A reduction method for semi-infinite programming by means of a global stochastic approach

Ana I.P.N. Pereira[1] and Edite M.G.P. Fernandes[2]

[1] Polytechnic Institute of Braganca, Braganca, Portugal, apereira@ipb.pt
[2] University of Minho, Braga, Portugal, emgpf@dps.uminho.pt

Summary. We describe a reduction algorithm for solving semi-infinite programming problems. The proposed algorithm uses the simulated annealing method equipped with a function stretching as a multi-local procedure, and a penalty technique for the finite optimization process. An exponential penalty merit function is reduced along each search direction to ensure convergence from any starting point. Our preliminary numerical results seem to show that the algorithm is very promising in practice.

Key words: Semi-infinite programming. Reduction method. Simulated annealing. Penalty method. Exponential function.

1 Introduction

We consider the semi-infinite programming (SIP) problem in the form:

$$\text{SIP}: \min f(x) \text{ subject to } g(x,t) \leq 0, \text{ for every } t \in T$$

where $T \subseteq \mathbb{R}^m$ is a compact set defined by $T = \{t \in \mathbb{R}^m : h_j(t) \geq 0, \text{ for } j \in J\}$, J is a set with finite cardinality, $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^n \times T \to \mathbb{R}$ are twice continuously differentiable functions with respect to x, g is a continuously differentiable function with respect to t and h_j are twice continuously differentiable functions with respect to t, for $j \in J$.

SIP problems appear in the optimization of a finite number of decision variables that are subject to an (potentially) infinite number of constraints. Such problems arise in many engineering areas, such as, computer aided design, air pollution control and production planning. For a more thorough review of applications the reader is referred to [9, 23, 25].

The numerical methods that are mostly used to solve SIP problems generate a sequence of finite problems. There are three main ways of generating the sequence: by discretization, exchange and reduction methods.

In a discretization method, a finite problem is obtained by replacing the infinite set by a finite subset of T (usually defined by a grid). As the process develops the grid is refined until a stopping criterion is satisfied. The increase of the number of constraints as the process develops is its main disadvantage (e.g. [9, 22, 29]).

In an exchange method (or semi-continuous method), a change of constraints is made, at each iteration, where some "old" constraints may be removed and/or some "new" constraints may be added (see e.g. [9, 22]).

A reduction method (or continuous method), under some assumptions, replaces the SIP problem by a locally reduced finite problem. The need to compute all the local maximizers of the constraint function is the main drawback of this type of method, unless an efficient multi-local algorithm is provided.

In this paper we propose a new reduction algorithm to find a stationary point of the SIP problem. It is based on a global stochastic method (simulated annealing) combined with a function stretching technique, as the multi-local procedure, and on a penalty technique as a finite optimization procedure. A line search with an exponential merit function is incorporated in the algorithm to promote global convergence. The novelty here is related to a local implementation of the stretching technique that coupled with the simulated annealing method is able to compute sequentially the local maxima of a multimodal function (the constraint function). We also propose a new continuous exponential merit function, denoted E_∞, to be used in the globalization procedure that performs well when compared with a standard exponential merit function.

The paper is organized as follows. In Section 2, we present the basic ideas behind the local reduction to finite problems. Section 3 is devoted to the global reduction method herein proposed. The general algorithm is presented in Subsection 3.1, the new multi-local algorithm is described in Subsection 3.2, and the penalty technique for solving the reduced finite problem is explained in Subsection 3.3. Subsection 3.4 describes the globalization procedure for the reduction method, and Subsection 3.5 presents the termination criteria of the iterative process. The Section 4 contains some numerical results, including a comparison with other reduction methods, and the conclusions and ideas for future work make Section 5.

2 Local reduction to a finite problem

A reduction method is based on the local reduction theory proposed by Hettich and Jongen [8]. The main idea of this theory is to replace the SIP problem by a sequence of finite problems. We shortly describe the main ideas of the local reduction theory.

For a given $\bar{x} \in \mathbb{R}^n$, consider the following so-called lower-level problem

$$O(\bar{x}) : \max_{t \in T} g(\bar{x}, t).$$

Let $T^{\bar{x}} = \{t^1, ..., t^{|L(\bar{x})|}\}$ be the set of its local solutions that satisfy the following condition

$$\left|g\left(\bar{x}, t^l\right) - g^*\right| \leq \delta^{ML}, l \in L(\bar{x}), \tag{1}$$

where $L(\bar{x})$ represents the index set of $T^{\bar{x}}$, δ^{ML} is a positive constant and g^* is the global solution value of $O(\bar{x})$. Condition (1) aims to generate a finite programming problem with few constraints that is, locally, equivalent to the SIP problem. Other conditions with similar goals are presented by [27, 30, 31].

Define the active index set, at a point $\bar{t}$, as $J_0(\bar{t}) = \{j \in J : h_j(\bar{t}) = 0\}$ and the Lagrangian function associated to the problem $O(\bar{x})$ as

$$\bar{L}(\bar{x}, t, u) = g(\bar{x}, t) + \sum_{j=1}^{|J|} u_j h_j(t),$$

where u_j represents the Lagrange multiplier associated to the constraint $h_j(t)$.

Throughout this section, we assume that the following condition is satisfied.

Condition 1 *The linear independence constraint qualification (LICQ) holds at $\bar{t} \in T$, i.e.,*

$$\{\nabla h_j(\bar{t})\}_{j \in J_0(\bar{t})}$$

are linearly independent.

Let $F(\bar{t}) = \{\xi \in \mathbb{R}^m : \xi^T \nabla h_j(\bar{t}) = 0, j \in J_0(\bar{t})\}$ be the tangent set.

Definition 1. ([11])

(a) *A point $\bar{t} \in T$ is called a critical point of $O(\bar{x})$ if there exist $\bar{u}_j$, $j \in J_0(\bar{t})$, such that*

$$\nabla_t \bar{L}(\bar{x}, \bar{t}, \bar{u}) = \nabla_t g(\bar{x}, \bar{t}) + \sum_{j \in J_0(\bar{t})} \bar{u}_j \nabla h_j(\bar{t}) = 0.$$

(b) *A point $\bar{t} \in T$ is called a nondegenerate critical point if the following conditions are satisfied:*

(i) *$\bar{t} \in T$ is a critical point;*

(ii) *$\bar{u}_j \neq 0$, for $j \in J_0(\bar{t})$, and*

(iii) *$\forall \xi \in F(\bar{t}) \backslash \{0\}$, $\xi^T \nabla^2_{tt} \bar{L}(\bar{x}, \bar{t}, \bar{u}) \xi \neq 0$.*

To be able to apply the theory of the local reduction to the SIP problem, the set $T^{\bar{x}}$ must be finite.

Definition 2. ([22]) *Problem $O(\bar{x})$, with $\bar{x} \in \mathbb{R}^n$, is said to be regular if all critical points of $O(\bar{x})$ are nondegenerate.*

When $\bar{x}$ is a feasible point and $O(\bar{x})$ is a regular problem, each local maximizer of the problem $O(\bar{x})$ is nondegenerate and consequently an isolated local maximizer. Since the set T is compact, then there exists a finite number of local maximizers of the problem $O(\bar{x})$. As each $t^l \in T^{\bar{x}}$ is an isolated point, the implicit function theorem can be applied.

Then there exist open neighborhoods $U(\bar{x})$, of $\bar{x}$, and $V(t^l)$, of t^l, and implicit functions $t^1(x), ..., t^{|L(\bar{x})|}(x)$ defined as:

i) $t^l : U(\bar{x}) \rightarrow V(t^l) \cap T$, for $l \in L(\bar{x})$;

ii) $t^l(\bar{x}) = t^l$, for $l \in L(\bar{x})$;

iii) $\forall x \in U(\bar{x})$, $t^l(x)$ is a nondegenerate and isolated local maximizer of the problem $O(\bar{x})$.

We may then conclude that $\{x \in U(\bar{x}) : g(x,t) \leq 0, \text{ for every } t \in T\} \Leftrightarrow \{x \in U(\bar{x}) : g^l(x) \equiv g(x, t^l(x)) \leq 0, l \in L(\bar{x})\}$.

So, when the problem $O(\bar{x})$ is regular, it is possible to replace the infinite constraints of the SIP problem by finite constraints that are, locally, sufficient to define the feasible region. This finite locally reduced problem is defined by

$$P_{red}(\bar{x}) : \min_{x \in U(\bar{x})} f(x) \text{ subject to } g^l(x) \leq 0, \ l \in L(\bar{x}).$$

For all $x \in U(\bar{x})$, $g^l(x)$ is a twice continuously differentiable function with respect to x, and x is a feasible point if and only if $g^l(x) \leq 0$. It can be proven that $x^* \in U(\bar{x})$ is a strict, isolated local minimizer of the SIP problem if and only if x^* is a strict, isolated local minimizer of the problem $P_{red}(\bar{x})$ [22].

3 A global reduction method for SIP

Consider the following condition. Let $x^k \in \mathbb{R}^n$ be an approximation to the solution of the SIP problem.

Condition 2 *The problem $O(x^k)$ is regular.*

In what follows, we assume that both Conditions 1 and 2 are satisfied. Thus a reduction type method for SIP resorts in an iterative process, indexed by k, with two major steps. Firstly, all local solutions of $O(x^k)$ that satisfy (1) should be computed; then some iterations of a finite programming method should be implemented to solve the reduced problem $P_{red}(x^k)$.

If a globalization technique is incorporated in the algorithm, the iterative process is known as a global reduction method (GRM).

Some procedures to find the local maximizers of the constraint function usually consist of two phases: first, a discretization of the set T is made and all maximizers are evaluated on that finite set; second, a local method is applied in order to increase the accuracy of the approximations found in the first phase (e.g. [1, 30, 31]). Recently, other strategies based on stochastic methods that incorporate descent methods have also been proposed (e.g. [13, 19]).

In reduction type methods, the most used finite programming algorithm is the sequential quadratic programming method [7] with augmented Lagrangian and BFGS updates [5], with L_1 and L_∞ merit functions or trust regions [1, 20, 27, 30].

3.1 General scheme for the GRM

The global reduction method consists of three main phases. These are the multi-local optimization procedure, the computation of a search direction solving a reduced finite optimization problem, and a line search with a merit function to ensure global convergence. The general scheme can be given as in Algorithm 1.

Algorithm 1

Given an initial approximation x^0. Set $k = 0$.

Step 1: *Compute the local solutions of $O(x^k)$.*

Step 2: *Solve $P_{red}(x^k)$.*

Step 3: *Analyze the globalization procedure.*

Step 4: *If the termination criteria are not satisfied go to Step 1 with $k = k + 1$.*

Details for each step of this algorithm follow.

3.2 Multi-local procedure

The multi-local procedure is used to compute all the local solutions of the problem $O(x^k)$ that satisfy (1). We propose to use a simulated annealing (SA) algorithm which is a stochastic global optimization method that does not require any derivative information and specific conditions on the objective function. This type of algorithm is well documented in the literature and we refer to [2, 3, 10, 17, 24, 26] for details. To guarantee convergence to a global solution with probability one we choose to implement the adaptive simulated annealing (ASA) variant proposed by [10].

In general, global optimization methods find just one global solution. To be able to compute multiple solutions, deflation techniques have to be incorporated in the algorithm. We propose the use of a SA algorithm equipped with a function stretching technique to be able to compute sequentially the local solutions of problem $O(x^k)$. Consider for simplicity the following notation $g(t) = g(x^k, t)$.

The function stretching technique was initially proposed in [16], in a particle swarm context, to provide a way to escape from a local solution, driving

the search to a global one. When a local (non-global) solution, $\widehat{t}$, is detected, this technique reduces by a certain amount the objective function values at all points t that verify $g(t) < g(\widehat{t})$, remaining $g(t)$ unchanged for all t such that $g(t) \geq g(\widehat{t})$. The process is repeated until the global solution is encountered.

In our case, the ASA algorithm guarantees the convergence to a global maximum of the objective function. However, as we need to compute global as well as local solutions, the inclusion of the function stretching technique aims to prevent the convergence of the ASA algorithm to an already detected solution. Let t^1 be the first computed global solution. At this point $T^{x^k} = \{t^1\}$. The function stretching technique is then applied, locally, in order to transform $g(t)$ in a neighborhood of t^1, say $V_\varepsilon(t^1)$, $\varepsilon > 0$. So, the decrease of $g(t)$ is carried out only on the region $V_\varepsilon(t^1)$ leaving all the other maxima unchanged. The maximum $g(t^1)$ disappears but all other maxima are left unchanged. The ASA algorithm is then applied to the new objective function to detect a global solution of the new problem. This process is repeated until no other global solution is found.

In this sequential simulated annealing algorithm we solve a sequence of global optimization problems with different objective functions. The mathematical formulation of the algorithm together with the transformations that are carried out are the following:

$$\max_{t \in T} \Psi(t) \equiv \begin{cases} h(t) \text{ if } t \in V_{\varepsilon^l}(t^l), l \in L(x^k) \\ g(t) \qquad \text{otherwise} \end{cases} \tag{2}$$

where $h(t)$ is defined as

$$h(t) = w(t) - \frac{\delta_2[\text{sgn}(g(t^l) - g(t)) + 1]}{2\tanh(\kappa(w(t^l) - w(t)))} \tag{3}$$

and

$$w(t) = g(t) - \frac{\delta_1}{2}\|t - t^l\|[\text{sgn}(g(t^l) - g(t)) + 1] \tag{4}$$

with δ_1, δ_2 and κ positive constants and sgn defines the well-known sign function. Transformations (4) and (3) stretch the neighborhood of t^l, with ray ε^l, downwards assigning lower function values to those points. The ε^l value should be chosen so that the condition

$$\left|g(t^l) - g(\widetilde{t})\right| \leq \delta^{ML}$$

is satisfied for a specific random point $\widetilde{t}$ at the boundary of $V_{\varepsilon^l}(t^l)$. This condition aims to prevent the stretching of the objective function at points that are candidate to local maximizers that satisfy (1). The ASA algorithm is again applied to problem (2) and whenever a global maximizer is found, it is added to the optimal set T^{x^k}. The choice of the ray ε^l for each computed solution t^l, at iteration k, is described in this general scheme for the multi-local algorithm:

Algorithm 2

Given ε^0, ε^{max} *and* δ^{ML}. *Set* $l = 1$ *and* $a = 0$.

Step 1: *Compute a global solution of problem (2) using ASA algorithm. Let* t^l *be the global maximizer.*

Step 2: *Set* $a = a + 1$ *and* $\Delta = a\varepsilon^0$. *For* $j = 1, \ldots, 2m$, *randomly generate* $\widetilde{t}^j$ *such that* $\widetilde{t}^j$ *belongs to the boundary of* $V_\Delta(t^l)$. *Let* $\widetilde{g}_{\max} = \max\{g(\widetilde{t}^1), ..., g(\widetilde{t}^{2m})\}$.

Step 3: *If* $\left|g\left(t^l\right) - \widetilde{g}_{\max}\right| > \delta^{ML}$ *and* $\Delta < \varepsilon^{max}$ *go to Step 2.*

Step 4: *Let* $T^{x^k} = T^{x^k} \cup \{t^l\}$ *and* $\varepsilon^l = \Delta$.

Step 5: *If the termination criterion is not satisfied go to Step 1 with* $l = l + 1$ *and* $a = 0$.

This multi-local algorithm is halted when the optimal set does not change for a fixed number of iterations, which by default is 4. This value has been shown to be adequate for all problems we have tested to date which are mostly small problems (in semi-infinite programming and global optimization context [18]).

Although there is no guarantee that all maximizers will be detected after a finite number of iterations, the asymptotical convergence of the ASA algorithm to a global maximizer and the inclusion of a scheme that sequentially eliminates previously detected maximizers, have been given high rates of success as we have observed in previous work with a set of multi-modal test problems [18]. We will return to this issue in the conclusions.

3.3 Finite optimization procedure

The finite optimization procedure is used to provide a descent direction for a merit function at x^k, solving the reduced problem $P_{red}(x^k)$. A penalty method based on the exponential function proposed by Kort and Bertsekas [12] is used. The algorithm still incorporates a local adaptation procedure as explained later on.

In a penalty method a solution to $P_{red}(x^k)$ is obtained by solving a sequence of unconstrained subproblems, indexed by i, of the form

$$F(x, \eta^i, \lambda^i) = f(x) + \frac{1}{\eta^i} \sum_{l=1}^{|L(x^k)|} \lambda_l^i \left[e^{\eta^i g^l(x)} - 1\right], \text{ for } i = 1, ..., n_k, \qquad (5)$$

for an increasing sequence of positive η^i values where n_k is the maximum number of iterations allowed in this inner iterative process. Each function $g^l(x)$ is obtained by incorporating the solution value t^l in $g(x, t)$.

For each η^i and λ^i, the minimum of $F(x, \eta^i, \lambda^i)$ is obtained by a BFGS quasi-Newton method. In this sequence process, the Lagrange multipliers are updated by

$$\lambda_l^{i+1} = \lambda_l^i e^{\eta^i g^l(x^{k,i})}, l \in L(x^k)$$

where $x^{k,i} = \arg\min F(x, \eta^i, \lambda^i)$ and η^i is the penalty parameter.

The classical definition of a reduction type method considers $n_k = 1$ to be able to guarantee that the optimal set T^{x^k} does not change. When $n_k > 1$, the values of the maximizers do probably change as $x^{k,i}$ changes along the process even if $|L(x^k)|$ does not change. Our experience with this reduction method has shown that we gain in efficiency if more than one iteration is made in solving the problem $P_{red}(x^k)$. This fact was already noticed by Haaren-Retagne [6] and later on by Hettich and Kortanek [9] and Gramlich, Hettich and Sachs [5]. However, in this case, a local adaptation procedure is incorporated in the algorithm.

This procedure aims to correct the maximizers, if necessary, each time a new approximation is computed, $x^{k,i}$. For each t^l, $l \in L(x^k)$, we compute $5m$ random points $\widetilde{t}^j = t^l + p$, where each component $p_i \sim U[-0.5, 0.5]$ $(i = 1, \ldots, m)$, $j = 1, \ldots, 5m$. Then, the one with largest g value, say $\widetilde{t}^s$, is compared with t^l. If $g(x^{k,i}, \widetilde{t}^s) > g(x^{k,i}, t^l)$ the point $\widetilde{t}^s$ replaces t^l.

3.4 Globalization procedure

To promote global convergence a line search procedure is implemented to ensure a sufficient decrease of a merit function. Two merit functions were tested. One is the Kort and Bertsekas [12] exponential penalty function

$$E_1(x, \mu, v) = f(x) + \frac{1}{\mu} \sum_{l=1}^{|L(x)|} v_l [e^{\mu g^l(x)} - 1]$$

where μ is a positive penalty parameter and v_l represents the Lagrange multiplier associated to the constraint $g^l(x)$. The cardinality of the set of maximizers, $|L(x)|$, is not constant over the iterative process, thus in the SIP context the function E_1 is not continuous.

Based on this exponential paradigm, we propose a new continuous extension of E_1 for SIP, herein denoted by E_∞ merit function, which is given by

$$E_\infty(x, \mu, \nu) = f(x) + \frac{\nu}{\mu} [e^{\mu\theta} - 1]$$

where $\theta(x) = \max_{t \in T} [g(x, t)]_+$, μ is a positive penalty parameter and $\nu \geq 0$. Clearly $\theta(x)$ is the infinity norm of the constraint violations, hence E_∞ is continuous for every $x \in \mathbb{R}^n$ [15].

When a new approximation to the solution of the problem $P_{red}(x^k)$ is computed, x^{k,n_k}, the maximizers of $g(x^{k,n_k}, t)$ for $t \in T$ and the cardinality of

the optimal set may change. Thus to be able to carry on with the globalization procedure, the multi-local procedure has to be called again to solve problem $O(x^{k,n_k})$. Let the optimal set be $\tilde{T} = \{\tilde{t}^1, \tilde{t}^2, \ldots, \tilde{t}^{|L(x^{k,n_k})|}\}$.

If $\tilde{T}$ agrees with T^{x^k} the set of constraints in the merit function are the same at x^k and x^{k,n_k}.

If there exists $\tilde{t}^j$ such that $\tilde{t}^j \in \tilde{T}$ and $\tilde{t}^j \notin T^{x^k}$, then $g(x^{k,n_k}, t)$ has one new maximizer that must be accounted for in the merit function. A new multiplier that is associated to the new constraint must then be initialized.

However, if there exists t^j such that $t^j \notin \tilde{T}$ and $t^j \in T^{x^k}$, then this point is no longer a solution of the problem $O(x^{k,n_k})$ that satisfies (1), the constraint associated to this maximizer no longer exists and the corresponding Lagrange multiplier should be removed.

To decide which direction could be used to define the next approximation to the SIP problem, we test the corresponding reduction in the merit function. First we assume that $x^{k+1} = x^{k,n_k}$ and $d = x^{k+1} - x^k$.

For the E_∞ merit function, if the sufficient descent condition

$$E_\infty^{k+1}(x^{k+1}, \mu^k, \nu^k) \leq E_\infty^k(x^k, \mu^k, \nu^k) + \sigma\alpha DE_\infty^k(x^k, \mu^k, \nu^k, d), \qquad (6)$$

for $0 < \sigma < 1$, holds with $\alpha = 1$, then we go to the next iteration. Otherwise, the algorithm recovers the direction $d^{k,1}$, selects α as the first element of the sequence $\{1, 1/2, 1/4, \ldots\}$ to satisfy (6) and sets $x^{k+1} = x^k + \alpha d^{k,1}$. $DE_\infty^k(x^k, \mu^k, \nu^k, d)$ is the directional derivative of the penalty function at x^k in the direction d.

For the merit function E_1, the more appropriate extended Armijo condition is used

$$E_1^{k+1}(x^{k+1}, \mu^k, v^k) \leq E_1^k(x^k, \mu^k, v^k) + \sigma\alpha DE_1^k(x^k, \mu^k, v^k, d) + \pi^k,$$

where $\pi^k = \min\{\tau^k \|\alpha d\|^2, \rho\pi^{k-1}\}$ for $0 < \tau^k \leq 0.5$ and $0 < \rho < 1$. The positive sequence $\{\pi^k\}_{k\geq 1}$ is such that $\lim_{k\to\infty} \pi^k = 0$.

The process continues with the updating of the penalty parameter

$$\mu^{k+1} = \min\left\{\mu^{max}, \Gamma^k\right\}$$

where μ^{max} is a positive constant and $\Gamma > 1$, and the Lagrange multipliers are updated as

$$v_l^{k+1} = v_l^k e^{\mu^k g^l(x^{k+1})}, l \in L(x^k)$$

and

$$\nu^{k+1} = \max\{v_l^{k+1}, l \in L(x^{k+1})\}$$

for E_1 and E_∞ respectively.

3.5 Termination criteria

As far as the termination criteria are concerned, the reduction algorithm stops at a point x^{k+1} if one of these conditions

C1: $|DE^{k+1}(x^{k+1}, \mu^k, ., d)| \leq \epsilon_{DE}$ and $\max\{g(x^{k+1}, t^l), l \in L(x^{k+1})\} < \epsilon_g$;

C2: the number of iterations exceeds k^{max}

holds. Other conditions could be used but this choice permits a comparison to be made with the results in the literature of other reduction methods.

We remark that for some merit functions, e.g., the L_∞ merit function used in [19, 21], if a point $\overline{x}$ is feasible and is a critical point of the merit function then $\overline{x}$ is also a stationary point of the SIP problem.

Although we have not proven yet this property for the two herein tested merit functions, we remark that the solutions reached when the condition C1 is satisfied are similar to the ones presented by [1].

4 Computational results

The proposed reduction method was implemented in the C programming language on a Pentium II, Celeron 466 Mhz with 64Mb of RAM. For the computational experiences we consider eight test problems from Coope and Watson [1] (problems 1, 2, 3, 4, 5, 6, 7, 14 ($c = 1.1$)). The used initial approximations are the ones reported by [1].

We fix the following constants: $\delta_1 = 100$, $\delta_2 = 1$, $\kappa = 10^{-3}$, $\varepsilon^0 = 0.25$, $\varepsilon^{max} = 1$, $\delta^{ML} = 5.0$, $\Gamma = 1.2$, $\sigma = 10^{-4}$, $\mu^{max} = 10^3$, $\rho = 0.95$, $\tau^k = \min\{\|H^k\|_\infty^{-1}, 2^{-1}\}$, $k^{max} = 100$, $\epsilon_x = 10^{-8}$, $\epsilon_g = 10^{-5}$ and $\epsilon_{DE} = 10^{-5}$. The matrix H^k is a BFGS quasi-Newton approximation to the Hessian of the penalty function (5).

We assign μ^k to the initial value of the penalty parameter η^1 and set $\lambda^1 = v^k$ in (5).

4.1 Comparison of the two merit functions

The following tables report on the number of the tested problem, the number of variables, n; the dimension of the set T, m; the number of maximizers satisfying (1) at the final iterate, $|T^*|$; the objective function value at the final iterate, f^*; the number of iterations needed by the presented variant of a reduction method, k_{RM}; the number of multi-local optimization calls needed, k_{ML}; the average number of iterations needed by the multi-local algorithm, k_a^{ML}; the average number of function evaluations in the multi-local procedure, FE_a^{ML}; the average number of iterations needed by the quasi-Newton method,

k_a^{QN}; and the termination criterion satisfied at the final iterate, TC. Tables 2 and 4 also present the average number of iterations needed in the penalty method, k_a^{PM}. Besides the limit on the number of iterations, n_k, the penalty algorithm terminates when the deviation between two consecutive iterates is smaller than ϵ_x. In the tables, columns headed D contain the magnitude of the directional derivative of the corresponding penalty function at the final iterate.

The Tables 1 and 2 present the numerical results obtained by the reduction method with the E_1 merit function for $n_k = 1$ and $n_k = 5$ respectively. To simplify the notation we use, for instance, $-2.51802(-1)$ instead of -2.51802×10^{-1}.

Tables 3 and 4 report on the numerical results obtained by the reduction method combined with E_∞ and $n_k = 1$ and $n_k = 5$, respectively.

Here we describe some modifications that we had to introduce in the default values defined above, for example, on the initial approximation and on the stopping tolerances, in order to obtain convergence.

On problem 2, the runs with E_1 and E_∞, $n_k = 1$, did not converge to the solution reported in the literature.

On problem 4 with $n = 6$, the algorithm only converged in the case E_∞ with $n_k = 5$. For both E_1 and E_∞, with $n_k = 1$, using $x^0 = (0, 0.5, 0, 0, 0, 0)^T$, the algorithm found the solution reported in [1]. In one of these runs we had to increase the tolerance ϵ_g to 10^{-3}. For the run with E_1 and $n_k = 5$, an increase of ϵ_{DE} to 10^{-4} solved the problem.

To be able to converge to the solution of problem 5 with both merit functions and $n_k = 5$, the following initial approximation $(0.5, 0.5, 0)$ and stopping tolerances $\epsilon_{DE} = 10^{-3}$ and $\epsilon_g = 10^{-2}$ had to be used.

When solving problems 4 with $n = 6$ (E_∞, $n_k = 1$) and 5 (E_∞, $n_k = 5$) with the default values, we observed that although the components of x were not changing in the first four decimal places the stopping criterion C1 was not satisfied and from a certain point on the algorithm starts to diverge. This is probably due to the Maratos effect.

Finally, on problem 1 (E_1 and E_∞, with $n_k = 1$) relaxing ϵ_g to 10^{-4} makes the termination criterion C1 to be satisfied.

Details of the corresponding results are reported in the tables.

When the process does not need to recover the first direction $d^{k,1}$, the number of iterations of the reduction method equals the number of multi-local optimization calls.

When comparing the presented four variants of the reduction method, we may conclude that the combination of E_∞ with $n_k = 5$ requires in general fewer iterations of the reduction algorithm as well as fewer multi-local optimization calls.

Table 1. Computational results with E_1 and $n_k = 1$

Problems	n	m	$\lvert T^*\rvert$	f^*	k_{RM}	k_{ML}	k_a^{ML}	FE_a^{ML}	k_a^{QN}	TC	D
1	2	1	2	$-2.51802(-1)$	100	1337	5	2897	3	C2	$7.0(-15)$
$1^{(1)}$	2	1	2	$-2.54318(-1)$	52	581	5	3062	3	C1	$1.6(-13)$
2	2	1	2	$4.02110(-1)$	7	13	7	4466	5	C1	$8.6(-13)$
3	3	1	2	$5.33515(+0)$	40	41	5	4429	10	C1	$9.3(-06)$
4	3	1	2	$6.57032(-1)$	9	186	5	8576	32	C1	$3.7(-10)$
4	6	1	1	$1.18871(+1)$	100	1070	5	3554	32	C2	$1.4(-03)$
$4^{(2)}$	6	1	2	$6.21080(-1)$	23	167	5	12137	41	C1	$9.3(-06)$
4	8	1	2	$6.19758(-1)$	7	110	4	3171	36	C1	$2.0(-14)$
5	3	1	2	$4.43969(+0)$	39	181	6	3159	14	C1	$1.8(-07)$
6	2	1	1	$9.71588(+1)$	51	53	4	2958	4	C1	$6.3(-08)$
7	3	2	1	$9.99999(-1)$	45	48	4	24046	4	C1	$7.9(-08)$
14	2	1	1	$2.22412(+0)$	47	173	4	3373	10	C1	$1.1(-12)$

(1) - $\epsilon_g = 10^{-4}$
(2) - $x^0 = (0, 0.5, 0, 0, 0, 0)^T$

Table 2. Computational results with E_1 and $n_k = 5$

Problems	n	m	$\lvert T^*\rvert$	f^*	k_{RM}	k_{ML}	k_a^{ML}	FE_a^{ML}	k_a^{PM}	k_a^{QN}	TC	D
1	2	1	2	$-2.51823(-1)$	51	345	5	3065	3	3	C1	$1.4(-07)$
2	2	1	2	$1.95428(-1)$	4	8	5	4214	2	4	C1	$4.8(-13)$
3	3	1	2	$5.34115(+0)$	18	56	5	3390	1	11	C1	$5.9(-10)$
4	3	1	2	$6.75377(-1)$	15	183	5	11180	1	28	C1	$1.2(-10)$
4	6	1	1	$9.34036(+7)$	100	2290	4	3735	1	24	C2	$9.7(+00)$
$4^{(1)}$	6	1	1	$6.60021(-1)$	84	2084	4	3632	1	22	C1	$2.9(-05)$
4	8	1	2	$6.28445(-1)$	6	47	4	4079	1	39	C1	$5.7(-15)$
5	3	1	2	$4.30057(+0)$	100	1190	5	8092	1	26	C2	$3.8(-03)$
$5^{(2)}$	3	1	2	$4.29811(+0)$	47	823	5	7112	1	27	C1	$5.1(-04)$
6	2	1	1	$9.71590(+1)$	11	52	4	3343	3	4	C1	$7.0(-09)$
7	3	2	1	$1.00011(+0)$	15	85	4	24130	2	4	C1	$1.2(-07)$
14	2	1	1	$2.20002(+0)$	12	79	4	3268	4	13	C1	$1.6(-09)$

(1) - $\epsilon_{DE} = 10^{-4}$
(2) - $x^0 = (0.5, 0.5, 0)^T$, $\epsilon_{DE} = 10^{-3}$, $\epsilon_g = 10^{-2}$

Table 3. Computational results with E_∞ and $n_k = 1$

Problems	n	m	$\lvert T^*\rvert$	f^*	k_{RM}	k_{ML}	k_a^{ML}	FE_a^{ML}	k_a^{QN}	TC	D
1	2	1	2	$-2.52545(-1)$	100	377	5	2854	3	C2	$4.9(-11)$
$1^{(1)}$	2	1	2	$-2.54386(-1)$	61	144	5	3081	4	C1	$1.4(-06)$
2	2	1	2	$4.76293(-1)$	3	38	6	4057	7	C1	$2.7(-13)$
$2^{(2)}$	2	1	2	$2.57526(-1)$	4	39	6	4324	8	C1	$5.7(-13)$
3	3	1	2	$5.34326(+0)$	21	22	6	5928	10	C1	$1.6(-12)$
4	3	1	1	$6.76364(-1)$	52	573	4	4165	27	C1	$1.4(-07)$
4	6	1	1	$2.21882(+9)$	100	481	4	5871	37	C2	$3.8(+01)$
$4^{(3)}$	6	1	1	$6.16754(-1)$	25	592	5	3662	32	C1	$3.2(-07)$
4	8	1	2	$6.18975(-1)$	15	157	5	6009	33	C1	$7.4(-13)$
5	3	1	2	$4.43589(+0)$	32	430	6	3616	9	C1	$2.3(-14)$
6	2	1	1	$9.71589(+1)$	44	55	4	2917	4	C1	$4.3(-07)$
7	3	2	1	$9.99999(-1)$	45	48	4	22753	4	C1	$5.3(-07)$
14	2	1	1	$2.21643(+0)$	40	95	4	3158	8	C1	$2.3(-08)$

(1) - $\epsilon_g = 10^{-4}$
(2) - $x^0 = (-1, 0)^T$
(3) - $x^0 = (0, 0.5, 0, 0, 0, 0)^T$, $\epsilon_g = 10^{-3}$

Table 4. Computational results with E_∞ and $n_k = 5$

Problems	n	m	$\lvert T^*\rvert$	f^*	k_{RM}	k_{ML}	k_a^{ML}	FE_a^{ML}	k_a^{PM}	k_a^{QN}	TC	D
1	2	1	2	$-2.50400(-1)$	48	60	4	4240	1	4	C1	$5.4(-08)$
2	2	1	2	$1.94620(-1)$	3	38	5	3101	3	4	C1	$8.8(-15)$
3	3	1	2	$5.33477(+0)$	3	13	5	2102	2	17	C1	$1.0(-06)$
4	3	1	1	$6.49848(-1)$	11	126	4	4986	1	23	C1	$1.2(-06)$
4	6	1	1	$6.16268(-1)$	74	1203	4	5687	1	26	C1	$1.0(-08)$
4	8	1	1	$6.15765(-1)$	54	928	5	6376	1	34	C1	$1.9(-06)$
5	3	1	1	$6.85226(+3)$	41	1000	5	6364	1	30	C1	$4.1(-11)$
$5^{(1)}$	3	1	2	$4.30743(+0)$	48	570	5	3985	1	20	C1	$7.2(-04)$
6	2	1	1	$9.71589(+1)$	7	8	4	2651	2	4	C1	$8.7(-06)$
7	3	2	1	$9.99997(-1)$	8	9	4	24117	2	3	C1	$4.0(-06)$
14	2	1	1	$2.20017(+0)$	10	95	4	3262	3	16	C1	$8.9(-15)$

(1) - $x^0 = (0.5, 0.5, 0)^T$, $\epsilon_{DE} = 10^{-3}$, $\epsilon_g = 10^{-2}$

4.2 Comparison with other reduction methods

We include Table 5 in order to allow a comparison between our reduction method and other well-known reduction methods. k_{RM} and k_{ML} represent the number of iterations needed by each reduction method and the number of multi-local optimization calls, respectively. The superscripts P, TFI and CW refer to the results obtained by Price [19], Tanaka, Fukushima and Ibaraki [28] and Coope and Watson [1], respectively.

For a straight comparison we include the results of our variant with E_∞ and $n_k = 5$ under the columns without the superscript. Although the results of problems 2, 3, 6 and 7 are quite satisfactory for an algorithm that relies on a quasi-Newton based method, some improvements may be obtained on the remaining problems if new strategies as explained in the next section are implemented.

Table 5. Numerical results obtained by reduction methods

Problems	k_{RM}	k_{ML}	D	k_{RM}^P	k_{ML}^P	D^P	k_{RM}^{TFI}	k_{ML}^{TFI}	D^{TFI}	k_{RM}^{CW}	D^{CW}
1	48	60	5.4(−08)	17	21	8.2(−06)	17	19	4.8(−07)	16	5.7(−06)
2	3	38	8.8(−15)	8	10	1.4(−08)	5	11	2.7(−08)	7	2.5(−10)
3	3	13	1.0(−06)	11	23	1.3(−06)	9	12	5.5(−08)	10	6.2(−12)
4	11	126	1.2(−06)	10	11	1.9(−06)	5	15	2.7(−07)	5	5.4(−08)
	74	1203	1.0(−08)	57	119	7.7(−06)	8	27	7.7(−06)	20	6.4(−06)
	54	928	1.9(−06)	84	164	1.0(−07)	3	14	3.4(−06)	16	7.4(−06)
5	48[(1)]	570	7.2(−04)	8	14	6.2(−06)	4	9	6.8(−07)	4	6.9(−06)
6	7	8	8.7(−06)	27	87	5.2(−06)	16	19	1.3(−18)	9	1.1(−08)
7	8	9	4.0(−06)	9	14	7.0(−09)	2	4	0.0(+00)	3	0.0(+00)
14	10	95	8.9(−15)	6	7	8.1(−06)	5	8	3.4(−07)	5	8.2(−07)

(1) - $x^0 = (0.5, 0.5, 0)^T$, $\epsilon_{DE} = 10^{-3}$, $\epsilon_g = 10^{-2}$

5 Conclusions and future work

We presented a global reduction method for solving semi-infinite programming problems. The algorithm relies on a stochastic approach, the simulated annealing method, which incorporates a sequence of local applications of a stretching technique, in order to compute the solutions of the lower-level problem. An approximation to the solution of a finite reduced optimization problem is computed by a penalty method. The algorithm also incorporates a globalization procedure based on a line search approach to be able to guarantee a sufficient decrease on a merit function. A new continuous E_∞ merit function is proposed with promising results.

The multi-local algorithm herein presented turns out well in detecting multiple solutions of a finite problem, although we are not able to guarantee that all required solutions are detected. It came to our knowledge that Topological Degree theory [15] may be used to obtain information on the number as well as the location of all the solutions.

It has been recognized that the choice of the penalty parameter value at each iteration, μ^k, is not a trivial matter. In particular with the two proposed exponential merit functions we have observed that a slight increase in its value rapidly generates some instability. We feel that this problem may be solved using a filter technique that has been proven to be efficient in the finite case [4].

Although our computed solutions can be considered reasonable it remains to be proven that a feasible point that is a critical point to the two merit functions used in this paper also is a stationary point of SIP. Another issue that should be addressed in the future is concerned with the inclusion of a second order correction to prevent the Maratos effect from occurring [14].

Acknowledgments

The authors wish to thank two anonymous referees for their careful reading of the manuscript and their fruitful comments and suggestions.

This work has been partially supported by the Algoritmi Research Center and by Portuguese FCT grant POCI/MAT/58957/2004.

References

1. I. D. Coope and G. A. Watson, *A projected Lagrangian algorithm for semi-infinite programming*, Mathematical Programming **32** (1985), 337–356.
2. A. Corana, M. Marchesi, C. Martini, and S. Ridella, *Minimizing multimodal functions of continuous variables with the "simulated annealing" algorithm*, ACM Transactions on Mathematical Software **13** (1987), no. 3, 262–280.
3. A. Dekkers and E. Aarts, *Global optimization and simulated annealing*, Mathematical Programming **50** (1991), 367–393.
4. R. Fletcher and S. Leyffer, *Nonlinear programming without a penalty function*, Mathematical Programming **91** (2002), 239–269.
5. G. Gramlich, R. Hettich, and E. W. Sachs, *Local convergence of SQP methods in semi-infinite programming*, SIAM Journal of Optimization **5** (1995), no. 3, 641–658.
6. E. Haaren-Retagne, *A semi-infinite programming algorithm for robot trajectory planning*, Ph.D. thesis, Universitat Trier, 1992.
7. R. Hettich and W. V. Honstede, *On quadratically convergent methods for semi-infinite programming*, Semi-Infinite Programming (R. Hettich, ed.), Springer-Verlag, 1979, pp. 97–111.

8. R. Hettich and H. Th. Jongen, *Semi-infinite programming: conditions of optimality and applications*, Lectures Notes in Control and Information Science, vol. 7, springer ed., 1978, pp. 1–11.
9. R. Hettich and K. O. Kortanek, *Semi-infinite programming: Theory, methods and applications*, SIAM Review **35** (1993), no. 3, 380–429.
10. L. Ingber, *Adaptive simulated annealing (ASA): Lessons learned*, Control and Cybernetics **25** (1996), no. 1, 33–54.
11. H. Th. Jongen and J.-J. Rückmann, *On stability and deformation in semi-infinite optimization*, Semi-Infinite Programming (R. Reemtsen and J.-J. Rückmann, eds.), Nonconvex Optimization and Its Applications, vol. 25, Kluwer Academic Publishers, 1998, pp. 29–67.
12. W. B. Kort and D. Bertsekas, *A new penalty function method for constrained minimization*, In Proceedings of the 1972 IEEE Conference on Decision and Control (1972), 162–166.
13. T. León, S. Sanmatías, and H. Vercher, *A multi-local optimization algorithm*, Top **6** (1998), no. 1, 1–18.
14. J. Nocedal and S. Wright, *Numerical optimization*, Springer Series in Operations Research, Springer, 1999.
15. J. M. Ortega and W. C. Rheinboldt, *Iterative solution of nonlinear equations in several variables*, Academic Press, 1970.
16. K. Parsopoulos, V. Plagianakos, G. Magoulas, and M. Vrahatis, *Objective function stretching to alleviate convergence to local minima*, Nonlinear Analysis **47** (2001), 3419–3424.
17. A. I. Pereira and E. M. Fernandes, *A study of simulated annealing variants*, In Proceedings of the XXVIII Congreso Nacional de Estadística e Investigacion Operativa, ISBN 84-609-0438-5, Cadiz (2004).
18. ———, *A new algorithm to identify all global maximizers based on simulated annealing*, Proceedings of the 6th World Congress on Structural and Multidisciplinary Optimization, J. Herskowitz, S. Mazorche and A. Canelas (Eds.), ISBN: 85-285-0070-5 (CD-ROM) (2005).
19. C. J. Price, *Non-linear semi-infinite programming*, Ph.D. thesis, University of Canterbury, 1992.
20. C. J. Price and I. D. Coope, *An exact penalty function algorithm for semi-infinite programmes*, BIT **30** (1990), 723–734.
21. ———, *Numerical experiments in semi-infinite programming*, Computational Optimization and Applications **6** (1996), 169–189.
22. R. Reemtsen and S. Görner, *Numerical methods for semi-infinite programming: a survey*, Semi-Infinite Programming (R. Reemtsen and J.-J. Rückmann, eds.), Nonconvex Optimizaion and Its Applications, vol. 25, Kluwer Academic Publishers, 1998, pp. 195–275.
23. R. Reemtsen and J.-J. Rückmann (eds.), *Semi-infinite programming*, Nonconvex Optimization and its Applications, Kluwer Academic Publishers, 1998.
24. H. Romeijn and R. Smith, *Simulated annealing for constrained global optimization*, Journal of Global Optimization **5** (1994), 101–126.
25. O. Stein, *Bi-level strategies in semi-infinite programming*, Nonconvex Optimization and Its Applications, vol. 71, Kluwer Academic, 2003.
26. H. Szu and R. Hartley, *Fast simulated annealing*, Physics Letters A **122** (1987), no. 3-4, 157–162.

27. Y. Tanaka, M. Fukushima, and T. Ibaraki, *A comparative study of several semi-infinite nonlinear programmnig algorithms*, European Journal of Operations Research **36** (1988), 92–100.
28. ———, *A globally convergence SQP method for semi-infinite non-linear optimization*, Journal of Computational and Applied Mathematics **23** (1988), 141–153.
29. A. I. F. Vaz, E. M. G. P. Fernandes, and M. P. S. F. Gomes, *Discretization methods for semi-infinite programming*, Investigação Operacional **21** (2001), 37–46.
30. G. A. Watson, *Globally convergent methods for semi-infinite programming*, BIT **21** (1981), 362–373.
31. ———, *Numerical experiments with globally convergent methods for semi-infinite programming problems*, Semi-Infinite Programming and Applications (A. V. Fiacco and K. O. Kortanek, eds.), Springer Verlag, 1983, pp. 193–205.