Chapter 4
A Penalty Approach for Solving Nonsmooth and Nonconvex MINLP Problems

M. Fernanda P. Costa, Ana Maria A. C. Rocha and Edite M. G. P. Fernandes

Abstract This paper presents a penalty approach for globally solving nonsmooth and nonconvex mixed-integer nonlinear programming (MINLP) problems. Both integrality constraints and general nonlinear constraints are handled separately by hyperbolic tangent penalty functions. Proximity from an iterate to a feasible promising solution is enforced by an oracle penalty term. The numerical experiments show that the proposed oracle-based penalty approach is effective in reaching the solutions of the MINLP problems and is competitive when compared with other strategies.

Keywords MINLP · Penalty function · DIRECT · Oracle

4.1 Introduction

In this paper, we address the solution of nonsmooth and nonconvex mixed-integer nonlinear programming (MINLP) problems by a penalty approach. It is assumed that the problem is in the form

$$\begin{array}{ll} \underset{x \in X \subset \mathbb{R}^n}{\text{glob min}} & f(x) \\ \text{subject to} & g_j(x) \leq 0, \; j = 1, \ldots, p \\ & h_l(x) = 0, \; l = 1, \ldots, m \\ & x_i \in \mathbb{R} \text{ for } i \in I_c \subseteq I \equiv \{1, \ldots, n\} \\ & x_j \in \mathbb{Z} \text{ for } j \in I_d \subseteq I \end{array} \tag{4.1}$$

M. F. P. Costa (✉)
Centre of Mathematics, University of Minho, Campus de Gualtar,
4710-057 Braga, Portugal
e-mail: mfc@math.uminho.pt

A. M. A. C. Rocha · E. M. G. P. Fernandes
Algoritmi Research Centre, University of Minho, Campus de Gualtar,
4710-057 Braga, Portugal
e-mail: arocha@dps.uminho.pt

E. M. G. P. Fernandes
e-mail: emgpf@dps.uminho.pt

A. I. F. Vaz et al. (eds.), *Operational Research*, Springer Proceedings
in Mathematics & Statistics 223, https://doi.org/10.1007/978-3-319-71583-4_4

where $f, g_j, h_l : \mathbb{R}^n \to \mathbb{R}$ are continuous possibly nonlinear functions in a compact subset of $\mathbb{R}^n$, herein defined as $X = \{x : -\infty < lb_i \leq x_i \leq ub_i < \infty, i = 1, \ldots, n\}$, $I_c \cap I_d = \emptyset$ and $I_c \cup I_d = I$. Thus, I_c is the index set of the continuous variables and I_d consists of the indices of the integer variables. Here, integer variables include binary variables. Let C be the following subset of $\mathbb{R}^n$, $C = \{x \in \mathbb{R}^n : g_j(x) \leq 0, j = 1, \ldots, p,\ h_l(x) = 0, l = 1, \ldots, m\}$, and let $W_c = C \cap X$ be a closed set. Consider the set D, which is the cartesian product of the sets D_j, $j \in I_d$, where

$$D_j = \{d \in \mathbb{Z} : lb_j \leq d \leq ub_j\}, j \in I_d, \tag{4.2}$$

let $\mathscr{I}$ be defined by $\mathscr{I} = \left\{x \in X : x_j \in \mathbb{Z} \text{ for } j \in I_d \subseteq I\right\}$ and let $W = C \cap \mathscr{I}$ be the nonempty feasible region of the problem (4.1). When a continuous relaxation of the integer variables is applied, $W \equiv W_c$. A continuous relaxation means that the integer variables can be treated as continuous variables, and all function (f, g and h) values can be computed for $x_j \in \mathbb{R}$, $j \in I_d$ (instead of $x_j \in \mathbb{Z}$, $j \in I_d$). The MINLP problem (4.1) is said to be convex if f and $g_1(x), \ldots, g_p(x)$ are convex functions and $h_1(x), \ldots, h_m(x)$ are affine functions over X. This means that by relaxing the integrality constraint on x_j, $j \in I_d$, a convex program is obtained (minimizing a convex function over a convex set). Otherwise, the MINLP is said to be nonconvex.

Most techniques available in the literature require the definition and the use of convex model functions and the continuous relaxations of the integer variables. However, some real-life MINLP problems that emerge in mechanical, electrical and chemical engineering applications involve nonsmooth and nonconvex functions and the specific integer variables cannot be relaxed [1]. Most exact methods for nonconvex MINLP are based on the branch-and-bound (BB) technique. Effective examples are the spatial-BB algorithm [2, 3], branch-and-reduce type algorithms [2, 4] and the α-BB algorithm [5].

Heuristics for nonconvex MINLP are also available in the literature. A heuristic approach extension of the boundary tracking optimization is presented in [6]. In [7], a variable neighborhood search heuristic is proposed and in [8], two heuristics are analyzed: the first aims to obtain an initial feasible solution, the second one searches for an improved solution within the neighborhood of a given point.

Extensions of the feasibility pump algorithm to nonconvex MINLP are available in [9]. A derivative-free method that relies on two search procedures, a line search strategy for the continuous variables and a local search for the discrete ones, is presented in [10]. Recently, penalty-based algorithms aiming to penalize integrality violation are available in the literature [11–13].

Metaheuristics are nowadays very popular and aim to compute fast and good approximations to optimal solutions of nonconvex MINLP problems. A mixed-integer hybrid differential evolution (MIHDE) [14] has been successfully applied to mixed-integer optimization problems and a particle swarm optimization is presented in [15]. A parameter free penalty approach with a genetic algorithm (GA) [16] and a filter technique combined with a GA [17] are analyzed when solving nonconvex MINLP. In [18], the BBMCSFilter method, which relies on a BB framework and a derivative-free methodology to solve nonsmooth and nonconvex NLP, is presented.

Two extended versions of the ant colony optimization framework are available in [19] and a new version of the firefly algorithm (FA), that uses four preference rules to select solutions that are feasible or have the least objective function values, is tested in [20]. Review on MINLP techniques and applications are available in [2, 21, 22]. A brief overview of the start-of-the-art in software for the solution of MINLP problems can be found in [23].

In this study, a penalty continuous formulation of the MINLP problem (4.1) is used. First, a penalty function has been selected from a class of penalty functions that are applied to general integer problems [11–13]. Second, two other penalty functions have been constructed in order to penalize the general constraints violation as well as to enforce convergence to a solution, denoted by the oracle, that is feasible and has the least function value found so far. Thus, after relaxing the integrality constraints on the variables and adding a particular penalty term to the objective function, $P_d(x; \varepsilon_d)$, aiming to penalize the integrality constraint violation, as well as by adding another penalty term, $P_c(x; \varepsilon_c)$, to penalize the general constraints violation, the following continuous bound constrained nonlinear programming (BCNLP) problem emerges

$$\begin{array}{rl} \operatorname*{glob\,min}_{x \in X} & \Psi(x; \varepsilon_d, \varepsilon_c) \equiv f(x) + P_d(x; \varepsilon_d) + P_c(x; \varepsilon_c) \\ \text{subject to} & x_i \in \mathbb{R},\ i = 1, \ldots, n, \end{array} \tag{4.3}$$

where $\varepsilon_d, \varepsilon_c \in \mathbb{R}^+$ are positive penalty parameters [24]. The motivation is that problem (4.1) is equivalent to the continuous BCNLP problem, in the sense that they have the same global minimizers. The optimal solution of the BCNLP problem can then be easily obtained by well-established and known solvers.

In the sequel, the herein presented work adds a new penalty term to the objective function in problem (4.3), aiming to enforce convergence to the oracle, represented by o^*, and defined as the best found feasible solution, aiming to predict a global optimum. The goal of the oracle penalty is to penalize solutions that move away from o^*. The new proposed algorithm is tested and compared with other nonconvex MINLP strategies.

Thus, our contribution in this article is directed to the combination of three penalty terms aiming to penalize the integrality violation, the nonlinear inequality and equality constraints violation and the distance to the oracle o^*. The penalty term for the integrality constraints is based on the hyperbolic tangent function, as proposed in [11], and the equality and inequality constraints are dealt with penalties also defined by the hyperbolic tangent function [24]. Similarly, the new penalty imposed on the distance of the current solution to the oracle is also based on the hyperbolic tangent function. The motivation for the use of the hyperbolic tangent function is that its boundedness property makes the BCNLP penalty problem easier to solve than with some of its competitors. The solution of the BCNLP problem is then obtained using the DIRECT algorithm [25], a deterministic and derivative-free algorithm for finding global solutions inside hyperrectangles. We illustrate the performance of the proposed penalty approach on a well-known set of MINLP test problems.

The remainder of the paper proceeds as follows. Section 4.2 introduces the penalty methodology and Sect. 4.3 addresses the implementation of the penalty terms and investigates the use of the penalty parameters and the oracle parameter. Section 4.4 contains the results of all the numerical experiments and the conclusions are summarized in Sect. 4.5.

4.2 Penalty Approaches

The following equivalence result based on a penalty approach will be used [11–13].

Property 4.1 *Assuming that W and W_c are compact sets, there exists a value $\bar{\varepsilon} > 0$ such that, for any $\varepsilon_d \in (0, \bar{\varepsilon}]$, the problems*

$$\min f(x), \quad \textit{subject to} \;\; x \in W$$

and

$$\min F(x; \varepsilon_d) \equiv f(x) + P_d(x; \varepsilon_d), \quad \textit{subject to} \;\; x \in W_c \tag{4.4}$$

where

$$P_d(x; \varepsilon_d) = \frac{1}{\varepsilon_d} \sum_{j \in I_d} \min_{d \in D_j} \tanh(|x_j - d|) \tag{4.5}$$

are equivalent in the sense that they have the same minimizers.

This property is a consequence of Property 2.5 in [11]. The below presented assumptions (A1)–(A3) on f and on the penalty $P_d(x; \varepsilon_d)$ (see (4.5)) are required to prove Property 4.1.

(A1) Function f is bounded on W_c and there exists an open set $A \supset W$ and real numbers $\alpha, L > 0$ such that for all $x, y \in A$, f satisfies

$$|f(x) - f(y)| \leq L\|x - y\|^{\alpha}.$$

(A2) For all $x, y \in W$ and for all $\varepsilon_d \in \mathbb{R}^+$,

$$P_d(x; \varepsilon_d) = P_d(y; \varepsilon_d).$$

(A3) There exists an $\bar{\varepsilon}$, and for all $z \in W$ there exists a neighborhood $S(z)$ such that

$$P_d(x; \varepsilon_d) - P_d(z; \varepsilon_d) \geq \bar{L}\|x - z\|^{\alpha}, \;\text{ for all } x \in S(z) \cap (W_c \backslash W), \varepsilon_d \in (0, \bar{\varepsilon}],$$

where $\bar{L} > L$ and α is chosen as in (A1). Furthermore, let $S = \bigcup_{z \in W} S(z)$, $\exists \bar{x} \notin S$ such that

$$\lim_{\varepsilon_d \to 0} (P_d(\bar{x}; \varepsilon_d) - P_d(z; \varepsilon_d)) = +\infty \text{ for all } z \in W,$$

$$P_d(x; \varepsilon_d) \geq P_d(\bar{x}; \varepsilon_d) \text{ for all } x \in W_c \backslash S \text{ and for all } \varepsilon_d > 0.$$

Problem (4.4) comes out by relaxing the integer constraints on the variables and adding a particular penalty term to the objective function f.

Let $P_c(\cdot; \varepsilon_c) : \mathbb{R}^n \to \mathbb{R}$ be a penalty term, that aims to penalize general equality and inequality constraints violation, defined by

$$P_c(x; \varepsilon_c) = \frac{1}{\varepsilon_c} \left(\sum_{j=1}^{p} \tanh(g_j^+(x)) + \sum_{l=1}^{m} \tanh(|h_l(x)|) \right), \tag{4.6}$$

where $g_j^+(x) = \max\{g_j(x), 0\}$ and $\varepsilon_c \in \mathbb{R}^+$ is the penalty parameter. We note that $P_c(x; \varepsilon_c) = 0$ when $x \in C$ and $P_c(x; \varepsilon_c) > 0$ when $x \notin C$. Generally speaking, under suitable assumptions on the objective function F of problem (4.4) and on the penalty $P_c(\cdot; \varepsilon_c)$, the problems

$$\min \Psi(x; \varepsilon_d, \varepsilon_c) \equiv F(x; \varepsilon_d) + P_c(x; \varepsilon_c), \quad \text{subject to } x \in X,$$

and (4.4) are equivalent (see Theorem 2.1 in [12]).

For the sake of simplicity, we define

$$P(x; \varepsilon_d, \varepsilon_c) = P_d(x; \varepsilon_d) + P_c(x; \varepsilon_c). \tag{4.7}$$

Both penalty terms in $P(x; \varepsilon_d, \varepsilon_c)$ are based on the hyperbolic tangent function, $\tanh : \mathbb{R} \to [-1, 1] \subset \mathbb{R}$, an odd function which is differentiable, strictly increasing on $\mathbb{R}$, and satisfies $\tanh(t) = 0$ iff $t = 0$ and

$$\lim_{t \to 0^+} \frac{\tanh(t)}{t} = 1, \quad \lim_{t \to +\infty} \tanh(t) = 1 \quad \text{and} \quad \lim_{t \to +\infty} \frac{d \tanh(t)}{dt} = 0.$$

Under some suitable assumptions on f and $P(x; \varepsilon_d, \varepsilon_c)$ (see Theorem 2.1 in [12], as well as Property 2.5 in [11] in the context of the hyperbolic tangent function) we may remark the following.

Remark 4.1 Under suitable assumptions on f and $P(x; \varepsilon_d, \varepsilon_c)$, let W and X ($W \subseteq X \subset \mathbb{R}^n$) be compact sets. Then, $\exists \tilde{\varepsilon} \in \mathbb{R}^+$ such that for all $\varepsilon_d, \varepsilon_c \in (0, \tilde{\varepsilon}]$, the problems (4.1) and (4.3) have the same global minimizers.

4.3 Oracle-Based Penalty Algorithm

The extension of the above presented penalty approach to solve MINLP problems is investigated.

We note here that the term $P_d(x; \varepsilon_d)$ (see (4.5)) penalizes the distance from x to a point z (in terms of the components $i \in I_d$) that satisfies $z := [x]_r \in \mathscr{I} \subset X$ where $z_i \in \mathbb{Z}$, $i \in I_d$ results from rounding x_i to the nearest integer and $z_l = x_l$ for $l \in I_c$, thus compelling x to come near z. However, since z may not be a global minimizer, our proposal considers a new penalty term that aims to reduce the distance from x to a very promising solution, o^* (ideally a global optimizer), that satisfies $o^* \in W$ and has an objective function value not greater than $f(z)$. The o^* is a parameter vector, herein also denoted by the oracle, likewise it is used in [26], due to its predictive nature. Although the original idea of the oracle penalty method corresponds to a transformation of the objective function f into an additional equality constraint $h_{m+1}(x) = f(x) - \gamma = 0$, where γ is the oracle parameter [26], our proposal is equivalent to having an extra equality constraint that aims to enforce the proximity of the current solution to the oracle. Thus, we add a new penalty term to P, measuring proximity from x to o^*, with the aim of finding a solution near the oracle with a lower objective function value $f(x) < f(o^*) \leq f(z)$

$$q(x; o^*) = \sum_{i=1}^{n} \tanh(|x_i - o_i^*|). \tag{4.8}$$

Remark 4.2 We note that, in the context of incorporating the function 'tanh' in the penalty terms, this corresponds to adding new equality constraints $x_i = o_i^*$ to the problem (4.1) and that the feasible set of the "new problem" is now $W_o = \{x \in W : x_i = o_i^*, i = 1, \ldots, n\}$. When the oracle parameter o^* is a global minimizer to the problem (4.1), a feasible solution to the "new problem" ($x \in W_o$) is the global solution of the MINLP problem.

Thus, the new proposed BCNLP problem for finding a global solution to a MINLP problem like (4.1) is

$$\begin{aligned} &\operatorname*{glob\,min}_{x \in X} \ \Psi(x; \varepsilon_d, \varepsilon_c, o^*) \equiv f(x) + P(x; \varepsilon_d, \varepsilon_c, o^*) \\ &\text{subject to} \ \ x_i \in \mathbb{R}, \ i = 1, \ldots, n, \end{aligned} \tag{4.9}$$

where the oracle penalty function reads as follows:

$$P(x; \varepsilon_d, \varepsilon_c, o^*) = P_d(x; \varepsilon_d) + P_c(x; \varepsilon_c) + \frac{1}{\varepsilon_c} q(x; o^*). \tag{4.10}$$

When there is a guess about the global minimizer, this information may be used to speed the convergence of the algorithm. To apply the oracle penalty function when there is no guess about the global minimizer, some modifications are required to make the method more robust regarding the oracle parameter selection. We assume that the two following conditions hold:

- $f(o^*) > f(x^*)$;
- there exists at least one $z^* \in W$ such that $f(o^*) = f(z^*) \geq f(x^*)$.

Thus, the oracle vector o^* should be updated whenever a solution better than o^* is produced, i.e., if a solution $z \in \mathscr{I}$ is found such that $f(z) \leq f(o^*)$ and $\Theta(z) \leq \Theta(o^*)$, where

$$\Theta(x) = \max_{j=1,\ldots,p;l=1,\ldots,m} \left\{ g_j^+(x), |h_l(x)| \right\} \tag{4.11}$$

represents the maximum general constraints violation, then the new value for the oracle is the following $o^* = z$.

The algorithm based on the proposed oracle penalty function, denoted by oracle-based penalty algorithm (ObPA), is shown in Algorithm 1. To initialize the oracle, we set $o^* = [x^0]_r$, where the initial approximation, x^0, is randomly generated in X.

Input: $x^0 \in X, \varepsilon > 0, \delta > 0, \eta > 0, \mu > 0, \varepsilon_d^1 > \varepsilon, \varepsilon_c^1 > \varepsilon, \delta^1 > \delta, \eta^1 > \eta, \mu^1 > \mu$;
Set $k = 1$;
Initialize the oracle as $o^* = z^0 = [x^0]_r$;
while *the stopping rule defined in (4.14) does not hold* **do**
  if $\Theta(z^{k-1}) \leq \Theta(o^*)$ *and* $f(z^{k-1}) \leq f(o^*)$ **then**
    Set $o^* = z^{k-1}$;
  end
  if $\Theta(o^*) \leq \eta^k$ **then**
    Compute x^k, an approximation to the solution of problem (4.9) such that

$$\Psi(x^k; \varepsilon_d^k, \varepsilon_c^k, o^*) \leq \Psi(x; \varepsilon_d^k, \varepsilon_c^k, o^*) + \delta^k \text{ for all } x \in X \tag{4.12}$$

  else
    Compute x^k, an approximation to the solution of problem (4.3) such that

$$\Psi(x^k; \varepsilon_d^k, \varepsilon_c^k) \leq \Psi(x; \varepsilon_d^k, \varepsilon_c^k) + \delta^k \text{ for all } x \in X \tag{4.13}$$

  end
  Set $z^k = [x^k]_r$;
  if $\|x^k - z^k\|_\infty > \mu^k$ **then**
    $\varepsilon_d^{k+1} = \max\{0.1\varepsilon_d^k, \varepsilon\}$; $\mu^{k+1} = \mu^k$; $\delta^{k+1} = \delta^k$;
  else
    $\varepsilon_d^{k+1} = \varepsilon_d^k$; $\mu^{k+1} = \max\{0.1\mu^k, \mu\}$; $\delta^{k+1} = \max\{0.9\delta^k, \delta\}$;
  end
  if $\Theta(x^k) > \eta^k$ **then**
    $\varepsilon_c^{k+1} = \max\{0.1\varepsilon_c^k, \varepsilon\}$; $\eta^{k+1} = \eta^k$; $\delta^{k+1} = \delta^k$;
  else
    $\varepsilon_c^{k+1} = \varepsilon_c^k$; $\eta^{k+1} = \max\{0.1\eta^k, \eta\}$; $\delta^{k+1} = \max\{0.9\delta^k, \delta\}$;
  end
  Set $k = k + 1$;
end

Algorithm 1: ObPA

In addition to forcing the integer variables to take integer values, another important issue is to reduce the overall general constraint violation measured by Θ. The ObPA has the ability to select the penalty objective function for the BCNLP problem. Either penalty (4.10) or (4.7) is used according to the general constraint feasibility level of the oracle. At iteration k, if $\Theta(o^*) \leq \eta^k$ then it is worth to penalize $|x_i - o_i^*|$ componentwise, so that an approximation near to the oracle is computed (and penalty (4.10) is used); otherwise, an approximation in the vicinity of the oracle is not of the upmost importance and the penalty (4.7) is used instead.

Besides the penalty parameters and the feasibility tolerance η^k, another parameter, μ^k, is required to check the level of integrality violation at the current solution x^k. Furthermore, the parameter δ^k represents the error bound which reflects the accuracy required for the current approximation x^k to the solution of the BCNLP problem.

Simple rules to control the reduction of parameters ε_d^k, ε_c^k, η^k, μ^k and δ^k are used and lower bounds are imposed to prevent the BCNLP problems of becoming very hard to solve. The penalty parameters ε_d^k and ε_c^k are reduced, using $\varepsilon_d^{k+1} = \max\{0.1\varepsilon_d^k, \varepsilon\}$ and $\varepsilon_c^{k+1} = \max\{0.1\varepsilon_c^k, \varepsilon\}$ respectively, when the corresponding violation measures ($\|x^k - z^k\|_\infty$ and $\Theta(x^k)$) at the computed approximation x^k are not satisfactory; otherwise, they are maintained.

The ObPA stops when an approximation x^k, which has a sufficiently small general constraints feasibility measure and is within an error of δ (in relative terms) of the known global solution, is computed. Thus, the stopping conditions are

$$\Theta(x^k) \leq \eta \text{ and } \frac{|f(x^k) - f^*|}{\max\{1, |f^*|\}} \leq \delta, \tag{4.14}$$

where η and δ are very small positive tolerances.

Remark 4.3 The use of the known global solution to stop the algorithm, during these preliminary tests, aims to analyze its effectiveness. In case f^* is not available, the second condition in (4.14) is replaced by the relative difference between the function values of two consecutive iterations less than or equal to the specified error tolerance, δ.

Finally, we now briefly elaborate on the global optimization method to solve the BCNLP problems formulated in (4.9) and (4.3). The deterministic algorithm DIRECT [25] is used. The problems to be addressed by DIRECT are defined in (4.9) and (4.3) in such a way that conditions (4.12) and (4.13) respectively are satisfied. The method does not require any derivative information and has been originally proposed to solve BCNLP problems, by producing finer and finer partitions of the hyperrectangles generated from X, and evaluating Ψ at their centers. The algorithm is a modification of the standard Lipschitzian approach that eliminates the need to specify the Lipschitz constant [25]. To perform a balance between global and local search, the algorithm makes use of two important concepts: potentially optimal hyperrectangle and grouping according to size. The center, c_i, the objective function value at the center point, $\Psi(c_i; \cdot)$, and the size, d_i, of each hyperrectangle i are used to define the groups of hyperrectangles, to select the potentially optimal hyperrectangles and to divide them into smaller ones, until a convergence condition is satisfied [27]. In the context of Algorithm 1, three stopping criteria were considered for DIRECT: (i) an error tolerance on the BCNLP objective penalty function value, δ^k, (ii) a maximum number of iterations, or (iii) a maximum number of function evaluations.

4.4 Numerical Experiments

To make a preliminary evaluation of the practical behavior of the proposed ObPA for solving nonconvex MINLP problems, we use a set of benchmark problems, identified as f1 to f29 in the subsequent tables (see [4, 17, 28]). The algorithm is implemented in Matlab™ (registered trademark of the MathWorks, Inc.) programming language. The algorithmic parameters are set as follows: $\eta = 1E-04$, $\delta = 1E-03$, $\mu = 1E-04$, $\varepsilon = 1E-05$, $\varepsilon_d^1 = 1$, $\varepsilon_c^1 = 0.1$, $\eta^1 = 0.1$, $\mu^1 = 0.1$. However, if the stopping conditions (4.14) do not hold for the given η and δ, ObPA is allowed to run for 30 iterations.

At each iteration k, when DIRECT is used to solve the BCNLP problems (4.9) or (4.3), by imposing the conditions (4.12) or (4.13) respectively, the error tolerance on the penalty function value is δ^k. We note that the parameter δ^1 is set to one, slowly decreases from one iteration to the other, until it reaches the value $\delta = 1E-03$. The maximum number of iterations is made to depend on the number of variables ($5n$ for f7; $10n$ for f3, f4, f8, f12, f14, f16, f18, f19, f24 and f26; $20n$ for f1, f5, f11 and f20; $50n$ for f9 and f17; $70n$ for f2, f22, f23, f28 and f29; $100n$ for f6, f13, f21 and f25; $150n$ for f15; $250n$ for f27; $300n$ for f10) and the maximum number of function evaluations is set to 50,000.

First, we compare the results produced by ObPA, as presented in Algorithm 1, with those obtained by a variant that does not use the oracle penalty, i.e., the BCNLP problem (4.3) is always solved in all iterations. See Table 4.1. The table shows the name of the problem, P, the best known optimal solution available in the literature, f^*, the solution produced by the algorithm, f_{sol}, the number of function evaluations required to achieved the reported solution, nfe, the number of iteration, nit, and the CPU time in seconds, T. From the results, it is possible to conclude that the proposed ObPA was able to find the global optimum for 20 of the 29 problems (according to the stopping conditions shown in (4.14) with $\eta = 1E-04$ and $\delta = 1E-03$). For the remaining nine problems, the algorithm run for 30 iterations. From the table, we may also conclude that the solutions obtained by the variant without the oracle penalty have been greatly deteriorated in three problems (f5, f7, f9) and slightly deteriorated in two (f11 and f12). The solutions for all the other problems are comparable, being f19 the only one with a slight improvement. Overall the results obtained by the proposed ObPA are superior to those of the tested variant.

Second, the results produced by ObPA are compared with those obtained by the BBMCSFilter, a BB-based multistart coordinate search filter method published in [18] and the results reported in [17], where a filter-based genetic algorithm (FGA) is presented. Table 4.2 shows the name of the problem, being the set f1–f12 also

Table 4.1 Numerical results produced by Algorithm 1 and by the variant without the oracle penalty

P	f^*	Algorithm 1				Variant without the oracle penalty			
		f_{sol}	nfe	nit	$T(sec.)$	f_{sol}	nfe	nit	T (sec.)
f1	2	2.000456	589	2	$1.29E-01$	2.000472	509	2	$1.10E-01$
f2	2.124	2.124481	5433	2	$2.54E+00$	2.124481	4891	2	$2.39E+00$
f3	1.07654	1.076392	1423	3	$5.96E-01$	1.076534	1233	3	$5.35E-01$
f4	99.239637	99.244695	629	2	$2.69E-01$	99.244695	523	2	$2.29E-01$
f5	3.557463	3.701380	103,049	30	$6.47E+01$	5.225669	87,569	30	$6.42E+01$
f6	4.579582	4.579600	88,843	3	$5.10E+01$	4.579600	77,111	3	$4.45E+01$
f7	−17	−16.691358	2039	30	$5.61E-01$	−10.333333	1757	30	$5.63E-01$
f8	−32217.4	−32215.640357	56,685	2	$4.39E+01$	−32215.640357	56,685	2	$4.62E+01$
f9	7.6671801	7.667232	20,523	4	$1.02E+01$	8.240213	198,977	30	$1.02E+02$
f10	−2.4444	−2.438023	354,975	30	$8.95E+01$	−2.438023	308,273	30	$7.97E+01$
f11	3.2361	3.236034	1417	2	$7.06E-01$	3.260172	21,901	30	$1.10E+01$
f12	1.125	1.125301	263	2	$5.94E-02$	1.132343	6911	30	$1.55E+00$
f13	87.5	89.500017	707,913	30	$3.22E+02$	89.500051	591,301	30	$2.74E+02$
f14	−6.666667	−6.666514	241	2	$8.68E-02$	−6.666514	223	2	$1.10E-01$
f15	−5.6848	−5.684732	14,315	3	$7.53E+00$	−5.684732	12,789	3	$6.79E+00$
f16	2.000	2.000119	1873	2	$8.24E-01$	2.000356	1549	2	$7.01E-01$
f17	3.4455	3.445514	5941	3	$1.19E+00$	3.445514	5235	3	$1.08E+00$
f18	2.2000	2.200032	5097	4	$2.28E+00$	2.200198	1445	2	$6.54E-01$
f19	6.00972	6.548438	39,871	30	$2.36E+01$	6.424818	35,395	30	$2.09E+01$
f20	−17.0000	−16.999953	16,871	6	$7.87E+00$	−16.999953	14,627	6	$7.03E+00$

(continued)

Table 4.1 (continued)

P	f^*	Algorithm 1				Variant without the oracle penalty			
		f_{sol}	nfe	nit	$T(sec.)$	f_{sol}	nfe	nit	T (sec.)
f21	−4.514202	−4.514198	25,999	4	$1.46E+01$	−4.514154	23,529	4	$1.39E+01$
f22	−13.401904	−13.401855	67,081	4	$3.60E+01$	−13.401855	36,605	3	$1.90E+01$
f23	−1.08333	−1.078680	206,889	30	$9.65E+01$	−1.078667	204,335	30	$9.44E+01$
f24	−0.94347	−0.664913	300,055	30	$1.98E+02$	−0.664913	267,527	30	$1.73E+02$
f25	189.3116	189.375606	14,855	4	$7.52E+00$	189.375388	13,161	4	$6.80E+00$
f26	31	31.000339	777	4	$1.85E-01$	31.001016	685	4	$1.60E-01$
f27	−32	−31.998899	34,169	4	$8.32E+00$	−31.998628	30,237	4	$6.90E+00$
f28	73.0353	78.769766	1425,125	30	$9.96E+02$	78.769766	1423,437	30	$1.00E+03$
f29	−1.923	−0.913446	991,839	30	$9.19E+02$	−0.913446	1451,577	30	$1.12E+03$

Table 4.2 Numerical results produced by the Algorithm 1, the BBMCSFilter in [18] and the FGA in [17]

		Algorithm 1		BBMCSFilter [†]			FGA [§]		
P	$(\|I_c\|,\|I_d\|)$	f_{sol}	nfe	f_{avg}	SD	nfe_{avg}	f_{avg}	SD	nfe_{avg}
f1	(1,1)	2.000456	589	2.000817	$3.6E-04$	3530	2.0000	$1.6E-06$	4530
f2	(1,1)	2.124481	5433	2.124590	$1.4E-06$	1259	2.1852	$6.1E-02$	3799
f3	(2,1)	1.076392	1423	1.081640	$8.1E-03$	5274	1.0769	$3.8E-04$	5752
f4	(2,1)	99.244695	629	99.239635	$1.0E-07$	670	99.5784	$3.4E-01$	9854
f5	(3,4)	3.701380	103,049	3.560848	$2.0E-03$	76,775	3.6822	$1.2E-01$	11,492
f6	(3,4)	4.579600	88,843	4.582322	$9.3E-04$	75,413	4.8048	$2.3E-01$	9937
f7	(1,1)	−16.691358	2039	−16.998054	$2.3E-03$	4296	−16.8267	$1.7E-01$	4147
f8	(3,2)	−32215.640357	56,685	−32217.428	$0.0E+00$	18,051	−32217	$2.7E-02$	6609
f9	(2,3)	7.667232	20,523	7.667583	$9.5E-04$	28,090	7.7472	$8.0E-02$	11,480
f10	(1,1)	−2.438023	354,975	−2.444444	$0.0E+00$	2736	−2.444	$4.4E-04$	4125
f11	(1,2)	3.236034	1417	3.236121	$8.7E-05$	41,635	3.3395	$1.0E-01$	5028
f12	(1,1)	1.125301	263	1.125115	$2.9E-04$	7770	1.125	$1.4E-06$	4757
f13	(2,2)	89.500017	707,913	87.507043	$1.7E-02$	41,852	–	–	–
f14	(1,1)	−6.6665143	241	−6.666131	$1.8E-04$	1122	–	–	–
f15	(1,2)	−5.684732	14,315	−5.651952	$2.6E-02$	393,345	–	–	–
f16	(2,2)	2.000119	1873	2.000000	$0.0E+00$	29,847	–	–	–
f17	(1,1)	3.445514	5941	3.445808	$2.1E-04$	5469	–	–	–
f18	(1,3)	2.200032	5097	2.200000	$0.0E+00$	11,182	–	–	–
f19	(4,2)	6.548438	39,871	6.010714	$6.6E-04$	37,132	–	–	–
f20	(2,3)	−16.999953	16,871	−16.994605	$5.5E-03$	27,149	–	–	–
f21	(1,3)	−4.514198	25,999	−4.513448	$6.8E-04$	50,146	–	–	–

(continued)

Table 4.2 (continued)

		Algorithm 1		BBMCSFilter †			FGA §		
P	$(\|I_c\|,\|I_d\|)$	f_{sol}	nfe	f_{avg}	SD	nfe_{avg}	f_{avg}	SD	nfe_{avg}
f22	(2,4)	−13.401855	67,081	−13.401930	$3.6E-04$	84,790	–	–	–
f23	(2,2)	−1.078680	206,889	−1.083245	$5.4E-05$	2458	–	–	–
f24	(3,8)	-0.664913	300,055	–	–	–	–	–	–
f25	(2,1)	189.375606	14,855	–	–	–	–	–	–
f26	(0,2)	31.000339	777	–	–	–	–	–	–
f27	(1,1)	−31.998899	34,169	–	–	–	–	–	–
f28	(6,5)	78.769766	1425,125	–	–	–	–	–	–
f29	(5,3)	−0.913446	991,839	–	–	–	–	–	–

† The NLP relaxation is stopped after 10 sample points are generated in the multistart algorithm and 30 runs are executed

§ The algorithm stops when a solution with error $1E-3$ is found or the number of function evaluations reaches 10,000; $P_s = 20$, $R = 50$

Table 4.3 Other numerical comparisons

P	Algorithm 1		EXP-MIP		4-rule FA $^\flat$		MIHDE §		ACOmi †		PSO ‡		pen-GA $^\natural$	
	f_{sol}	$nfe\ (nit)$	f_{exp}	# nod.	f_{avg}	nfe_{avg}	f_{avg}	nfe_{avg}	f_{avg}	nfe_{avg}	% suc.	nfe_{avg}	% suc.	nfe_{avg}^{suc}
f1	2.000011	1589 (2)	–	–	2.0000	3409	–	13,104	–	–	–	–	84	172
f2	2.124476	13,449 (2)	–	–	2.7149	5253	–	29,166	–	–	100	3500	85	64
f3	1.076392	1423 (3)	1.076	0	1.0767	5178	–	28,455	1.1459	4250	–	–	43	18,608
f4	99.244695	629 (2)	–	–	–	–	–	60,950	–	–	100	4000	59	7447
f5	3.701662	38,287 (11)	–	–	–	–	–	12,375	–	–	–	–	41	3571
f6	4.579600	33,859 (3)	4.579	2	4.7758	12,157	–	–	4.5796	731	100	30,000	–	–
f7	−16.998720	1501 (2)	−17	1	−16.9998	3243	–	983	−17	307	–	–	–	–
f8	−32215.640357	56,685 (2)	–	–	–	–	–	50,976	–	–	–	–	100	100
f9	7.667232	20,523 (4)	7.667	2	8.0695	8622	–	–	7.6672	363	–	–	–	–
f10	−2.438023	12,395 (3)	–	–	−2.4380	3501	–	–	−2.4444	270	–	–	–	–
f11	3.236034	1397 (2)	–	–	3.2361	4405	–	–	23.475	1180	–	–	–	–
f24	−0.686926	62,391 (3)	−0.912	1	–	–	–	–	–	–	–	–	93	258
f26	31.000339	801 (4)	31	1	–	–	–	–	–	–	–	–	–	–
f29	−1.393493	36,191 (5)	–	–	–	–	–	–	–	–	88	40,000	–	–

$^\flat$ Termination conditions: $|f^k - f^*|/|f^*| \le 1E-04$ and violation $\le 1E-03$; $P_s = 20,\ R = 30$
§ Termination condition: $|f^{k+20} - f^k| < 1E-05$ or a maximum of 2000 iterations; $P_s = 3,\ R = 10$
† Algorithm stops when a solution with error $1E-03$ is reached or a maximum of 10, 000 function evaluations is attained; $P_s = 20,\ R = 30$
‡ Termination conditions: $|f^{k+50} - f^k| < 1E-05$ or a maximum of 200 iterations; $P_s = 50,\ R = 100$
$^\natural$ Termination conditions: $|f^k - f^*| \le 1E-02$ or a maximum of 200 iterations; $P_s = 10n,\ R = 100$

used in [17] and the set f1–f23 used in [18]. In the second column of the table, the pair inside parenthesis corresponds to $(|I_c|,|I_d|)$. The remaining columns contain: the solution produced by ObPA, f_{sol}, and the number of function evaluations, nfe, the average value of the objective function values produced by all the executed runs (with BBMCSFilter and FGA), f_{avg}, the standard deviation of the function values, SD, and the average number of function evaluations (over all the runs), nfe_{avg}. The character '–' in the tables means that the information is not available in the cited papers, 'P_s' is the size of the population and 'R' gives the number of independent executed runs. From the comparison, we may conclude that the produced solutions are of good quality. For most problems, the number of required function evaluations is moderate when compared with the numbers produced by the other algorithms, with the exception in nine problems where it is much higher. In seven of these problems, the algorithm reached 30 iterations since one of the conditions in (4.14) was not satisfied. Thus, from the comparison with the BBMCSFilter and FGA, the ObPA proves to be competitive either in terms of the quality of the found solutions or in the number of function evaluations.

Finally, using a small subset of the problems, we compare our results with those reported by other strategies. Table 4.3 reports the solution produced by Algorithm 1, f_{sol}, the number of function evaluations, nfe, and the number of iterations, nit. The algorithm is made to stop when a solution with an error of $1E-03$ is reached or a maximum of $5000n$ function evaluations is attained. The other results in the table are collected from the exact penalty for mixed-integer programs (EXP-MIP) in [13], the 4-rule FA in [20], the MIHDE in [14], the extended version of the ant colony optimization (ACOmi) in [19], the particle swarm optimization (PSO) in [15] and the penalty GA (pen-GA) in [16]. The table also shows the solution found by EXP-MIP, f_{exp}, and the number of nodes (corresponding to the number of branch and reduce iterations), '# nod.'.

As far as the stochastic heuristics are concerned, Table 4.3 shows: the average of the objective function values (over all the executed runs), f_{avg}, the average number of function evaluations, nfe_{avg}, the percentage of successful runs (according to the stopping condition based on the proximity of f to f^*), % suc., and the average number of function evaluations of the successful runs alone, nfe^{suc}_{avg}. From the results we may conclude that the proposed ObPA performs reasonably well.

4.5 Conclusions

In this paper, an oracle-based penalty approach for solving nonsmooth and nonconvex MINLP problems is proposed. A continuous reformulation BCNLP problem is solved by the deterministic DIRECT solver. The penalty function to be optimized involves a combination of penalty terms to penalize the integrality constraints, the equality and inequality constraints and the distance to the oracle, based on hyperbolic tangent penalty functions. The numerical experiments show that the proposed algorithm gives competitive results when compared with other methods in the literature.

Future developments will be directed to improve the efficiency of the oracle-based penalty algorithm, in terms of the number of function evaluations, by using an alternative deterministic and derivative-free global optimizer to solve the continuous BCNLP problems.

Acknowledgements The authors would like to thank two anonymous referees for their valuable comments and suggestions to improve the paper.

This work has been supported by COMPETE: POCI-01-0145-FEDER-007043 and FCT - Fundação para a Ciência e Tecnologia, within the projects UID/CEC/00319/2013 and UID/MAT/00013/2013.

References

1. O. Exler, T. Lehmann, K. Schittkowski, A comparative study of SQP-type algorithms for nonlinear and nonconvex mixed-integer optimization. Math. Program. Comput. **4**(4), 383–412 (2012)
2. S. Burer, A.N. Letchford, Non-convex mixed-integer nonlinear programming: a survey. Surv. Oper. Res. Manag. Sci. **17**(2), 97–106 (2012)
3. S. Lee, I.E. Grossmann, A global optimization algorithm for nonconvex generalized disjunctive programming and applications to process systems. Comput. Chem. Eng. **25**(11), 1675–1697 (2001)
4. H.S. Ryoo, N.V. Sahinidis, Global optimization of nonconvex NLPs and MINLPs with applications in process design. Comput. Chem. Eng. **19**(5), 551–566 (1995)
5. C.S. Adjiman, I.P. Androulakis, C.A. Floudas, Global optimization of mixed-integer nonlinear problems. AIChE J. **46**(9), 1769–1797 (2000)
6. V.K. Srivastava, A. Fahim, An optimization method for solving mixed discrete-continuous programming problems. Comput. Math. Appl. **53**(10), 1481–1491 (2007)
7. L. Liberti, G. Nannicini, N. Mladenović, A good recipe for solving MINLPs, in *Matheuristics: Hybridizing Metaheuristics and Mathematical Programming*, vol. 10, ed. by V. Maniezzo, T. Stützle, S. Voß (Springer, US, 2010), pp. 231–244
8. G. Nannicini, P. Belotti, Rounding-based heuristics for nonconvex MINLPs. Math. Program. Comput. **4**(1), 1–31 (2012)
9. C. D'Ambrosio, A. Frangioni, L. Liberti, A. Lodi, A storm of feasibility pumps for nonconvex MINLP. Math. Program. **136**(2), 375–402 (2012)
10. G. Liuzzi, S. Lucidi, F. Rinaldi, Derivative-free methods for bound constrained mixed-integer optimization. Comput. Optim. Appl. **53**(2), 505–526 (2012)
11. M.F.P. Costa, A.M.A.C. Rocha, R.B. Francisco, E.M.G.P. Fernandes, Firefly penalty-based algorithm for bound constrained mixed-integer nonlinear programming. Optimization **65**(5), 1085–1104 (2016)
12. S. Lucidi, F. Rinaldi, Exact penalty functions for nonlinear integer programming problems. J. Optim. Theory Appl. **1**45(3), 479–488 (2010)
13. S. Lucidi, F. Rinaldi, An exact penalty global optimization approach for mixed-integer programming problems. Optim. Lett. **7**(2), 297–307 (2013)
14. Y.-C. Lin, K.-S. Hwang, F.-S. Wang, A mixed-coding scheme of evolutionary algorithms to solve mixed-integer nonlinear programming problems. Comput. Math. Appl. **47**(8–9), 1295–1307 (2004)
15. L. Yiqing, Y. Xigang, L. Yongjian, An improved PSO algorithm for solving non-convex NLP/MINLP problems with equality constraints. Comput. Chem. Eng. **31**(3), 153–162 (2007)
16. K. Deep, K.P. Singh, M. L. Kansal, C. Mohan, A real coded genetic algorithm for solving integer and mixed integer optimization problems. Appl. Math. Comput. **212**(2), 505–518 (2009)

17. A. Hedar, A. Fahim, Filter-based genetic algorithm for mixed variable programming. Numer. Algebr. Control Optim. **1**(1), 99–116 (2011)
18. F.P. Fernandes, M.F.P. Costa, E.M.G.P. Fernandes, Branch and bound based coordinate search filter algorithm for nonsmooth nonconvex mixed-integer nonlinear programming problems, in *Computational Science and Its Applications – ICCSA 2014, Part II*, LNCS, vol. 8580, ed. by B. Murgante, S. Misra, A.M.A.C. Rocha, C. Torre, J.G. Rocha, M.I. Falcão, D. Taniar, B.O. Apduhan, O. Gervasi (Springer, Berlin, 2014), pp. 140–153
19. M. Schlüter, J.A. Egea, J.R. Banga, Extended ant colony optimization for non-convex mixed integer nonlinear programming. Comput. Oper. Res. **36**(7), 2217–2229 (2009)
20. M.F.P. Costa, A.M.A.C. Rocha, R.B. Francisco, E.M.G.P. Fernandes, Extension of the firefly algorithm and preference rules for solving MINLP problems, in *International Conference of Numerical Analysis and Applied Mathematics (ICNAAM 2016)*, AIP. Conf. Proc. **1863**, 270003-1–270003-4 (2017)
21. P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, A. Mahajan, Mixed-integer nonlinear optimization. Acta Numer. **22**, 1–131 (2013)
22. F. Boukouvala, R. Misener, C.A. Floudas, Global optimization advances in mixed-integer nonlinear programming, MINLP, and constrained derivative-free optimization, CDFO. Eur. J. Oper. Res. **252**(3), 701–727 (2016)
23. M.R. Bussieck, S. Vigerske, MINLP solver software, in *Wiley Encyclopedia of Operations Research and Management Science*, ed. by J.J. Cochran, L.A. Cox, P. Keskinocak, J.P. Kharoufeh, J.C. Smith (Wiley, New York, 2011)
24. A.M.A.C. Rocha, M.F.P. Costa, E.M.G.P. Fernandes, Solving MINLP problems by a penalty framework, in *Proceedings of XIII Global Optimization Workshop*, ed. by A.M. Rocha, M.F. Costa, E. Fernandes, (2016), pp. 97–100
25. D.R. Jones, C.D. Perttunen, B.E. Stuckman, Lipschitzian optimization without the Lipschitz constant. J. Optim. Theory Appl. **79**(1), 157–181 (1993)
26. M. Schlüter, M. Gerdts, The oracle penalty method. J. Glob. Optim. **47**(2), 293–325 (2010)
27. D.E. Finkel, DIRECT Optimization Algorithm User Guide, Center for Research in Scientific Computation. (CRSC-TR03-11, North Carolina State University, Raleigh, NC 27695-8205, March 2003)
28. C.A. Floudas, P.M. Pardalos, C. Adjiman, W.R. Esposito, Z.H. Gümüs, S.T. Harding, J.L. Klepeis, C.A. Meyer, C.A. Schweiger, *Handbook of Test Problems in Local and Global Optimization*, Nonconvex Optimization and its Applications (Springer Science & Business Media, Dordrecht, 1999)