with affine $\mathbf{g}$, problem (DC) becomes

$$\begin{aligned} \sup_{\mathbf{y}^L, \mathbf{y}^U} \ & \prod_j (y_j^U - y_j^L) \\ \text{s.t. } & \mathbf{c}_i^{\mathrm{T}} \mathbf{y} - b_i \leq 0, \quad \forall \mathbf{y} \in [\mathbf{y}^L, \mathbf{y}^U], \quad \forall i \in \{1, \ldots, m\}, \\ & \mathbf{y}^L \leq \mathbf{y}^U, \\ & (\mathbf{y}^L, \mathbf{y}^U) \in \mathbb{R}^{n_y} \times \mathbb{R}^{n_y}. \end{aligned} \tag{7}$$

The constraints $\mathbf{y}^L \leq \mathbf{y}^U$ ensure that $D(\mathbf{y}^L, \mathbf{y}^U)$ is nonempty.

The reformulation of this problem has been considered in [7, 57].

Theorem 4. *Consider the linearly-constrained NLP*

$$\begin{aligned} \sup_{\mathbf{y}^L, \mathbf{y}^U} \ & \prod_j (y_j^U - y_j^L) \\ \text{s.t. } & \mathbf{c}_i^{\mathrm{T}} \mathbf{M}_i^L \mathbf{y}^L + \mathbf{c}_i^{\mathrm{T}} \mathbf{M}_i^U \mathbf{y}^U \leq b_i, \quad \forall i, \\ & \mathbf{y}^L \leq \mathbf{y}^U, \\ & (\mathbf{y}^L, \mathbf{y}^U) \in \mathbb{R}^{n_y} \times \mathbb{R}^{n_y}, \end{aligned} \tag{8}$$

where $\mathbf{M}_i^L$ and $\mathbf{M}_j^U$ are diagonal n_y by n_y matrices where the j^{th} element of the diagonals, $m_{i,j}^L$ and $m_{i,j}^U$, respectively, are given by

$$m_{i,j}^L = \begin{cases} 1 & c_{i,j} < 0, \\ 0 & c_{i,j} \geq 0, \end{cases} \quad \textit{and} \quad m_{i,j}^U = \begin{cases} 0 & c_{i,j} < 0, \\ 1 & c_{i,j} \geq 0. \end{cases}$$

Problem (8) *is equivalent to problem* (7).

Proof. Begin by analyzing the lower-level programs of (7): $g_i^*(\mathbf{y}^L, \mathbf{y}^U) = \sup\{\mathbf{c}_i^{\mathrm{T}} \mathbf{y} : \mathbf{y} \in [\mathbf{y}^L, \mathbf{y}^U]\} - b_i$. Again, if $(\mathbf{y}^L, \mathbf{y}^U)$ is feasible in (7), then $g_i^*(\mathbf{y}^L, \mathbf{y}^U) \leq 0$. Further, the lower-level programs are linear programs with box constraints, and consequently can be solved by inspection: an optimal solution $\mathbf{y}^i$ of the i^{th} lower-level program can be constructed by letting $y_j^i = y_j^U$ if $c_{i,j} \geq 0$ and $y_j^i = y_j^L$ otherwise (where $c_{i,j}$ denotes the j^{th} component of $\mathbf{c}_i$).

The constraint in (7) that $\mathbf{c}_i^{\mathrm{T}} \mathbf{y} - b_i \leq 0$, for all $\mathbf{y} \in [\mathbf{y}^L, \mathbf{y}^U]$, is equivalent (when $\mathbf{y}^L \leq \mathbf{y}^U$) to $\sup\{\mathbf{c}_i^{\mathrm{T}} \mathbf{y} : \mathbf{y} \in [\mathbf{y}^L, \mathbf{y}^U]\} \leq b_i$, which by the previous discussion is equivalent to $\mathbf{c}_i^{\mathrm{T}} \mathbf{M}_i^L \mathbf{y}^L + \mathbf{c}_i^{\mathrm{T}} \mathbf{M}_i^U \mathbf{y}^U \leq b_i$, which are the constraints in (8). Finally, both formulations include the constraints $\mathbf{y}^L \leq \mathbf{y}^U$, thus the feasible sets of Problems (8) and (7) are equivalent. Since the objective functions are the same, the problems are equivalent. □

A more numerically favorable restriction of (8) is possible, at the expense of the restriction potentially being infeasible if G is "thin."

Corollary 5. *Consider the convex program*

$$\begin{aligned} \sup_{\mathbf{y}^L,\mathbf{y}^U} \ & \sum_j \ln(y_j^U - y_j^L) \\ \text{s.t. } & \mathbf{c}_i^{\mathrm{T}}\mathbf{M}_i^L\mathbf{y}^L + \mathbf{c}_i^{\mathrm{T}}\mathbf{M}_i^U\mathbf{y}^U \le b_i, \quad \forall i, \\ & \mathbf{y}^U - \mathbf{y}^L \ge \epsilon\mathbf{1}, \\ & (\mathbf{y}^L, \mathbf{y}^U) \in \mathbb{R}^{n_y} \times \mathbb{R}^{n_y}, \end{aligned} \tag{9}$$

where $\epsilon > 0$, and $\mathbf{M}_i^L$ and $\mathbf{M}_j^U$ are defined as in Theorem 4. If an optimal solution $(\mathbf{y}^{L,*}, \mathbf{y}^{U,*})$ of problem (7) satisfies $\mathbf{y}^{U,*} - \mathbf{y}^{L,*} \ge \epsilon\mathbf{1}$, then this is also a solution of problem (9).

Proof. Since $\ln(\cdot)$ is a nondecreasing concave function and for each j, $(y_j^L, y_j^U) \mapsto (y_j^U - y_j^L)$ is a concave function, then the objective function $\sum_j \ln(y_j^U - y_j^L)$ is concave on the convex feasible set, and so this maximization problem is indeed a convex program.

Denote the feasible set of problem (9) by X_R. Denote the objective function of (7) by $f(\mathbf{y}^L, \mathbf{y}^U) = \prod_j (y_j^U - y_j^L)$. Note that f is positive on X_R. If an optimal solution $(\mathbf{y}^{L,*}, \mathbf{y}^{U,*})$ of problem (7) satisfies $\mathbf{y}^{U,*} - \mathbf{y}^{L,*} \ge \epsilon\mathbf{1}$, then clearly $(\mathbf{y}^{L,*}, \mathbf{y}^{U,*}) \in X_R$. Since $\ln(\cdot)$ is increasing, $\arg\max\{f(\mathbf{y}^L, \mathbf{y}^U) : (\mathbf{y}^L, \mathbf{y}^U) \in X_R\} = \arg\max\{\ln(f(\mathbf{y}^L, \mathbf{y}^U)) : (\mathbf{y}^L, \mathbf{y}^U) \in X_R\}$, and so $(\mathbf{y}^{L,*}, \mathbf{y}^{U,*})$ is a solution of problem (9). □

4 Convex LLP

For the most part, in this section it is assumed that the LLPs are convex programs (and so $\mathbf{g}$ is a *concave* function), but to be more accurate, the main focus of this section is when duality results hold for each LLP. When this is the case, a number of reformulations provide a way to solve (GSIP) via methods for NLPs or SIPs. In §4.1.4, the LLPs have a specific form and the concavity of $\mathbf{g}$ is not necessary. Reformulation results from the literature are reviewed and numerical examples considered.

Before continuing, we mention a connection to another field. As mentioned, the main focus of this section is when duality results hold for each LLP. A similar approach is taken in the study of the "robust" formulation of mathematical programs with uncertain data [8, 9]. In that work, the robust mathematical program is typically a semi-infinite program. These SIPs are reduced to finite NLPs through similar techniques that are employed in this section.

4.1 Reformulation

4.1.1 KKT conditions

In the literature, one of the first reformulations of (GSIP) when the LLPs are convex programs comes from replacing the LLPs with algebraic constraints which are necessary and sufficient for a maximum; i.e., their KKT conditions. This approach can be found in [60, 61], for instance. The following result establishes the equivalence of (GSIP) with a mathematical program with complementarity constraints (MPCC), a type of NLP. Refer to MPCC (10) as the "KKT reformulation."

Proposition 6. *Suppose Y is a nonempty, open, convex set. Suppose $D(\mathbf{x})$ is compact for each $\mathbf{x} \in X$ and $D(\mathbf{x}) = \{\mathbf{y} \in Y : \mathbf{h}(\mathbf{x}, \mathbf{y}) \le \mathbf{0}\}$ for some $\mathbf{h} : X \times Y \to \mathbb{R}^{n_h}$ where $\mathbf{h}(\mathbf{x}, \cdot)$ is convex and differentiable for each $\mathbf{x} \in X$. Suppose that for each $i \in \{1, \ldots, m\}$ and each $\mathbf{x} \in X$ the Slater*

condition holds for (LLP i)*: there exists a* $\mathbf{y}_s \in Y$ *such that* $\mathbf{h}(\mathbf{x}, \mathbf{y}_s) < \mathbf{0}$*. Suppose that* $\mathbf{g}$ *is concave and differentiable. Then* (GSIP) *is equivalent to the MPCC*

$$\begin{aligned} \sup_{\mathbf{x},\mathbf{y}^1,\boldsymbol{\mu}^1,\ldots,\mathbf{y}^m,\boldsymbol{\mu}^m} \quad & \operatorname{vol}(D(\mathbf{x})) \\ \text{s.t. } & g_i(\mathbf{y}^i) \le 0, \quad \forall i, \\ & \mathbf{h}(\mathbf{x}, \mathbf{y}^i) \le \mathbf{0}, \quad \forall i, \\ & \nabla g_i(\mathbf{y}^i) - \nabla_{\mathbf{y}} \mathbf{h}(\mathbf{x}, \mathbf{y}^i)\boldsymbol{\mu}^i = \mathbf{0}, \quad \forall i, \\ & \boldsymbol{\mu}^i \ge \mathbf{0}, \quad \mathbf{y}^i \in Y, \quad \forall i, \\ & \mu_j^i h_j(\mathbf{x}, \mathbf{y}^i) = 0, \quad \forall (i,j), \\ & \mathbf{x} \in X. \end{aligned} \tag{10}$$

Proof. See [60, §3]. □

Note that the assumptions imply that $D(\mathbf{x})$ is nonempty and a subset of Y for all $\mathbf{x} \in X$; thus Assumption 1 is still satisfied. For each $i \in \{1, \ldots, m\}$, the hypotheses imply that (LLP i) is a differentiable convex program (satisfying a constraint qualification) which achieves its maximum; thus there exists a $\boldsymbol{\mu}^i \in \mathbb{R}^{n_h}$ such that $(\mathbf{y}^i, \boldsymbol{\mu}^i)$ is a KKT point iff $\mathbf{y}^i$ is a global optimum of (LLP i). It can be shown that the result in Proposition 6 holds under weaker conditions than the Slater condition for the LLPs, as in [60]. However, the Slater condition has a natural interpretation in design centering problems; a design space must have some minimum size or afford some minimum amount of operational flexibility. The Slater condition is common to many reformulations in this work. Indeed, a Slater-like condition has already been used in Corollary 5 and will be used throughout the rest of §4.

The constraints $\mu_j^i h_j(\mathbf{x}, \mathbf{y}^i) = 0$, $\mu_j^i \ge 0$, and $h_j(\mathbf{x}, \mathbf{y}^i) \le 0$ in NLP (10) are the complementarity constraints which give the class of MPCC its name. Unfortunately, there are numerical difficulties involved in solving MPCCs. This relates to the fact that the Mangasarian-Fromovitz Constraint Qualification is violated everywhere in its feasible set [18]. This motivates the reformulation of §4.1.3.

4.1.2 Lagrangian dual

It is helpful to analyze a reformulation of (GSIP) based on Lagrangian duality at this point. Assume $D(\mathbf{x}) = \{\mathbf{y} \in Y : \mathbf{h}(\mathbf{x}, \mathbf{y}) \le \mathbf{0}\}$ for some $\mathbf{h} : X \times Y \to \mathbb{R}^{n_h}$. Define the dual function $q_i(\mathbf{x}, \cdot)$ and its effective domain by

$$\begin{aligned} q_i(\mathbf{x}, \boldsymbol{\mu}) &= \sup\{g_i(\mathbf{y}) - \boldsymbol{\mu}^{\mathrm{T}} \mathbf{h}(\mathbf{x}, \mathbf{y}) : \mathbf{y} \in Y\}, \\ \operatorname{dom}(q_i(\mathbf{x}, \cdot)) &= \{\boldsymbol{\mu} \in \mathbb{R}^{n_h} : q_i(\mathbf{x}, \boldsymbol{\mu}) < +\infty\}. \end{aligned}$$

Then define the (Lagrangian) dual problem of (LLP i) by

$$q_i^*(\mathbf{x}) = \inf\left\{q_i(\mathbf{x}, \boldsymbol{\mu}) : \boldsymbol{\mu} \ge \mathbf{0}, \boldsymbol{\mu} \in \operatorname{dom}(q_i(\mathbf{x}, \cdot))\right\}. \tag{11}$$

Under appropriate assumptions, one can establish that $g_i^*(\mathbf{x}) = q_i^*(\mathbf{x})$ (known as strong duality) for each $\mathbf{x}$. This forms the basis for the following results. The first result establishes that one can always obtain an SIP restriction of (GSIP). The set M is included to facilitate certain numerical developments; see §5.1.2.

Proposition 7. *(Proposition 3.1 in [28]) Suppose* $D(\mathbf{x}) = \{\mathbf{y} \in Y : \mathbf{h}(\mathbf{x}, \mathbf{y}) \leq \mathbf{0}\}$ *for some* $\mathbf{h} : X \times Y \to \mathbb{R}^{n_h}$*. For any* $M \subset \mathbb{R}^{n_h}$ *and for any* $(\mathbf{x}, \boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^m)$ *feasible in the SIP*

$$\begin{aligned} \sup_{\mathbf{x}, \boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^m} \quad & \operatorname{vol}(D(\mathbf{x})) \\ \text{s.t. } & g_i(\mathbf{y}) - (\boldsymbol{\mu}^i)^{\mathrm{T}} \mathbf{h}(\mathbf{x}, \mathbf{y}) \leq 0, \quad \forall \mathbf{y} \in Y, \quad \forall i, \\ & \boldsymbol{\mu}^i \geq \mathbf{0}, \quad \boldsymbol{\mu}^i \in M, \quad \forall i, \\ & \mathbf{x} \in X, \end{aligned} \tag{12}$$

$\mathbf{x}$ *is feasible in* (GSIP)*.*

The next result establishes that SIP (12) is equivalent to (GSIP) under hypotheses similar to those in Proposition 6.

Proposition 8. *Suppose* Y *is convex. Suppose* $D(\mathbf{x}) = \{\mathbf{y} \in Y : \mathbf{h}(\mathbf{x}, \mathbf{y}) \leq \mathbf{0}\}$ *for some* $\mathbf{h} : X \times Y \to \mathbb{R}^{n_h}$*. For all* $\mathbf{x} \in X$*, suppose* $\mathbf{g}$ *is concave,* $\mathbf{h}(\mathbf{x}, \cdot)$ *is convex,* $g_i^*(\mathbf{x})$ *(defined by* (LLP i)*) is finite for all* i*, and there exists a* $\mathbf{y}_s(\mathbf{x}) \in Y$ *such that* $\mathbf{g}(\mathbf{y}_s(\mathbf{x})) > -\mathbf{g}_b$ *for some* $\mathbf{g}_b > \mathbf{0}$ *and* $\mathbf{h}(\mathbf{x}, \mathbf{y}_s(\mathbf{x})) \leq -\mathbf{h}_b$ *for some* $\mathbf{h}_b > \mathbf{0}$*. Then for compact* $M = [\mathbf{0}, \mathbf{b}^*] \subset \mathbb{R}^{n_h}$ *(where* $b_j^* = \max_i\{g_{b,i}\}/h_{b.j}$*),* (GSIP) *is equivalent to SIP* (12)*.*

Proof. Follows from Lemmata 3.3 and 3.4 and Theorem 3.2 in [28]. □

4.1.3 Wolfe dual

To obtain a more numerically tractable reformulation than the KKT reformulation (10), one follows the ideas in [18] to obtain a reformulation of (GSIP) which does not have complementarity constraints. This follows by looking at the dual function $q_i(\mathbf{x}, \cdot)$ and noting that if Y is a nonempty open convex set and g_i and $-\mathbf{h}(\mathbf{x}, \cdot)$ are concave and differentiable, then for $\boldsymbol{\mu} \geq \mathbf{0}$, the supremum defining the dual function is achieved at $\mathbf{y}$ if and only if $\nabla g_i(\mathbf{y}) - \nabla_{\mathbf{y}} \mathbf{h}(\mathbf{x}, \mathbf{y})\boldsymbol{\mu} = \mathbf{0}$. Consequently, one obtains the Wolfe dual problem of (LLP i):

$$q_i^W(\mathbf{x}) = \inf \left\{ g_i(\mathbf{y}) - \boldsymbol{\mu}^{\mathrm{T}} \mathbf{h}(\mathbf{x}, \mathbf{y}) : \mathbf{y} \in Y, \boldsymbol{\mu} \geq \mathbf{0}, \nabla g_i(\mathbf{y}) - \nabla_{\mathbf{y}} \mathbf{h}(\mathbf{x}, \mathbf{y})\boldsymbol{\mu} = \mathbf{0} \right\}. \tag{13}$$

Under suitable assumptions (namely, that (LLP i) achieves its supremum and a Slater condition), strong duality holds. An alternate proof follows, based on (much better established) Lagrangian duality results. See also Section 6.3 in [22].

Lemma 9. *Suppose* Y *is a nonempty open convex set. For a given* $\mathbf{x} \in X$ *and* $i \in \{1, \ldots, m\}$*, suppose the following:* $D(\mathbf{x})$ *is compact and* $D(\mathbf{x}) = \{\mathbf{y} \in Y : \mathbf{h}(\mathbf{x}, \mathbf{y}) \leq \mathbf{0}\}$ *for some* $\mathbf{h}(\mathbf{x}, \cdot) : Y \to \mathbb{R}^{n_h}$ *which is convex and differentiable. Suppose that the Slater condition holds for* (LLP i) *(there exists a* $\mathbf{y}_s \in Y$ *such that* $\mathbf{h}(\mathbf{x}, \mathbf{y}_s) < \mathbf{0}$*). Suppose* g_i *is concave and differentiable. Then there exists* $(\mathbf{y}^i, \boldsymbol{\mu}^i)$ *satisfying* $\boldsymbol{\mu}^i \geq \mathbf{0}$*,* $\mathbf{y}^i \in Y$*,* $\nabla g_i(\mathbf{y}^i) - \nabla_{\mathbf{y}} \mathbf{h}(\mathbf{x}, \mathbf{y}^i)\boldsymbol{\mu}^i = \mathbf{0}$*, and* $g_i^*(\mathbf{x}) = g_i(\mathbf{y}^i) - (\boldsymbol{\mu}^i)^{\mathrm{T}} \mathbf{h}(\mathbf{x}, \mathbf{y}^i)$*. Further,* $q_i^W(\mathbf{x}) = g_i^*(\mathbf{x})$*.*

Proof. First, it is established that the Wolfe dual is weaker than the Lagrangian dual (11) (i.e. $q_i^W(\mathbf{x}) \geq q_i^*(\mathbf{x})$, thus establishing weak duality between (LLP i) and the Wolfe dual). As before, since Y is a nonempty open convex set and g_i and $-\mathbf{h}$ are concave and differentiable, then for $\widetilde{\boldsymbol{\mu}} \geq \mathbf{0}$, $\sup\{g_i(\mathbf{y}) - \widetilde{\boldsymbol{\mu}}^{\mathrm{T}} \mathbf{h}(\mathbf{x}, \mathbf{y}) : \mathbf{y} \in Y\}$ is achieved at $\widetilde{\mathbf{y}} \in Y$ if and only if $\nabla g_i(\widetilde{\mathbf{y}}) - \nabla_{\mathbf{y}} \mathbf{h}(\mathbf{x}, \widetilde{\mathbf{y}})\widetilde{\boldsymbol{\mu}} = \mathbf{0}$. Let

$$F_W = \{(\widetilde{\mathbf{y}}, \widetilde{\boldsymbol{\mu}}) : \widetilde{\boldsymbol{\mu}} \geq \mathbf{0}, \widetilde{\mathbf{y}} \in Y, \nabla g_i(\widetilde{\mathbf{y}}) - \nabla_{\mathbf{y}} \mathbf{h}(\mathbf{x}, \widetilde{\mathbf{y}})\widetilde{\boldsymbol{\mu}} = \mathbf{0}\}.$$

Thus, for all $(\widetilde{\mathbf{y}}, \widetilde{\boldsymbol{\mu}}) \in F_W$, we have

$$\sup\{g_i(\mathbf{y}) - \widetilde{\boldsymbol{\mu}}^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \mathbf{y}) : \mathbf{y} \in Y\} = g_i(\widetilde{\mathbf{y}}) - \widetilde{\boldsymbol{\mu}}^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \widetilde{\mathbf{y}}),$$

which also implies that $\widetilde{\boldsymbol{\mu}} \in \mathrm{dom}(q_i(\mathbf{x}, \cdot))$. It follows that for all $(\widetilde{\mathbf{y}}, \widetilde{\boldsymbol{\mu}}) \in F_W$, we have $\widetilde{\boldsymbol{\mu}} \geq \mathbf{0}$, $\widetilde{\boldsymbol{\mu}} \in \mathrm{dom}(q_i(\mathbf{x}, \cdot))$, and $q_i(\mathbf{x}, \widetilde{\boldsymbol{\mu}}) = g_i(\widetilde{\mathbf{y}}) - \widetilde{\boldsymbol{\mu}}^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \widetilde{\mathbf{y}})$. Therefore, by definition of the dual problem (11), for all $(\widetilde{\mathbf{y}}, \widetilde{\boldsymbol{\mu}}) \in F_W$, we have

$$q_i^*(\mathbf{x}) \leq g_i(\widetilde{\mathbf{y}}) - \widetilde{\boldsymbol{\mu}}^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \widetilde{\mathbf{y}}).$$

Consequently, taking the infimum over all $(\widetilde{\mathbf{y}}, \widetilde{\boldsymbol{\mu}}) \in F_W$ yields $q_i^*(\mathbf{x}) \leq q_i^W(\mathbf{x})$, by the definition of the Wolfe dual. Note that F_W may be empty, in which case the infimum in the definition of $q_i^W(\mathbf{x})$ is over an empty set, and the inequality $q_i^*(\mathbf{x}) \leq q_i^W(\mathbf{x})$ holds somewhat trivially.

Next we establish $q_i^W(\mathbf{x}) \leq g_i^*(\mathbf{x})$, using strong duality for the Lagrangian dual. Since g_i is differentiable on Y, it is continuous on Y and since $D(\mathbf{x})$ is compact, (LLP i) achieves its supremum (since $D(\mathbf{x})$ is nonempty under the Slater condition). Under the convexity assumptions and Slater conditions, strong duality holds for (LLP i); i.e. $q_i^*(\mathbf{x}) = g_i^*(\mathbf{x})$ (see for instance Proposition 5.3.1 in [12]). Further, a duality multiplier exists; that is, there exists $\boldsymbol{\mu}^i \geq \mathbf{0}$ such that $g_i^*(\mathbf{x}) = \sup\{g_i(\mathbf{y}) - (\boldsymbol{\mu}^i)^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \mathbf{y}) : \mathbf{y} \in Y\}$. Since (LLP i) achieves its supremum, there exists a maximizer $\mathbf{y}^i$ of (LLP i). Because a duality multiplier exists, by Proposition 5.1.1 in [11], we have $\mathbf{y}^i \in \arg\max\{g_i(\mathbf{y}) - (\boldsymbol{\mu}^i)^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \mathbf{y}) : \mathbf{y} \in Y\}$, and thus

$$g_i^*(\mathbf{x}) = g_i(\mathbf{y}^i) - (\boldsymbol{\mu}^i)^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \mathbf{y}^i).$$

Again, since Y is a nonempty open set we have $\nabla g_i(\mathbf{y}^i) - \nabla_{\mathbf{y}}\mathbf{h}(\mathbf{x}, \mathbf{y}^i)\boldsymbol{\mu}^i = \mathbf{0}$. In other words, $(\mathbf{y}^i, \boldsymbol{\mu}^i) \in F_W$ defined before, which establishes the first claim. Finally, applying the definition of $q_i^W(\mathbf{x})$ as an infimum over F_W, we get that $g_i^*(\mathbf{x}) \geq q_i^W(\mathbf{x})$. But since $g_i^*(\mathbf{x}) = q_i^*(\mathbf{x})$ and $q_i^*(\mathbf{x}) \leq q_i^W(\mathbf{x})$ (established above), we have $g_i^*(\mathbf{x}) = q_i^W(\mathbf{x})$. □

With this, one can establish an NLP reformulation of (GSIP) which does not have complementarity constraints. Refer to NLP (14) as the "Wolfe reformulation."

Proposition 10. *Suppose Y is a nonempty open convex set. Suppose $D(\mathbf{x})$ is compact for each $\mathbf{x} \in X$ and $D(\mathbf{x}) = \{\mathbf{y} \in Y : \mathbf{h}(\mathbf{x}, \mathbf{y}) \leq \mathbf{0}\}$ for some $\mathbf{h} : X \times Y \to \mathbb{R}^{n_h}$ where $\mathbf{h}(\mathbf{x}, \cdot)$ is convex and differentiable for each $\mathbf{x} \in X$. Suppose that for each $i \in \{1, \ldots, m\}$ and each $\mathbf{x} \in X$ the Slater condition holds for* (LLP i)*: there exists a $\mathbf{y}_s \in Y$ such that $\mathbf{h}(\mathbf{x}, \mathbf{y}_s) < \mathbf{0}$. Suppose $\mathbf{g}$ is concave and differentiable. Then* (GSIP) *is equivalent to the NLP*

$$\begin{aligned} \sup_{\mathbf{x}, \mathbf{y}^1, \boldsymbol{\mu}^1, \ldots, \mathbf{y}^m, \boldsymbol{\mu}^m} \quad & \mathrm{vol}(D(\mathbf{x})) \\ \text{s.t. } & g_i(\mathbf{y}^i) - (\boldsymbol{\mu}^i)^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \mathbf{y}^i) \leq 0, \quad \forall i, \\ & \nabla g_i(\mathbf{y}^i) - \nabla_{\mathbf{y}}\mathbf{h}(\mathbf{x}, \mathbf{y}^i)\boldsymbol{\mu}^i = \mathbf{0}, \quad \forall i, \\ & \boldsymbol{\mu}^i \geq \mathbf{0}, \quad \mathbf{y}^i \in Y, \quad \forall i, \\ & \mathbf{x} \in X. \end{aligned} \tag{14}$$

Proof. Choose $\mathbf{x} \in X$ feasible in (GSIP), then $g_i^*(\mathbf{x}) \leq 0$ for each i. By Lemma 9, there exists $(\mathbf{y}^i, \boldsymbol{\mu}^i)$ such that $\boldsymbol{\mu}^i \geq \mathbf{0}$, $\mathbf{y}^i \in Y$, $\nabla g_i(\mathbf{y}^i) - \nabla_{\mathbf{y}}\mathbf{h}(\mathbf{x}, \mathbf{y}^i)\boldsymbol{\mu}^i = \mathbf{0}$, and $g_i^*(\mathbf{x}) = g_i(\mathbf{y}^i) - (\boldsymbol{\mu}^i)^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \mathbf{y}^i)$. In other words, $(\mathbf{x}, \mathbf{y}^1, \boldsymbol{\mu}^1, \ldots, \mathbf{y}^m, \boldsymbol{\mu}^m)$ is feasible in (14).

Conversely, choose $(\mathbf{x}, \mathbf{y}^1, \boldsymbol{\mu}^1, \ldots, \mathbf{y}^m, \boldsymbol{\mu}^m)$ feasible in (14). Again, by Lemma 9 (in fact, weak duality between (LLP i) and the Wolfe dual (13) suffices), we must have $g_i^*(\mathbf{x}) \leq 0$ for each i, which establishes that $\mathbf{x}$ is feasible in (GSIP). Equivalence follows. □

One notes that indeed NLP (14) does not have the complementarity constraints that make MPCC (10) numerically unfavorable. Proposition 10 is similar to Corollary 2.4 in [18]. The difference is that the latter result assumes that $-\mathbf{g}$ and $\mathbf{h}(\mathbf{x}, \cdot)$ are convex on all of $Y = \mathbb{R}^{n_y}$. As this is a rather strong assumption, this motivates the authors of [18] to weaken this, and merely assume that $\mathbf{g}$ is concave on $D(\mathbf{x})$ for each $\mathbf{x}$. They then obtain an NLP which adds the constraints $\mathbf{h}(\mathbf{x}, \mathbf{y}^i) \leq \mathbf{0}$ to NLP (14). In design centering applications, assuming that $\mathbf{g}$ is concave on Y versus assuming $\mathbf{g}$ is concave on $D(\mathbf{x})$ for all $\mathbf{x}$ is typically not much stronger anyway.

4.1.4 General quadratically constrained quadratic LLP

When $Y = \mathbb{R}^{n_y}$, $D(\mathbf{x})$ is defined in terms of a ball given by a weighted 2-norm, and each g_i is quadratic, a specific duality result can be used to reformulate (GSIP). It should be stressed that this does not require that the LLPs are convex programs, despite the fact that this is part of a section titled "Convex LLP." This duality result applies to the general case of a quadratic program with a single quadratic constraint; given $\mathbf{A}_0$, $\mathbf{A} \in \mathbb{S}^{n_y \times n_y}$, $\mathbf{b}_0$, $\mathbf{b} \in \mathbb{R}^{n_y}$, and c_0, $c \in \mathbb{R}$, define

$$p^* = \sup\{\mathbf{y}^{\mathrm{T}}\mathbf{A}_0\mathbf{y} + 2\mathbf{b}_0^{\mathrm{T}}\mathbf{y} + c_0 : \mathbf{y} \in \mathbb{R}^{n_y}, \mathbf{y}^{\mathrm{T}}\mathbf{A}\mathbf{y} + 2\mathbf{b}^{\mathrm{T}}\mathbf{y} + c \leq 0\}. \tag{15}$$

The (Lagrangian) dual of this problem is

$$\begin{aligned} q^Q &: \mu \mapsto \sup\{\mathbf{y}^{\mathrm{T}}(\mathbf{A}_0 - \mu\mathbf{A})\mathbf{y} + 2(\mathbf{b}_0 - \mu\mathbf{b})^{\mathrm{T}}\mathbf{y} + c_0 - \mu c : \mathbf{y} \in \mathbb{R}^{n_y}\}, \\ d^* &= \inf\{q^Q(\mu) : \mu \in \mathrm{dom}(q^Q), \mu \geq 0\}. \end{aligned} \tag{16}$$

Noting that the Lagrangian of (15) is a quadratic function, for given $\mu \geq 0$ the supremum defining the dual function q^Q is achieved at $\mathbf{y}^*$ iff the second-order conditions $\mathbf{A}_0 - \mu\mathbf{A} \preceq \mathbf{0}$, $(\mathbf{A}_0 - \mu\mathbf{A})\mathbf{y}^* = -(\mathbf{b}_0 - \mu\mathbf{b})$, are satisfied; otherwise the supremum is $+\infty$. This leads to

$$\begin{aligned} d^* = \inf_{\mu, \mathbf{y}} \ & \mathbf{y}^{\mathrm{T}}(\mathbf{A}_0 - \mu\mathbf{A})\mathbf{y} + 2(\mathbf{b}_0 - \mu\mathbf{b})^{\mathrm{T}}\mathbf{y} + c_0 - \mu c \\ \text{s.t. } & (\mathbf{A}_0 - \mu\mathbf{A})\mathbf{y} = -(\mathbf{b}_0 - \mu\mathbf{b}), \\ & \mathbf{A}_0 - \mu\mathbf{A} \preceq \mathbf{0}, \\ & \mu \geq 0, \quad \mathbf{y} \in \mathbb{R}^{n_y} \end{aligned} \tag{17}$$

(note the similarity to the Wolfe dual in §4.1.3). Whether or not program (15) is convex, strong duality holds assuming (15) has a Slater point. That is to say, $p^* = d^*$ (and the dual solution set is nonempty) assuming there exists $\mathbf{y}_s$ such that $\mathbf{y}_s^{\mathrm{T}}\mathbf{A}\mathbf{y}_s + 2\mathbf{b}^{\mathrm{T}}\mathbf{y}_s + c < 0$. A proof of this can be found in Appendix B of [16]. The proof depends on the somewhat cryptically named "S-procedure," which is actually a theorem of the alternative. A review of results related to the S-procedure or S-lemma can be found in [48]. The required results are stated formally in the following.

Lemma 11. *Consider the quadratically constrained quadratic program* (15) *and its dual* (16). *Suppose there exists* $\mathbf{y}_s$ *such that* $\mathbf{y}_s^{\mathrm{T}}\mathbf{A}\mathbf{y}_s + 2\mathbf{b}^{\mathrm{T}}\mathbf{y}_s + c < 0$. *Then* $p^* = d^*$. *Further, if* p^* *is finite, then the solution set of the dual problem* (16) *is nonempty.*

Proof. See [16, §B.4] □

Lemma 12. *Consider problem* (15) *and its dual* (16). *If* $p^* = d^*$ *(strong duality holds), and there exists* $(\mu^*, \mathbf{y}^*)$ *with* μ^* *in the solution set of the dual* (16) *and* $\mathbf{y}^*$ *in the solution set of problem* (15), *then* $(\mu^*, \mathbf{y}^*)$ *is optimal in problem* (17).

Proof. Since there is no duality gap, μ^* is a duality multiplier (see Proposition 5.1.4 in [11]). Thus, any optimal solution of the primal problem (15) maximizes the Lagrangian for this fixed μ^*, i.e. $q^Q(\mu^*) = (\mathbf{y}^*)^{\mathrm{T}}(\mathbf{A}_0 - \mu^*\mathbf{A})\mathbf{y}^* + 2(\mathbf{b}_0 - \mu^*\mathbf{b})^{\mathrm{T}}\mathbf{y}^* + c_0 - \mu^* c$ (see Proposition 5.1.1 in [11]). Thus the second-order conditions $\mathbf{A}_0 - \mu^*\mathbf{A} \preceq \mathbf{0}$, $(\mathbf{A}_0 - \mu^*\mathbf{A})\mathbf{y}^* = -(\mathbf{b}_0 - \mu^*\mathbf{b})$, are satisfied. Since $d^* = q^Q(\mu^*)$, $(\mu^*, \mathbf{y}^*)$ must be optimal in (17). □

A reformulation of (GSIP) when the LLPs are quadratically constrained quadratic programs follows.

Proposition 13. *Suppose $Y = \mathbb{R}^{n_y}$, and that for $i \in \{1, \ldots, m\}$ there exist $(\mathbf{A}_i, \mathbf{b}_i, c_i) \in \mathbb{S}^{n_y \times n_y} \times \mathbb{R}^{n_y} \times \mathbb{R}$ such that $g_i : \mathbf{y} \mapsto \mathbf{y}^{\mathrm{T}}\mathbf{A}_i\mathbf{y} + 2\mathbf{b}_i^{\mathrm{T}}\mathbf{y} + c_i$. Suppose $X \subset \{(\mathbf{P}, \mathbf{y}_c) \in \mathbb{S}^{n_y \times n_y} \times \mathbb{R}^{n_y} : \mathbf{P} \succ \mathbf{0}\}$. Suppose that $D(\mathbf{P}, \mathbf{y}_c) = \{\mathbf{y} : (\mathbf{y} - \mathbf{y}_c)^{\mathrm{T}}\mathbf{P}(\mathbf{y} - \mathbf{y}_c) \leq 1\}$ and $\mathrm{vol}(D(\mathbf{P}, \mathbf{y}_c)) = \det(\mathbf{P})^{-1}$. Then* (GSIP) *is equivalent to the program*

$$
\begin{aligned}
\sup_{\mathbf{P}, \mathbf{y}_c, \boldsymbol{\mu}, \mathbf{y}^1, \ldots, \mathbf{y}^m} \quad & \det(\mathbf{P})^{-1} \\
\text{s.t. } & (\mathbf{y}^i)^{\mathrm{T}}\mathbf{A}_i\mathbf{y}^i + 2\mathbf{b}_i^{\mathrm{T}}\mathbf{y}^i + c_i \\
& \qquad - \mu_i \left((\mathbf{y}^i - \mathbf{y}_c)^{\mathrm{T}}\mathbf{P}(\mathbf{y}^i - \mathbf{y}_c) - 1\right) \leq 0, \quad \forall i, \\
& (\mathbf{A}_i - \mu_i\mathbf{P})\mathbf{y}^i = -(\mathbf{b}_i + \mu_i\mathbf{P}\mathbf{y}_c), \quad \forall i, \\
& \mathbf{A}_i - \mu_i\mathbf{P} \preceq \mathbf{0}, \quad \forall i, \\
& \mu_i \geq 0, \quad \mathbf{y}^i \in \mathbb{R}^{n_y}, \quad \forall i, \\
& (\mathbf{P}, \mathbf{y}_c) \in X.
\end{aligned}
\tag{18}
$$

Proof. Note that $D(\mathbf{P}, \mathbf{y}_c) = \{\mathbf{y} : \mathbf{y}^{\mathrm{T}}\mathbf{P}\mathbf{y} - 2\mathbf{y}_c^{\mathrm{T}}\mathbf{P}\mathbf{y} + \mathbf{y}_c^{\mathrm{T}}\mathbf{P}\mathbf{y}_c - 1 \leq 0\}$. Also, for all $(\mathbf{P}, \mathbf{y}_c) \in X$, a solution exists for (LLP i) for each i since $\mathbf{P}$ is constrained to be positive definite and so $D(\mathbf{P}, \mathbf{y}_c)$ is compact. Further, by assumption on X, $\mathbf{y}_c$ is a Slater point for each LLP for all $(\mathbf{P}, \mathbf{y}_c) \in X$. Consequently, by Lemma 11, strong duality holds for each LLP and an optimal dual solution exists.

Choose $(\mathbf{P}, \mathbf{y}_c)$ feasible in (GSIP). Then for all i, $g_i^*(\mathbf{P}, \mathbf{y}_c) \leq 0$. Then by Lemma 12, there exists $(\mu_i, \mathbf{y}^i)$ optimal in the dual of (LLP i) written in the form (17), and combined with strong duality $(\mathbf{P}, \mathbf{y}_c, \boldsymbol{\mu}, \mathbf{y}^1, \ldots, \mathbf{y}^m)$ is feasible in (18). Conversely, choose $(\mathbf{P}, \mathbf{y}_c, \boldsymbol{\mu}, \mathbf{y}^1, \ldots, \mathbf{y}^m)$ feasible in (18). Weak duality establishes that $g_i^*(\mathbf{P}, \mathbf{y}_c) \leq 0$ for each i, and so $(\mathbf{P}, \mathbf{y}_c)$ is feasible in (GSIP). Equivalence follows. □

Note that program (18) contains nonlinear matrix inequalities. Consequently, many general-purpose software for the solution of NLP cannot handle this problem. Choosing $\mathbf{P}$ by some heuristic leads to a more practical reformulation. Further, Y can be restricted to a subset of $\mathbb{R}^{n_y}$ by taking advantage of strong duality. Refer to NLP (19) as the "Quadratic reformulation."

Corollary 14. *Suppose that for $i \in \{1, \ldots, m\}$ there exist $(\mathbf{A}_i, \mathbf{b}_i, c_i) \in \mathbb{S}^{n_y \times n_y} \times \mathbb{R}^{n_y} \times \mathbb{R}$ such that $g_i : \mathbf{y} \mapsto \mathbf{y}^{\mathrm{T}}\mathbf{A}_i\mathbf{y} + 2\mathbf{b}_i^{\mathrm{T}}\mathbf{y} + c_i$. Suppose $\mathbf{P} \in \mathbb{S}^{n_y \times n_y}$ is given and $\mathbf{P} \succ \mathbf{0}$, and further $X \subset \{(\mathbf{y}_c, \delta) : \mathbf{y}_c \in \mathbb{R}^{n_y}, \delta \geq \epsilon\}$ for some $\epsilon > 0$. Suppose that $D(\mathbf{y}_c, \delta) = \{\mathbf{y} : (\mathbf{y} - \mathbf{y}_c)^{\mathrm{T}}\mathbf{P}(\mathbf{y} - \mathbf{y}_c) \leq \delta^2\}$, $\mathrm{vol}(D(\mathbf{y}_c, \delta)) = \delta$, and that $Y \subset \mathbb{R}^{n_y}$ satisfies $D(\mathbf{y}_c, \delta) \subset Y$ for all $(\mathbf{y}_c, \delta) \in X$. Then* (GSIP) *is*

equivalent to the program

$$\begin{aligned}\sup_{\mathbf{y}_c,\delta,\boldsymbol{\mu},\mathbf{y}^1,\ldots,\mathbf{y}^m} \quad & \delta \\ \text{s.t. } & g_i(\mathbf{y}^i) - \mu_i\left((\mathbf{y}^i-\mathbf{y}_c)^{\mathrm{T}}\mathbf{P}(\mathbf{y}^i-\mathbf{y}_c)-\delta^2\right) \le 0, \quad \forall i, \\ & (\mathbf{A}_i-\mu_i\mathbf{P})\mathbf{y}^i = -(\mathbf{b}_i+\mu_i\mathbf{P}\mathbf{y}_c), \quad \forall i, \\ & \mathbf{A}_i-\mu_i\mathbf{P} \preceq \mathbf{0}, \quad \forall i, \\ & \mu_i \ge 0, \quad \mathbf{y}^i \in Y, \quad \forall i, \\ & (\mathbf{y}_c,\delta) \in X.\end{aligned} \tag{19}$$

Proof. The proof is similar to that of Proposition 13. The added constraint that the $\mathbf{y}^i$ components of the solutions of (19) are in Y does not change the fact that (19) is a "restriction" of (GSIP) (i.e. for $(\mathbf{y}_c,\delta,\boldsymbol{\mu},\mathbf{y}^1,\ldots,\mathbf{y}^m)$ feasible in (19), $(\mathbf{y}_c,\delta)$ is feasible in (GSIP)).

We only need to check that for $(\mathbf{y}_c,\delta)$ feasible in (GSIP), that there exist $(\mu_i,\mathbf{y}^i)$ such that $(\mathbf{y}_c,\delta,\boldsymbol{\mu},\mathbf{y}^1,\ldots,\mathbf{y}^m)$ is feasible in (19) (specifically that $\mathbf{y}^i \in Y$ for each i). But this must hold since we can take $\mathbf{y}^i$ and μ_i to be optimal solutions of (LLP i) and its dual, respectively, for each i. Thus $\mathbf{y}^i \in D(\mathbf{y}_c,\delta) \subset Y$ for each i, and so by strong duality and Lemma 12, $(\mathbf{y}_c,\delta,\boldsymbol{\mu},\mathbf{y}^1,\ldots,\mathbf{y}^m)$ is feasible in (19). □

The Quadratic reformulation (19) is still nonlinear and nonconvex, but the matrix inequality constraints are linear, and as demonstrated by the example in §4.2.2, these can sometimes be reformulated as explicit constraints on $\boldsymbol{\mu}$.

4.1.5 Convex quadratic constraints

This section discusses a special case of what was considered in §4.1.4, when $D(\mathbf{x})$ is defined in terms of a ball given by a weighted 2-norm, and each g_i is *convex* and quadratic. In this case, a convex reformulation is possible. This case has the geometric interpretation of inscribing the maximum volume ellipsoid in the intersection of ellipsoids, and has been considered in [10, §4.9.1], [15, §3.7.3], [16, §8.5]. The representation of the design space is a little different from what has been considered so far; it depends on the inverse of the symmetric square root of a positive semidefinite matrix.

Proposition 15. *Suppose $Y = \mathbb{R}^{n_y}$, and that for $i \in \{1,\ldots,m\}$ there exist $(\mathbf{A}_i,\mathbf{b}_i,c_i) \in \mathbb{S}^{n_y\times n_y} \times \mathbb{R}^{n_y}\times\mathbb{R}$, with $\mathbf{A}_i \succ \mathbf{0}$, such that $g_i : \mathbf{y} \mapsto \mathbf{y}^{\mathrm{T}}\mathbf{A}_i\mathbf{y}+2\mathbf{b}_i^{\mathrm{T}}\mathbf{y}+c_i$. Suppose $X \subset \{(\mathbf{P},\mathbf{y}_c) \in \mathbb{S}^{n_y\times n_y}\times\mathbb{R}^{n_y} : \mathbf{P} \succ \mathbf{0}\}$. Suppose that $D(\mathbf{P},\mathbf{y}_c) = \{\mathbf{y} : (\mathbf{y}-\mathbf{y}_c)^{\mathrm{T}}\mathbf{P}^{-2}(\mathbf{y}-\mathbf{y}_c) \le 1\}$ and $\mathrm{vol}(D(\mathbf{P},\mathbf{y}_c)) = \det(\mathbf{P})$. Then* (GSIP) *is equivalent to the program*

$$\begin{aligned}\sup_{\mathbf{P},\mathbf{y}_c,\boldsymbol{\mu}} \quad & \det(\mathbf{P}) \\ \text{s.t. } & \boldsymbol{\mu} \ge 0, \quad (\mathbf{P},\mathbf{y}_c) \in X, \\ & \begin{bmatrix} -\mathbf{A}_i^{-1} & -\mathbf{A}_i^{-1}\mathbf{b}_i-\mathbf{y}_c & \mathbf{P} \\ (-\mathbf{A}_i^{-1}\mathbf{b}_i-\mathbf{y}_c)^{\mathrm{T}} & \mu_i-(\mathbf{b}_i^{\mathrm{T}}\mathbf{A}_i^{-1}\mathbf{b}_i-c_i) & \mathbf{0} \\ \mathbf{P} & \mathbf{0} & -\mu_i\mathbf{I} \end{bmatrix} \preceq \mathbf{0}, \quad \forall i.\end{aligned} \tag{20}$$

Proof. The proof depends on the following characterization of (in essence) the dual function of (LLP i) in this case: Assuming $\mathbf{P}$, $\mathbf{A}_i$ are positive definite symmetric matrices and $\mu_i \ge 0$, we have

$$\sup\left\{\mathbf{y}^{\mathrm{T}}\mathbf{A}_i\mathbf{y}+2\mathbf{b}_i^{\mathrm{T}}\mathbf{y}+c_i-\mu_i\left((\mathbf{y}-\mathbf{y}_c)^{\mathrm{T}}\mathbf{P}^{-2}(\mathbf{y}-\mathbf{y}_c)-1\right) : \mathbf{y}\in\mathbb{R}^{n_y}\right\} \le 0 \tag{21}$$

if and only if

$$\begin{bmatrix} -\mathbf{A}_i^{-1} & -\mathbf{A}_i^{-1}\mathbf{b}_i - \mathbf{y}_c & \mathbf{P} \\ (-\mathbf{A}_i^{-1}\mathbf{b}_i - \mathbf{y}_c)^{\mathrm{T}} & \mu_i - (\mathbf{b}_i^{\mathrm{T}}\mathbf{A}_i^{-1}\mathbf{b}_i - c_i) & \mathbf{0} \\ \mathbf{P} & \mathbf{0} & -\mu_i\mathbf{I} \end{bmatrix} \preceq \mathbf{0}.$$

A proof of this equivalence can be found in §3.7.3 of [15].

Choosing $(\mathbf{P}, \mathbf{y}_c)$ feasible in (GSIP), by strong duality (Lemma 11) we have that for each i there exists $\mu_i \geq 0$ such that the dual function is nonpositive (i.e. Inequality (21) holds). Thus $(\mathbf{P}, \mathbf{y}_c, \boldsymbol{\mu})$ is feasible in problem (20). Conversely, for $(\mathbf{P}, \mathbf{y}_c, \boldsymbol{\mu})$ feasible in problem (20), by the above equivalence and weak duality $(\mathbf{P}, \mathbf{y}_c)$ is feasible in (GSIP). □

The matrix constraints in problem (20) are linear inequalities, and reformulating the objective along the lines of the discussion in §3.2 yields an SDP. As another practical note, the explicit matrix inverses in problem (20) could be removed, for instance, by introducing new variables $(\mathbf{E}_i, \mathbf{d}_i, f_i)$ and adding the linear constraints $\mathbf{A}_i\mathbf{E}_i = \mathbf{I}$, $\mathbf{A}_i\mathbf{d}_i = \mathbf{b}_i$, $\mathbf{b}_i^{\mathrm{T}}\mathbf{d}_i = f_i$, for each i.

4.2 Numerical examples

In this section global NLP solvers are applied to the reformulations of design centering problems discussed in the previous sections. The studies are performed in GAMS version 24.3.3 [21]. Deterministic global optimizers BARON version 14.0.3 [68, 52] and ANTIGONE version 1.1 [38] are employed. Algorithm 2.1 in [39], a global, feasible-point method, is applied to the SIP reformulation. An implementation is coded in GAMS, employing BARON for the solution of the subproblems (see also the discussion in §5.1.2). The initial parameters are $\epsilon^{g,0} = 1$, $r = 2$, and $Y^{LBP,0} = Y^{UBP,0} = \varnothing$. These examples have a single infinite constraint (single LLP), and so the subscripts on g and solution components are dropped. All numerical studies were performed on a 64-bit Linux virtual machine allocated a single core of a 3.07 GHz Intel Xeon processor and 1.28 GB RAM.

4.2.1 Convex LLP with interval design space

The following design centering problem is considered:

$$\begin{aligned} &\sup_{\mathbf{y}^L, \mathbf{y}^U} \ \mathrm{vol}([\mathbf{y}^L, \mathbf{y}^U]) \\ &\quad \text{s.t. } g(\mathbf{y}) = -(y_1+1)^2 - (y_2-1)^2 + 1 \leq 0, \quad \forall \mathbf{y} \in [\mathbf{y}^L, \mathbf{y}^U], \end{aligned} \tag{22}$$

where $\mathrm{vol}([\mathbf{y}^L, \mathbf{y}^U]) = (y_1^U - y_1^L)(y_2^U - y_2^L)$.

For the KKT and Wolfe reformulations, letting $Y = (-2,2)\times(-2,2)$, $X = \{(\mathbf{y}^L, \mathbf{y}^U) \in [-1,1]^2 \times [-1,1]^2 : y_1^U - y_1^L \geq 0.002, y_2^U - y_2^L \geq 0.002\}$, and

$$\mathbf{h} : X \times Y \ni (\mathbf{y}^L, \mathbf{y}^U, \mathbf{y}) \mapsto \begin{bmatrix} -\mathbf{I} \\ \mathbf{I} \end{bmatrix} \mathbf{y} + \begin{bmatrix} \mathbf{y}^L \\ -\mathbf{y}^U \end{bmatrix}$$

it is clear that the hypotheses of Propositions 6 and 10 hold. The corresponding reformulations are solved to global optimality with BARON and ANTIGONE. The relative and absolute optimality tolerances are both 10^{-4}. The solution obtained in each case is $(\mathbf{y}^L, \mathbf{y}^U) = (0, -1, 1, 1)$. The other components of the solution and the solution times are in Table 1.

Meanwhile, the SIP reformulation from Proposition 8 holds for $M = [\mathbf{0}, \mathbf{b}^*]$, where $\mathbf{b}^* = (18 \times 10^3)\mathbf{1}$ (for instance, by noting that $g(\mathbf{y}) \geq -18$ for all $\mathbf{y} \in Y$ and taking $h_{b,i} = 0.001$). However, to apply the feasible-point, global solution method for SIP from [39], Y needs to be compact, and so

Table 1: Solution times and solutions for problem (22) by various reformulations. The SIP reformulation was solved by the SIP method from [39].

Method	Solution Time (s)		Solution			
	BARON	ANTIGONE	$\mathbf{y}^L$	$\mathbf{y}^U$	$\boldsymbol{\mu}$	$\mathbf{y}$
KKT reformulation (10)	0.07	0.05	$(0,-1)$	$(1,1)$	$(2,0,0,0)$	$(0,1)$
Wolfe reformulation (14)	0.65	0.29	$(0,-1)$	$(1,1)$	$(2,0,0,0)$	$(0,1)$
SIP reformulation (12)	10.5		$(-1,-1)$	$(1,0)$	$(0,0,0,2)$	

let $Y = [-2,2] \times [-2,2]$ for the purposes of this reformulation. This method requires continuous objective, g, and $\mathbf{h}$, and this clearly holds. The overall relative and absolute optimality tolerances for the method in [39] each equal 10^{-4}, and the subproblems required by the method are all solved with relative and absolute optimality tolerances equal to 10^{-5}. The method terminates in 28 iterations and the solution obtained is $(\mathbf{y}^L, \mathbf{y}^U) = (-1,-1,1,0)$. Although different from what what obtained with the NLP reformulations, the optimal objective value is the same and it is still optimal. The solution time is included in Table 1.

As expected, the NLP reformulations are quicker to solve than the SIP reformulation. What is somewhat surprising is that the KKT reformulation, which is an MPCC, solves more quickly than the Wolfe reformulation, which omits the complementarity constraints. This is perhaps due to the nature of the global solvers BARON and ANTIGONE, which can recognize and efficiently handle the complementarity constraints [52], and overall make use of the extra constraints to improve domain reduction through constraint propagation.

However, note that the KKT and Wolfe reformulations involve the derivatives of $\mathbf{g}$ and $\mathbf{h}$. Subsequently, solving these reformulations with implementations of global methods such as BARON and ANTIGONE requires explicit expressions for these derivatives. In a general-purpose modeling language such as GAMS, supplying these derivative expressions typically must be done by hand which is tedious and error prone. In contrast, the solution method for the SIP (12) from [39] involves the solution of various NLP subproblems which are defined in terms of the original functions $\mathbf{g}$ and $\mathbf{h}$.

4.2.2 Nonconvex quadratic LLP

The following design centering problem is considered:

$$\begin{aligned} &\sup_{\mathbf{y}_c,\delta} \delta \\ &\text{s.t. } g(\mathbf{y}) = y_1^2 - y_2^2 \leq 0, \quad \forall \mathbf{y} : \|\mathbf{y} - \mathbf{y}_c\|_2 \leq \delta. \end{aligned} \tag{23}$$

Note that g is a nonconvex quadratic function, and that the lower-level program is a quadratically-constrained quadratic program. Lemma 11 asserts that strong duality holds for the lower-level program assuming a Slater point exists. Each of the reformulations of §4 are considered. This is to demonstrate what happens when strong duality holds for the lower-level program, but an inappropriate reformulation is used. It will be seen that the KKT and Wolfe reformulations fail to give a correct answer, while the SIP reformulation succeeds.

Let $X = \{(\mathbf{y}_c, \delta) : \|\mathbf{y}_c\|_2 \leq 2, 0.1 \leq \delta \leq 2\}$. Let

$$\mathbf{A} = \left[\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right] \quad \text{and} \quad \mathbf{P} = \mathbf{I},$$

Table 2: Solution times and solutions for problem (23) by various reformulations. The SIP reformulation was solved by the SIP method from [39].

Method	Solution Time (s)		Solution			
	BARON	ANTIGONE	$\mathbf{y}_c$	δ	μ	$\mathbf{y}$
Quadratic reformulation (19)	2.07	17.08	$(0,-2)$	1.414	1	$(4,-1)$
KKT reformulation (10)	0.01	0.01	$(0,0)$	2	0	$(0,0)$
Wolfe reformulation (14)	0.01	0.01	$(0,0)$	2	0	$(0,0)$
SIP reformulation (12)	22.76		$(0,2)$	1.414	1	

where $\mathbf{I}$ is the identity matrix, so that $g(\mathbf{y}) = \mathbf{y}^{\mathrm{T}}\mathbf{A}\mathbf{y}$ and $D(\mathbf{y}_c, \delta) = \{\mathbf{y} : (\mathbf{y}-\mathbf{y}_c)^{\mathrm{T}}\mathbf{P}(\mathbf{y}-\mathbf{y}_c) \leq \delta^2\}$. With $Y = [-4,4]\times[-4,4]$, the Quadratic reformulation (19) is applicable by Corollary 14. Further, since $\mathbf{A}$ and $\mathbf{P}$ are diagonal, the matrix inequality constraint $\mathbf{A} - \mu\mathbf{P} \preceq \mathbf{0}$ in that problem can be reformulated as nonpositivity of the diagonal elements of $\mathbf{A} - \mu\mathbf{P}$, which reduces to $\mu \geq 1$ (and $\mu \geq -1$, but this is redundant). General-purpose solvers can handle this form of the constraint more easily.

The Quadratic reformulation (19) is solved to global optimality with BARON and ANTIGONE. The relative and absolute optimality tolerances are both 10^{-4}. The solution obtained in each case is $\mathbf{y}_c = (0,-2)$, $\delta = 1.414$. The other components of the solution and solution statistics are in Table 2.

The SIP reformulation is also applicable in this case. The center of the design space $\mathbf{y}_c$ is a Slater point for the lower-level program for all $(\mathbf{y}_c, \delta) \in X$, and so $g(\mathbf{y}_c) \geq -4$ for all $(\mathbf{y}_c, \delta) \in X$. Further, let $h(\mathbf{y}_c, \delta, \mathbf{y}) = (\mathbf{y}-\mathbf{y}_c)^{\mathrm{T}}\mathbf{P}(\mathbf{y}-\mathbf{y}_c) - \delta^2$, so that $h(\mathbf{y}_c, \delta, \mathbf{y}_c) = -\delta^2 \leq -0.01$ for all $(\mathbf{y}_c, \delta) \in X$. With Lemma 3.4 in [28] and the strong duality result from Lemma 11, the conclusion of Proposition 8 holds with $M = [0, 400]$. The overall relative and absolute optimality tolerances for the method in [39] each equal 10^{-4}, and the subproblems required by the method are all solved with relative and absolute optimality tolerances equal to 2×10^{-5}. The method terminates in 33 iterations and the solution obtained is $(\mathbf{y}_c, \delta) = (0, 2, 1.414)$, a different optimal solution for (23). What is interesting to note is that the SIP solution method is competitive with the solution of the Quadratic reformulation for this example; see Table 2.

Not surprisingly, the KKT and Wolfe reformulations fail to provide even a feasible solution. Quite simply, this is due to the omission of the constraint $\mu \geq 1$, which is the only difference between the Wolfe reformulation (14) and the Quadratic reformulation (19). Consequently, even when strong duality holds, one must be careful if attempting to apply the KKT and Wolfe reformulations when global optima are sought. Meanwhile, in [18], the authors apply reformulations similar to (10) and (14) to a problem with quadratic LLPs, but do not include the constraint on the duality multiplier for a nonconvex LLP. However, the focus of that work is on local solutions and methods. In particular, for the numerical results, convergence of a local solution method to an infeasible point might not be observed if the starting point is sufficiently close to a local minimum.

5 Nonconvex LLP

In this section, approaches for finding a feasible solution of (GSIP) are considered, with optimality being a secondary concern. To this end, the focus is on constructing restrictions of (GSIP) which can be solved to global optimality practically. This has the benefit that these methods do not rely on initial guesses that typically must be supplied to a local optimization method.

Furthermore, the motivation of this section are those instances of (GSIP) when $\mathbf{g}$ might not be explicitly defined. For instance, in many engineering applications, the design constraints $\mathbf{g}$ may be defined implicitly by the solution of a system of algebraic or differential equations; the example in §5.2.1 provides an instance. In this case, many solution methods are impractical if not impossible to apply. For example, in the previous section, global solution of the NLP reformulations (10) and (14) typically requires explicit expressions for the derivative of $\mathbf{g}$. Thus applying these reformulations does not lead to a practical method. In the context of SIP, [64] provides a good discussion of why many methods (for SIP) cannot be applied practically to infinitely-constrained problems in engineering applications.

5.1 Methods

5.1.1 Interval restriction with branch and bound

In this section, a method for solving a restriction of (GSIP) is described. The restriction is constructed by noting that the constraint $g_i(\mathbf{y}) \leq 0$ for all $\mathbf{y} \in D(\mathbf{x})$ is equivalent to $g_i^*(\mathbf{x}) \leq 0$, where g_i^* is defined by (LLP i). To describe the restriction and solution method, make the following assumption; as in §3.3, it is assumed that a candidate design space is an interval parameterized by its endpoints.

Assumption 2. *Let $\mathbb{I}Y$ denote the set of all nonempty interval subsets of Y ($\mathbb{I}Y = \{[\mathbf{v}, \mathbf{w}] \subset Y : [\mathbf{v}, \mathbf{w}] \neq \varnothing\}$). Suppose that $X \subset \{(\mathbf{v}, \mathbf{w}) \in \mathbb{R}^{n_y} \times \mathbb{R}^{n_y} : \mathbf{v} \leq \mathbf{w}\}$ and denote the upper variables of* (GSIP) *as $\mathbf{x} = (\mathbf{y}^L, \mathbf{y}^U)$ and let $D(\mathbf{y}^L, \mathbf{y}^U) = [\mathbf{y}^L, \mathbf{y}^U]$. Assume that for interval A,* $\mathrm{vol}(A)$ *equals the volume of A. Assume that for each i, the function $g_i^U : \mathbb{I}Y \to \mathbb{R}$ satisfies $g_i^U(D(\mathbf{x})) \geq g_i^*(\mathbf{x})$, for all $\mathbf{x} \in X$. Assume that each g_i^U is monotonic in the sense that $g_i^U(A) \leq g_i^U(B)$ for all A, $B \in \mathbb{I}Y$ with $A \subset B$.*

Under Assumption 2, the following program is a restriction of (GSIP):

$$\begin{aligned} \sup_{\mathbf{x}} \ & \mathrm{vol}(D(\mathbf{x})) \\ \text{s.t. } & g_i^U(D(\mathbf{x})) \leq 0, \forall i, \\ & \mathbf{x} \in X. \end{aligned} \tag{24}$$

One could take $g_i^U(A) = \max\{g_i(\mathbf{y}) : \mathbf{y} \in A\}$ (so that $g_i^U(D(\mathbf{x})) = g_i^*(\mathbf{x})$ trivially). There exist a number of results dealing with such mappings; [5, 6, 34, 49] are among a few dealing with the continuity and differentiability properties of such maps. Then one approach to solve (GSIP) might be to analyze (24) as an NLP using some of these parametric optimization results. This characterizes the "local reduction" approaches in [30, 32, 51, 62], as well as a local method for SIP which takes into account the potential nonsmoothness of g_i^* in [47].

The subject of this section is a different approach, where the restriction (24) is solved globally with branch and bound. In this case, one can take advantage of g_i^U which are cheap to evaluate. For instance, let G_i be an interval-valued mapping on $\mathbb{I}Y$. G_i is said to be inclusion monotonic if it satisfies $G_i(A) \subset G_i(B)$ for all A, $B \in \mathbb{I}Y$ with $A \subset B$. Further, G_i is said to be an inclusion function of g_i if it satisfies $G_i(A) \ni g_i(\mathbf{y})$, for all $\mathbf{y} \in A$, $A \in \mathbb{I}Y$. Then letting g_i^U be the upper bound of G_i, we have that g_i^U satisfies Assumption 2. See [43] for an introduction to interval arithmetic and inclusion functions. The benefit is that the interval-valued inclusion functions are typically cheaper to evaluate than the global optimization problems defining g_i^*. The idea of using interval arithmetic to construct a restriction is related to the method in [50], except that the optimization

approach in that work is based on an "evolutionary" optimization algorithm. Conceptually similar are the methods for the global solution of SIP in [13, 14].

To solve problem (24) via branch and bound, one needs to be able to obtain lower bounds and upper bounds on the optimal objective values of the subproblems

$$f_k^* = \sup\{\text{vol}(D(\mathbf{x})) : \mathbf{x} \in X_k, g_i^U(\mathbf{x}) \leq 0, \forall i\},$$

where $X_k \subset X$. For this discussion assume X_k is a nonempty interval subset of X. This means X_k will have the form $[\mathbf{v}_k^L, \mathbf{v}_k^U] \times [\mathbf{w}_k^L, \mathbf{w}_k^U]$ for $\mathbf{v}_k^L$, $\mathbf{v}_k^U$, $\mathbf{w}_k^L$, $\mathbf{w}_k^U \in \mathbb{R}^{n_y}$. Under Assumption 2, one always has $\mathbf{y}^L \leq \mathbf{y}^U$ for $(\mathbf{y}^L, \mathbf{y}^U) = \mathbf{x} \in X$. Thus, if X_k is a subset of X, one has $\mathbf{v}_k^U \leq \mathbf{w}_k^L$. Furthermore,

$$[\mathbf{v}_k^U, \mathbf{w}_k^L] \subset [\mathbf{y}^L, \mathbf{y}^U] \subset [\mathbf{v}_k^L, \mathbf{w}_k^U], \quad \forall (\mathbf{y}^L, \mathbf{y}^U) \in [\mathbf{v}_k^L, \mathbf{v}_k^U] \times [\mathbf{w}_k^L, \mathbf{w}_k^U] \subset X.$$

Consequently, $UB_k = \text{vol}([\mathbf{v}_k^L, \mathbf{w}_k^U])$ is an upper bound for the optimal subproblem objective f_k^*. Meanwhile, any feasible point provides a lower bound. In the context of the current problem, there are two natural choices:

1. The point $(\mathbf{v}_k^L, \mathbf{w}_k^U)$ represents the "largest" candidate design space possible in X_k. Consequently, if feasible, it gives the best lower bound for this node.
2. The point $(\mathbf{v}_k^U, \mathbf{w}_k^L)$ represents the "smallest" candidate design space possible in X_k, and thus is more likely to be feasible, and thus to provide a nontrivial lower bound. However, if it is infeasible, i.e. if $g_i^U(\mathbf{v}_k^U, \mathbf{w}_k^L) > 0$ for some i, then $g_i^U(\mathbf{y}^L, \mathbf{y}^U) > 0$ for all $(\mathbf{y}^L, \mathbf{y}^U) = \mathbf{x} \in X_k$ (by the monotonicity property in Assumption 2), and thus X_k can be fathomed by infeasibility.

As noted, with the definition $g_i^U(D(\mathbf{x})) = g_i^*(\mathbf{x})$, determining whether either of these points is feasible requires evaluating g_i^*, which is still a global optimization problem. In contrast, if the g_i^U are cheap to evaluate, these lower and upper bounds are also cheap to obtain.

Under mild assumptions, if (GSIP) has a solution, so does the restriction (24), although it may be a somewhat trivial solution. Assume $g_i^U([\mathbf{y}, \mathbf{y}]) = g_i(\mathbf{y})$ for all $\mathbf{y}$ (as is the case when g_i^U is the upper bound of an interval extension of g_i). Let $D(\mathbf{x}^*) = [\mathbf{y}^{L,*}, \mathbf{y}^{U,*}]$ be a solution of (GSIP); then for any $\widehat{\mathbf{y}} \in D(\mathbf{x}^*)$, one has $g_i(\widehat{\mathbf{y}}) \leq 0$ for all i, and so $[\widehat{\mathbf{y}}, \widehat{\mathbf{y}}]$ is feasible in the restriction (24), assuming $(\widehat{\mathbf{y}}, \widehat{\mathbf{y}}) \in X$. Although $[\widehat{\mathbf{y}}, \widehat{\mathbf{y}}]$ would violate the Slater condition that has been present in many results, it is unnecessary in this approach.

Numerical experiments (see §5.2.1) show that the branch and bound algorithm applied to this problem can be slow. To try to explain why this might be the case, consider the lower and upper bounds described above. For a nonempty interval subset $X_k = [\mathbf{v}_k^L, \mathbf{v}_k^U] \times [\mathbf{w}_k^L, \mathbf{w}_k^U]$ of X, in the worst case $f_k^* = \text{vol}(D(\mathbf{v}_k^U, \mathbf{w}_k^L))$, while the upper bound is $UB_k = \text{vol}(D(\mathbf{v}_k^L, \mathbf{w}_k^U))$. In one dimension ($n_y = 1$), for example, $\text{vol}(D(y^L, y^U)) = y^U - y^L$, so $f_k^* = w_k^L - v_k^U$ and $UB_k = w_k^U - v_k^L$. Meanwhile, the width (or diameter) of X_k is $\text{diam}(X_k) = \max\{(w_k^U - w_k^L), (v_k^U - v_k^L)\}$. Thus one has

$$UB_k - f_k^* = (w_k^U - w_k^L) + (v_k^U - v_k^L) \leq 2\,\text{diam}(X_k).$$

Thus, the bounding procedure described is at least first-order convergent (see Definition 2.1 in [70], noting that the present problem is a maximization). See Appendix A for the general case. However, also note that for all $\alpha > 0$, there exists nonempty $\widetilde{X} = [\widetilde{v}^L, \widetilde{v}^U] \times [\widetilde{w}^L, \widetilde{w}^U]$ sufficiently small so that $\widetilde{w}^U - \widetilde{w}^L > \alpha(\widetilde{w}^U - \widetilde{w}^L)^2$ and $\widetilde{v}^U - \widetilde{v}^L > \alpha(\widetilde{v}^U - \widetilde{v}^L)^2$ which implies

$$\begin{aligned}(\widetilde{w}^U - \widetilde{w}^L) + (\widetilde{v}^U - \widetilde{v}^L) &> \alpha\big((\widetilde{w}^U - \widetilde{w}^L)^2 + (\widetilde{v}^U - \widetilde{v}^L)^2\big) \\ &\geq \alpha \max\{(\widetilde{w}^U - \widetilde{w}^L)^2, (\widetilde{v}^U - \widetilde{v}^L)^2\} \\ &= \alpha(\max\{(\widetilde{w}^U - \widetilde{w}^L), (\widetilde{v}^U - \widetilde{v}^L)\})^2 = \alpha\,\text{diam}(\widetilde{X})^2.\end{aligned}$$

This establishes that the method is not, in general, second-order convergent. When the solution is unconstrained, a convergence order of two or greater is required to avoid the "cluster problem" when applying the branch and bound method; this refers to a phenomenon that hinders the efficiency of the branch and bound method (see [19, 71]). A deeper understanding of these issues might be wise if attempting to develop the method in this section further.

5.1.2 SIP restriction

Proposition 7 provides the inspiration for another restriction-based method; for $M \subset \mathbb{R}^{n_h}$ with nonempty intersection with the nonnegative orthant, SIP (12) is a restriction of (GSIP) (this follows from weak duality). If this SIP restriction is feasible, subsequently solving it with a feasible-point method yields a feasible solution of (GSIP).

In contrast, the other duality-based reformulations of (GSIP), such as those in Propositions 6 and 10, do not provide restrictions if the assumption of convexity of the LLPs is dropped. Furthermore, as mentioned in the discussion in §4.2.1, solving these reformulations globally would require explicit expressions for the derivatives of $\mathbf{g}$. Since the overall goal of this section is to be able to handle robust design problems where $\mathbf{g}$ might be defined implicitly by the solution of systems of algebraic or differential equations, such information can be difficult to obtain.

The discussion in §4.2.2 also demonstrates that the SIP reformulation can take advantage of strong duality even in the cases that the specific hypotheses of Proposition 8 fail. In other words, if strong duality happens to hold for the LLPs of a specific problem, there is hope that the global solution of SIP (12) will also be the global solution of the original problem (GSIP).

For simplicity assume $m = 1$ (so there is a single LLP) and drop the corresponding index. Global solution of SIP (12) by the feasible-point method from [39] (at a specific iteration of the method) requires the solution of the subproblems

$$\begin{aligned} &\sup_{\mathbf{x},\boldsymbol{\mu}} \operatorname{vol}(D(\mathbf{x})) \\ &\text{s.t. } g(\mathbf{y}) - \boldsymbol{\mu}^{\mathrm{T}}\mathbf{h}(\mathbf{x},\mathbf{y}) \leq 0, \quad \forall \mathbf{y} \in Y^{UBP}, \\ &\quad \boldsymbol{\mu} \geq \mathbf{0}, \quad \boldsymbol{\mu} \in M \\ &\quad \mathbf{x} \in X, \end{aligned} \tag{UBP}$$

$$\begin{aligned} &\sup_{\mathbf{x},\boldsymbol{\mu}} \operatorname{vol}(D(\mathbf{x})) \\ &\text{s.t. } g(\mathbf{y}) - \boldsymbol{\mu}^{\mathrm{T}}\mathbf{h}(\mathbf{x},\mathbf{y}) \leq -\epsilon^g, \quad \forall \mathbf{y} \in Y^{LBP}, \\ &\quad \boldsymbol{\mu} \geq \mathbf{0}, \quad \boldsymbol{\mu} \in M, \\ &\quad \mathbf{x} \in X, \end{aligned} \tag{LBP}$$

and for given $(\mathbf{x}, \boldsymbol{\mu})$,

$$q(\mathbf{x}, \boldsymbol{\mu}) = \sup\{g(\mathbf{y}) - \boldsymbol{\mu}^{\mathrm{T}}\mathbf{h}(\mathbf{x},\mathbf{y}) : \mathbf{y} \in Y\}, \tag{SIP LLP}$$

for finite subsets $Y^{LBP} \subset Y$, $Y^{UBP} \subset Y$, and $\epsilon^g > 0$. Note that each subproblem is a finite NLP that is defined in terms of the original functions g and $\mathbf{h}$, and not their derivatives. As their names suggest, (UBP) and (LBP) aim to furnish upper and lower bounds, respectively, on SIP (12) that converge as the algorithm iterates.

A source of numerical difficulty that can arise in applying this method follows. A part of the algorithm is determining the feasibility (in SIP (12)) of the optimal solution $(\mathbf{x}, \boldsymbol{\mu})$ of either

(UBP) or (LBP). This requires solving (SIP LLP) and checking that $q(\mathbf{x}, \boldsymbol{\mu}) \leq 0$. One must either guarantee that $q(\mathbf{x}, \boldsymbol{\mu}) \leq 0$, or else find $\mathbf{y} \in Y$ such that $g(\mathbf{y}) - \boldsymbol{\mu}^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \mathbf{y}) > 0$. In the latter case, $\mathbf{y}$ is added to the discretization set Y^{UBP} (or Y^{LBP}). Typically global optimization methods find such guaranteed bounds and feasible points, but in practice one can often have the situation on finite termination that the approximate solution $\mathbf{y}$ of (SIP LLP) and its guaranteed upper bound UB^{llp} satisfy $g(\mathbf{y}) - \boldsymbol{\mu}^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \mathbf{y}) \leq 0 < UB^{llp}$. In this case, one cannot guarantee that the point $(\mathbf{x}, \boldsymbol{\mu})$ is feasible in SIP (12). Meanwhile, adding $\mathbf{y}$ to the discretization set Y^{UBP} does not eliminate the previous optimal solution $(\mathbf{x}, \boldsymbol{\mu})$, since $g(\mathbf{y}) - \boldsymbol{\mu}^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \mathbf{y}) \leq 0$. Consequently, it is highly likely that the numerical method used to solve (UBP) produces the same solution in the next iteration, and the same ambiguity arises when solving (SIP LLP). The cycle repeats, and the upper bound that the method provides fails to improve. A similar effect can occur with the lower-bounding problem (LBP).

This effect can be overcome by redefining g by adding a constant tolerance to its value. Consider that the pathological case $\widetilde{g} = g(\mathbf{y}) - \boldsymbol{\mu}^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \mathbf{y}) \leq 0 < UB^{llp}$ occurs, where $\mathbf{y}$ is the approximate solution found for (SIP LLP). Assume that the relative and absolute optimality tolerances for the global optimization method used are $\varepsilon_{rtol} \leq 1$ and ε_{atol}, respectively. In this case $UB^{llp} - \widetilde{g} > \varepsilon_{rtol}\,|\widetilde{g}|$. So assuming that the termination criterion of the global optimization procedure is $UB^{llp} - \widetilde{g} \leq \max\{\varepsilon_{atol}, \varepsilon_{rtol}\,|\widetilde{g}|\}$, it is easy to see that one must have $UB^{llp} - \widetilde{g} \leq \varepsilon_{atol}$ and thus $UB^{llp} \leq \widetilde{g} + \varepsilon_{atol}$.

To determine if $(\mathbf{x}, \boldsymbol{\mu})$ is feasible, solve (SIP LLP) and let the solution be $\mathbf{y}$. Then, if $g(\mathbf{y}) + \varepsilon_{atol} - \boldsymbol{\mu}^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \mathbf{y}) \leq 0$, by the preceding discussion one can guarantee that $(\mathbf{x}, \boldsymbol{\mu})$ is feasible in the SIP (12). However, if $g(\mathbf{y}) + \varepsilon_{atol} - \boldsymbol{\mu}^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \mathbf{y}) > 0$, adding the point $\mathbf{y}$ to the discretization set Y^{UBP} actually does restrict the upper-bounding problem (UBP), where g is redefined as $g \equiv g + \varepsilon_{atol}$. In effect, the following restriction of SIP (12)

$$\begin{aligned} &\sup_{\mathbf{x},\boldsymbol{\mu}} \operatorname{vol}(D(\mathbf{x})) \\ &\text{s.t. } g(\mathbf{y}) + \varepsilon_{atol} - \boldsymbol{\mu}^{\mathrm{T}}\mathbf{h}(\mathbf{x}, \mathbf{y}) \leq 0, \quad \forall \mathbf{y} \in Y, \\ &\quad \boldsymbol{\mu} \geq \mathbf{0}, \quad \boldsymbol{\mu} \in M, \\ &\quad \mathbf{x} \in X, \end{aligned} \tag{25}$$

is being solved with a method which does not quite guarantee feasibility (in (25)) of the solutions found. However, the solutions found *are* feasible in the original SIP (12).

For concreteness, further aspects of this approach are discussed in the context of the example considered in §5.2.1.

5.1.3 Generically valid solution

Methods for the general case of GSIP, and related problems such as bi-level programs, with non-convex lower-level programs have been presented in [41, 42, 69]. We investigate the method from [42] as another approach to solving a robust design problem.

We assume the same setting as in §4.1.2 and the previous section: for $\mathbf{x} \in X$, $D(\mathbf{x}) = \{\mathbf{y} \in Y : \mathbf{h}(\mathbf{x}, \mathbf{y}) \leq \mathbf{0}\}$ for some mapping $\mathbf{h} : X \times Y \to \mathbb{R}^{n_h}$. Then the work in [42] is based on the reformulation of (GSIP) as

$$\begin{aligned} &\sup_{\mathbf{x} \in X} \operatorname{vol}(D(\mathbf{x})) \\ &\text{s.t. } (g_i(\mathbf{y}) \leq 0 \vee \exists j : h_j(\mathbf{x}, \mathbf{y}) > 0), \quad \forall \mathbf{y} \in Y, \forall i. \end{aligned} \tag{26}$$

This is a mathematical program with an infinite number of disjunctive constraints. These constraints are encoding the same information as in (GSIP); a given $\mathbf{x}$ is feasible if for each i, and for each $\mathbf{y} \in Y$, either $g_i(\mathbf{y}) \leq 0$ or $\mathbf{y}$ is infeasible in (LLP i) (in this case, if some constraint $h_j(\mathbf{x}, \mathbf{y}) \leq 0$ is violated). The presence of the strict inequality makes numerical solution difficult, and so (26) is relaxed to

$$\begin{aligned} &\sup_{\mathbf{x} \in X} \operatorname{vol}(D(\mathbf{x})) \\ &\text{s.t. } (g_i(\mathbf{y}) \leq 0 \vee \exists j : h_j(\mathbf{x}, \mathbf{y}) \geq 0), \quad \forall \mathbf{y} \in Y, \forall i. \end{aligned} \tag{27}$$

This problem is then equivalent to the following SIP with nonsmooth constraints

$$\begin{aligned} &\sup_{\mathbf{x} \in X} \operatorname{vol}(D(\mathbf{x})) \\ &\text{s.t. } \max\left\{g_i(\mathbf{y}), \max_j\{-h_j(\mathbf{x}, \mathbf{y})\}\right\} \leq 0, \quad \forall \mathbf{y} \in Y, \forall i. \end{aligned} \tag{28}$$

The numerical solution method in [42] is based on applying the SIP method from [39] (upon which the method of §5.1.2 is based) to the SIP (28).

Strictly, then, the numerical method is solving a relaxation of (GSIP), and there may be some concern that an infeasible point is found. However, we can say that the feasible set of (27) (or (28)) is "typically" the closure of the feasible set of (GSIP). The technical term is that the property is *generically* valid; see [25, 26]. Further, assuming that $\mathbf{x} \mapsto \operatorname{vol}(D(\mathbf{x}))$ is continuous, maximizing it over the original feasible set or its closure yields the same optimal objective value. Further, the method is still a feasible point method, and the feasibility of candidate solutions is validated by solving the original lower-level programs (LLP i) of (GSIP) (and *not*, for instance, the lower-level program of the relaxation (28)). Thus, we refer to the method of this section as the "generically valid" solution.

The method in [42] for the generically valid solution of (GSIP) requires the solution of the subproblems at a specific iteration (again, assuming $m = 1$ and dropping the subscripts)

$$\begin{aligned} &\sup_{\mathbf{x} \in X} \operatorname{vol}(D(\mathbf{x})) \\ &\text{s.t. } (g(\mathbf{y}) \leq 0 \vee \exists j : h_j(\mathbf{x}, \mathbf{y}) \geq 0), \quad \forall \mathbf{y} \in Y^{UBP}, \end{aligned} \tag{GSIP-UBP}$$

$$\begin{aligned} &\sup_{\mathbf{x} \in X} \operatorname{vol}(D(\mathbf{x})) \\ &\text{s.t. } (g(\mathbf{y}) \leq -\epsilon_g \vee \exists j : h_j(\mathbf{x}, \mathbf{y}) \geq \epsilon_h), \quad \forall \mathbf{y} \in Y^{LBP}, \end{aligned} \tag{GSIP-LBP}$$

and for given $\mathbf{x}$,

$$\sup\{g(\mathbf{y}) : \mathbf{h}(\mathbf{x}, \mathbf{y}) \leq \mathbf{0}, \mathbf{y} \in Y\} \tag{LLP}$$

and

$$\begin{aligned} &\inf_{\mathbf{y} \in Y} \max_j\{h_j(\mathbf{x}, \mathbf{y})\} \\ &\text{s.t. } g(\mathbf{y}) \geq \alpha g^*(\mathbf{x}), \end{aligned} \tag{LLP-AUX}$$

where $\alpha \in (0, 1)$ is a fixed parameter, and again, Y^{LBP} and Y^{UBP} are finite subsets of Y and ϵ_g and ϵ_h are positive parameters that may change from iteration to iteration.

In practice, the disjunctive constraints (or equivalent nonsmooth constraints) in (GSIP-UBP) and (GSIP-LBP) are reformulated by introducing binary variables as in [42]. We obtain

$$\begin{aligned}
\sup_{\mathbf{x}, z_g(\mathbf{y}), z_j(\mathbf{y})} \quad & \mathrm{vol}(D(\mathbf{x})) && \text{(GSIP-UBP-B)}\\
\text{s.t. } & z_g(\mathbf{y}) \leq 1 - \frac{g(\mathbf{y})}{g^{\max}}, \quad \forall \mathbf{y} \in Y^{UBP},\\
& z_j(\mathbf{y}) \leq 1 + \frac{h_j(\mathbf{x}, \mathbf{y})}{\bar{h}^{\max}}, \quad \forall j, \forall \mathbf{y} \in Y^{UBP},\\
& z_g(\mathbf{y}) + \sum_j z_j(\mathbf{y}) \geq 1, \quad \forall \mathbf{y} \in Y^{UBP},\\
& \mathbf{x} \in X,\\
& z_g(\mathbf{y}) \in \{0,1\},\ z_j(\mathbf{y}) \in \{0,1\}, \forall j, \quad \forall \mathbf{y} \in Y^{UBP},
\end{aligned}$$

and

$$\begin{aligned}
\sup_{\mathbf{x}, z_g(\mathbf{y}), z_j(\mathbf{y})} \quad & \mathrm{vol}(D(\mathbf{x})) && \text{(GSIP-LBP-B)}\\
\text{s.t. } & z_g(\mathbf{y}) \leq 1 - \frac{g(\mathbf{y}) + \epsilon_g}{g^{\max} + \epsilon_g}, \quad \forall \mathbf{y} \in Y^{LBP},\\
& z_j(\mathbf{y}) \leq 1 - \frac{-h_j(\mathbf{x}, \mathbf{y}) + \epsilon_h}{\bar{h}_j^{\max} + \epsilon_h}, \quad \forall j, \forall \mathbf{y} \in Y^{LBP},\\
& z_g(\mathbf{y}) + \sum_j z_j(\mathbf{y}) \geq 1, \quad \forall \mathbf{y} \in Y^{LBP},\\
& \mathbf{x} \in X,\\
& z_g(\mathbf{y}) \in \{0,1\},\ z_j(\mathbf{y}) \in \{0,1\}, \forall j, \quad \forall \mathbf{y} \in Y^{LBP},
\end{aligned}$$

where $g^{\max} \geq \sup\{g(\mathbf{y}) : \mathbf{y} \in Y\}$ and $\bar{h}_j^{\max} \geq \sup\{-h_j(\mathbf{x}, \mathbf{y}) : (\mathbf{x}, \mathbf{y}) \in X \times Y\}$. Since Y^{UBP} is finite, for instance, the notation $z_g(\mathbf{y})$ merely means that we are indexing a vector of binary variables $\mathbf{z}_g$ by $\mathbf{y} \in Y^{UBP}$. Note that this implies that the number of variables in the upper and lower bounding problems grows as the discretized sets Y^{UBP} and Y^{LBP} grow (which they do from iteration to iteration).

5.2 Numerical examples

We look at two different examples in which the design constraints are implicitly defined; in one, $\mathbf{g}$ is defined by a dynamic system, and in the another $\mathbf{g}$ is defined by the solution of a system of algebraic equations. These numerical studies were performed on a 64-bit Linux (Ubuntu 14.04) laptop with a 2.53GHz Intel Core2 Duo processor (P9600) and 3.8 GB RAM. Any compiled code is compiled with GCC version 4.8.4 with the -O3 optimization flag. Some of the methods require GAMS; version 24.3.3 is used [21], which employs BARON version 14.0.3 [68, 52].

5.2.1 LLP defined by dynamic system

Consider the following robust design problem. In a batch chemical reactor, two chemical species A and B react according to mass-action kinetics with an Arrhenius dependence on temperature to form chemical species C. However, A and C also react according to mass-action kinetics with

a dependence on temperature to form chemical species D. The initial concentrations of A and B vary from batch to batch, although it can be assumed they are never outside the range $[0.5, 2]$ (M). Although temperature can be controlled by a cooling element, it too might vary from batch to batch; it can be assumed it never leaves the range $[300, 800]$ (K). What are the largest acceptable ranges for the initial concentrations of A and B and the operating temperature to ensure that the mole fraction of the undesired side product D is below 0.05 at the end of any batch operation?

As a mathematical program this problem is written as (since there is one LLP the subscript on g is dropped)

$$\begin{aligned} \sup_{\mathbf{y}^L, \mathbf{y}^U} \; & \sum_j \ln(y_j^U - y_j^L) \\ \text{s.t. } & g(\mathbf{y}) = \frac{z_{\mathrm{D}}(t_f, \mathbf{y})}{\mathbf{1}^{\mathrm{T}} \mathbf{z}(t_f, \mathbf{y})} - 0.05 \leq 0, \quad \forall \mathbf{y} \in [\mathbf{y}^L, \mathbf{y}^U], \\ & \mathbf{y}^U - \mathbf{y}^L \geq (0.01)\mathbf{1}, \\ & \mathbf{y}^L, \mathbf{y}^U \in [0.5, 2] \times [0.5, 2] \times [3, 8], \end{aligned} \tag{29}$$

where $\mathbf{z} = (z_{\mathrm{A}}, z_{\mathrm{B}}, z_{\mathrm{C}}, z_{\mathrm{D}})$ is a solution of the initial value problem in parametric ordinary differential equations

$$\begin{aligned} \dot{z}_{\mathrm{A}}(t, \mathbf{y}) &= -k_1(100y_3) z_{\mathrm{A}}(t, \mathbf{y}) z_{\mathrm{B}}(t, \mathbf{y}) - k_2(100y_3) z_{\mathrm{A}}(t, \mathbf{y}) z_{\mathrm{C}}(t, \mathbf{y}), \\ \dot{z}_{\mathrm{B}}(t, \mathbf{y}) &= -k_1(100y_3) z_{\mathrm{A}}(t, \mathbf{y}) z_{\mathrm{B}}(t, \mathbf{y}), \\ \dot{z}_{\mathrm{C}}(t, \mathbf{y}) &= k_1(100y_3) z_{\mathrm{A}}(t, \mathbf{y}) z_{\mathrm{B}}(t, \mathbf{y}) - k_2(100y_3) z_{\mathrm{A}}(t, \mathbf{y}) z_{\mathrm{C}}(t, \mathbf{y}), \\ \dot{z}_{\mathrm{D}}(t, \mathbf{y}) &= k_2(100y_3) z_{\mathrm{A}}(t, \mathbf{y}) z_{\mathrm{C}}(t, \mathbf{y}), \end{aligned}$$

on the time interval $[t_0, t_f]$, with initial conditions $\mathbf{z}(t_0, \mathbf{y}) = (y_1, y_2, 0, 0)$. See Table 3 for a summary of the parameter values used and the expressions for the kinetic parameters k_1 and k_2. Note that the variables y_1 and y_2 correspond to the initial concentrations of A and B, respectively, while y_3 is a scaled temperature. This scaling helps overcome some numerical issues. Further, note the use of the logarithm to "convexify" the objective, which requires the constraint that any candidate design space have a minimum size. The concave objective can help speed up some of the methods.

Table 3: Parameter values for problem (29).

Symbol	Value/Expression
$[t_0, t_f]$	$[0, 0.1]$ (h)
A_1	150 (M^{-1}h)
A_2	80 (M^{-1}h)
E_1	4×10^3 (J/mol)
E_2	15×10^3 (J/mol)
R	8.3145 (J/K $\cdot$ mol)
k_1	$k_1 : T \mapsto A_1 \exp(-E_1/(RT))$
k_2	$k_2 : T \mapsto A_2 \exp(-E_2/(RT))$

SIP restriction

First consider applying the SIP restriction method discussed in §5.1.2. Let

$$\begin{aligned} Y &= [0.5, 2] \times [0.5, 2] \times [3, 8], \\ X &= \{(\mathbf{y}^L, \mathbf{y}^U) \in Y \times Y : \mathbf{y}^U - \mathbf{y}^L \geq (0.01)\mathbf{1}\}, \\ M &= [\mathbf{0}, (10)\mathbf{1}], \end{aligned}$$

and

$$\mathbf{h} : X \times Y \ni (\mathbf{y}^L, \mathbf{y}^U, \mathbf{y}) \mapsto \begin{bmatrix} -\mathbf{I} \\ \mathbf{I} \end{bmatrix} \mathbf{y} + \begin{bmatrix} \mathbf{y}^L \\ -\mathbf{y}^U \end{bmatrix}.$$

Thus it is clear that X, Y, and M are compact, and that the objective and constraints of the SIP restriction (12) are continuous; the latter (specifically, the continuity of g) follows from standard parametric analysis of initial value problems from, for instance, Chapter II of [37]. The value of M is somewhat arbitrary; the SIP restriction is still valid for any M. If M is too large, there can be numerical issues and solving the subproblems can be slow, but as M becomes smaller, the restriction can become more conservative.

Consider the subproblems (LBP), (UBP), and (SIP LLP) that must be solved. The lower-level program of the SIP restriction for this example is a global dynamic optimization problem. This problem must be solved repeatedly in the course of the SIP solution algorithm as a test for feasibility. Although this may seem daunting, the computational time is reasonable. Here, the "direct method" approach of control parameterization (sometimes called single-shooting) is taken; see §1.1 of [54]. A C++ code implementing dynamic relaxations and branch-and-bound along the lines of [54, Ch. 2], [55, Ch. 10], is used to solve (SIP LLP) at the required values of $(\mathbf{x}, \boldsymbol{\mu})$.

Next consider (UBP) and (LBP). Although the function g appears in the constraints of these problems, this does not make them dynamically-constrained problems for this specific example. For this example, only the finite set of values $\{g(\mathbf{y}) : \mathbf{y} \in Y^{LBP} \cup Y^{UBP}\}$ is required at any iteration. These values can be obtained without too much extra effort in the course of solving the (SIP LLP) (since Y^{UBP} and Y^{LBP} are populated with the maximizers of the lower-level program). Meanwhile, $\mathbf{h}$ is a known, explicit function of $\mathbf{x}$ and $\mathbf{y}$, as are the objectives of (UBP) and (LBP), and overall their solution is no more arduous than in the cases considered in §4. The (UBP) and (LBP) are solved in GAMS with BARON. The user-defined extrinsic function capability of GAMS is used to communicate with the C++ code solving (SIP LLP).

Meanwhile, suppose that a ball-valued design space was being used in this example, and contrast this with the situation of trying to solve the SIP reformulation from Proposition 2. Since the SIP method from [39] holds for general SIP, one could attempt to solve an exact SIP reformulation similar to the form (2). However, the upper bounding problem for that reformulation would be

$$\begin{aligned} &\sup_{\mathbf{y}_c, \delta} \ \delta \\ &\text{s.t. } g(\mathbf{y}_c + \delta \mathbf{y}_d) \leq 0, \quad \forall \mathbf{y}_d \in \widetilde{B}_1^{UBP}, \\ &\quad (\mathbf{y}_c, \delta) \in X. \end{aligned}$$

The complication here is that the upper-level variables $(\mathbf{y}_c, \delta)$ are required to evaluate g; that is, in contrast to above, a finite set of values does not suffice, and in fact the upper and lower bounding problems become dynamic optimization problems in addition to the lower-level program. However, applying the solution method from [64] to the SIP reformulation from Proposition 2 might lead to a successful method for design centering problems when $\mathbf{g}$ is defined by the solution of a system of algebraic equations.

For the results in §5.2.1, the initial discretizations are $Y^{LBP,0} = Y^{UBP,0} = \varnothing$, the initial right-hand side restriction parameter is $\varepsilon^{g,0} = 1$, the right-hand side restriction parameter reduction factor is $r = 2$, and the subproblems (UBP), (LBP), and (SIP LLP) are solved with relative and absolute tolerances of 5×10^{-4} and 10^{-4}, respectively. (thus $\varepsilon_{atol} = 10^{-4}$ in (25) and the preceding discussion). We will not use any overall relative or absolute optimality termination criteria, but rather look at the best objective value for various solution times.

Generically valid solution

The same definitions as for the SIP restriction method are used for X, Y and $\mathbf{h}$.

Note that, similarly to (LBP) and (UBP), the subproblem (GSIP-LBP) (or its equivalent reformulation with binary variables (GSIP-LBP-B)) only depends on a finite list of values $\{g(\mathbf{y}) : \mathbf{y} \in Y^{LBP}\}$, and similarly for the upper bounding problem. Meanwhile, it is assumed we have the functional form of $\mathbf{h}$, and so the subproblems are mixed-integer nonlinear programs (MINLP), and they are solved to global optimality in GAMS with BARON. (LLP) and (LLP-AUX) are solved similarly to (SIP LLP), with a C++ code employing branch-and-bound.

Note that since $\mathbf{h}$ is linear in $\mathbf{x}$, the constraints of (GSIP-LBP-B) and (GSIP-UBP-B) are *linear*. Combined with the concave objective, these subproblems are mixed-integer convex programs and are fairly quick to solve, despite the fact that the number of variables and constraints grows with each iteration. This actually makes this method quite attractive for interval design centering.

Since the algorithm for solving SIP from [39] is similar to the algorithm for solving GSIP from [42], similar parameters are used as for the SIP restriction method; the initial discretizations are $Y^{LBP,0} = Y^{UBP,0} = \varnothing$, the initial right-hand side restriction parameter is $\varepsilon^{g,0} = 1$, the right-hand side restriction parameter reduction factor is $r = 2$, and the subproblems (GSIP-LBP), (GSIP-UBP), (LLP-AUX) and (LLP) are solved with relative and absolute tolerances of 5×10^{-4} and 10^{-4}, respectively. Additionally, we have $\alpha = 0.5$ in (LLP-AUX), and in (GSIP-LBP-B) and (GSIP-UBP-B) we use $g^{\max} = 11$ and $\bar{h}_j^{\max} = 9$ for each j. As above, we will not use any overall relative or absolute optimality termination criteria, but rather look at the best objective value for various solution times.

Interval restriction with branch and bound

To apply the interval restriction method from §5.1.1, Assumption 2 must be satisfied. The main challenge is defining the upper bound function g^U. In the robust design problem (29), g is defined in terms of the solution of an initial value problem in ordinary differential equations, and so the dynamic bounding method from [56] is used. This method provides interval bounds on each component of $\mathbf{z}(t_f, \cdot)$, the solution of the embedded differential equations. Combined with interval arithmetic, one obtains an inclusion monotonic interval extension (and thus an inclusion function) of g. Subsequently, the upper bound of this inclusion function (denote it g^U) satisfies the relevant parts of Assumption 2. A C++ implementation of the dynamic bounding method from [56] is used in conjunction with the interval arithmetic capabilities of MC++ [17, 40] to calculate the value of g^U.

The same C++ code implementing the branch-and-bound framework is used as in the previous section for solving (SIP LLP). The bounding scheme discussed in §5.1.1 is implemented with this branch-and-bound framework. The "convexified" objective does not help much with this bounding scheme. Thus the objective used is just $\prod_j (y_j^U - y_j^L)$. Again, we will not use any overall relative or absolute optimality termination criteria, but rather look at the best objective value for various solution times.

Table 4: Best solutions for various methods applied to (29).

Method	Solution		Objective Value
	$\mathbf{y}^L$	$\mathbf{y}^U$	
Interval restriction (§5.1.1)	$(0.51, 1.26, 3.04)$	$(0.92, 1.99, 4.80)$	0.52
SIP restriction (§5.1.2)	$(0.50, 0.50, 3.00)$	$(2.0, 2.0, 3.74)$	1.66
Generically valid solution (§5.1.3)	$(0.50, 0.50, 3.00)$	$(1.75, 2.0, 4.25)$	2.34

Careful inspection of the dynamic bounding method used to construct g^U reveals that the interval restriction method is in fact a restriction for the problem

$$\begin{aligned} \sup_{\mathbf{y}^L, \mathbf{y}^U} \ & \prod_j (y_j^U - y_j^L) \\ \text{s.t. } & \frac{z_{\mathrm{D}}(t_f, \mathbf{y})}{\mathbf{1}^{\mathrm{T}} \mathbf{z}(t_f, \mathbf{y})} - 0.05 \leq 0, \quad \forall (y_1, y_2, y_3) \in [y_1^L, y_1^U] \times [y_2^L, y_2^U] \times \mathcal{U}(y_3^L, y_3^U), \\ & \mathbf{y}^U - \mathbf{y}^L \geq (0.01)\mathbf{1}, \\ & \mathbf{y}^L, \mathbf{y}^U \in [0.5, 2] \times [0.5, 2] \times [3, 8], \\ & \mathcal{U}(y_3^L, y_3^U) = \left\{ u \in L^1([t_0, t_f], \mathbb{R}) : u(t) \in [y_3^L, y_3^U], a.e.\ t \in [t_0, t_f] \right\}. \end{aligned} \tag{30}$$

In words, one is looking for the largest acceptable ranges for the initial concentrations of A and B and range for the temperature *control profile* to ensure that the mole fraction of D is below the threshold. This is due to the fact that the bounding method from [56] indeed produces an interval $[\mathbf{z}^L(\mathbf{y}^L, \mathbf{y}^U), \mathbf{z}^U(\mathbf{y}^L, \mathbf{y}^U)]$ satisfying

$$\mathbf{z}(t_f, \mathbf{y}) \in [\mathbf{z}^L(\mathbf{y}^L, \mathbf{y}^U), \mathbf{z}^U(\mathbf{y}^L, \mathbf{y}^U)], \quad \forall \mathbf{y} \in [y_1^L, y_1^U] \times [y_2^L, y_2^U] \times \mathcal{U}(y_3^L, y_3^U).$$

Continuing the evaluation of g in interval arithmetic then guarantees that $g(\mathbf{y}) \leq g^U([\mathbf{y}^L, \mathbf{y}^U])$ for all $\mathbf{y} \in [y_1^L, y_1^U] \times [y_2^L, y_2^U] \times \mathcal{U}(y_3^L, y_3^U)$. Thus the restriction (24) still holds. Consequently, this restriction, when applied to dynamic problems, has ties to robust dynamic optimization problems, such as those considered in [31].

Results and discussion

The results of the three methods applied to the robust design problem (29) are summarized in Table 4 and Figure 1. Table 4 lists the best solutions for each method for the alotted solution time. In this case, each method was given approximately 10 minutes. But since each method is a feasible point method, it is interesting to see how the objective evolves over time, and in particular, with respect to a guaranteed upper bound which would give the optimality gap. This is summarized in Figure 1.

From Figure 1, it is clear that the generically valid method produces better feasible points for any solution time; indeed, all of its feasible points have an objective value greater than the upper bounds of the other methods. However, the best upper bound of the generically valid method is 3.5, and so there is a fair amount of gap left. Meanwhile, the SIP restriction method converges in about 70 seconds, when the upper-bounding problem (UBP) produces a feasible, and thus optimal (for the restriction) point.

Finally, the interval restriction method has an upper bound that is much lower than the best objectives of the other methods. It does make a fair amount of progress to closing the optimality

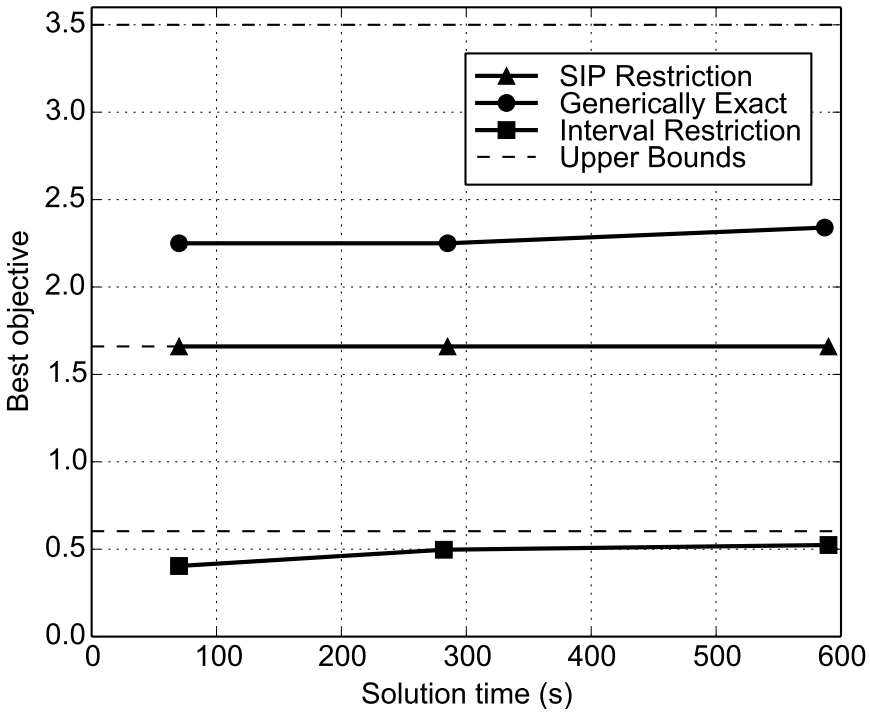
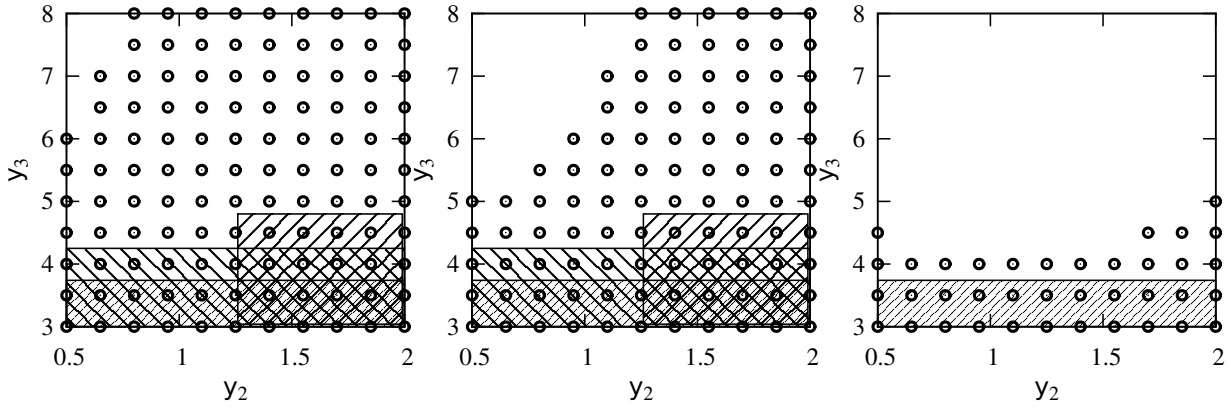


Figure 1: Best feasible objective value ($\prod_j (y_j^U - y_j^L)$) versus time for various methods applied to problem (29). The best upper bound for each method is plotted as a dotted line.



(a) $y_1 = 0.6$ (b) $y_1 = 0.9$ (c) $y_1 = 2$

Figure 2: Sampling of $G = \{\mathbf{y} \in Y : g(\mathbf{y}) \leq 0\}$ for various fixed values of y_1 for problem (29); points in G are marked with a circle. The various solutions from Table 4 are also plotted.

gap in the 10 minutes of solution time. However, one could argue that problem (30) is a harder problem than (29); problem (30) has a lower-level variable that takes values in a function space. If such a robust dynamic optimization problem is of interest, the interval restriction method provides an approach to obtaining a feasible point.

5.2.2 LLP defined by implicit function

We investigate the performance of the SIP restriction and the generically valid solution approaches on a system in which the design constraints are given by an implicit function solving a system of algebraic equations.

We consider steady-state operation of a continuously-stirred tank reactor with constant heat capacity of its contents. Reactant A flows in at a fixed concentration. The volume flow rate of this feed can be set in the range $[0.05, 2]$ (Ls^{-1}). In the reactor, A isomerizes to form species B, which in turn can isomerize to form species C. Both isomerization reactions are exothermic, so the reactor is jacketed and cooling water at a fixed temperature can be used to cool the reactor. The volume flow rate of the cooling water can be set in the range $[0.4, 2]$ (Ls^{-1}). What is the maximum acceptable ranges on the reactant feed flow rate and cooling water flow rate to ensure that the temperature of the reactor does not exceed 350 (K)?

First, the state of the system is given by the value of the vector $\mathbf{z}$; z_1 is the concentration of A; z_2 is the concentration of B; z_3 is the concentration of C; z_4 is the temperature inside the reactor; z_5 is the temperature of the cooling water flowing out of the jacketing; and z_6 is an artificial variable equalling the difference between z_4 and z_5. These variables satisfy the system of equations $\mathbf{0} = \mathbf{f}(\mathbf{p}, \mathbf{z})$, where p_1 is the volume flow of the feed and p_2 is the volume flow of the cooling water, and

$$
\begin{aligned}
f_1(\mathbf{p}, \mathbf{z}) &= 20p_1 - p_1 z_1 - V z_1 r_1(z_4), \\
f_2(\mathbf{p}, \mathbf{z}) &= -p_1 z_2 + V z_1 r_1(z_4) - V z_2 r_2(z_4), \\
f_3(\mathbf{p}, \mathbf{z}) &= -p_1 z_3 + V z_2 r_2(z_4), \\
f_4(\mathbf{p}, \mathbf{z}) &= 300p_1 - p_1 z_4 - V H_1 z_1 r_1(z_4) - V H_2 z_2 r_2(z_4) - p_2(z_5 - 300), \\
f_5(\mathbf{p}, \mathbf{z}) &= p_2(z_5 - 300) - 0.35 \frac{(z_5 - 300)}{\log\left({}^{(z_4 - 300)}/_{z_6}\right)}, \\
f_6(\mathbf{p}, \mathbf{z}) &= z_6 - (z_4 - z_5).
\end{aligned}
$$

See [35, §3] for a similar system. Other parameters and expressions are given in Table 5. We will assume that $\mathbf{z}$ takes values in the set $Z = [0, 20] \times [0, 20] \times [0, 20] \times [302, 400] \times [301, 399] \times [1, 200]$.

The function g defining the design constraint that temperature remain below a certain limit is then given by $g : (\mathbf{p}, \mathbf{z}) \mapsto z_4 - 350$, where it is implied that the domain of g is the set of $(\mathbf{p}, \mathbf{z})$ such that $\mathbf{f}(\mathbf{p}, \mathbf{z}) = \mathbf{0}$. Consequently, define $P = [0.05, 2] \times [0.4, 2]$ and

$$
Y = \{(\mathbf{p}, \mathbf{z}) \in P \times Z : \mathbf{f}(\mathbf{p}, \mathbf{z}) = \mathbf{0}\}.
$$

Table 5: Parameter values for problem (31).

Symbol	Value/Expression
V	5 (L)
H_1	-5 (LK(mol)$^{-1}$)
H_2	-5 (LK(mol)$^{-1}$)
A_1	2.7×10^8 (s^{-1})
A_2	160 (s^{-1})
r_1	$r_1 : T \mapsto A_1 \exp(-6000/T)$
r_2	$r_2 : T \mapsto A_2 \exp(-4500/T)$

We note Y is compact, since $\mathbf{f}$ is continuous and Y is defined in terms of a preimage of $\mathbf{f}$.

Then, as a mathematical program, the robust design problem is formulated as

$$\begin{aligned} \sup_{\mathbf{p}^L, \mathbf{p}^U} \ & \sum_j \ln(p_j^U - p_j^L) \\ \text{s.t. } & g(\mathbf{p}, \mathbf{z}) \leq 0, \forall (\mathbf{p}, \mathbf{z}) \in Y : \mathbf{p} \in [\mathbf{p}^L, \mathbf{p}^U], \\ & \mathbf{p}^U - \mathbf{p}^L \geq (0.01)\mathbf{1}, \\ & (\mathbf{p}^L, \mathbf{p}^U) \in P \times P. \end{aligned} \tag{31}$$

It might not be immediately apparent that this can be put into the same form as (GSIP); however, take

$$\begin{aligned} &\mathbf{x} = (\mathbf{p}^L, \mathbf{p}^U), \\ &X = \{(\mathbf{p}^L, \mathbf{p}^U) \in P \times P : \mathbf{p}^U - \mathbf{p}^L \geq (0.01)\mathbf{1}\}, \\ &D : (\mathbf{p}^L, \mathbf{p}^U) \mapsto \{(\mathbf{p}, \mathbf{z}) \in Y : \mathbf{p} \in [\mathbf{p}^L, \mathbf{p}^U]\}. \end{aligned}$$

We note that the main difference is that only a subset of the lower-level variables (namely, $\mathbf{p}$) are considered in the design space.

The SIP restriction and generically valid methods discussed in Sections 5.1.2 and 5.1.3 apply to the present problem. Define

$$\mathbf{h} : (\mathbf{p}^L, \mathbf{p}^U, \mathbf{p}) \mapsto \begin{bmatrix} -\mathbf{I} \\ \mathbf{I} \end{bmatrix} \mathbf{p} + \begin{bmatrix} \mathbf{p}^L \\ -\mathbf{p}^U \end{bmatrix}.$$

Given the form of Y, (SIP LLP), for instance, becomes

$$\begin{aligned} \sup_{\mathbf{p}, \mathbf{z}} \ & g(\mathbf{p}, \mathbf{z}) - \boldsymbol{\mu}^{\mathrm{T}} \mathbf{h}(\mathbf{p}^L, \mathbf{p}^U, \mathbf{p}) \\ \text{s.t. } & \mathbf{f}(\mathbf{p}, \mathbf{z}) = \mathbf{0}, \\ & (\mathbf{p}, \mathbf{z}) \in P \times Z. \end{aligned}$$

Since Y^{UBP} in subproblem (UBP) is populated with optimal solutions of (SIP LLP), we must have $Y^{UBP} \subset Y$. Thus no changes are required of (LBP) and (UBP). Similar reasoning holds for the subproblems (GSIP-UBP) and (GSIP-LBP) in the generically valid method.

For both methods, all necessary subproblems are solved with BARON in GAMS. For the SIP restriction, take $M = [\mathbf{0}, (100)\mathbf{1}]$. The algorithmic parameters in both cases are similar to before; the

initial discretizations are $Y^{LBP,0} = Y^{UBP,0} = \varnothing$, the initial right-hand side restriction parameter is $\varepsilon^{g,0} = 1$, the right-hand side restriction parameter reduction factor is $r = 2$, and the overall relative and absolute optimality tolerances of 0.05 and 0.01, respectively, while the subproblems (UBP), (LBP), (SIP LLP), (GSIP-UBP), (GSIP-LBP), (LLP-AUX) and (LLP) are solved with relative and absolute tolerances of 5×10^{-5} and 10^{-5}, respectively. Additionally, we have $\alpha = 0.5$ in (LLP-AUX), and in (GSIP-LBP-B) and (GSIP-UBP-B) we use $g^{\max} = 1.393$ and $\bar{h}_j^{\max} = 4.55$ for each j.

The SIP restriction method requires 13 iterations, totalling 680 seconds, converging to the solution $[0.05, 0.23] \times [0.4, 2.0]$ with objective 0.290. The generically valid method requires 146 iterations, totalling 290 seconds, converging to the solution $[0.05, 0.25] \times [0.53, 2.0]$ with objective 0.295. Both methods perform comparably, although the generically valid method is certainly better. It is interesting to note that the SIP restriction method requires much fewer iterations, with a much more expensive lower-level program (solution of (SIP LLP) requires over 400 seconds of the total solution time). Meanwhile, the generically valid method requires many more iterations, but which are much cheaper (for instance, the total solution time of (LLP) and (LLP-AUX) is only 30 seconds).

6 Conclusions and future work

This work has discussed a number of approaches to solving design centering problems, motivated by the specific instance of robust design in engineering applications. Reformulations to simpler problems were reviewed; many of these are inspired by duality-based reformulations from the GSIP literature. Two approaches for determining a feasible solution of a design centering problem were discussed and applied to an engineering application of robust design. The two methods are successful, with the SIP restriction-based approach (§5.1.2) performing better for this example.

One aspect of the restriction-based approaches in §5 that was not considered was the case of multiple infinite constraints (i.e. multiple LLPs). The interval restriction (§5.1.1) can likely handle multiple constraints in practice without much numerical difficulty. Meanwhile, [39] mentions a potential way to extend the basic SIP algorithm; this simply depends on using separate discretization sets and restriction parameters for each LLP. Unfortunately, numerical experiments show that such an extension to the method performs poorly for design centering problems. The upper-bounding problem (UBP) has trouble finding a feasible point, and so the algorithm is slow to converge. Extending the SIP method from [39] and the GSIP method [42] merits further investigation.

A Convergence of Bounding Method from §5.1.1

Let $[\mathbf{v}^U, \mathbf{w}^L] \subset [\mathbf{v}^L, \mathbf{w}^U] \subset [\bar{\mathbf{y}}^L, \bar{\mathbf{y}}^U] \subset \mathbb{R}^{n_y}$. Let

$$Y_j^v = [v_1^L, w_1^U] \times \cdots \times [v_{j-1}^L, w_{j-1}^U] \times [v_j^L, v_j^U] \times [v_{j+1}^L, w_{j+1}^U] \times \cdots \times [v_{n_y}^L, w_{n_y}^U],$$
$$Y_j^w = [v_1^L, w_1^U] \times \cdots \times [v_{j-1}^L, w_{j-1}^U] \times [w_j^L, w_j^U] \times [v_{j+1}^L, w_{j+1}^U] \times \cdots \times [v_{n_y}^L, w_{n_y}^U],$$

for each $j \in \{1, \ldots, n_y\}$. Then it is easy to see that the "outer" interval $[\mathbf{v}^L, \mathbf{w}^U]$ is a subset of $[\mathbf{v}^U, \mathbf{w}^L] \cup \left(\bigcup_j (Y_j^v \cup Y_j^w)\right)$; for any $\mathbf{y} \in [\mathbf{v}^L, \mathbf{w}^U]$, each component y_j lies in one of $[v_j^L, v_j^U]$, $[v_j^U, w_j^L]$, or $[w_j^L, w_j^U]$, which is included in the definition of one of Y_j^v, $[\mathbf{v}^U, \mathbf{w}^L]$, or Y_j^w.

Thus

$$\begin{aligned}\operatorname{vol}([\mathbf{v}^L, \mathbf{w}^U]) &\leq \operatorname{vol}([\mathbf{v}^U, \mathbf{w}^L]) + \sum_j \operatorname{vol}(Y_j^v) + \operatorname{vol}(Y_j^w) \\ &= \operatorname{vol}([\mathbf{v}^U, \mathbf{w}^L]) + \sum_j ((v_j^U - v_j^L) + (w_j^U - w_j^L)) \prod_{k \neq j} (w_k^U - v_k^L).\end{aligned}$$

Let $\alpha = \operatorname{diam}([\bar{\mathbf{y}}^L, \bar{\mathbf{y}}^U])$. Then $(w_k^U - v_k^L) \leq \alpha$ for each k. If $X' = [\mathbf{v}^L, \mathbf{v}^U] \times [\mathbf{w}^L, \mathbf{w}^U]$, then $(v_j^U - v_j^L) + (w_j^U - w_j^L) \leq 2 \operatorname{diam}(X')$ for each j. Putting all these inequalities together one obtains

$$\operatorname{vol}([\mathbf{v}^L, \mathbf{w}^U]) - \operatorname{vol}([\mathbf{v}^U, \mathbf{w}^L]) \leq 2 n_y \alpha^{n_y - 1} \operatorname{diam}(X').$$

Thus assuming $[\mathbf{y}^L, \mathbf{y}^U] \subset [\bar{\mathbf{y}}^L, \bar{\mathbf{y}}^U]$ for all $(\mathbf{y}^L, \mathbf{y}^U) \in X$, this establishes that the bounding method described in §5.1.1 is at least first-order convergent.

References

[1] MOSEK. `http://www.mosek.com/`, 2015.

[2] SeDuMi: Optimization over symmetric cones. `http://sedumi.ie.lehigh.edu/`, 2015.

[3] YALMIP. `http://users.isy.liu.se/johanl/yalmip/`, 2015.

[4] H. L. Abdel-Malek and A. K. S. O. Hassan. The ellipsoidal technique for design centering and region approximation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 10(8):1006–1014, Aug 1991.

[5] Jean-Pierre Aubin and Helene Frankowska. *Set-Valued Analysis*. Birkhauser, Boston, 1990.

[6] B. Bank, J. Guddat, Diethard Klatte, Bernd Kummer, and K. Tammer. *Non-Linear Parametric Optimization*. Birkhauser, Boston, 1983.

[7] Alberto Bemporad, Carlo Filippi, and Fabio D. Torrisi. Inner and outer approximations of polytopes using boxes. *Computational Geometry*, 27(2):151–178, February 2004.

[8] A. Ben-Tal and Arkadii Nemirovski. Robust Convex Optimization. *Mathematics of Operations Research*, 23(4):769–805, 1998.

[9] A. Ben-Tal and Arkadii Nemirovski. Robust solutions of uncertain linear programs. *Operations Research Letters*, 25(1):1–13, August 1999.

[10] Aharon Ben-Tal and Arkadii Nemirovski. *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*. SIAM, Philadelphia, 2001.

[11] Dimitri P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, Massachusetts, second edition, 1999.

[12] Dimitri P. Bertsekas. *Convex Optimization Theory*. Athena Scientific, Belmont, Massachusetts, 2009.

[13] Binita Bhattacharjee, William H. Green, and Paul I. Barton. Interval Methods for Semi-Infinite Programs. *Computational Optimization and Applications*, 30(1):63–93, January 2005.

[14] Binita Bhattacharjee, Panayiotis Lemonidis, William H. Green, and Paul I. Barton. Global solution of semi-infinite programs. *Mathematical Programming, Series B*, 103:283–307, 2005.

[15] Stephen Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*. SIAM, Philadelphia, 1994.

[16] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, 2004.

[17] Benoît Chachuat. MC++: A Versatile Library for McCormick Relaxations and Taylor Models. `http://www.imperial.ac.uk/people/b.chachuat/research.html`, 2015.

[18] M. Diehl, B. Houska, Oliver Stein, and P. Steuermann. A lifting method for generalized semi-infinite programs based on lower level Wolfe duality. *Computational Optimization and Applications*, 54(1):189–210, June 2013.

[19] Kaisheng Du and R. Baker Kearfott. The cluster problem in multivariate global optimization. *Journal of Global Optimization*, 5(3):253–265, 1994.

[20] Christodoulos A. Floudas, Zeynep H. Gümüş, and Marianthi G. Ierapetritou. Global Optimization in Design under Uncertainty: Feasibility Test and Flexibility Index Problems. *Industrial & Engineering Chemistry Research*, 40:4267–4282, 2001.

[21] GAMS Development Corporation. GAMS: General Algebraic Modeling System. `http://www.gams.com`, 2014.

[22] A. M. Geoffrion. Duality in Nonlinear Programming: A Simplified Applications-Oriented Development. *SIAM Review*, 13(1):1–37, January 1971.

[23] Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, Stephen Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008.

[24] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, March 2014.

[25] Harald Günzel, Hubertus Th. Jongen, and Oliver Stein. On the closure of the feasible set in generalized semi-infinite programming. *Central European Journal of Operations Research*, 15(3):271–280, 2007.

[26] Harald Günzel, Hubertus Th. Jongen, and Oliver Stein. Generalized semi-infinite programming: on generic local minimizers. *Journal of Global Optimization*, 42(3):413–421, 2008.

[27] K. P. Halemane and Ignacio E. Grossmann. Optimal Process Design Under Uncertainty. *AIChE Journal*, 29(3):425–433, 1983.

[28] Stuart M. Harwood and Paul I. Barton. Lower level duality and the global solution of generalized semi-infinite programs. *Optimization*, 65(6):1129–1149, 2016.

[29] Eligius M.T. Hendrix, Carmen J. Mecking, and Theo H.B. Hendriks. Finding robust solutions for product design problems. *European Journal of Operational Research*, 92(1):28–36, July 1996.

[30] R. Hettich and K. O. Kortanek. Semi-infinite programming: theory, methods, and applications. *SIAM Review*, 35(3):380–429, 1993.

[31] Boris Houska, Filip Logist, Jan Van Impe, and Moritz Diehl. Robust optimization of nonlinear dynamic systems with application to a jacketed tubular reactor. *Journal of Process Control*, 22(6):1152–1160, 2012.

[32] N. Kanzi and S. Nobakhtian. Necessary optimality conditions for nonsmooth generalized semi-infinite programming problems. *European Journal of Operational Research*, 205(2):253–261, September 2010.

[33] Leonid G. Khachiyan and Michael J. Todd. On the complexity of approximating the maximal inscribed ellipsoid for a polytope. *Mathematical Programming*, 61(1-3):137–159, 1993.

[34] Diethard Klatte and Bernd Kummer. Stability properties of infima and optimal solutions of parametric optimization problems. *Lecture Notes in Economics and Mathematical Systems*, 255:215–229, 1985.

[35] C. Loeblein and J.D. Perkins. Economic analysis of different structures of on-line process optimization systems. *Computers & chemical engineering*, 22(9):1257–1269, 1998.

[36] J. Lofberg. YALMIP: A toolbox for modeling and optimization in MATLAB. *2004 IEEE International Conference on Computer Aided Control Systems Design*, pages 284–289, 2004.

[37] Robert Mattheij and Jaap Molenaar. *Ordinary Differential Equations in Theory and Practice*. SIAM, Philadelphia, 2002.

[38] Ruth Misener and Christodoulos A. Floudas. ANTIGONE: Algorithms for coNTinuous / Integer Global Optimization of Nonlinear Equations. *Journal of Global Optimization*, 59(2-3):503–526, 2014.

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

[40] Alexander Mitsos, Benoît Chachuat, and Paul I. Barton. McCormick-based relaxations of algorithms. *SIAM Journal on Optimization*, 20(2):573–601, 2009.

[41] Alexander Mitsos, Panayiotis Lemonidis, and Paul I. Barton. Global solution of bilevel programs with a nonconvex inner program. *Journal of Global Optimization*, 42(4):475–513, December 2008.

[42] Alexander Mitsos and Angelos Tsoukalas. Global optimization of generalized semi-infinite programs via restriction of the right hand side. *Journal of Global Optimization*, 61(1):1–17, 2014.

[43] Ramon E. Moore, R. Baker Kearfott, and Michael J. Cloud. *Introduction to Interval Analysis*. SIAM, Philadelphia, 2009.

[44] Yurii Nesterov and Arkadii Nemirovski. *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM, Philadelphia, 1994.

[45] V. Hien Nguyen and Jean-Jacques Strodiot. Computing a global optimal solution to a design centering problem. *Mathematical Programming*, 53(1-3):111–123, January 1992.

[46] A. Parkinson, C. Sorensen, and N. Pourhassan. A general approach for robust optimal design. *Journal of Mechanical Design*, 115(1):74–80, 1993.

[47] Elijah Polak. On the mathematical foundations of nondifferentiable optimization in engineering design. *SIAM Review*, 29(1):21–89, 1987.

[48] Imre Polik and Tamas Terlaky. A Survey of the S-Lemma. *SIAM Review*, 49(3):371–418, 2007.

[49] Daniel Ralph and S. Dempe. Directional derivatives of the solution of a parametric nonlinear program. *Mathematical Programming*, 70:159–172, 1995.

[50] Claudio M. Rocco, José Al Moreno, and Néstor Carrasquero. Robust design using a hybrid-cellular-evolutionary and interval-arithmetic approach: a reliability application. *Reliability Engineering & System Safety*, 79(2):149–159, February 2003.

[51] Jan-Joachim Rückmann and Alexander Shapiro. First-Order Optimality Conditions in Generalized Semi-Infinite Programming. *Journal of Optimization Theory and Applications*, 101(3):677–691, 1999.

[52] N. V. Sahinidis. BARON 14.0.3: Global Optimization of Mixed-Integer Nonlinear Programs, User's Manual. `http://www.minlp.com/downloads/docs/baronmanual.pdf`, 2014.

[53] Daniel E. Salazar A. and Claudio M. Rocco S. Solving advanced multi-objective robust designs by means of multiple objective evolutionary algorithms (MOEA): A reliability application. *Reliability Engineering & System Safety*, 92(6):697–706, June 2007.

[54] Spencer D. Schaber. *Tools for dynamic model development.* PhD thesis, Massachusetts Institute of Technology, 2014.

[55] Joseph K. Scott. *Reachability Analysis and Deterministic Global Optimization of Differential-Algebraic Systems.* PhD thesis, Massachusetts Institute of Technology, 2012.

[56] Joseph K. Scott and Paul I. Barton. Bounds on the reachable sets of nonlinear control systems. *Automatica*, 49(1):93–100, 2013.

[57] Abbas Seifi, K. Ponnambalam, and Jiri Vlach. A unified approach to statistical design centering of integrated circuits with correlated parameters. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 46(1):190–196, 1999.

[58] Oliver Stein. A semi-infinite approach to design centering. In S. Dempe and V. Kalashnikov, editors, *Optimization with Mulitvalued Mappings*, chapter 1, pages 209–228. Springer, 2006.

[59] Oliver Stein. How to solve a semi-infinite optimization problem. *European Journal of Operational Research*, 223(2):312–320, June 2012.

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

[61] Oliver Stein and Anton Winterfeld. Feasible Method for Generalized Semi-Infinite Programming. *Journal of Optimization Theory and Applications*, 146(2):419–443, March 2010.

[62] Georg Still. Generalized semi-infinite programming : Theory and methods. *European Journal of Operational Research*, 119:301–313, 1999.

[63] Matthew D. Stuber and Paul I. Barton. Robust simulation and design using semi-infinite programs with implicit functions. *International Journal of Reliability and Safety*, 5(3-4):378–397, 2011.

[64] Matthew D. Stuber and Paul I. Barton. Semi-Infinite Optimization with Implicit Functions. *Industrial & Engineering Chemistry Research*, 54(5):307–317, 2015.

[65] Jos F. Sturm. Using SeDuMi 1.02, A Matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(1-4):625–653, 1999.

[66] R. E. Swaney and Ignacio E. Grossmann. An Index for Operational Flexibility in Chemical Process Design - Part I: Formulation and Theory. *AIChE Journal*, 31(4):621–630, 1985.

[67] R. E. Swaney and Ignacio E. Grossmann. An Index for Operational Flexibility in Chemical Process Design - Part II: Computational Algorithms. *AIChE Journal*, 31(4):631–641, 1985.

[68] M. Tawarmalani and N. V. Sahinidis. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, 103(2):225–249, 2005.

[69] Angelos Tsoukalas, Berç Rustem, and Efstratios N. Pistikopoulos. A global optimization algorithm for generalized semi-infinite, continuous minimax with coupled constraints and bi-level problems. *Journal of Global Optimization*, 44(2):235–250, July 2009.

[70] Achim Wechsung. *Global optimization in reduced space*. PhD thesis, Massachusetts Institute of Technology, 2013.

[71] Achim Wechsung, Spencer D. Schaber, and Paul I. Barton. The cluster problem revisited. *Journal of Global Optimization*, 58(3):429–438, 2014.

[72] Anton Winterfeld. Application of general semi-infinite programming to lapidary cutting problems. *European Journal of Operational Research*, 191(3):838–854, December 2008.