Global Optimization in Generalized Geometric Programming

Costas D. Maranas[1] and Christodoulos A. Floudas[2]
Department of Chemical Engineering,
Princeton University,
Princeton, NJ 08544-5263
January 1996

Abstract

A deterministic global optimization algorithm is proposed for locating the global minimum of generalized geometric (signomial) problems **(GGP)**. By utilizing an exponential variable transformation the initial nonconvex problem **(GGP)** is reduced to a **(DC)** programming problem where both the constraints and the objective are decomposed into the difference of two convex functions. A convex relaxation of problem **(DC)** is then obtained based on the linear lower bounding of the concave parts of the objective function and constraints inside some box region. The proposed branch and bound type algorithm attains finite ϵ–convergence to the global minimum through the successive refinement of a convex relaxation of the feasible region and/or of the objective function and the subsequent solution of a series of nonlinear convex optimization problems. The efficiency of the proposed approach is enhanced by eliminating variables through monotonicity analysis, by maintaining tightly bound variables through rescaling, by further improving the supplied variable bounds through convex minimization, and finally by transforming each inequality constraint so as the concave part lower bounding is as tight as possible. The proposed approach is illustrated with a large number of test examples and robust stability analysis problems.

Keywords: global optimization, generalized geometric programming, signomials, robust stability analysis.

1 Introduction

Generalized geometric or signomial programming **(GGP)** problems are characterized by an objective function and constraints which are the difference of two *posynomials*. A posynomial $G(\mathbf{x})$ is simply the sum of a number of *posynomial terms* or *monomials*

[1]Current Address: Department of Chemical Engineering, The Pennsylvania State University, University Park, PA 16802.

[2]Author to whom all correspondence should be addressed.

$g_k(\mathbf{x}),\ k = 1, \ldots, K$ multiplied by some positive real constants $c_k,\ k = 1, \ldots, K$. Each monomial $g_k(\mathbf{x})$ is in turn the product of a number of positive variables each of them raised to some real power,

$$g_k(\mathbf{x}) = x_1^{d_{1,k}} x_2^{d_{2,k}} \cdots x_n^{d_{N,k}}, \quad k = 1, \ldots, K$$

where $d_{1,k}, d_{2,k}, \ldots, d_{N,k} \in \Re$ and are not necessarily integers. The term geometric programming was adopted because of the key role that the well known arithmetic–geometric inequality played in the initial developments. Generalized geometric problems were first introduced and studied by Passy and Wilde [1] and Blau and Wilde [2] when existing (posynomial) geometric programming **(GP)** formulations failed to account for the presence of negatively signed monomials in models for important engineering applications. These applications are extensively reviewed in [3] and [4]. Chemical engineering applications include heat exchanger network design [5], chemical reactor design [2, 6], optimal condenser design [7], oxygen production [8], membrane separation process design [27], optimal design of cooling towers [4], chemical equilibrium problems [10], optimal control [11], batch plant modeling [12, 13], optimal location of hydrogen supply centers [14] and many more.

By grouping together monomials with identical sign, the generalized geometric **(GGP)** problem can be formulated as the following nonlinear optimization problem:

$$\begin{array}{lll} \min\limits_{\mathbf{t}} & G_0(\mathbf{t}) = G_0^+(\mathbf{t}) - G_0^-(\mathbf{t}) & \\ \text{subject to} & G_j(\mathbf{t}) = G_j^+(\mathbf{t}) - G_j^-(\mathbf{t}) \leq 0, \quad j = 1, \ldots, M & \quad (\mathbf{GGP}) \\ & t_i \geq 0, \quad i = 1, \ldots, N & \\ \text{where} & G_j^+(\mathbf{t}) = \sum\limits_{k \in K_j^+} c_{jk} \prod\limits_{i=1}^{N} t_i^{\alpha_{ijk}}, \quad j = 0, \ldots, M & \\ & G_j^-(\mathbf{t}) = \sum\limits_{k \in K_j^-} c_{jk} \prod\limits_{i=1}^{N} t_i^{\alpha_{ijk}}, \quad j = 0, \ldots, M & \end{array}$$

where $\mathbf{t} = (t_1, \ldots, t_N)$ is the positive variable vector; $G_j^+, G_j^-,\ j = 0, \ldots, M$ are positive posynomial functions in $\mathbf{t}$; α_{ijk} are arbitrary real constant exponents; and c_{jk} are *positive* coefficients. Also, sets K_j^+, K_j^- count how many positively/negatively signed monomials form posynomials G_j^+, G_j^- respectively. In general, formulation **(GGP)** corresponds to a nonlinear optimization problem with nonconvex objective function and/or constraint set. Note that if we set $K_j^- = 0$ for all $j = 0, \ldots, M$ then the mathematical model for **(GGP)** reduces to the (posynomial) geometric programming **(GP)** formulation which laid the foundation for the theory of generalized geometric **(GGP)** problems.

Unlike (posynomial) **(GP)** problems, **(GGP)** problems remain nonconvex in both their primal and dual representation and no known transformation can convexify them. They may involve multiple local minima and/or nonconvex feasible regions and therefore are much more difficult problems to solve. Local optimization approaches for solving **GGP** problems include bounding procedures based on *posynomial condensation* [15, 16, 18, 40, 11]; iterative solution of **KKT** conditions [6, 19, 20]; and adaptations of general purpose nonlinear programming methods [21, 3, 22, 23, 24, 25, 26]. A computational comparison of available codes for signomial programming is given in [27, 20]. While local optimization methods for solving **GGP** problems are ubiquitous, application of specialized global optimization algorithms on **GGP** problems is scarce. Existing global optimization algorithms such as **GOP** [28, 29, 30, 31] and α**BB** [32, 33, 34, 35] are not currently designed so as to handle efficiently the special structure of signomial terms. Falk [36] proposed such a global optimization algorithm based on the exponential variable transformation of **(GGP)** and the convex relaxation and branch and bounding on the space of exponents of negative monomials ($j = 1, \ldots, M$ and $k \in K_j^-$). In this paper, (i) an alternative partitioning in the typically smaller space of variables $i = 1, \ldots, N$ is investigated; (ii) a number of features aimed at improving the efficiency of the branch and bound procedure are discussed; (iii) the convergence properties of the proposed approach are analyzed; (iv) and a number of example problems in the areas of engineering design and stability analysis are considered.

2 Analysis

2.1 Difference of two Convex Functions Transformation

The objective function and constraints in the original formulation **GGP** are in general nonconvex functions. Based on an eigenvalue analysis, it is quite straightforward to show that the Hessian matrices of these nonlinear functions may involve eigenvalues of either sign implying that in general they are neither convex nor concave. However, by applying the transformation [15],

$$t_i \;=\; \exp z_i, \quad i = 1, \ldots, N$$

to the original formulation **(GGP)** we obtain the following programming problem **(DC)**.

$$\begin{aligned}
\min_{\mathbf{z}} \quad & G_0(\mathbf{z}) \;=\; G_0^+(\mathbf{z}) \;-\; G_0^-(\mathbf{z}) \\
\text{subject to} \quad & G_j(\mathbf{z}) \;=\; G_j^+(\mathbf{z}) \;-\; G_j^-(\mathbf{z}) \;\le\; 0, \quad j = 1, \ldots, M \qquad \textbf{(DC)} \\
& z_i^L \;\le\; z_i \;\le\; z_i^U, \quad i = 1, \ldots, N \\
\text{where} \quad & G_j^+(\mathbf{z}) \;=\; \sum_{k \in K_j^+} c_{jk} \exp\left\{ \sum_{i=1}^{N} \alpha_{ijk}\, z_i \right\}, \quad j = 0, \ldots, M
\end{aligned}$$

$$G_j^-(\mathbf{z}) \;=\; \sum_{k\in K_j^-} c_{jk}\exp\left\{\sum_{i=1}^{N} \alpha_{ijk}\, z_i\right\}, \;\; j=0,\ldots,M$$

Because the the exponentiation of a linear expression is convex, both the objective function and constraints of formulation **(DC)** are the difference of two convex functions. Also, since $z_i^L = \log(t_i^L)$ it is necessary that the lower bounds t_i^L of all t_i must be strictly positive for such a reformulation to exist. This problem can be overcome by appropriately scaling all problematic original variables t_i so as

$$t_i' = t_i + \max\left\{0, -t_i^L + \epsilon\right\}, \;\; \epsilon > 0.$$

Note that if $G_j^-(\mathbf{z}) \;=\; 0$ for every $j = 0,\ldots,M$ then problem **(DC)** becomes a convex programming problem.

2.2 Lower Bounding

A lower bound on the solution of problem **(DC)** can be obtained by solving a convex relaxation of **(DC)**. Such a convex relaxation can be realized by underestimating every concave function, $-G_j^-(\mathbf{z})$ with a linear function $-L_j^-(\mathbf{z})$ for every $j = 0,\ldots,M$. This linear function is constructed by underestimating every implicitly separable term $-\exp\left\{\sum_{i=1}^{N} \alpha_{ijk}\, z_i\right\}$ with a linear function. This defines the following relaxed convex programming problem **(R)** whose solution provides a lower bound on the solution of **(DC)**.

$$\begin{array}{lll}
\min\limits_{\mathbf{z}} & G_0^{conv}(\mathbf{z}) \;=\; G_0^+(\mathbf{z}) \;-\; L_0^-(\mathbf{z}) & \\[2ex]
\text{subject to} & G_j^{conv}(\mathbf{z}) \;=\; G_j^+(\mathbf{z}) \;-\; L_j^-(\mathbf{z}) \;\le\; 0, \quad j=1,\ldots,M & \textbf{(R)} \\[1ex]
 & z_i^L \;\le\; z_i \;\le\; z_i^U, \quad i=1,\ldots,N & \\[2ex]
\text{where} & G_j^+(\mathbf{z}) \;=\; \sum\limits_{k\in K_j^+} c_{jk}\exp\left\{\sum\limits_{i=1}^{N} \alpha_{ijk}\, z_i\right\}, \;\; j=0,\ldots,M & \\[2ex]
 & L_j^-(\mathbf{z}) \;=\; \sum\limits_{k\in K_j^-} c_{jk}\left\{A_{jk} + B_{jk}\left(\sum\limits_{i=1}^{N}\alpha_{ijk} z_i\right)\right\}, \;\; j=0,\ldots,M, \;\; k\in K_j^- & \\[2ex]
\text{and} & A_{jk} \;=\; \dfrac{Y_{jk}^U\exp(Y_{jk}^L) - Y_{jk}^L\exp(Y_{jk}^U)}{Y_{jk}^U - Y_{jk}^L}, \;\; B_{jk} \;=\; \dfrac{\exp(Y_{jk}^U) - \exp(Y_{jk}^L)}{Y_{jk}^U - Y_{jk}^L}, & \\[2ex]
 & Y_{jk}^L \;=\; \sum\limits_{i=1}^{N}\min\left(\alpha_{ijk} z_i^L, \alpha_{ijk} z_i^U\right), \;\; Y_{jk}^U \;=\; \sum\limits_{i=1}^{N}\max\left(\alpha_{ijk} z_i^L, \alpha_{ijk} z_i^U\right) &
\end{array}$$

Note that the linear underestimator of $-G_j^-$, $-L_j^-(\mathbf{z})$, is composed by the sum of a number of linear functions, each one of which is a lower bound on an implicitly univariate function of the form $-\exp(Y)$. Clearly, the smaller the difference between the original functions $G_j^-(\mathbf{z})$ and the linearizations $L_j^-(\mathbf{z})$ the closer the solution of **(R)** will be to the solution of **(DC)**. The quality of this lower bounding can be analyzed by examining the tightness of underestimation of every concave term of the form $-\exp(Y)$ with a linear function inside some interval $[Y^L, Y^U]$. Let $\Delta(Y)$ be the separation between the concave function $-\exp(Y)$ and its linear underestimator inside the interval $[Y^L, Y^U]$. $\Delta(Y)$ is concave in Y and it reaches its single maximum at

$$Y^* \;=\; \log\left(\frac{\exp(Y_{jk}^U) - \exp(Y_{jk}^L)}{Y_{jk}^U - Y_{jk}^L}\right)$$

with a value

$$\Delta_{max} \;=\; \exp(Y^L)\,(1 - Z + Z\log(Z))$$

$$\text{where } Z \;=\; \frac{\exp(\delta) - 1}{\delta},\;\; \delta \;=\; Y^U - Y^L.$$

Note that as the interval width $\delta = Y^U - Y^L$ goes to zero, Z approaches one and therefore the maximum separation goes to zero as well, $\delta \longrightarrow 0$, $Z \longrightarrow 1$, and $\Delta_{max} \longrightarrow 0$. The rate at which this maximum separation goes to zero can be determined by the Taylor expansion of $\Delta_{max}(\delta)$ at $\delta = 0$.

$$\frac{\Delta_{max}}{\exp(Y^L)} \;=\; \frac{\delta^2}{8} + \frac{\delta^3}{16} + \frac{11\delta^4}{576} + \frac{5\delta^5}{1152} + \frac{41\delta^6}{51840} + \frac{5\delta^7}{41472} + \mathcal{O}(\delta^8)$$

By considering only the first leading term of the positive series expansion we deduce that the rate at which Δ_{max} approaches zero as δ goes to zero is $\Delta_{max} \approx \mathcal{O}(\delta^2)$. On the other hand, as δ goes to infinity, Δ_{max} goes to infinity as $\Delta_{max} \approx \mathcal{O}(\exp(\delta))$. The scaled maximum separation $\Delta_{max}/\exp(Y^L)$ for different values of δ is shown in Table 1. Note that for small values of $\delta <\approx 0.5$ the maximum separation Δ_{max} goes as $\mathcal{O}(\delta^2)$ following the theoretically derived predictions. Therefore, if for example a maximum separation of 10^{-4} is required an interval width of just 10^{-2} suffices. On the other hand, for larger values of $\delta >\approx 2$ the maximum separation grows exponentially with δ rendering this underestimating scheme inefficient.

2.3 Variable Scaling

To maintain a tight convex underestimation of the concave terms it is important to keep the ranges of the linear terms participating as narrow as possible. To this end, a variable rescaling procedure is proposed on the original formulation **(GGP)**. *This variable scaling can be accomplished if all α_{ijk}'s are integrands or rational numbers that can be reduced into integrands.* Note that, by preserving all constants in the transformed objective function and constraints the same convergence and feasibility tolerances can be maintained. The proposed linear rescaling is the following

$$t_i \longleftarrow t_i^L + \frac{t_i^U - t_i^L}{t_i^{U,new} - t_i^{L,new}} \left(t_i^{new} - t_i^{L,new} \right), \quad i = 1, \ldots, N$$

where $t_i^{L,new}, t_i^{U,new}$ are selected so that $\log(t_i^{U,new}) - \log(t_i^{L,new})$ is small. The effect of this scaling on the maximum separation can be realized by considering the following simple example:

$$\begin{array}{rl} \min & -t \\ \text{subject to} & 0 < t^L = \exp(Y^L) \leq t \leq \exp(Y^U) = t^U \end{array}$$

It was shown earlier that the maximum separation between the transformed concave objective function and the linear underestimation is

$$\Delta_{max} = \exp(Y^L) \left[1 - \frac{\exp(\delta) - 1}{\delta} + \frac{\exp(\delta) - 1}{\delta} \log \frac{\exp(\delta) - 1}{\delta} \right].$$

The single variable t of the problem can be scaled as follows:

$$t \longleftarrow t^L + \frac{t^U - t^L}{t^{U,new} - t^{L,new}} \left(t^{new} - t^{L,new} \right).$$

Without loss of generality we select $t^{L,new} = \exp(Y^{L,new}) = 1$ which means that $Y^{L,new} = 0$ and therefore $Y^{U,new} = \delta^{new}$. The original minimization problem now becomes,

$$\begin{array}{rl} \min & -\left[\exp(Y^L) + \dfrac{\exp(Y^U) - \exp(Y^L)}{\exp(\delta^{new}) - 1} (t^{new} - 1) \right] \\ \text{subject to} & 1 \leq t \leq \exp(\delta^{new}). \end{array}$$

The maximum separation after rescaling is now

$$\Delta_{max}^{new} = \exp(Y^L) \frac{\exp(\delta) - 1}{\exp(\delta^{new}) - 1} \left[1 - \frac{\exp(\delta^{new}) - 1}{\delta^{new}} + \frac{\exp(\delta^{new}) - 1}{\delta^{new}} \log \frac{\exp(\delta^{new}) - 1}{\delta^{new}} \right].$$

The ratio of the maximum separation for the scaled example over the maximum separation for the original one is therefore,

$$R(\delta, \delta^{new}) = \frac{\Delta_{max}^{new}}{\Delta_{max}} = \left\{ \frac{\frac{1}{\exp(\delta^{new}) - 1} - \frac{1}{\delta^{new}} + \frac{1}{\delta^{new}} \log \frac{\exp(\delta^{new}) - 1}{\delta^{new}}}{\frac{1}{\exp(\delta) - 1} - \frac{1}{\delta} + \frac{1}{\delta} \log \frac{\exp(\delta) - 1}{\delta}} \right\}$$

It must be emphasized, however, that *the proposed variable scaling does not affect the scaling of the constraints.* This implies that the same feasibility tolerance ϵ_f can still be utilized before and after the variable scaling. The effect of this rescaling procedure on the global optimization algorithm is illustrated in the motivating example of subsection 4.1.

2.4 Transformation of Inequalities

The feasible region of the original problem **(GGP)** is defined by the fraction of the intersection of the nonlinear inequality constraints

$$G_j(\mathbf{t}) \;=\; G_j^+(\mathbf{t}) - G_j^-(\mathbf{t}) \;\leq\; 0, \;\; j = 1, \ldots, M$$

which lies within the hyperectangle defined by the box constraints

$$0 \;<\; t_i^L \;\leq\; t_i \;\leq\; t_i^U, \;\; i = 1, \ldots, N.$$

The convex relaxation **(R)** of **(GGP)** approximates the original feasible region with a relaxation of its convex hull. The closer this relaxation is to its convex hull the tighter the lower bounding of the objective function will be. This motivates the need to "preprocess" the original nonlinear constraints so that the employed relaxation will be as tight as possible. This can be accomplished by taking advantage of the mathematical structure of each original constraint $G_j(\mathbf{t})$ and the fact that all variables $\mathbf{t}$ are strictly positive. The positivity of $\mathbf{t}$ implies that for every nonlinear inequality constraint $G_j(\mathbf{t})$ there exists a family of nonlinear constraints $\mathcal{G}_j(\mathbf{t})$ whose elements are completely interchangeable with $G_j(\mathbf{t})$ and are derived by multiplying the original constraint by $t_1^{d_{1j}} t_2^{d_{2j}} \cdots t_N^{d_{Nj}}$ where $d_{ij}, \; i = 1, \ldots, N, \; j = 1, \ldots, M$ are arbitrarily selected real parameters.

$$\mathcal{G}_j(\mathbf{t}) \;=\; \left\{ G_j(\mathbf{t}, \mathbf{d}_j) \;=\; G_j(\mathbf{t}) \left(\prod_{i=1}^{N} t_i^{d_{ij}} \right) \;:\; d_{ij} \;\in\; \Re, \; i = 1, \ldots, N \right\}$$

Note that even though the values of the elements of $\mathcal{G}_j$ might not be identical for every $\mathbf{t}$, because $t_1^{d_{1j}}, \ldots, t_N^{d_{Nj}} \;>\; 0$, they always maintain the same sign as $G_j(\mathbf{t})$. This implies that every element of $\mathcal{G}_j$

$$G_j(\mathbf{t}, \mathbf{d}_j) \;=\; \sum_{k \in K_j^+} c_{jk} \prod_{i=1}^{N} t_i^{\left(\alpha_{ijk} + d_{ij}\right)} \;-\; \sum_{k \in K_j^-} c_{jk} \prod_{i=1}^{N} t_i^{\left(\alpha_{ijk} + d_{ij}\right)}$$

is an equivalent representation of the original constraint $G_j(\mathbf{t})$ and therefore it can replace it in formulation **(GGP)**. Clearly, unlike the variable scaling presented in the previous subsection, this multiplicative transformation affects the scaling of the problem. Therefore, appropriate changes have to be made on the feasibility as well as convergence tolerances to account for the changed scaling of the objective function and constraints. After applying the exponential transformation on the constraint $G_j(\mathbf{z}, \mathbf{t}_j)$ we obtain,

$$G_j(\mathbf{z}, \mathbf{d}_j) \;=\; \sum_{k \in K_j^+} c_{jk} \exp\left[\sum_{i=1}^{N} \left(\alpha_{ijk} + d_{ij}\right) z_i \right] \;-\; \sum_{k \in K_j^-} c_{jk} \exp\left[\sum_{i=1}^{N} \left(\alpha_{ijk} + d_{ij}\right) z_i \right],$$

which can replace the constraint $G_j(\mathbf{z})$ in problem **(DC)**. Even though all constraints $G_j(\mathbf{z}, \mathbf{d}_j) \in \mathcal{G}_j$ are interchangeable, this is not the case for their convex relaxations. By linearizing the concave terms $G_j^-(\mathbf{z}, \mathbf{d}_j)$, different elements of $\mathcal{G}_j$ give rise to distinctly different convex relaxed constraints $G_j^{conv}(\mathbf{z}, \mathbf{d}_j)$ whose domain of feasibility depends on the selection of $\mathbf{d}_j$.

$$G_j^{conv}(\mathbf{z}, \mathbf{d}_j) \;=\; \sum_{k \in K_j^+} c_{jk} \exp\left[\sum_{i=1}^{N} (\alpha_{ijk} + d_{ij})\, z_i\right] - \sum_{k \in K_j^-} c_{jk} \left\{ A_{jk} + B_{jk} \left[\sum_{i=1}^{N} (\alpha_{ijk} + d_{ij})\, z_i\right]\right\}$$

Because the tightest lower bounding of $G_j(\mathbf{z}_j)$ is sought, for a given $\mathbf{z}$ the best selection of $\mathbf{d}_j$ is the one for which $G_j(\mathbf{z}, \mathbf{d}_j)$ is maximized,

$$\mathbf{d}_j^*(\mathbf{z}) \;=\; \arg \max_{\mathbf{d}_j} \; G_j(\mathbf{z}, \mathbf{d}_j).$$

Note that it is not possible to consider the convex relaxations of all elements of $\mathcal{G}_j$ because their number is infinite. Unfortunately, the maximization problem not only is parametric in $\mathbf{z}$ but also for a given $\mathbf{z}$ is as difficult to solve as the original problem **(DC)**. Therefore, instead the following two criteria will be utilized to guide the selection of a good value for $\mathbf{d}_j$ for every constraint j. The first criterion is to minimize the number of variables z_i participating in every concave function $G_j^-(\mathbf{z}, \mathbf{d}_j)$. This requirement reflects the primary concern of any deterministic global optimization algorithm which is to keep the number of variables where branching is required at a minimum. The second criterion is associated with the observation that for a given constraint j, a very important consideration when selecting $\mathbf{d}_j$ is to maintain $Y_{jk}^U - Y_{jk}^L$ as small as possible for every $k \in K_j^-$. Although the rescaling of variables aims at this, it is sometimes necessary to complement this rescaling with an appropriate selection of $\mathbf{d}_j$. Because the primary concern is not to allow any $Y_{jk}^U - Y_{jk}^L$ to become larger than $\approx 2-5$ rather than minimizing a weighted sum of intervals $Y_{jk}^U - Y_{jk}^L$, it appears to be more meaningful to minimize the maximum $Y_{jk}^U - Y_{jk}^L$ over all $k \in K_j^-$ (worst case analysis). Furthermore, in practice, more than one convex relaxed function $G_j^{conv}(\mathbf{z}, \mathbf{d}_j)$ per constraint j can be incorporated in formulation $(\mathbf{R})$ to tighter approximate the convex hull of constraint $G_j(\mathbf{z})$. Although no rigorous way is provided for selecting the optimum selection of the set of $\mathbf{d}_j$i's per constraint, computational experience reveals that by utilizing the proposed criteria convex underestimation of every constraint j is improved.

2.5 Reduction of Rectangular Partitions

A discussed in an earlier subsection, the convex relaxation $(\mathbf{R})$ of the problem **(DC)** is aimed at deriving a lower bound to the solution of **(DC)**. This lower bound is calculated by solving the convex programming problem $(\mathbf{R})$ inside some hyperectangle defined by:

$$z_i^L \;\leq\; z_i \;\leq\; z_i^U, \;\; i = 1, \ldots, N.$$

Clearly, the smaller this hyperectangle, the tighter the convex lower bounding of all $G_j(\mathbf{z}),\ j = 0, \ldots, M$ and therefore the closer the solution of **(R)** will be to the solution of **(DC)**. Deterministic global optimization algorithms improve the quality of the obtained lower bounds by partitioning the initial hyperectangle into smaller ones, and repeatedly solving **(R)** inside each one of them. A number of branching rules are utilized which make the subdivision process as efficient as possible. However, the employed branch and bound procedure is exponential in nature and therefore potentially very computationally consuming. This means that one would desire to fathom all or part of a hyperectangle region without having to resort to the CPU demanding branch and bound whenever possible.

To meet this objective, one can attempt to locate the minimum/maximum value for every variable z_i for which the nonlinear constraint set of **(DC)** remains feasible and also the value of the objective function is at least as good as the best current upper bound U.

$$
\begin{array}{ll}
\min / \max & z_{i'}, \quad i' = 1, \ldots, N \\
\text{subject to} & G_j(\mathbf{z}) \ \leq \ 0, \ \ j = 1, \ldots, M \\
& G_0(\mathbf{z}) \ \leq \ U \\
& z_i^L \ \leq \ z_i \ \leq \ z_i^U, \ \ i = 1, \ldots, N.
\end{array}
$$

However, this is a nonconvex problem of equivalent difficulty with the initial problem **(DC)**. Instead, one can solve a relaxation of this problem by replacing all nonlinear constraints and objective function with their respective convex relaxations.

$$
\begin{array}{ll}
\min & z_{i'}, \quad i' = 1, \ldots, N \\
\text{subject to} & G_j^{conv}(\mathbf{z}) \ \leq \ 0, \ \ j = 1, \ldots, M \\
& G_0^{conv}(\mathbf{z}) \ \leq \ U \\
& z_i^L \ \leq \ z_i \ \leq \ z_i^U, \ \ i = 1, \ldots, N.
\end{array}
\qquad \textbf{(BR)}
$$

The solution of these convex bound reducing **(BR)** problems provide valid lower/upper bounds on all z_i so that all relaxed nonlinear constraints are feasible and the relaxed objective function is less than some upper bound U. Note that if **(BR)** is infeasible this implies the intersection of the feasible region of problem **(DC)** with the domain where the objective function is at least as good as some prespecified bound U is the empty set. In this case, the corresponding rectangular partition can be eliminated because the global solution is guaranteed not to be situated inside it. As shown in Figure 1 by minimizing/maximizing all z_i, the feasible region of **(R)** is inscribed inside the smallest possible rectangle defining a new refined set of bounds for $\mathbf{z}$. Because the reduction of the set of box constraints results in tighter underestimating functions $G_j^{conv}(\mathbf{z})$ and $G_0^{conv}(\mathbf{z})$ the minimization/maximization of all z_i can be repeated by dynamically updating the bounds z_i^L, z_i^U for more than one iteration until no further significant improvement on the bounds is achieved. This defines the following bound improving procedure:

STEP 0. Set $i' \longleftarrow 1$

STEP 1. Solve

$$\begin{array}{lll} & \min & z_{i'} \\ \text{subject to} & G_j^{conv}(\mathbf{z}) \leq 0, & j = 1, \ldots, M \\ & G_0^{conv}(\mathbf{z}) \leq U & \\ & z_i^L \leq z_i \leq z_i^U, & i = 1, \ldots, N. \end{array} \qquad (\mathbf{BRmin})$$

STEP 2. Set

$$\begin{array}{rcl} z_{i'}^{L,old} & \longleftarrow & z_{i'}^L \\ z_{i'}^L & \longleftarrow & z_{i'}^* \end{array}$$

STEP 3. Solve

$$\begin{array}{lll} & \max & z_{i'} \\ \text{subject to} & G_j^{conv}(\mathbf{z}) \leq 0, & j = 1, \ldots, M \\ & G_0^{conv}(\mathbf{z}) \leq U & \\ & z_i^L \leq z_i \leq z_i^U, & i = 1, \ldots, N. \end{array} \qquad (\mathbf{BRmax})$$

STEP 4. Set

$$\begin{array}{rcl} z_{i'}^{U,old} & \longleftarrow & z_{i'}^U \\ z_{i'}^U & \longleftarrow & z_{i'}^* \end{array}$$

STEP 5. If $i' < N$ then $i' \longleftarrow i' + 1$. Go to **STEP 1.**

STEP 6. If $\frac{\prod_{i=1}^{N}\left(z_i^L - z_i^U\right)}{\prod_{i=1}^{N}\left(z_i^{L,old} - z_i^{U,old}\right)} \geq r$ then go to **STEP 0**. Otherwise **END**

Parameter $r < 1$ typically $r \approx 0.6 - 0.9$, is simply the minimum hyperectangle volume reduction per iteration which is required to further continue the bound improving procedure. It must be noted, however, that with this procedure $2N$ convex programming problems must be solved per iteration. In this work this procedure is applied only in the initial partition. For intermediate rectangular partitions a relaxation of this procedure is utilized for detecting whether the current partition element can be fathomed. This alternative check is based on first deriving a lower bound for the objective function and every constraint of formulation **(R)** using monotonicity analysis principles. If the lower bound of the objective function is greater than U or if the lower bound of any of the constraints is greater than zero then the corresponding partition element can be eliminated since it is guaranteed either not to include the global minimum solution or to be infeasible. Note that the form of the objective and constraints in formulation **(R)** is

$$G_j^{conv}(\mathbf{z}) \;=\; \sum_{k \in K_j^+} c_{jk} \exp\left(\sum_{i=1}^{N} Y_{jk}\right) - \sum_{k \in K_j^-} c_{jk}\left[A_{jk} + B_{jk}\left(\sum_{i=1}^{N} Y_{jk}\right)\right], \;\; j = 0, \ldots, M$$

Note that because the function $\exp(x)$ is monotonically increasing, a lower bound $G_j^{conv,L}$ of $G_j^{conv}(\mathbf{z})$ can be obtained by fixing all $Y_{jk},\ k \in K_j^+$ at their lower bounds Y_{jk}^L, and all $Y_{jk},\ k \in K_j^-$ at their respective upper bounds Y_{jk}^U since $B_{jk} \geq 0$.

$$G_j^{conv,L} \;=\; \sum_{k \in K_j^+} c_{jk} \exp\left(\sum_{i=1}^{N} Y_{jk}^L\right) - \sum_{k \in K_j^-} c_{jk}\left[A_{jk} + B_{jk}\left(\sum_{i=1}^{N} Y_{jk}^U\right)\right], \;\; j = 0, \ldots, M$$

The corresponding partition element can be eliminated if

$$G_0^{conv,L} \;\geq\; U, \;\; \text{or} \;\; \exists\, j = 1, \ldots, M \text{ such that } G_j^{conv,L} \;\geq\; 0.$$

These feasibility checks can be applied to every subdivision element before the convex minimization of problem **(R)** is performed and the associated computational effort is negligible. Note that the relaxation of the bounding procedure is a computationally efficient check for detecting infeasibility of the constraint set especially at earlier stages of the algorithm. Computational experience has shown that this results in significant reduction in the number of required convex minimizations ($10\% - 50\%$).

2.6 Monotonicity Analysis

Monotonicity analysis principles can be utilized to improve the convex relaxation of **(DC)** by dictating that some nonlinear inequality constraints must be satisfied as equality constraints at the global minimum solution.

$$\begin{aligned} \min_{\mathbf{z}} \quad & G_0(\mathbf{z}) \\ \text{subject to} \quad & G_j(\mathbf{z}) \;\leq\; 0, \quad j = 1, \ldots, M \end{aligned} \tag{DC}$$

Hansen *et al.* [37] pointed out that if there exists a global minimum solution for **(DC)** and the objective function $G_0(\mathbf{z})$ is monotonically increasing(decreasing) in some z_i, then at least one constraint which is monotonically decreasing(increasing) with respect to z_i will be active at the global minimum solution. Note also that if z_i is not present in the objective function then at least one constraint involving z_i will be active at the global minimum solution. The conclusions from the monotonicity analysis can be more conveniently expressed by associating a boolean variable a_j for every constraint $j = 1, \ldots, M$. If the constraint j is active at the global minimum solution then $a_j = 1$, otherwise $a_j = 0$. Let also J_i^-, J_i^+, J_i^0 denote the sets of indices of constraints j involving variables z_i which are monotonically decreasing, monotonically increasing, and not monotonous or with unknown monotonicity in z_i respectively. The monotonicity principles can then be expressed mathematically as follows:

$$\sum_{j \in J_i^+ \cup J_i^0} a_j \ \geq \ 1, \quad \text{if } G_0(\mathbf{z}) \text{ is monotonically decreasing in } z_i$$

$$\sum_{j \in J_i^- \cup J_i^0} a_j \ \geq \ 1, \quad \text{if } G_0(\mathbf{z}) \text{ is monotonically increasing in } z_i$$

$$\sum_{j \in J_i^+ \cup J_i^0} a_j \ \geq \ 1 \ \text{ or } \ \sum_{j \in J_i^- \cup J_i^0} a_j \ \geq \ 1, \quad \text{if } G_0(\mathbf{z}) \text{ is independent of } z_i$$

Clearly, the smaller the sets J_i^0, the less the number of a_j's that will participate in inequality relations, and therefore it will be more likely to have some a_j's fixed at one. Note that if $a_j = 1$, then the corresponding constraint j is satisfied as an equality. This can lead to elimination of a variable or addition of an extra constraint.

A constraint $G_j(\mathbf{z})$ of formulation **(DC)**,

$$G_j(\mathbf{z}) \ = \ \sum_{k \in K_j^+} c_{jk} \exp\left(\sum_{i=1}^{N} \alpha_{ijk} z_i\right) - \sum_{k \in K_j^-} c_{jk} \exp\left(\sum_{i=1}^{N} \alpha_{ijk} z_i\right),$$

is guaranteed to be monotonic in z_i if

$$\begin{aligned} \alpha_{ijk} \ &\geq \ 0, \ \forall\, k \in K_j^+ \text{ and } \alpha_{ijk} \ \leq \ 0, \ \forall\, k \in K_j^- \\ \text{or} \quad \alpha_{ijk} \ &\leq \ 0, \ \forall\, k \in K_j^+ \text{ and } \alpha_{ijk} \ \geq \ 0, \ \forall\, k \in K_j^- \end{aligned}$$

However, by appropriately replacing the initial constraint $G_j(\mathbf{z})$ with one from the family $\mathcal{G}_j$

$$G_j(\mathbf{z}, \mathbf{d}_j) \ = \ \sum_{k \in K_j^+} c_{jk} \exp\left[\sum_{i=1}^{N} \left(\alpha_{ijk} + d_{ij}\right) z_i\right] \ - \ \sum_{k \in K_j^-} c_{jk} \exp\left[\sum_{i=1}^{N} \left(\alpha_{ijk} + d_{ij}\right) z_i\right],$$

one can come up with less restrictive conditions for monotonicity. More specifically,

$$\begin{aligned} G_j(\mathbf{z}, \mathbf{d}_j) \ &\text{is monotonically increasing in } z_i \text{ if, } \min_{k \in K_j^+} \alpha_{ijk} \ \geq \ \max_{k \in K_j^-} \alpha_{ijk} \\ &\text{and} \quad d_{ij} \ \in \ \left[\max_{k \in K_j^-} \alpha_{ijk}, \ \min_{k \in K_j^+} \alpha_{ijk}\right] \\ G_j(\mathbf{z}, \mathbf{d}_j) \ &\text{is monotonically decreasing in } z_i \text{ if, } \max_{k \in K_j^+} \alpha_{ijk} \ \leq \ \min_{k \in K_j^-} \alpha_{ijk} \\ &\text{and} \quad d_{ij} \ \in \ \left[\max_{k \in K_j^+} \alpha_{ijk}, \ \min_{k \in K_j^-} \alpha_{ijk}\right] \end{aligned}$$

These monotonicity analysis principles can be utilized to eliminate variables or further constraint formulation **(DC)**. In the next section, a branch and bound type global optimization algorithm is introduced for solving problem **(DC)**.

3 Global Optimization

3.1 Description

A global optimization algorithm is proposed for locating the global minimum solution of **(DC)** based on the refinement of converging lower and upper bounds through the solution of convex programming problems **(R)**. Clearly the value of the objective function $G_0(\mathbf{z})$ at any feasible point $\mathbf{z}$ provides an upper bound on the global minimum G_0^*. Lower bounds on G_0^* within some box constraints are derived by solving the convex lower bounding optimization problem **(R)**.

Since **(R)** is convex, its single global minimum within some box constraints can be routinely found with a commercially available optimization algorithm (e.g. MINOS 5.4 [38]) and will always underestimate the global minimum of **(DC)** within the same box constraints. Assuming that this global minimum solution of **(R)** is a feasible point for **(DC)**, an upper bound on G_0^* can then be obtained by simply calculating $G_0(\mathbf{z})$ at the global minimum point of **(R)**. Based on the analysis performed in subsection 2.2, the gap between the upper bound and the lower bound on G_0^* will be at most,

$$\begin{aligned}
\Delta_{0,max} &= \sum_{k \in K_0^-} c_{0k} \exp\left(Y_{0k}^L\right) \left[1 - \frac{\exp\left(\delta_{0k}\right) - 1}{\delta_{0k}} + \frac{\exp\left(\delta_{0k}\right) - 1}{\delta_{0k}} \log \frac{\exp\left(\delta_{0k}\right) - 1}{\delta_{0k}}\right] \\
\text{where} \quad \delta_{0k} &= Y_{0k}^U - Y_{0k}^L \\
\text{and} \quad Y_{0k}^L &= \sum_{i=1}^{N} \min\left(\alpha_{i0k} z_i^L, \alpha_{i0k} z_i^U\right), \quad Y_{0k}^U = \sum_{i=1}^{N} \max\left(\alpha_{i0k} z_i^L, \alpha_{i0k} z_i^U\right)
\end{aligned}$$

Note that $\Delta_{0,max}$ goes to zero as all $\delta_{0k}, \ k \in K_0^-$ approach zero. Furthermore, because

$$\lim_{\delta_i = z_i^U - z_i^L \to 0^+, \, \forall \, i=1,\ldots,N} \delta_{0k} = 0, \quad \forall \, k \in K_0^-$$

we have

$$\lim_{\delta_i = z_i^U - z_i^L \to 0^+, \, \forall \, i=1,\ldots,N} \Delta_{0,max} = 0.$$

This implies that as the current box constraints $[\mathbf{z}^L, \mathbf{z}^U]$ collapse into a point the maximum separation $\Delta_{0,max}$ between the original objective function of **(DC)** and its convex relaxation in **(R)** becomes zero. Because $\Delta_{0,max}$ is a continuous function of $\delta_{0k} \geq 0, \ k \in K_0^-$ and thus of $\delta_i \geq 0, \ i = 1, \ldots, N$, for every positive number ϵ_c there will always be a positive number δ^* such that if all $\delta_{0k} \leq \delta^*$ we have $\Delta_{0,max} \leq \epsilon_c$. This result reflects the fact that by reducing the bounds $[\mathbf{z}^L, \mathbf{z}^U]$ the objective function $G_0(\mathbf{z})$ of **(DC)** can become arbitrarily close to its convex relaxation $G_0^{conv}(\mathbf{z})$ of **(R)**.

By following the same argument the maximum separations $\Delta_{j,max}$ between constraints $G_j(\mathbf{z})$ and the corresponding convex relaxations $G_j^{conv}(\mathbf{z})$

$$\Delta_{j,max} = \sum_{k \in K_j^-} c_{jk} \exp\left(Y_{jk}^L\right) \left[1 - \frac{\exp\left(\delta_{jk}\right) - 1}{\delta_{jk}} + \frac{\exp\left(\delta_{jk}\right) - 1}{\delta_{jk}} \log \frac{\exp\left(\delta_{jk}\right) - 1}{\delta_{jk}}\right]$$

$$\text{where} \qquad \delta_{jk} = Y_{jk}^U - Y_{jk}^L$$

$$\text{and} \qquad Y_{jk}^L = \sum_{i=1}^{N} \min\left(\alpha_{ijk} z_i^L, \alpha_{ijk} z_i^U\right), \;\; Y_{jk}^U = \sum_{i=1}^{N} \max\left(\alpha_{ijk} z_i^L, \alpha_{ijk} z_i^U\right)$$

go to zero as all δ_{jk}, $\forall\, k \in K_j^-$ or alternatively $\delta_i = z_i^U - z_i^L \to 0^+$, $\forall\, i = 1, \ldots, N$ approach zero. Furthermore, $\Delta_{j,max}$ are continuous functions of δ_{jk}, $k \in K_j^-$ and also of δ_i, $i = 1, \ldots, N$. This implies that for every positive number ϵ_f there will always be a positive number δ^* such that by selecting $\delta_{jk} \leq \delta^*$ we have

$$\max_{j \in K_j^-} \Delta_{j,max} \leq \epsilon_f.$$

The interpretation of this result is that by reducing the bounds $[\mathbf{z}^L, \mathbf{z}^U]$ on $\mathbf{z}$, differences between the feasible region of the original problem **(DC)** and its convex relaxation **(R)** become arbitrarily small. Therefore, any feasible point $\mathbf{z}^c$ of problem **(R)** (even the global minimum solution) becomes at least ϵ_f–feasible for problem **(DC)** by sufficiently tightening the bounds on $\mathbf{z}$ around this point.

After establishing upper and lower bounds on the global minimum inside some rectangular domain, the next step is to tighten them. This is accomplished by restricting the rectangular size. Tighter box constraints can be realized by partitioning the rectangle that the initial box constraints define into a number of smaller rectangles. The minimum number of variables along which subdivision is required is equal to the number of *linearly independent* $Y'_{jk}s$, $j = 0, \ldots, M$, $k \in K_j^-$. Typically this number is equal to the number of z_i participating in any of the terms $G_j^-(\mathbf{z})$. This defines the following set of variables where branching is required.

$$\mathcal{Z} \;=\; \left\{ z_{i^*} \;:\; \exists\; \alpha_{i^*jk} \;\neq\; 0,\; j = 0, \ldots, M, \;\text{and}\, k \in K_j^- \right\}$$

In the great majority of cases the number of $z_i \in \mathcal{Z}$ is much smaller than the total number of terms Y_{jk}, $k \in K_j^-$ in formulation **(DC)**. Therefore in this work subdivision is performed on the set of variables z_i belonging in $\mathcal{Z}$.

One way of partitioning is to successively divide the current rectangle in two subrectangles by halving on the middle point of the longest side of the initial rectangle (bisection). Presumably, at each iteration the lower bound of G_0^* is simply the minimum over all the minima of problem **(R)** in every subrectangle composing the initial rectangle. Therefore, a straightforward (bound improving) way of tightening the lower bound is to halve at each iteration, only the subrectangle responsible for the infimum of the minima of **(R)** over all subrectangles, according to the rules discussed earlier. This procedure generates a *nondecreasing* sequence for the lower bound of G_0^*. Furthermore, we construct a *nonincreasing* sequence for the upper bound by selecting it to be the infenum over all the previously recorded upper bounds. Clearly, if the single minimum of **(R)** in any subrectangle is greater than the current upper bound we can safely ignore this subrectangle because the global minimum of $G_0(\mathbf{z})$ cannot be situated inside it (fathoming step).

The next question is how small these subrectangles must become before the upper and lower bounds of $G_0(\mathbf{z})$ are within ϵ_c and also the feasible region of **(DC)** is ϵ_f-close to the feasible region of **(R)**. Because ϵ_c and ϵ_f are very small numbers the maximum separations $\Delta_{j,max}$, will be proportional to the square of the diagonal δ of the current subrectangle ($\approx \mathcal{O}(\delta^2)$). This means that the required for convergence value of δ is proportional to the square root of ϵ_c. Therefore, if for example ϵ_c, ϵ_f are set to be 0.0001, it suffices for δ to be proportional to 0.01.

The basic steps of the proposed global optimization algorithm are summarized in the following section.

3.2 Steps of the Global Optimization Algorithm

STEP 1 - *Initialization*

A convergence tolerance ϵ_c and a feasibility tolerance ϵ_f are selected and the iteration counter $Iter$ is set to one. Appropriate global bounds z_i^{LBD}, z_i^{UBD} on the z_i's are calculated based on the bound improving procedure discussed in subsection 2.5 and local bounds $z_i^{L,Iter}, z_i^{U,Iter}$ for the first iteration are set to be equal to the global ones. The initial constraints $G_j(\mathbf{z})$, $j = 1, \ldots, M$ are replaced by an appropriately selected (based on the analysis of subsection 2.3) set of constraints $G_j(\mathbf{z}, \mathbf{d}_j) \in \mathcal{G}$. In this description of the algorithm the parameters $\mathbf{d}_j$ are omitted from the constraints for the sake of simplicity. Finally, lower and upper bounds G_0^{LBD}, G_0^{UBD} on the global minimum G_0^* are initialized and an initial current point $z_i^{c,Iter}$ is selected.

STEP 2 - *Feasibility Check and Update of Upper Bound* G_0^{UBD}

If the maximum over all constraints G_j calculated at the current point $z_i^{c,Iter}$ is less than ϵ_f,

$$\max_{j=1,\ldots,M} \; G_j(\mathbf{z}^{c,Iter}) \;\leq\; \epsilon_f$$

then the constraint set is ϵ_f–feasible at the current point. If so, the objective function G_0 is calculated at the current point $\mathbf{z}^{c,Iter}$ and the upper bound G_0^{UBD} is updated as follows,

$$G_0^{UBD} \;=\; \min\left(G_0^{UBD}, G_0(\mathbf{z}^{c,Iter})\right)$$

STEP 3 - *Partitioning of Current Rectangle*

The current rectangle $\left[z_i^{L,iter}, z_i^{U,Iter}\right]$, $\; i = 1, \ldots, N$ is partitioned into the following two rectangles ($r = 1, 2$):

$$\begin{bmatrix} z_1^{L,Iter} & z_1^{U,Iter} \\ \vdots & \vdots \\ z_{l^{Iter}}^{L,Iter} & \frac{\left(z_{l^{Iter}}^{L,Iter} + z_{l^{Iter}}^{U,Iter}\right)}{2} \\ \vdots & \vdots \\ z_N^{L,Iter} & z_N^{U,Iter} \end{bmatrix}, \quad \begin{bmatrix} z_1^{L,Iter} & z_1^{U,Iter} \\ \vdots & \vdots \\ \frac{\left(z_{l^{Iter}}^{L,Iter} + z_{l^{Iter}}^{U,Iter}\right)}{2} & z_{l^{Iter}}^{U,Iter} \\ \vdots & \vdots \\ z_N^{L,Iter} & z_N^{U,Iter} \end{bmatrix}$$

where l^{Iter} corresponds to the variable with the longest side in the initial rectangle,

$$l^{Iter} \;=\; \arg\max_i \left(z_i^{U,Iter} \;-\; z_i^{L,Iter}\right)$$

STEP 4 - *Feasibility Checks of Convex Relaxation* **(R)**

Update in both subrectangle ($r = 1, 2$) the parameters Y_{jk}^L, Y_{jk}^U, A_{jk}, B_{jk} as follows:

$$Y_{jk}^L \;=\; \sum_{i=1}^{N} \min\left(\alpha_{ijk} z_i^L, \alpha_{ijk} z_i^U\right),\;\; Y_{jk}^U \;=\; \sum_{i=1}^{N} \max\left(\alpha_{ijk} z_i^L, \alpha_{ijk} z_i^U\right)$$

$$A_{jk} \;=\; \frac{Y_{jk}^U \exp\left(Y_{jk}^L\right) - Y_{jk}^L \exp\left(Y_{jk}^U\right)}{Y_{jk}^U - Y_{jk}^L},\;\; B_{jk} \;=\; \frac{\exp\left(Y_{jk}^U\right) - \exp\left(Y_{jk}^L\right)}{Y_{jk}^U - Y_{jk}^L}$$

If there exists $j = 0, \ldots, M$ such that one of the lower bounds

$$G_j^{conv,L} \;=\; \sum_{k \in K_j^+} c_{jk} \exp\left(\sum_{i=1}^{N} Y_{jk}^L\right) - \sum_{k \in K_j^-} c_{jk} \left[A_{jk} + B_{jk}\left(\sum_{i=1}^{N} Y_{jk}^U\right)\right],\;\; j = 0, \ldots, M$$

satisfy

$$G_0^{conv,L} \;\geq\; G_0^{UBD} \;\;\text{or}\;\; G_j^{conv,L} \;\geq\; 0, \;\text{for some}\;\; j = 1, \ldots, M$$

then the corresponding subrectangle is eliminated (fathoming), go to **STEP 6.** Otherwise, proceed with **STEP 5**.

STEP 5 - *Solution of Convex Problems Inside Subrectangles*

Solve the following convex optimization problem **(R)** in both subrectangles ($r = 1, 2$) by using any convex nonlinear solver (e.g. MINOS 5.4 [38]).

$$\begin{aligned} \min_{\mathbf{z}} \quad & G_0^{conv}(\mathbf{z}) \\ \text{subject to} \quad & G_j^{conv}(\mathbf{z}) \;\leq\; 0, \;\; j = 1, \ldots, M \\ & \mathbf{z}^L \;\leq\; \mathbf{z} \;\leq\; \mathbf{z}^U \end{aligned}$$

If a solution $G_{0,sol}^{r,Iter}$ is less than the current upper bound, G_0^{UBD} then it is stored along with the value of the variables $\mathbf{z}$ at the solution point $z_{i,sol}^{r,Iter}$.

STEP 6 - *Update Iteration Counter Iter and Lower Bound* G_0^{LBD}
The iteration counter is increased by one,

$$Iter \longleftarrow Iter + 1$$

and the lower bound G_0^{LBD} is updated to the minimum solution over the stored ones from previous iterations. Furthermore, the selected solution is erased from the stored set.

$$\begin{aligned} G_0^{LBD} &= G_{0,sol}^{r',Iter'} \\ \text{where} \quad G_{0,sol}^{r',Iter'} &= \min_{r,I} G_{0,sol}^{r,I}, \quad r = 1,2, \quad I = 1, \ldots, Iter - 1. \end{aligned}$$

STEP 7 - *Update Current Point* $\mathbf{z}^{c,Iter}$ *and Current Bounds* $\mathbf{z}^{L,Iter}, \mathbf{z}^{U,Iter}$ *on* $\mathbf{z}$
The current point is selected to be the solution point of the previously found minimum solution in **STEP 6**,

$$\mathbf{z}^{c,Iter} = \mathbf{z}_{i,sol}^{r',Iter'}$$

and the current rectangle becomes the subrectangle containing the previously found solution,

$$\left[z_i^{L,Iter}, z_i^{U,Iter}\right] = \begin{bmatrix} z_1^{L,Iter'} & z_1^{U,Iter'} \\ \vdots & \vdots \\ z_{l_{Iter'}}^{L,Iter'} & \frac{\left(z_{l_{Iter'}}^{L,Iter'} + z_{l_{Iter'}}^{U,Iter'}\right)}{2} \\ \vdots & \vdots \\ z_N^{L,Iter'} & z_N^{U,Iter'} \end{bmatrix}, \quad \text{if} \quad r' = 1$$

$$\left[z_i^{L,Iter}, z_i^{U,Iter}\right] = \begin{bmatrix} z_1^{L,Iter'} & z_1^{U,Iter'} \\ \vdots & \vdots \\ \frac{\left(z_{l_{Iter'}}^{L,Iter'} + z_{l_{Iter'}}^{U,Iter'}\right)}{2} & z_{l_{Iter'}}^{U,Iter'} \\ \vdots & \vdots \\ z_N^{L,Iter'} & z_N^{U,Iter'} \end{bmatrix}, \quad \text{if} \quad r' = 2$$

STEP 8 - *Check for Convergence*

IF $\left(G_0^{UBD} - G_0^{LBD}\right) > \epsilon_c$, then return to **STEP 2**

Otherwise, ϵ_c–convergence has been reached and the global minimum solution, and solution point are:

$$\begin{array}{rcl} G_0^* & \longleftarrow & G_0(\mathbf{z}^{c,Iter''}) \\ \mathbf{z}^* & \longleftarrow & \mathbf{z}^{c,Iter''} \\ \text{where} \quad Iter'' & = & \arg_I \left\{ G_0(\mathbf{z}^{c,I}) \; = \; G_0^{UBD}, \quad I = 1, \ldots, Iter \right\}. \end{array}$$

In the following section, a mathematical proof that the proposed global optimization algorithm converges to the the global minimum is given based on the analysis of a standard deterministic global optimization algorithm presented in [39].

3.3 Proof of Convergence to the Global Minimum

Convergence properties of a global optimization algorithm depend on:

(i) the limit behavior of the difference $G_j^{UBD} - G_j^{LBD}$, $j = 0, \ldots, M$ for unfathomed successively refined partitions,

(ii) the subdivision process of the current partitions, and

(iii) the employed selection process of the partition(s) that have to be further refined.

In the employed global optimization algorithm, a lower bound $G_{0,L}^{r,Iter}$ is obtained as the solution of the convex minimization of **(R)** for every partition element $(r, Iter)$. Also an upper bound $G_{0,U}^{r,Iter}$ is obtained as the value of G_0 at the single solution of problem **(R)** assuming that it is an ϵ_f–feasible point for **(DC)**. The partition element involving the minimum lower bound $G_{0,L}^{r,Iter}$ is selected for further refining according to the bisection rule and partition elements whose lower bound $G_{0,L}^{r,Iter}$ is greater than the current upper bound G_0^{UBD} are fathomed. A sufficient condition for a global optimization algorithm to be convergent to the global minimum, stated in [39], requires that the bounding operation must be *consistent* and the selection operation *bound improving*.

A bounding operation is called **consistent** if at every step any unfathomed partition can be further refined, and if any infinitely decreasing sequence of successively refined partition elements satisfies,

$$\lim_{Iter \to \infty} \left(G_{0,U}^{Iter} \; - \; G_{0,L}^{r,Iter} \right) \; = \; 0,$$

where $G_{0,L}^{r,Iter}$ is a lower bound of G_0^* inside the $(r, Iter)$ partition element and $G_{0,U}^{Iter}$ is the best upper bound at iteration $Iter$ not necessarily occurring inside the $(r, Iter)$ partition element. In practice, the requirement $\lim_{Iter \to \infty} \left(G_{0,U}^{Iter} \; - \; G_{0,L}^{r,Iter} \right) \; = \; 0$ for any infinitely decreasing sequence of successively refined partition elements is difficult to verify because $G_{0,U}^{Iter}$ is not necessarily attained at the partition element $(r, Iter)$. Therefore, in view of the inequality $G_{0,U}^{r,Iter} \geq G_{0,U}^{Iter} \geq G_{0,L}^{r,Iter}$, the following, at least equally strong condition, suffices to be shown:

$$\lim_{Iter\to\infty}\left(G_{0,U}^{r,Iter} \;-\; G_{0,L}^{r,Iter}\right) \;=\; 0.$$

In subsection 3.1 we have shown that by reducing the size of the partition element $[\mathbf{z}^L, \mathbf{z}^U]$ where we assume that **(R)** is feasible then **(DC)** is ϵ_f–feasible and the objective function $G_0(\mathbf{z})$ becomes arbitrarily close to its convex relaxation $G_0^{conv}(\mathbf{z})$. This means that,

$$\lim_{\delta_i = z_i^U - z_i^L \to 0^+,\, \forall\, i=1,\ldots,N} \quad G_{0,U}^{r,Iter} \;-\; G_{0,L}^{r,Iter} \;=\; 0$$

Therefore, it suffices to show that

$$\lim_{Iter\to\infty}\; \max_i\; \delta_i \;=\; 0$$

This is equivalent with requiring that any unfathomed infinitely decreasing sequence of successively refined partition elements is *exhaustive* [39]. The employed subdivision process is the *bisection* where every partition element $(r, Iter)$ is subdivided into two subrectangles $r = 1,\ 2$ of equal volume by halving at the midpoint of the longest side. By scaling all sides of any partition element with the sides of the rectangle that the initial global constraints define, the scaled sides of the initial rectangle are all equal to one. Therefore, the condition of always subdividing along the longest side can be satisfied by simply subdividing first along the side $k = 1$, then along the side $k = 2$, etc. until the last side $k = K$ is encountered when the subdivision starts again from $k = 1$. By partitioning in this orderly manner each side of every successively refined partition element is halved *exactly* once every K subdivisions. Consequently, after K subdivisions the diagonal of the resulting partition elements is one half the diagonal of the original partition element. Therefore, as the number of successive subdivisions of a partition element goes to infinity, the diagonals of the resulting partition elements go to zero. This implies that the employed subdivision process is *exhaustive*.

Consequently, for any infinitely decreasing sequence of successively refined partition elements

$$\lim_{Iter\to\infty}\left(G_{0,U}^{r,Iter} \;-\; G_{0,L}^{r,Iter}\right) \;=\; 0$$

meaning that the employed bounding operation is *consistent*.

A *selection operation* is called *bound improving* if at least one partition element where the actual lower bound is attained is selected for further partition after a finite number of refinements. Clearly, the employed selection operation is *bound improving* because the partition element where the actual lower bound is attained is selected for further partition in the immediately following iteration.

In summary, we have shown that the bounding operation is *consistent* and that the selection operation is *bound improving*, therefore according to Theorem IV.3. in [39] the employed global optimization algorithm is *convergent* to the global minimum. In the next section the proposed global optimization algorithm is applied to a number of example problems.

4 Computational Results

First, a prototype generalized geometric problem is addressed and the effect of rescaling on the total number of required for convergence iterations is examined and shown to follow the theoretical predictions. Next, a number of engineering design problems are addressed, two of them in detail, and ϵ–convergence to the global minimum is achieved in all cases. Finally, in the last subsection a model for checking the stability of given control strategies is presented which requires the solution of **(GGP)** problems and a number of stability analysis examples are considered. The proposed deterministic global optimization algorithm has been implemented in GAMS and computational times are reported for all examples on a HP–730 workstation. More efficient implementations in C are expected to result in significant CPU reductions.

4.1 Motivating Example

The following motivating example, taken from [40], involves the minimization of a single variable subject to two nonlinear constraints.

$$\min_{x_1,x_2} \quad x_1$$

$$\begin{aligned} \text{subject to} \quad & \frac{1}{4}x_1 + \frac{1}{2}x_2 - \frac{1}{16}x_1^2 - \frac{1}{16}x_2^2 - 1 \leq 0 \\ & \frac{1}{14}x_1^2 + \frac{1}{14}x_2^2 + 1 - \frac{3}{7}x_1 - \frac{3}{7}x_2 \leq 0 \end{aligned} \qquad (\mathbf{P})$$

$$\begin{aligned} 1 \leq\ & x_1 \leq 5.5 \\ 1 \leq\ & x_2 \leq 5.5 \end{aligned}$$

These two nonlinear constraints define a nonconvex crescent–shaped feasible region shown in Figure 2 and even though the objective function is linear there still exists three distinct KKT points A,B and C. Therefore, local optimization approaches can converge to either one of them at best depending on which basin of attraction the initial point is inside.

The following linear variable scaling is then applied on both x_1, x_2 and its effect is measured on the total number of iterations for convergence.

$$x_1 \longleftarrow 1 + \frac{4.5}{U-1}\left(x_1^{new} - 1\right) \quad x_2 \longleftarrow 1 + \frac{4.5}{U-1}\left(x_2^{new} - 1\right).$$

This transformation defines a one to one mapping of the original variables x_1, x_2 onto the new variables x_1^{new}, x_2^{new} whose upper bound U is a manipulated parameter.

$$x_1^{new} = 1 + \frac{U-1}{4.5}\left(x_1 - 1\right) \quad x_2 = 1 + \frac{U-1}{4.5}\left(x_2 - 1\right)$$

$$1 \leq x_1^{new}, x_2^{new} \leq U$$

After substituting the expressions for x_{11}, x_2 into **(P)** we obtained the following transformed problem, parametric in U:

$$\min_{x_1^{new}, x_2^{new}} \quad 1 + \frac{4.5}{U-1}\left(x_1^{new} - 1\right)$$

$$\begin{aligned}
\text{subject to} \quad & \frac{63+18U}{32(U-1)^2}x_1^{new} + \frac{27+54U}{32(U-1)^2}x_2^{new} - \frac{81}{64(U-1)^2}(x_1^{new})^2 \\
& -\frac{81}{64(U-1)^2}(x_2^{new})^2 - \frac{21+48U+12U^2}{32(U-1)^2} \leq 0 \\
& \frac{81}{56(U-1)^2}(x_1^{new})^2 + \frac{81}{56(U-1)^2}(x_2^{new})^2 + \frac{17+56U+8U^2}{28(U-1)^2} \\
& -\frac{45+36U}{28(U-1)^2}x_1^{new} - \frac{45+36U}{28(U-1)^2}x_2^{new} \leq 0 \\
& 1 \leq x_1^{new}, x_2^{new} \leq U
\end{aligned} \qquad (\mathbf{P_U})$$

Clearly, for any value of the new upper bound $U \geq 1$, problem $(\mathbf{P_U})$ has the same solution with **(P)**. Furthermore, the performed variable scaling does not affect the scaling of the constraints. Note that even though the scaling of the new variables x_1^{new}, x_2^{new} depends on U, the value of the objective function and constraints remains unchanged in $(\mathbf{P_U})$. However, the solution of the convex relaxation $(\mathbf{R_U})$ of $(\mathbf{P_U})$ is a function of U. This implies that different values for U yield different number of required for convergence iterations for $(\mathbf{P_U})$. After selecting the convergence as well as the feasibility tolerances to be equal to $\epsilon_c = \epsilon_f = 10^{-4}$, the proposed global optimization procedure is applied and convergence to the global minimum point $(x_1^* = 1.178, x_2^* = 2.178)$ is achieved from all starting points for different values of U. The average number of iterations as a function of U are summarized in Table 2. It appears that the total number of iterations for solving $(\mathbf{P_U})$ is a monotonically increasing function of U. This implies that the tighter the rescaling, the better the underestimation will be. This is consistent with the theoretical results presented in subsection 2.2. Note that as U approaches one, numerical stability problems offset the benefits of marginally better underestimators.

4.2 Alkylation process design

This example involves the design of an alkylation unit [41]. The objective is to improve the octane number of some olefin feed by reacting it with isobutane in the presence of acid. The product of the reaction is distilled and the unreacted isopropane is recycled back to the reactor. All equality constraints are eliminated and the problem has been formulated as a signomial optimization problem.

$$\min \quad c_1x_1 + c_2x_1x_6 + c_3x_3 + c_4x_2 + c_5 - c_6x_3x_5$$

$$\begin{array}{lrcl}
\text{subject to} & c_7x_6^2 + c_8x_1^{-1}x_3 - c_9x_6 & \leq & 1 \\
& c_{10}x_1x_3^{-1} + c_{11}x_1x_3^{-1}x_6 - c_{12}x_1x_3^{-1}x_6^2 & \leq & 1 \\
& c_{13}x_6^2 + c_{14}x_5 - c_{15}x_4 - c_{16}x_6 & \leq & 1 \\
& c_{17}x_5^{-1} + c_{18}x_5^{-1}x_6 + c_{19}x_4x_5^{-1} - c_{20}x_5^{-1}x_6^2 & \leq & 1 \\
& c_{21}x_7 + c_{22}x_2x_3^{-1}x_4^{-1} - c_{23}x_2x_3^{-1} & \leq & 1 \\
& c_{24}x_7^{-1} + c_{25}x_2x_3^{-1}x_7^{-1} - c_{26}x_2x_3^{-1}x_4^{-1}x_7^{-1} & \leq & 1 \\
& c_{27}x_5^{-1} + c_{28}x_5^{-1}x_7 & \leq & 1 \\
& c_{29}x_5 - c_{30}x_7 & \leq & 1 \\
& c_{31}x_3 - c_{32}x_1 & \leq & 1 \\
& c_{33}x_1x_3^{-1} + c_{34}x_3^{-1} & \leq & 1 \\
& c_{35}x_2x_3^{-1}x_4^{-1} - c_{36}x_2x_3^{-1} & \leq & 1 \\
& c_{37}x_4 + c_{38}x_2^{-1}x_3x_4 & \leq & 1 \\
& c_{39}x_1x_6 + c_{40}x_1 - c_{41}x_3 & \leq & 1 \\
& c_{42}x_1^{-1}x_3 + c_{43}x_1^{-1} - c_{44}x_6 & \leq & 1
\end{array}$$

$$\begin{array}{rcccr}
1500 & \leq & x_1 & \leq & 2000 \\
1 & \leq & x_2 & \leq & 120 \\
3000 & \leq & x_3 & \leq & 3500 \\
85 & \leq & x_4 & \leq & 93 \\
90 & \leq & x_5 & \leq & 95 \\
3 & \leq & x_6 & \leq & 12 \\
145 & \leq & x_7 & \leq & 162
\end{array}$$

where $c_i,\ i = 1, \ldots, 44$ are positive parameters given in Table 3. This problem involves seven nonlinear variables subject to 12 nonlinear and two linear inequality constraints. Note that there exists a large number of bilinear, trilinear, and even fractional terms in the objective function and constraints. First, the bound improving procedure yields the following reduced variable bounds:

$$\begin{array}{rcccr}
1660.00 & \leq & x_1 & \leq & 2000.00 \\
23.54 & \leq & x_2 & \leq & 120.00 \\
3000.00 & \leq & x_3 & \leq & 3278.68 \\
85.00 & \leq & x_4 & \leq & 93.00 \\
92.18 & \leq & x_5 & \leq & 95.00 \\
3.78 & \leq & x_6 & \leq & 10.68 \\
145.00 & \leq & x_7 & \leq & 153.53
\end{array}$$

Convergence to the global minimum point,

$$\begin{aligned} x_1^* &= 1698.18 \\ x_2^* &= 53.66 \\ x_3^* &= 3031.30 \\ x_4^* &= 90.11 \\ x_5^* &= 95 \\ x_6^* &= 10.50 \\ x_7^* &= 153.53 \end{aligned}$$

having an objective function value of $f^* = 1227.00$ is reached from every initial point in about 200 iterations requiring 30 seconds of CPU time.

4.3 CSTR Sequence Design with Capital Cost Constraints

This example was first proposed in [42] and involves the design of a sequence of two CSTR reactors where the consecutive reaction $A \rightarrow B \rightarrow C$ takes place. The concentration C_{b2} of B in the outlet stream of the second reactor needs to be maximized subject to an upper bound on the investment cost. First order kinetics are assumed for both reactions and the reaction constants for the first (a) and second (b) reaction in reactors (1) and (2) are

$$\begin{aligned} k_{a1} &= 9.6540\ 10^{-2}\ s^{-1}, \\ k_{b1} &= 3.5272\ 10^{-2}\ s^{-1}, \\ k_{a2} &= 9.7515\ 10^{-2}\ s^{-1}, \\ k_{b2} &= 3.9191\ 10^{-2}\ s^{-1}. \end{aligned}$$

The inlet concentration of A is $c_{a0} = 1.0/, mol/l$, and zero for both B and C. If V_1, V_2 are the residence times for the two reactors and $c_{a1}, c_{b1}, c_{a2}, c_{b2}$ are the corresponding exit concentrations then the reactor design problem can be formulated as the following nonconvex optimization problem.

$$\min \quad -c_{b2}$$

$$\begin{aligned} \text{subject to} \qquad (c_{a1} - c_{a0}) + k_{a1} c_{a1} V_1 &= 0 \\ (c_{a2} - c_{a1}) + k_{a2} c_{a2} V_2 &= 0 \\ (c_{b1} + c_{a1} - c_{a0}) + k_{b1} c_{b1} V_1 &= 0 \\ (c_{b2} - c_{b1} + c_{a2} - c_{a1}) + k_{b2} c_{b2} V_2 &= 0 \\ V_1^{0.5} + V_2^{0.5} &\leq 4 \end{aligned} \tag{2}$$

$$0 \leq c_{a1}, c_{b1}, c_{a2}, c_{b2} \leq 1 \tag{3}$$

$$1 \leq V_1, V_2 \leq 100 \tag{4}$$

Assuming that the capital cost of a reactor is proportional to the square root of its residence time, constraint (2) provides an upper bound on the total investment costs. Also constraints (3) and (4) define some initial bounds on the variables. This problem has been reported to involve two local minima [42].

First, by applying the bound improving procedure described in subsection 2.5 the initial bounds are refined into the following:

$$\begin{array}{rcccl} 0.4090 & \leq & c_{a1} & \leq & 0.9119 \\ 0.2128 & \leq & c_{a2} & \leq & 0.8309 \\ 0.0845 & \leq & c_{b1} & \leq & 0.4501 \\ 0.0571 & \leq & c_{b2} & \leq & 0.7099 \\ 1.0000 & \leq & V_1 & \leq & 9.0000 \\ 1.0000 & \leq & V_2 & \leq & 9.0000 \end{array}$$

Application of the proposed global optimization procedure yields the following global optimum solution

$$\begin{array}{rcl} c_{a1}^* & = & 0.6587 \\ c_{a2}^* & = & 0.5161 \\ c_{b1}^* & = & 0.2869 \\ c_{b2}^* & = & 0.3866 \\ V_1^* & = & 5.3666 \\ V_2^* & = & 2.8338 \end{array}$$

from every initial point in about 400 iterations and CPU of 20 seconds with convergence and feasibility tolerances of 10^{-3}.

Next, the bounds of the reactor volumes are expanded to:

$$10^{-6} \leq V_1, V_2 \leq 16$$

and all concentration variables c_{a1}, c_{a2}, c_{b1} and c_{b2} are eliminated from the initial formulation for the reactor design problem. The resulting compact formulation involves only two variables V_1, V_2.

$$\min \quad -c_{b2} \;=\; -c_{a0}\frac{k_{a2}V_2\,(1+k_{b1}V_1)+k_{a1}V_1\,(1+k_{a2}V_2)}{(1+k_{a1}V_1)\,(1+k_{b1}V_1)\,(1+k_{a2}V_2)\,(1+k_{b2}V_2)}$$

$$\begin{array}{rrcl} \text{subject to} & V_1^{0.5}+V_2^{0.5} & \leq & 4 \\ & 10^{-6} \;\leq\; V_1, V_2 & \leq & 16 \end{array}$$

Note that unlike methods based on the bilinearization of the original problem [42, 43] the proposed approach can readily handle ratios of polynomial expressions and thus take advantage of the reduction of the total number of variables from six to only two. After transforming the fractional objective into a signomial form it takes only about 282 iterations and 17 seconds of CPU to locate the global optimum solution

($V_1^* = 15.992 \approx 16, V_2^* = 10^{-6} \approx 0$). This corresponds to over an order of magnitude reduction in the computational requirements over those reported in [42], (7950 iterations and 7950 seconds on an Apollo DN10000). This is attributed to the fact that not only no additional variables are introduced but also elimination of existing variables is possible. Consequently, the obtained lower bounds are much tighter and the branch and bound procedure is performed on a smaller set of variables.

Recently, Ryoo and Sahinidis [43] solved globally the same problem by convex lower bounding the bilinear terms for a different set of reaction constants. A number of variations on the main algorithm were considered and the best one yielded a CPU of 23 seconds on a SPARC 2.

$$\begin{aligned} k_{a1} &= 9.755988\ 10^{-2}\ s^{-1}, \\ k_{b1} &= 3.919080\ 10^{-2}\ s^{-1}, \\ k_{a2} &= 9.658428\ 10^{-2}\ s^{-1}, \\ k_{b2} &= 3.527172\ 10^{-2}\ s^{-1}. \end{aligned}$$

The approach proposed here, converged to the global optimum ($V_1^* = 3.037, V_2^* = 5.096$) after 299 iterations and about 20 seconds of CPU time. This is competitive with the CPU requirements in [43].

Computational results on a number of additional examples corresponding to optimal reactor design, heat exchanger design as well as standard test examples are shown in Table 4.

4.4 Robust Stability Analysis of Nonlinear Systems

Robust stability analysis of nonlinear systems involves the identification of the largest possible region in the uncertain model parameter space for which the controller manages to attenuate any disturbances in the system. The stability of a feedback structure is determined by the roots of the closed loop characteristic equation:

$$\det\left(I + P(s,\mathbf{q})C(s,\mathbf{q})\right) \;=\; 0$$

where $\mathbf{q}$ is the vector of the uncertain model parameters, and $P(s), C(s)$ the transfer functions of the plant and controller respectively. After expanding the determinant we have:

$$P(s,\mathbf{q}) \;=\; a_n(\mathbf{q})s^n + a_{n-1}(\mathbf{q})s^{n-1} + \cdots + a_1(\mathbf{q})s + a_0(\mathbf{q}) \;=\; 0$$

where the coefficients $a_i(\mathbf{q}),\ i = 0, \ldots, n$ are typically multivariable polynomial functions. The "zero exclusion condition" (ZEC) implies that a system with characteristic equation $P(\mathbf{q}, s) = 0$ is stable only if it does not have any roots on the imaginary axis for any realization of the $\mathbf{q}$'s in the uncertain model parameter space $\mathcal{Q}$.

$$0 \;\notin\; P(j\omega,\mathbf{q}),\ \ \forall\, \mathbf{q} \in \mathcal{Q},\ \text{and}\ \forall\, \omega \;\in\; [0,\infty]$$

A stability margin k_m can then be defined as follows:

$$k_m(j\omega) \;=\; \inf\left\{k \;:\; P\left(j\omega,\; \mathbf{q}(k)\right) \;=\; 0,\; \forall\, \mathbf{q} \;\in\; \mathcal{Q}\right\}$$

Robust stability for this model is then guaranteed if and only if:

$$k_m \;\geq\; 1$$

Geometrically, k_m expands the initial uncertain parameter region $\mathcal{Q}$ as much as possible without loosing stability (See Figure 3). Note that, typically real parameter uncertainty is expressed as bounds on the real parameters of the model.

Checking the stability of a particular system with characteristic equation $P(j\omega, \mathbf{q})$ involves the solution of the following nonconvex optimization problem.

$$\min_{q_i, k \geq 0, \omega \geq 0} \qquad k \qquad\qquad (\mathbf{S})$$

$$\begin{aligned}
\text{subject to} \quad Re\left[P(j\omega, \mathbf{q})\right] \;&=\; 0\\
Im\left[P(j\omega, \mathbf{q})\right] \;&=\; 0\\
q_i^N - \Delta q_i^-\, k \;\leq\; q_i \;&\leq\; q_i^N + \Delta q_i^+\, k, \quad i = 1, \ldots, n
\end{aligned}$$

where $\mathbf{q}^N$ is a stable nominal point for the uncertain parameters and $\Delta\mathbf{q}^+, \Delta\mathbf{q}^-$ are estimated bounds. Note that it is important to be able to always locate the global minimum of $(\mathbf{S})$, otherwise the stability margin might be overestimated. This overestimation can sometimes lead to the erroneous conclusion that a system is stable when it is not. Because for most problems without time delays $a_i(\mathbf{q}),\; i = 0, \ldots, n$ are multivariable polynomial functions, formulation $(\mathbf{S})$ corresponds to a generalized geometric problem. In the following, a number of stability analysis examples are considered.

4.4.1 Stability Analysis Examples

Problem 1

This example was studied in [45] and [46]. The plant has three uncertain parameters and the characteristic equation is:

$$P(s, q_1, q_2, q_3) \;=\; s^4 + (10 + q_2 + q_3)\, s^3 + (q_2 q_3 + 10 q_2 + 10 q_3)\, s^2 + (1 - q_2 q_3 + q_1)\, s + 2 q_1$$

The nominal values of the parameters of the system are

$$q_1^N = 800,\; q_2^N = 4,\; q_3^N = 6$$

and the bounded perturbations

$$\Delta q_1^+ = \Delta q_1^- = 800,\ \Delta q_2^+ = \Delta q_2^- = 2,\ \Delta q_3^+ = \Delta q_3^- = 3$$

After eliminating ω the zero exclusion formulation becomes:

$$\min \quad k$$

$$\begin{array}{ll} \text{subject to} & 10q2^2q_3^3 + 10q_2^3q_3^2 + 200q_2^2q_3^2 + 100q_2^3q_3 \\ & +100q_2q_3^3 + q_1q_2q_3^2 + q_1q_2^2q_3 + 1000q_2q_3^2 \\ & +8q_1q_3^2 + 1000q_2^2q_3 + 8q_1q_2^2 + 6q_1q_2q_3 \\ & +60q_1q_3 + 60q_1q_2 - q_1^2 - 200q_1 \ \leq \ 0 \end{array}$$

$$\begin{array}{rcccl} 800 - 800k & \leq & q_1 & \leq & 800 + 800k \\ 4 - 2k & \leq & q_2 & \leq & 4 + 2k \\ 6 - 3k & \leq & q_3 & \leq & 6 + 3k \end{array}$$

The stability margin is found to be equal to $k_m = 0.3417$ which implies that the system is unstable. Furthermore, the first instability occurs at:

$$\begin{array}{rcr} q_1^* & = & 1073.4 \\ q_2^* & = & 3.318 \\ q_3^* & = & 4.975 \end{array}$$

Computational requirements were 15 iterations and 0.5 seconds of CPU time.

Problem 2

This example examines the l_∞ stability margin for the closed–loop system described in [47]. The characteristic polynomial is:

$$P(s, q_1, q_2, q_3) \ = \ s^4 + \left(q_1^3q_2\right)s^3 + \left(q_1^2q_2^2q_3\right)s^2 + \left(q_1q_2^3q_3^2\right)s + q_3^3$$

The nominal parameter values are

$$q_1^N = 1.4,\ q_2^N = 1.5,\ q_3^N = 0.8$$

and the bounded perturbations

$$\Delta q_1^+ = \Delta q_1^- = 0.25,\ \Delta q_2^+ = \Delta q_2^- = 0.20,\ \Delta q_3^+ = \Delta q_3^- = 0.20$$

Again after eliminating ω the zero exclusion formulation becomes:

$$\min \quad k$$

$$\text{subject to} \qquad q_1^4 q_2^4 - q_1^4 - q_2^4 q_3 \;=\; 0$$

$$\begin{array}{rcccl} 1.4 - 0.25k & \leq & q_1 & \leq & 1.4 + 0.25k \\ 1.5 - 0.20k & \leq & q_2 & \leq & 1.5 + 0.20k \\ 0.8 - 0.20k & \leq & q_3 & \leq & 0.8 + 0.20k \end{array}$$

The stability margin is found to be equal to $k_m = 1.089$ implying that the system is stable. Furthermore, the closest to the nominal point unstable point is:

$$\begin{array}{rcl} q_1^* & = & 1.1275 \\ q_2^* & = & 1.2820 \\ q_3^* & = & 1.0179 \end{array}$$

Computational requirements were 26 iterations and 1.4 seconds of CPU time.

Problem 3

This example involves affine coefficient functions and it was fist proposed by [48]. The closed–loop characteristic polynomial is:

$$\begin{array}{rcl} P(s, q_1, q_2, q_3) & = & 16s^4 + (16q_1 - 24)\, s^3 + (-9.625q_1 - 16q_2 + 78)\, s^2 \\ & + & (19q_1 + 8q_2 + q_3 - 44)\, s + (q_3 - q_2 - 4q_1 + 12) \end{array}$$

The nominal parameter values are

$$q_1^N = 2.25,\ q_2^N = 1.5,\ q_3^N = 1.5$$

and the bounded perturbations

$$\Delta q_1^+ = \Delta q_1^- = 0.25,\ \Delta q_2^+ = \Delta q_2^- = 0.50,\ \Delta q_3^+ = \Delta q_3^- = 1.50$$

In this example ω is not eliminated and therefore the formulation becomes:

$$\min \ k$$

$$\begin{array}{lrcl} \text{subject to} & q_3 + 9.625q_1\omega + 16q_2\omega + 16\omega^2 + 12 - 4q_1 - q_2 - 78\omega & = & 0 \\ & 16q_1\omega + 44 - 19q_1 - 8q_2 - q_3 - 24\omega & = & 0 \end{array}$$

$$\begin{array}{rcccl} 2.25 - 0.25k & \leq & q_1 & \leq & 2.25 + 0.25k \\ 1.5 - 0.50k & \leq & q_2 & \leq & 1.5 + 0.50k \\ 1.5 - 1.50k & \leq & q_3 & \leq & 1.5 + 1.50k \end{array}$$

The stability margin is found to be equal to $k_m = 0.8175$ which means that the system is stable. Note that application of a local solver [38] consistently overestimated the stability margin as $k_m = 0.8765$. The critical parameter values for instability are:

$$\begin{aligned} q_1^* &= 2.4544 \\ q_2^* &= 1.9088 \\ q_3^* &= 2.7263 \\ \omega^* &= 1.3510 \end{aligned}$$

Computational requirements were 51 iterations and 5 seconds of CPU time.

Problem 4

This example studies the stability of the mechanical system proposed in [49]. The corresponding closed–loop characteristic polynomial is:

$$P(s, \mathbf{q}) = a_4(\mathbf{q})s^4 + a_3(\mathbf{q})s^3 + a_2(\mathbf{q})s^2 + a_1(\mathbf{q})s^1 + a_0(\mathbf{q})$$

where
$$\begin{aligned} a_4(\mathbf{q}) &= q_3^2 q_2 \left(4q_2 + 7q_1\right) \\ a_3(\mathbf{q}) &= 7q_4 q_3^2 q_2 - 64.918 q_3^2 q_2 + 380.067 q_3 q_2 + 3q_5 q_2 + 3q_5 q_1 \\ a_2(\mathbf{q}) &= 3\left(-9.81 q_3 q_2^2 - 9.81 q_3 q_1 q_2 - 4.312 q_3^2 q_2 + 264.896 q_3 q_2 + q_4 q_5 - 9.274 q_5\right) \\ a_1(\mathbf{q}) &= \frac{1}{5}\left(-147.15 q_4 q_3 q_2 + 1364.67 q_3 q_2 - 27.72 q_5\right) \\ a_0(\mathbf{q}) &= 54.387 q_3 q_2 \end{aligned}$$

The nominal parameter values are

$$q_1^N = 10.0,\ q_2^N = 1.0,\ q_3^N = 1.0,\ q_4^N = 0.2,\ q_5^N = 0.05$$

and the parameter perturbations

$$\Delta q_1^+ = \Delta q_1^- = 1.0, \Delta q_2^+ = \Delta q_2^- = 0.1, \Delta q_3^+ = \Delta q_3^- = 0.1, \Delta q_4^+ = \Delta q_4^- = 0.01, \Delta q_5^+ = \Delta q_5^- = 0.005$$

The zero exclusion formulation is:

$$\min \ k$$

$$\text{subject to} \qquad \begin{aligned} a_4(\mathbf{q})\omega^4 - a_2(\mathbf{q})\omega^2 + a_0(\mathbf{q}) &= 0 \\ a_3(\mathbf{q})\omega^2 - a_1(\mathbf{q}) &= 0 \end{aligned}$$

$$
\begin{array}{rcccl}
10.0 - 1.0k & \leq & q_1 & \leq & 10.0 + 1.0k \\
1.0 - 0.1k & \leq & q_2 & \leq & 1.0 + 0.1k \\
1.0 - 0.1k & \leq & q_3 & \leq & 1.0 + 0.1k \\
0.2 - 0.01k & \leq & q_4 & \leq & 0.2 + 0.01k \\
0.05 - 0.005k & \leq & q_5 & \leq & 0.05 + 0.005k
\end{array}
$$

Computational requirements were 3076 iterations and 485.75 seconds of CPU time.

Problem 5

This example addresses the stability of a wire guided Daimler Benz 0305 bus [48]. The transfer function of the linearized system is

$$
G(s, q_1, q_2) = \frac{q_1\,(48280 q_1 + 388600 s + 609.8 q_1 q_2 s^2)}{(16.8 q_1^2 q_2 + 270000 + 1077 q_1 q_2 s + q_1^2 q_2^2 s^2)\, s^3}.
$$

The transfer of the controller is

$$
C(s) = \frac{9375 + 10938 s + 2344 s^2}{15625 + 1250 s + 50 s^2 + s^3}.
$$

The closed–loop characteristic polynomial is

$$
P(s, q_1, q_2) = \sum_{i=0}^{8} a_i(q_1, q_2) s^i
$$

$$
\begin{array}{rrcl}
\text{where} & a_0(q_1, q_2) & = & 453\,10^6 q_1^2 \\
& a_1(q_1, q_2) & = & 528\,10^6 q_1^2 + 3640\,10^6 q_1 \\
& a_2(q_1, q_2) & = & 5.72\,10^6 q_1^2 q_2 + 113\,10^6 q_1^2 + 4250\,10^6 q_1 \\
& a_3(q_1, q_2) & = & 6.93\,10^6 q_1^2 q_2 + 911\,10^6 q_1 + 4220\,10^6 \\
& a_4(q_1, q_2) & = & 1.45\,10^6 q_1^2 q_2 + 16.8\,10^6 q_1 q_2 + 338\,10^6 \\
& a_5(q_1, q_2) & = & 15.6\,10^3 q_1^2 q_2^2 + 840 q_1^2 q_2 + 1.35\,10^6 q_1 q_2 + 13.5\,10^6 \\
& a_6(q_1, q_2) & = & 1.25\,10^3 q_1^2 q_2^2 + 16.8 q_1^2 q_2 + 53.9\,10^3 q_1 q_2 + 270\,10^3 \\
& a_7(q_1, q_2) & = & 50 q_1^2 q_2^2 + 1080 q_1 q_2 \\
& a_8(q_1, q_2) & = & q_1^2 q_2^2
\end{array}
$$

The nominal parameter values are

$$
q_1^N = 17.5,\ q_2^N = 20.0
$$

and the parameter perturbations

$$
\Delta q_1^+ = \Delta q_1^- = 14.5,\ \Delta q_2^+ = \Delta q_2^- = 15
$$

The zero exclusion formulation is:

$$\min \quad k$$

$$\begin{aligned}\text{subject to} \quad a_8(\mathbf{q})\omega^8 - a_6(\mathbf{q})\omega^6 + a_4(\mathbf{q})\omega^4 - a_2(\mathbf{q})\omega^2 + a_0(\mathbf{q}) &= 0 \\ a_7(\mathbf{q})\omega^6 - a_5(\mathbf{q})\omega^4 + a_3(\mathbf{q})\omega^2 - a_1(\mathbf{q}) &= 0\end{aligned}$$

$$\begin{aligned}17.5 - 14.5k \ \leq \ q_1 \ \leq \ 17.5 + 14.5k \\ 20.0 - 15.0k \ \leq \ q_2 \ \leq \ 20.0 + 15.0k\end{aligned}$$

The system was found to be stable, $k_m > 1.0$, for the given range of uncertain parameters. Computational requirements were 109 iterations and 22 seconds of CPU time.

Problem 6

In this example, an analysis of the stability margin of the spark ignition engine Fiat Dedra [50, 51] is carried out for a continuous range of operating conditions involving seven uncertain parameters. The closed–loop characteristic polynomial of seventh degree is:

$$P(s,\mathbf{q}) = \sum_{i=0}^{7} a_i(\mathbf{q}) s^i$$

where the characteristic polynomial coefficients $a_i(\mathbf{q})$, $i = 0,\ldots,7$ are polynomial functions of q_i, $i = 0,\ldots,7$.

$$a_7(\mathbf{q}) = q_7^2$$

$$a_6(\mathbf{q}) = 0.1586q_1q_7^2 + 2q_2q_7^2 + 2q_5q_7 + 0.1826q_6q_7 + 0.0552q_7^2$$

$$\begin{aligned}a_5(\mathbf{q}) = \ & 0.0189477q_1q_7^2 + 0.11104q_2q_7^2 + 0.1826q_5q_6 + 0.1104q_5q_7 + 0.0237398q_6q_7 \\ & +q_2^2q_7^2 + 0.1586q_1q_2q_7^2 + 0.0872q_1q_4q_7 + 0.0215658q_1q_6q_7 + 0.3652q_2q_6q_7 \\ & +2q_3q_4q_7 - 0.0848q_3q_6q_7 + q_5^2 + 7.61760\ 10^{-4}q_7^2 + 0.3172q_1q_5q_7 \\ & +4q_2q_5q_7\end{aligned}$$

$$\begin{aligned}a_4(\mathbf{q}) = \ & 0.1586q_1q_5^2 + 4.02141\ 10^{-4}q_1q_7^2 + 2q_2q_5^2 + 0.00152352q_2q_7^2 + 0.0237398q_5q_6 \\ & +0.00152352q_5q_7 + 5.16120\ 10^{-4}q_6q_7 + 0.0552q_2^2q_7^2 + 0.01898477q_1q_2q_7^2 \\ & +0.0872q_1q_4q_5 + 0.034862q_1q_4q_7 + 0.0215658q_1q_5q_6 + 0.00287416q_1q_6q_7 \\ & +0.0474795q_2q_6q_7 + 2q_3q_4q_5 + 0.1826q_3q_4q_6 + 0.1104q_3q_4q_7 - 0.0848q_3q_5q_6 \\ & -0.00234048q_3q_6q_7 + 2q_2^2q_5q_7 + 0.1826q_2^2q_6q_7 + 0.0872q_1q_2q_4q_7 + 0.3172q_1q_2q_5q_7 \\ & +0.0215658q_1q_2q_6q_7 + 0.1586q_1q_3q_4q_7 + 2q_2q_3q_4q_7 - 0.0848q_2q_3q_6q_7 + 0.0552q_5^2 \\ & +0.3652q_2q_5q_6 + 0.0378954q_1q_5q_7 + 0.2208q_2q_5q_7\end{aligned}$$

$$
\begin{aligned}
a_3(\mathbf{q}) \;=\;& 0.0189477 q_1 q_5^2 + 0.1104 q_2 q_5^2 + 5.16120\ 10^{-4} q_5 q_6 + q_2^2 q_5^2 \\
& +7.61760\ 10^{-4} q_2^2 q_7^2 + q_3^2 q_4^2 + 0.1586 q_1 q_2 q_5^2 + 4.02141\ 10^{-4} q_1 q_2 q_7^2 \\
& +0.0872 q_1 q_3 q_4^2 + 0.034862 q_1 q_4 q_5 + 0.00336706 q_1 q_4 q_7 + 0.00287416 q_1 q_5 q_6 \\
& +6.28987\ 10^{-5} q_1 q_6 q_7 + 0.00103224 q_2 q_6 q_7 + 0.1104 q_3 q_4 q_5 + 0.0237398 q_3 q_4 q_6 \\
& +0.00152352 q_3 q_4 q_7 - 0.00234048 q_3 q_5 q_6 + 0.1826 q_2^2 q_5 q_6 + 0.1104 q_2^2 q_5 q_7 \\
& +0.0237398 q_2^2 q_6 q_7 - 0.0848 q_3^2 q_4 q_6 + 0.0872 q_1 q_2 q_4 q_5 + 0.034862 q_1 q_2 q_4 q_7 \\
& +0.0215658 q_1 q_2 q_5 q_6 + 0.0378954 q_1 q_2 q_5 q_7 + 0.00287416 q_1 q_2 q_6 q_7 \\
& +0.1586 q_1 q_3 q_4 q_5 + 0.0215658 q_1 q_3 q_4 q_6 + 0.0189477 q_1 q_3 q_4 q_7 + 2 q_2 q_3 q_4 q_5 \\
& +0.1826 q_2 q_3 q_4 q_6 + 0.1104 q_2 q_3 q_4 q_7 - 0.0848 q_2 q_3 q_5 q_6 - 0.00234048 q_2 q_3 q_6 q_7 \\
& +7.61760\ 10^{-4} q_5^2 + 0.0474795 q_2 q_5 q_6 + 8.04282\ 10^{-4} q_1 q_5 q_7 + 0.00304704 q_2 q_5 q_7
\end{aligned}
$$

$$
\begin{aligned}
a_2(\mathbf{q}) \;=\;& 4.02141\ 10^{-4} q_1 q_5^2 + 0.00152352 q_2 q_5^2 + 0.0552 q_2^2 q_5^2 + 0.0552 q_3^2 q_4^2 \\
& +0.0189477 q_1 q_2 q_5^2 + 0.034862 q_1 q_3 q_4^2 + 0.00336706 q_1 q_4 q_5 \\
& +6.82079\ 10^{-5} q_1 q_4 q_7 + 6.28987\ 10^{-5} q_1 q_5 q_6 + 0.00152352 q_3 q_4 q_5 \\
& +5.16120\ 10^{-4} q_3 q_4 q_6 - 0.00234048 q_3^2 q_4 q_6 + 0.034862 q_1 q_2 q_4 q_5 \\
& +0.0237398 q_2^2 q_5 q_6 + 0.00152352 q_2^2 q_5 q_7 + 5.16120\ 10^{-4} q_2^2 q_6 q_7 \\
& +0.00336706 q_1 q_2 q_4 q_7 + 0.00287416 q_1 q_2 q_5 q_6 + 8.04282\ 10^{-4} q_1 q_2 q_5 q_7 \\
& +6.28987\ 10^{-5} q_1 q_2 q_6 q_7 + 0.0189477 q_1 q_3 q_4 q_5 + 0.00287416 q_1 q_3 q_4 q_6 \\
& +4.02141\ 10^{-4} q_1 q_3 q_4 q_7 + 0.1104 q_2 q_3 q_4 q_5 + 0.0237398 q_2 q_3 q_4 q_6 \\
& +0.00152352 q_2 q_3 q_4 q_7 - 0.00234048 q_2 q_3 q_5 q_6 + 0.00103224 q_2 q_5 q_6
\end{aligned}
$$

$$
\begin{aligned}
a_1(\mathbf{q}) \;=\;& 7.61760\ 10^{-4} q_2^2 q_5^2 + 7.61760\ 10^{-4} q_3^2 q_4^2 + 4.02141\ 10^{-4} q_1 q_2 q_5^2 \\
& +0.00336706 q_1 q_3 q_4^2 + 6.82079\ 10^{-5} q_1 q_4 q_5 + 5.16120\ 10^{-4} q_2^2 q_5 q_6 \\
& +0.00336706 q_1 q_2 q_4 q_5 + 6.82079\ 10^{-5} q_1 q_2 q_4 q_7 + 6.28987\ 10^{-5} q_1 q_2 q_5 q_6 \\
& +4.02141\ 10^{-4} q_1 q_3 q_4 q_5 + 6.28987\ 10^{-5} q_1 q_3 q_4 q_6 + 0.00152352 q_2 q_3 q_4 q_5 \\
& +5.16120\ 10^{-4} q_2 q_3 q_4 q_6
\end{aligned}
$$

$$
a_0(\mathbf{q}) \;=\; 6.82079\ 10^{-5} q_1 q_3 q_4^2 + 6.82079\ 10^{-5} q_1 q_2 q_4 q_5
$$

The nominal parameter values and perturbations are

$$
\begin{array}{llllllll}
q_1^N &=& 3.4329, & \Delta q_1^+ &=& 0, & \Delta q_1^- &=& 1.2721 \\
q_2^N &=& 0.1627, & \Delta q_2^+ &=& 0, & \Delta q_2^- &=& 0.06 \\
q_3^N &=& 0.1139, & \Delta q_3^+ &=& 0, & \Delta q_3^- &=& 0.0782 \\
q_4^N &=& 0.2539, & \Delta q_4^+ &=& 0.3068, & \Delta q_4^- &=& 0 \\
q_5^N &=& 0.0208, & \Delta q_5^+ &=& 0, & \Delta q_5^- &=& 0.0108 \\
q_6^N &=& 2.0247, & \Delta q_6^+ &=& 2.4715, & \Delta q_6^- &=& 0 \\
q_7^N &=& 1.0000, & \Delta q_7^+ &=& 9.0000, & \Delta q_7^- &=& 0
\end{array}
$$

The zero exclusion formulation yields:

$$\min \quad k$$

$$\begin{array}{rrcl}\text{subject to} & -a_6(\mathbf{q})\omega^6 + a_4(\mathbf{q})\omega^4 + -a_2(\mathbf{q})\omega^2 + a_0(\mathbf{q}) & = & 0 \\ & a_7(\mathbf{q})\omega^6 - a_5(\mathbf{q})\omega^4 + a_3(\mathbf{q})\omega^2 - a_1(\mathbf{q}) & = & 0\end{array}$$

$$\begin{array}{rcccl} 3.4329 - 1.2721k & \leq & q_1 & \leq & 3.4329 \\ 0.1627 - 0.06k & \leq & q_2 & \leq & 0.1627 \\ 0.1139 - 0.0782k & \leq & q_3 & \leq & 0.1139 \\ 0.2539 & \leq & q_4 & \leq & 0.2539 + 0.3068k \\ 0.0208 - 0.0108k & \leq & q_5 & \leq & 0.0208 \\ 2.0247 & \leq & q_6 & \leq & 2.0247 + 2.4715k \\ 1.0000 & \leq & q_7 & \leq & 1.0000 + 9.0000k \end{array}$$

The system was found to be stable $k_m > 1.0$, throughout the entire range of the uncertain parameters. Computational requirements were 4,896 iterations and about 10,000 seconds of CPU time.

5 Summary and Conclusions

In this paper a deterministic branch and bound type global optimization algorithm was proposed for solving generalized geometric problems (signomials) **(GGP)**. This class of optimization problems has been extensively used to model a host of different engineering design and robust stability problems. An exponential variable transformation was employed to the initial nonconvex problem **(GGP)** to reduced it into a **(DC)** programming problem. A convex relaxation **(R)** of problem **(DC)** is then obtained based on the linear lower bounding of the concave parts of the objective function and constraints. The proposed algorithm was shown to attain finite $\epsilon-$convergence to the global minimum through the successive refinement of a convex relaxation of the feasible region and/or of the objective function and the subsequent solution of a series of nonlinear convex optimization problems. A number of procedures aimed at improving the efficiency of the proposed approach are also discussed. The proposed approach was applied to a number of small to medium size engineering design problems and a number of test as well as real size robust stability analysis problems. In all cases, convergence to the global minimum was achieved. Convergence was particularly expedient for problems with a small number of variables participating in negative monomial terms and preferably a small number of negative monomial terms.

Acknowledgments

Financial support from the National Science Foundation under Grants CBT–8857013, CTS–9221411, the Binational Science Foundation as well as Exxon Co. and Mobil Corporation is gratefully acknowledged.

References

[1] Passy U. and D.J. Wilde, Generalized polynomial optimizations, *SIAM J. Appl. Math.*, **15**, 1344, (1967).

[2] Blau G.E. and D.J. Wilde, Generalized Polynomial Programming, *Can. J. Chem. Eng.*, **47**, 317, (1969).

[3] Rijckaert M.J. and X.M. Martens, Bibliographical note on geometric programming, *JOTA*, **2**, 325, (1978).

[4] Ecker J.G. and R.D. Wiebking, Optimal design of a dry–type natural–draft cooling tower by geometric programming, *JOTA*, **26**, 305, (1978).

[5] Duffin R.J. and E.L. Peterson, Duality theory for geometric programming, *SIAM J. Appl. Math.*, **14**, 1307, (1966).

[6] Blau G.E. and D.J. Wilde, A lagrangian algorithm for equality constrained generalized polynomial optimization, *AIChE*, **17**, 235, (1971).

[7] Avriel M. and D.J. Wilde, Optimal Condenser Design by Geometric Programming, *I&EC Proc. Des. Dev.*, **6**, 256, (1967).

[8] Hellinckx L.J. and M.J. Rijckaert, Optimal Capacities of Production Facilities: An Application of geometric programming, *Can. J. Chem. Eng.*, **50**, 148, (1972).

[9] Dembo R.S, Optimal design of a membrane separation process using signomial programming, *Math. Prog.*, **15**, 12, (1978).

[10] Passy U. and D.J. Wilde, A geometric programming algorithm for solving chemical equilibrium problems, *SIAM Review*, **11**, 89, (1967).

[11] Jefferson T.R. and C.H. Scott, Generalized geometric programming applied to problems of optimal control: I. Theory, *JOTA*, **26**, 117, (1978).

[12] Salomone H.E. and O.A. Iribarren, Posynomial modeling of batch plants: A procedure to include process decision variables, *Computers Chem. Engnr*, **16**, 173, (1992).

[13] Hellinckx L.J. and M.J. Rijckaert, Minimization of capital investment for batch processes, *I&EC Proc. Des. Dev.*, **10**, 422, (1971).

[14] Avriel M. and V. Gurovich, Optimal Condenser Design by Geometric Programming, *I&EC Proc. Des. Dev.*, **6**, 256, (1967).

[15] Duffin R.J., Linearizing geometric problems, *SIAM Review*, **12**, 211, (1970).

[16] Avriel M. and A.C. Williams, Complementary geometric programming, *SIAM J. Appl. Math.*, **19**, 125, (1970).

[17] Avriel M., R. Dembo and U. Passy, Solution of generalized geometric programs, *SIAM Intern. J. Numer. Methods Engnr.*, **26**, 291, (1975).

[18] Duffin R.J. and E.L. Peterson, Reversed geometric programming treated by harmonic means, *Indiana Univ. Math. J.*, **22**, 531, (1972).

[19] Kuester J.L. and J.H. Mize, Optimization Techniques with Fortran, McGraw–Hill Book Company, New York, NY, (1973).

[20] Rijckaert and X.M. Martens, Comparison of generalized geometric programming algorithms, *JOTA*, **26**, 205, (1978).

[21] Ratner M., L.S. Lasdon and A. Jain, Solving geometric problems using GRG: Results and comparisons, *JOTA*, **26**, 253, (1978).

[22] Kochenberger A., R.E.D. Woolsey and B.A. McCarl, On the solution of geometric programming via separable programming, *Oper. Res. Quart.*, **24**, 285, (1973).

[23] Abrams R.A. and C.T. Wu, Projection and restriction methods in geometric and related problems, *JOTA*, **26**, 59, (1978).

[24] J. Bradley, An algorithm for the numerical solution of prototype geometric programs, Institute of Industrial Research and Standards, Dublin, Ireland, (1973).

[25] Beck P.A. and J.G. Ecker, A modified concave simplex algorithm for geometric programming, *JOTA*, **15**, 189, (1975).

[26] Haarhoff P.C. and J.D. Buys, A new method for the optimization of a nonlinear function subject to nonlinear constraints, *Comp. J.*, **13**, 178, (1970).

[27] Dembo R.S., Current state of the art of algorithms and computer software for geometric programming, *JOTA*, **26**, 149, (1978).

[28] Floudas, C.A. and V. Visweswaran, A global optimization algorithm (GOP) for certain classes of nonconvex NLP's: I. Theory., *Comput. chem. engng.*, **12**, 14, 1397, (1990).

[29] Floudas, C.A. and V. Visweswaran, A primal–relaxed dual global optimization approach, *Journal of Optimization Theory and Applications*, **78**, 2, 187, (1993).

[30] Visweswaran, V. and C.A. Floudas, New properties and computational improvement of the GOP algorithm for problems with quadratic objective functions and constraints, *Journal of Global Optimization*, **3**, 439, (1993).

[31] Liu, W.B. and C.A. Floudas, A remark on the GOP algorithm for global optimization, *Journal of Global Optimization*, **3**, 519, (1994).

[32] Maranas, C.D. and C.A. Floudas A Global Optimization Approach for Lennard–Jones Microclusters, *J. Chem. Phys.*, **97**, 10, 7667, (1992).

[33] Maranas, C.D. and C.A. Floudas Global Optimization for Molecular Conformation Problems, *Annals of Operations Research*, **42**, 85, (1993).

[34] Maranas, C.D. and C.A. Floudas Global Minimum Potential Energy Conformations of Small Molecules, *Journal of Global Optimization*, **4**, 135, (1994).

[35] Maranas, C.D. and C.A. Floudas A Deterministic Global Optimization Approach for Molecular Structure Determination, *J. Chem. Phys.*, **100** 2, 1247, (1994).

[36] Falk J.E., Global Solutions of Signomial Programs, *Report*, The George Washington University, Program in Logistics, (1973).

[37] Hansen P., B. Jaumard and S.H. Lu, Some Further Results on Monotonicity in Globally Optimal Design, *Journal of Mechanisms, Transmissions, and Automation in Design*, **111**, 345, (1989).

[38] B.A. Murtagh and M.A. Saunders, *MINOS 5.3 User's Guide*, (Systems Optimization Laboratory, Department of Operations Research, Stanford University, 1987).

[39] Horst R. and H. Tuy, *Global Optimization, Deterministic Approaches*, (Springer–Verlag, Berlin 1990).

[40] Avriel M., R. Dembo and U. Passy, Solution of Generalized Geometric Programs, *International Journal for Numerical Methods in Engineering*, **9**, 149, (1975).

[41] Dembo R.S., A Set of Geometric Programming Test Problems and their Solutions, *Math. Prog.*, **10**, 192, (1976).

[42] Manousiouthakis V. and D. Sourlas, A Global Optimization Approach to Rationally Constrained Rational Programming, *Che. Eng. Comm.*, **115**, 127, (1992).

[43] Ryoo H.S. and N.V. Sahinidis, Global Optimization of Nonconvex NLPs and MINLPs with Applications in Process Design, *Computers chem. Engng*, **19**(5), 551, (1995).

[44] Avriel M. and A.C. Williams, An Extension of Geometric Programming with Applications in Engineering Optimization, *Journal of Engineering Mathematics*, **5**(3), 187, (1971).

[45] Daston R.R.E. and M.G. Safonov, Exact Calculation of the Multi–Loop Stability Margin, *IEEE Trans. Automat. Contr.*, **33**, 68, (1988).

[46] Psarris P. and C.A. Floudas, Robust Stability Analysis of Linear and Nonlinear Systems with Real Parameter Uncertainty, submitted, (1994).

[47] Sideris A. and R.S. Sànchez Peña, Fast Computation of the Multivariable Stability Margin for Real Interrelated Uncertain Parameters, in Proc. ACC, Atlanta, Ga, (1988).

[48] Ackermann J., D. Kaesbauer and R. Muench, Robust Gamma-Stability Analysis in a Plant Parameter Space, *Automatica*, **27**, 75, (1991.

[49] Vicino A., A. Tesi and M. Milanese Computation of Nonconservative Stability Perturbation Bounds for Systems with Nonlinearly Correlated Uncertainties, *IEEE Trans. Autom. Contr.*, **35**, 835, (1990).

[50] Barmish, B.R., *New Tools for Robustness of Linear Systems* (Macmillan Publishing Company, New York, 1994).

[51] Abate M., B.R. Barmish, C. Murillo–Sanchez, R. Tempo, Application of Some New Tools to Robust Stability Analysis of Spark Ignition Engines: A Case Study, *IEEE Trans. Contr. Syst. Tech.*, **2**, 1, 22, (1994).

Table 1: Maximum separation as a function of interval width

$\delta = Y^U - Y^L$	$\Delta_{max} / \exp\left(Y^L\right)$
0.00026	10^{-8}
0.00275	10^{-6}
0.02808	10^{-4}
0.26449	10^{-2}
1.75079	1
2	1.51572
5	71.2807
10	14752.3

Table 2: Number of Iterations as a function of U

U	Iterations
1.1	17
1.5	18
2	19
5.5	20
10	21
100	27
1000	58

Table 3: Coefficients for alkylation example

i	c_i	i	c_i	i	c_i
1	1.715	16	0.19120592 E-1	31	0.00061000
2	0.035	17	0.56850750 E+2	32	0.0005
3	4.0565	18	1.08702000	33	0.81967200
4	10.000	19	0.32175000	34	0.81967200
5	3000.0	20	0.03762000	35	24500.0
6	0.063	21	0.00619800	36	250.0
7	0.59553571 E-2	22	0.24623121 E+4	37	0.10204082 E-1
8	0.88392857	23	0.25125634 E+2	38	0.12244898 E-4
9	0.11756250	24	0.16118996 E+3	39	0.00006250
10	1.10880000	25	5000.0	40	0.00006250
11	0.13035330	26	0.48951000 E+6	41	0.00007625
12	0.00660330	27	0.44333333 E+2	42	1.22
13	0.66173269 E-3	28	0.33000000	43	1.0
14	0.17239878 E-1	29	0.02255600	44	1.0
15	0.56595559 E-2	30	0.00759500		

Table 4: Computational results on additional examples.

Example	# Var.	# Neg. Mon.	Solution	Iter.	CPU
Heat exchanger design [44]	8	5	7049.24	1600	100
Optimal reactor design [41]	8	2	3.9511	71	6.8
Colville's test problem [20]	5	4	1.1436	30	2.0
Problem 10 of [20]	3	2	-83.254	50	17
Problem 11 of [20]	4	1	-5.7398	7	0.4
Problem 12 of [20]	8	2	-6.0482	82	12
Problem 14 of [20]	10	2	1.1436	290	22
Problem 17 of [20]	11	5	0.1406	2950	427

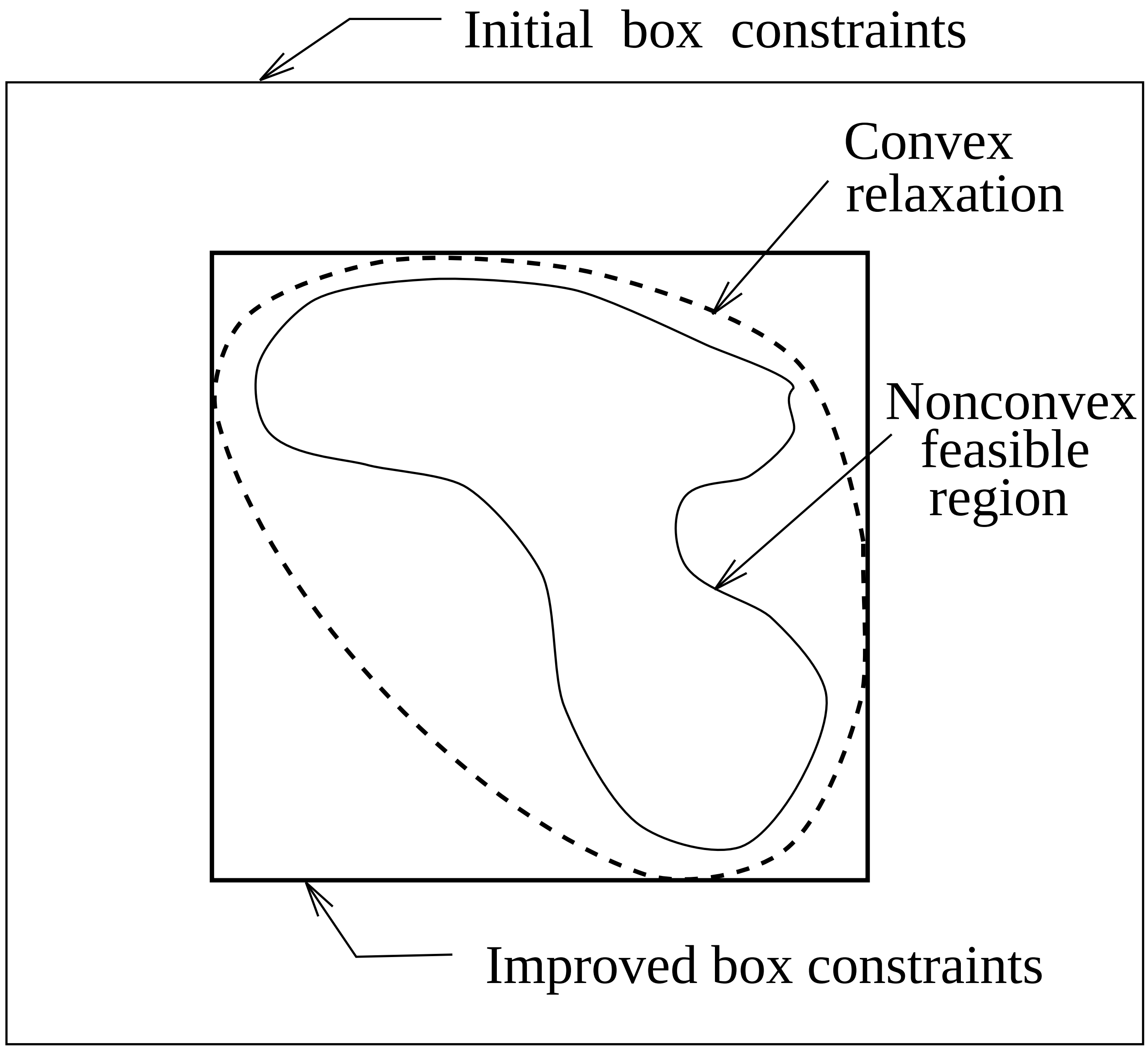

Figure 1: Initial bounds and feasible region, convex relaxation and improved bounds

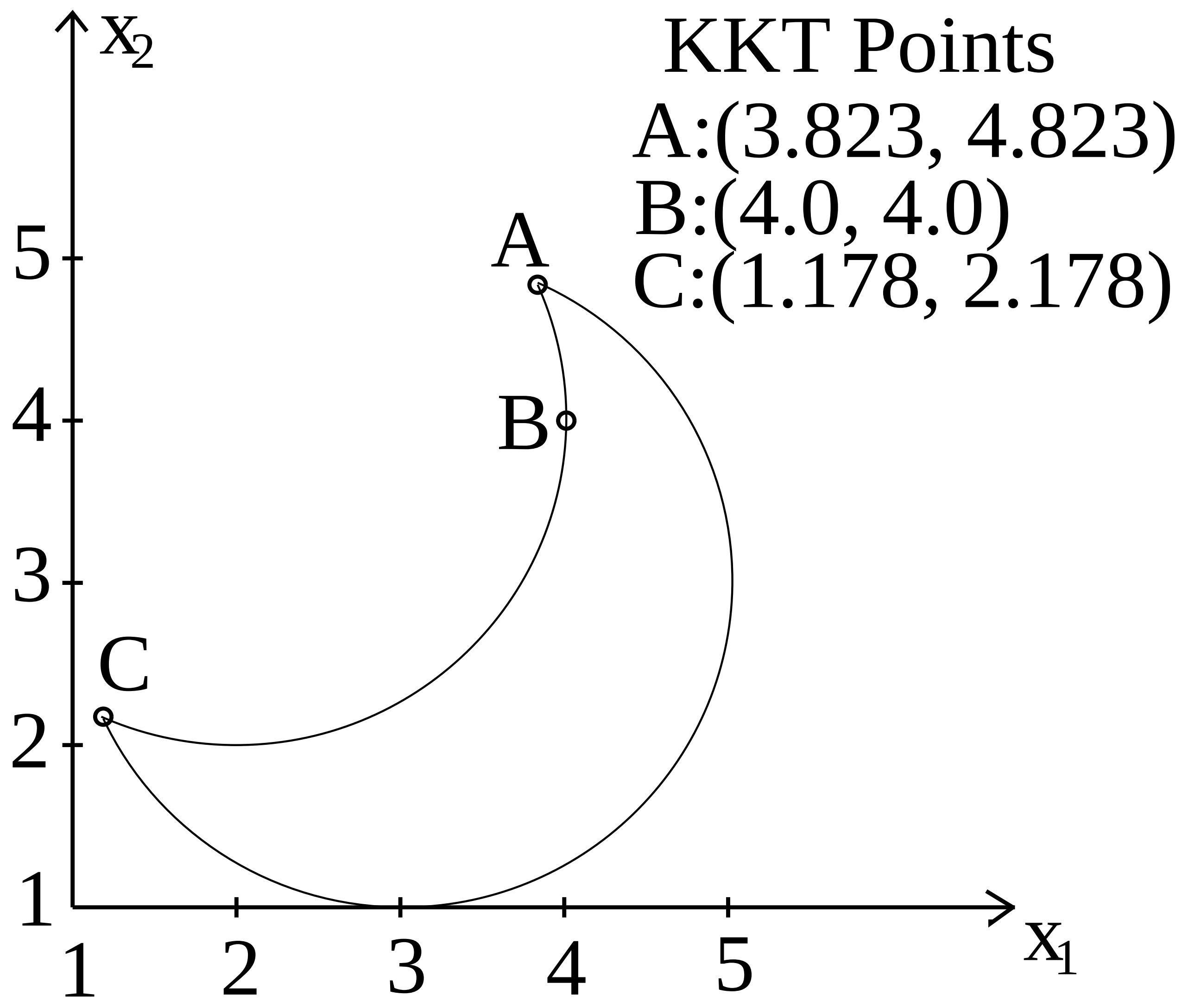

Figure 2: Feasible region and KKT points of motivating example

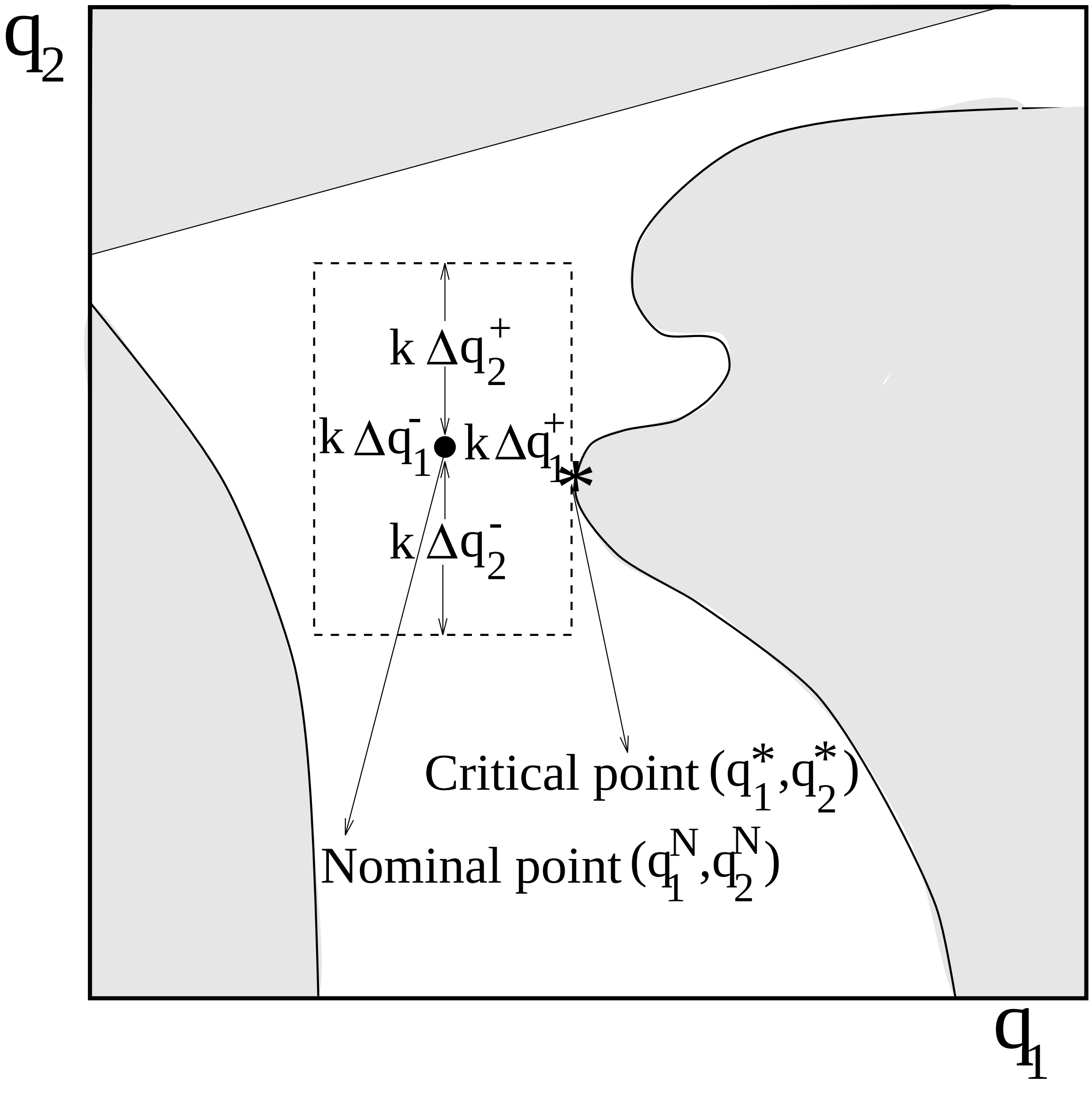

Figure 3: Stability Margin