On mixed integer nonsmooth optimization

Eronen, Ville-Pekka; Westerlund, Tapio; Mäkelä, Marko M.

Published in:
Numerical Nonsmooth Optimization

DOI:
10.1007/978-3-030-34910-3_16

Published: 01/01/2020

Document Version
Accepted author manuscript

Document License
Publisher rights policy

On mixed integer nonsmooth optimization

Ville-Pekka Eronen, Tapio Westerlund and Marko M. Mäkelä

Abstract In this chapter we review some deterministic solution methods for convex mixed integer nonsmooth optimization problems. The methods are branch and bound, outer approximation, extended cutting plane, extended supporting hyperplane and extended level bundle method. Nonsmoothness is taken into account by using Clarke subgradients as a substitute for the classical gradient. Ideas for convergence proofs are given as well as references where the details can be found. We also consider how some algorithms can be modified in order to solve nonconvex problems including f°-pseudoconvex functions or even f°-quasiconvex constraints.

1 Introduction

In mixed integer optimization some variables are continuous and some are integers. The difficulty in dealing with integer variables is that the feasible set is not necessarily connected nor convex. This causes finding descent direction and doing line searches less fruitful than in continuous optimization. The deterministic methods to solve mixed integer nonlinear programming (MINLP) problems with differentiable functions include branch and bound [10], outer approximation [13] and cutting plane [48] methods. An important

Ville-Pekka Eronen
Department of Mathematics and Statistics, University of Turku, Turku, Finland, e-mail: vpoero@utu.fi

Tapio Westerlund
Department of Mathematics and Statistics, University of Turku, Turku, Finland e-mail: twesterl@abo.fi

Marko M. Mäkelä
Department of Mathematics and Statistics, University of Turku, Turku, Finland e-mail: makela@utu.fi

special case of an MINLP problem is when integers are binary variables having only possible values $\{0, 1\}$. In this case the variables can model yes or no type decisions, making these kind of problems important in various applications. MINLP models have been used in many fields including chemical and mechanical engineering, physics, medicine and, for example, in design of water/gas network [2, 3, 28, 29, 39].

In nonsmooth optimization (NSO) some functions are not necessarily continuously differentiable. A typical reason for nonsmoothness is functions like max, $|\cdot|$ or $\|\cdot\|$. We will only consider the case where functions are locally Lipschitz continuous (LLC). Lipschitz continuity implies that the set of nondifferentiable points has zero measure [1]. Furthermore, while we cannot calculate the classical gradient, a Clarke subgradient belonging to the Clarke subdifferential exists at any point [1, 7]. The problem in solving NSO is finding a descent direction, which is not as readily available as in the continuously differentiable case. In addition, identifying a minimum in which objective function is not differentiable may be problematic. Nonsmooth problems can be found in many applications for example in optimal control problems, data analysis and economics [1]. In general NSO problems can be solved with subgradient (See e.g. [43] and Chapter A in this book) and bundle methods (See e.g. [15, 35] and Chapter B in this book). In addition, derivative free methods (See e.g. [41] and Chapter **??** in this book) or gradient sampling methods (See e.g. Chapter **??** in this book) can be used to solve nonsmooth problems as well but we do not study these methods here.

In this chapter we review some deterministic methods that solve convex mixed integer nonsmooth optimization problems (MINSO) with help of the Clarke subdifferential. In these problems the objective and constraint functions are convex and the integer relaxed feasible set is convex. Furthermore, we assume that we can evaluate a subgradient of any function at any given point. Some of the methods can be generalized to solve problems with f°-pseudoconvex functions or even f°-quasiconvex constraint functions. Methods to solve MINSO problems has been studied recently in [11, 14, 18, 20, 46, 51], among others.

MINSO problems can be found for example in electrical engineering [6], gas network planning [42], statistics [52] and facility layout problems [5]. In these articles the problems were solved with metaheuristics and reformulation techniques. Other techniques, not covered here, are derivative free algorithms to solve MINSO problems containing black box functions. These methods can be found, for example, in [33, 36, 37] and references therein.

One reason for the lack of research on methods for mixed integer NSO is that typical nonsmooth functions, like max and $|\cdot|$, can be modelled with auxiliary variables leading to a smooth MINLP problem. However, transformation sometimes leads to a more difficult problem as was noticed in [51]. In that case transforming a mixed integer NSO problem by means of a pseudoconvex objective function resulted in a nonconvex MINLP problem. Another

way to deal with function nonsmoothness is to use smoothing techniques. For example $|x| \approx \sqrt{x^2 + \tau}$ for a small $\tau > 0$ [42].

In Section 2 we present several deterministic algorithms to solve convex MINSO problems. Ideas of the convergence proofs are given as well as references to articles where more details can be found. In Section 3 the algorithms are illustrated by solving a simple example problem.

2 Deterministic algorithms for convex mixed integer NSO

The considered MINSO problem can be formulated as follows

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{x}, \mathbf{y}) \\ \text{subject to} \quad & g_j(\mathbf{x}, \mathbf{y}) \leq 0,\ j \in \mathcal{J} \qquad \text{(MINSO)} \\ & (\mathbf{x}, \mathbf{y}) \in L, \mathbf{x} \in \mathbb{R}^n, \mathbf{y} \in \mathbb{Z}^m, \end{aligned}$$

where $\mathcal{J} = \{1, 2, \ldots, J\}$. The set L is defined by linear constraints and it is assumed to be compact. Hence, we assume that there are finitely many feasible integer vectors $\mathbf{y}$. Define the finite set

$$Y = \{\mathbf{y} \in \mathbb{Z}^m \mid (\mathbf{x}, \mathbf{y}) \in L, \mathbf{x} \in \mathbb{R}^n\}.$$

Sometimes it is convenient not to separate continuous and integer variables and due to this we define $\mathbf{z} = (\mathbf{x}, \mathbf{y})$ and set

$$Z = \{\mathbf{z} = (\mathbf{x}, \mathbf{y}) \mid \mathbf{x} \in \mathbb{R}^n, \mathbf{y} \in \mathbb{Z}^m\}.$$

Denote also

$$G(\mathbf{z}) = \max_{j \in \mathcal{J}} \{g_j(\mathbf{z})\} \quad \text{and} \quad N = \{\mathbf{z} \mid G(\mathbf{z}) \leq 0\}.$$

With these notations we can formulate the problem (MINSO) as follows

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{z}) \\ \text{s.t} \quad & \mathbf{z} \in N \cap L \cap Z. \end{aligned}$$

If not otherwise stated, the functions f and g_j, $j \in \mathcal{J}$ are assumed to be convex and thus LLC. Hence, the problem is convex: nonlinear functions are convex and the feasible set is convex when integer variables are relaxed to be continuous variables.

Some of the algorithms can be generalized to solve problems with f°-pseudoconvex functions and even f°-quasiconvex constraint functions.

Definition 1. A LLC function $f : \mathbb{R}^n \to \mathbb{R}$ is $f^\circ - pseudoconvex$ ($f^\circ - quasiconvex$) if for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$

$$f(\mathbf{y}) < (\leq) f(\mathbf{x}) \qquad \text{implies} \qquad f^\circ(\mathbf{x}; \mathbf{y} - \mathbf{x}) < (\leq) 0.$$

f°-pseudoconvexity is a straightforward generalization of the classical pseudoconvexity to LLC functions. f°-quasiconvexity is slightly more restrictive than quasiconvexity for LLC functions [1]. A convex function is always f°-pseudoconvex, which in turn, is always f°-quasiconvex. Moreover, an f°-quasiconvex function is always quasiconvex. An important feature of an f°-pseudoconvex function is that a local minimum is also a global one. Another important property, that holds for f°-quasiconvex functions as well, is that the level sets $\{\mathbf{x} \in \mathbb{R}^n \mid f(\mathbf{x}) \leq a\}$ are convex for any $a \in \mathbb{R}$. From the definition of generalized directional derivative and Definition 1 we can see that inequality

$$\boldsymbol{\xi}^T(\mathbf{x}_1 - \mathbf{x}_2) \leq 0, \tag{1}$$

holds true for any $\boldsymbol{\xi} \in \partial f(\mathbf{x}_2)$ and $\mathbf{x}_1 \leq \mathbf{x}_2$ such that $f(\mathbf{x}_1) < f(\mathbf{x}_2)$ if f is f°-quasiconvex. This inequality will prove to be useful when studying problems with f°-quasiconvex constraint functions.

Articles [2, 24] describe several deterministic methods to solve the MINLP in case all the functions are continuously differentiable. The following deterministic methods for the nonsmooth case (MINSO) are mainly based on those. Most of the methods require that we are able to calculate the values of the nonlinear functions and an arbitrary subgradient at the points where the algorithm visits. These points are assumed to belong to the set L.

2.1 NSO branch and bound

Branch and bound (B&B) method for mixed integer linear programming (MILP) problems was developed in 1960 [31] and it is a general framework to deal with integer variables. The method was generalized for MINLP problems in [10]. In B&B method only integer relaxed problems are solved, that is, problems where integer variables are treated as continuous ones. To cut off the previous non-integer solutions, bounds to the integer variables are added to the subsequent problems. The problems to be solved can also be seen as nodes of a tree.

The B&B method solves the problem (MINSO) as a sequence of continuous NSO subproblems. The first subproblem is

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{x}, \mathbf{y}) \\ \text{s.t} \quad & g_j(\mathbf{x}, \mathbf{y}) \leq 0,\ j \in \mathcal{J} \qquad \text{(BB-NSO)} \\ & (\mathbf{x}, \mathbf{y}) \in L, \mathbf{x} \in \mathbb{R}^n, \mathbf{y} \in \mathbb{R}^m. \end{aligned}$$

If at iteration k the solution of the subproblem is not integer feasible, *branching* will be applied. In branching we pick an integer variable y_i that was not integer at the solution point $(\mathbf{x}^k, \mathbf{y}^k)$. Then we create two subproblems with all constraints from the previously solved problem and add the constraint $y_i \leq \lfloor y_i^k \rfloor$ to one subproblem and $y_i \geq \lceil y_i^k \rceil$ to the other. Clearly, any of these constraints will cut off the previous solution point. The branching does not occur in three cases:

- The problem has no feasible solution
- The solution is integer feasible. In this case the solution is feasible in the original MINSO problem and, thus, the objective function value at the solution is an upper bound.
- The solution is not integer feasible but the value of the objective function is greater than the current best upper bound.

In these cases the node will be pruned and it becomes a leaf of the tree. B&B continues to solve previously created subproblems until none is left. As noted in [2] the subproblems can be uniquely identified by the bounds $(\mathbf{l}, \mathbf{u})$ given to the integer vectors. For example, problem NSO$(\mathbf{l}, \mathbf{u})$ corresponds to subproblem with bounds $\mathbf{l} \leq \mathbf{y} \leq \mathbf{u}$ and NSO$(-\infty, \infty)$ corresponds to the first subproblem or the root node. The NSO branch and bound algorithm is presented in Algorithm 1.

Algorithm 1: The NSO branch and bound algorithm

Step 1 Set upper bound $U = \infty$, $k = 0$ and initiate the list of unsolved subproblems $\mathcal{L} = \{\text{NSO}(-\infty, \infty)\}$.

Step 2 If $\mathcal{L} = \emptyset$ the current upper bound is the global minimum. Otherwise, choose a subproblem NSO$(\mathbf{l}, \mathbf{u})$ from the list $\mathcal{L}$ and update $\mathcal{L} = \mathcal{L} \setminus \{\text{NSO}(\mathbf{l}, \mathbf{u})\}$.

Step 3 Solve NSO$(\mathbf{l}, \mathbf{u})$. If NSO$(\mathbf{l}, \mathbf{u})$ is infeasible go to Step 2. Otherwise, denote $(\mathbf{x}^k, \mathbf{y}^k)$ the solution of NSO$(\mathbf{l}, \mathbf{u})$ and set $k = k + 1$.

Step 4 Suppose $\mathbf{y}^k$ is an integer vector. If $f(\mathbf{x}^k, \mathbf{y}^k) < U$ set $U = f(\mathbf{x}^k, \mathbf{y}^k)$. Go to Step 2.

Step 5 Suppose that $\mathbf{y}^k$ is not an integer vector. If $f(\mathbf{x}^k, \mathbf{y}^k) > U$, go to Step 2. Otherwise, take an integer variable y_i that was not integer at $\mathbf{y}^k$. Set $\mathbf{l}^- = \mathbf{l}$, $l_i^- = \lceil y_i^k \rceil$, $\mathbf{u}^+ = \mathbf{u}$ and $u_i^+ = \lfloor y_i^k \rfloor$. Add two new subproblems to the list $\mathcal{L} = \mathcal{L} \cup \{\text{NSO}(\mathbf{l}^-, \mathbf{u}), \text{NSO}(\mathbf{l}, \mathbf{u}^+)\}$. Go to Step 2.

Theorem 1. *If the NSO subproblems are solved to a global minimizer, Algorithm 1 finds a global minimizer of (MINSO) after solving a finite number of NSO subproblems.*

Proof. Since there are finitely many feasible integer vectors the tree will be finite as well. If every NSO subproblem is solved to a global minimum, the minimum of the original problem will be found. □

In the worst case the B&B method solves more NSO problems than there are integer vectors in Y. For example if Y consists of binary vectors $\{0,1\}^m$ there are 2^m possible binary vectors but in the worst case B&B method solves $2^{m+1}-1$ NSO problems. Thus, the efficiency of B&B is based on ruling out regions from the feasible set by a good upper bound or by finding integer feasible solutions from the relaxed problems.

There is a couple of degrees of freedom in Algorithm 1. The choice of the branching variable and the choice of the problem to be solved at Step 3 is discussed for example in [2]. Good choices will ideally result in a smaller search tree. The choice of branching variable aims to find a variable causing the minimum of the resulting NSO problem to be greatest making it probable to prune it. The choice of NSO problem aims to find a good upper bound fast and then increase the lower bound in order to prove the optimality of the current upper bound.

The NSO algorithm itself is free to choose as long as it can solve the NSO problems to the global minimum. Nonsmooth algorithms like in [15, 35,] could be used to solve the NSO problems we consider. However, to authors' knowledge there exists no implementation of these with B&B. There exists, though, an implementation of B&B with a derivative free NLP algorithm [21] that can deal with the nonsmooth functions.

Being a general way to handle integer variables the B&B method can be used to solve also MILP problems. This can be done by using a linear programming solver instead of the NSO solver in Algorithm 1. In fact many popular LP solvers, like CPLEX [8] and Gurobi [25], utilize sophisticated versions of B&B and can solve MILP problems very efficienly. MILP is an important class of problems since the other methods we consider to solve the problem (MINSO) generate MILP subproblems as a part of their solution strategy.

2.2 Outer approximation

The outer approximation method was developed in 80s-90s [13, 22]. The linear outer approximation method is not as readily applicable for nonsmooth problems as the NLP branch and bound. Hence, we will present the method for the smooth case first.

In the linear outer approximation method an MILP master problem is created by removing nonlinearities from the original MINLP problem. The nonlinear functions are taken into account by adding linear approximations of them to the MILP master problem. The points at which these approximations are made are found by solving NLP problems. Thus, the method alternates between solving NLP and MILP problems. At iteration k the NLP problem corresponds to the original MINLP problem where integer vector has a fixed value $\mathbf{y}^k$ obtained as an initial point or from the solution point of the MILP

master problem. The NLP problem can be written as

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{x}, \mathbf{y}^k) \\ \text{s.t} \quad & g_j(\mathbf{x}, \mathbf{y}^k) \leq 0,\ j \in \mathcal{J} \quad \text{(OA-NLP}(\mathbf{y}^k)\text{)} \\ & (\mathbf{x}, \mathbf{y}^k) \in L, \mathbf{x} \in \mathbb{R}^n. \end{aligned}$$

The solution of this problem will provide an upper bound for the original MINLP problem. Denote U^k the smallest upper bound found after solving the kth NLP or feasibility problem. If the problem (OA-NLP($\mathbf{y}^k$)) turns out to be infeasible we need to solve a feasibility problem instead. One possible feasibility problem is

$$\begin{aligned} \text{minimize} \quad & \mu \\ \text{s.t} \quad & g_j(\mathbf{x}, \mathbf{y}^k) \leq \mu,\ j \in \mathcal{J} \quad \text{(OA-F}(\mathbf{y}^k)\text{)} \\ & (\mathbf{x}, \mathbf{y}^k) \in L, \mathbf{x} \in \mathbb{R}^n. \end{aligned}$$

In the both cases, linearizations of the nonlinear functions at the solution point $(\mathbf{x}^k, \mathbf{y}^k)$ will be added to the MILP master problem

$$\begin{aligned} \text{minimize} \quad & \eta \\ \text{s.t} \quad & \eta < U^k - \varepsilon \\ & \eta \geq f(\mathbf{x}^i, \mathbf{y}^i) + \nabla f(\mathbf{x}^i, \mathbf{y}^i)^T \begin{pmatrix} \mathbf{x} - \mathbf{x}^i \\ \mathbf{y} - \mathbf{y}^i \end{pmatrix}, i \in T^k \quad \text{(OA-MILP}_k\text{)} \\ & 0 \geq g_j(\mathbf{x}^i, \mathbf{y}^i) + \nabla g_j(\mathbf{x}^i, \mathbf{y}^i)^T \begin{pmatrix} \mathbf{x} - \mathbf{x}^i \\ \mathbf{y} - \mathbf{y}^i \end{pmatrix}, i \in T^k \cup S^k, j \in \mathcal{J} \\ & (\mathbf{x}, \mathbf{y}) \in L, \mathbf{x} \in \mathbb{R}^n, \mathbf{y} \in \mathbb{Z}^m, \eta \in \mathbb{R}. \end{aligned}$$

Here $\varepsilon > 0$ is a small user given parameter and the set T^k collects indices when (OA-NLP($\mathbf{y}^k$)) is feasible whereas S^k collects indices when it is not.

Algorithm 2: The OA algorithm

Data Let feasible integer solution $\mathbf{y}^1 \in \mathbb{Z}^m$ be given.
Step 1 Set $k = 1$, $T^0 = \emptyset$, $S^0 = \emptyset$ and $U^0 = \infty$.
Step 2 Solve (OA-NLP($\mathbf{y}^k$)), or the feasibility problem (OA-F($\mathbf{y}^k$)), if (OA-NLP($\mathbf{y}^k$)) is infeasible, and let the solution be $\mathbf{x}^k$.
Step 3 Linearize the objective and constraint functions on $(\mathbf{x}^k, \mathbf{y}^k)$. Set $T^k = T^{k-1} \cup \{k\}$ or $S^k = S^{k-1} \cup \{k\}$ as appropriate.
Step 4 If (OA-NLP($\mathbf{y}^k$)) is feasible and $f(\mathbf{x}^k, \mathbf{y}^k) < U^{k-1}$ then update $U^k = f(\mathbf{x}^k, \mathbf{y}^k)$. Otherwise, set $U^k = U^{k-1}$
Step 5 Solve (OA-MILP$_k$), giving a new integer vector $\mathbf{y}^{k+1}$. If (OA-MILP$_k$) is infeasible stop: the current upper bound is the global minimum. Otherwise, set $k = k + 1$ and go to Step 2.

To guarantee the optimality of the solution we need to assume that KKT-conditions hold at the minimizer of (OA-NPL($\mathbf{y}^k$)) for each feasible $\mathbf{y}^k$. This assumption holds if an appropriate constraint qualification is satisfied at $(\mathbf{x}^k, \mathbf{y}^k)$ one example being that the gradients of the active constraint functions are linearly independent [22]. Due to variable μ the gradients of constraint functions of (OA-F($\mathbf{y}^k$)) are never $\mathbf{0}$. Thus, the Cottle constraint qualification [1] holds at any point, implying that the classical KKT-conditions hold at the minimum without any additional assumptions.

When the assumption holds, the linearizations made at $(\mathbf{x}^k, \mathbf{y}^k)$ will make the integer value $\mathbf{y}^k$ infeasible in the subsequent MILP master problems. The convexity of f and g_j for all $j \in \mathcal{J}$ implies that no feasible point in which f attains smaller value than $U^k - \varepsilon$ will be cut off. The solution is found after a finite number of iterations since Y is finite. Then U^k is the global minimum as every (OA-NLP($\mathbf{y}^k$)) is solved to the global minimum.

Theorem 2. *Algorithm 2 solves the problem (MINSO) to a global minimizer when functions are convex and continuously differentiable and KKT-conditions hold at minimizer of (OA-NLP($\mathbf{y}^k$)) for every feasible $\mathbf{y}^k$.*

Proof. The proof can be found in [22]. □

Outer approximation has been generalized to handle continuously differentiable quasiconvex constraints [4, 26]. With these constraints the integer relaxed feasible set will be convex. Three difficulties arises with this generalization. The first one is that the linearizations in (OA-MILP$_k$) may cut off feasible points when the corresponding functions are not convex. This is solved by doing linearizations from quasiconvex functions at point $(\mathbf{x}^k, \mathbf{y}^k)$ only if they are active, that is, $g_j(\mathbf{x}^k, \mathbf{y}^k) = 0$. Then the linearization reduces to

$$\nabla g_j(\mathbf{x}^k, \mathbf{y}^k)^T \begin{pmatrix} \mathbf{x} - \mathbf{x}^k \\ \mathbf{y} - \mathbf{y}^k \end{pmatrix} \leq 0. \tag{2}$$

Due to the properties of quasiconvexity linearization (2) will not cut off any points where g_j attains value less than or equal to $g_j(\mathbf{x}^k, \mathbf{y}^k) = 0$. The active constraints are the important ones when proving that the linearizations in (OA-MILP$_k$) will cut off the old solution $(\mathbf{x}^k, \mathbf{y}^k)$. Hence, this property holds when linearizations (2) are used in (OA-MILP$_k$) for quasiconvex functions.

Another difficulty is the feasibility problem (OA-F($\mathbf{y}^k$)) that will now have a nonquasiconvex constraint functions. Instead of the feasibility problem we may solve problem

$$\begin{aligned} \text{minimize} \quad & \theta(\mathbf{y}; \mathbf{y}^k) := \left\| \mathbf{y} - \mathbf{y}^k \right\|^2 \\ \text{s.t} \quad & g_j(\mathbf{x}, \mathbf{y}) \leq 0, \ j \in \mathcal{J} \qquad \text{(FP-NLP}(\mathbf{y}^k)\text{)} \\ & (\mathbf{x}, \mathbf{y}) \in L, \mathbf{x} \in \mathbb{R}^n, \mathbf{y} \in \mathbb{R}^m. \end{aligned}$$

Note that $\mathbf{y}$ is considered continuous variable in this problem. (FP-NLP($\mathbf{y}^k$)) finds $(\overline{\mathbf{x}}, \overline{\mathbf{y}})$ that minimizes distance of $\mathbf{y}^k$ and the projection of the integer

relaxed feasible set $N \cap L$ onto $\mathbb{R}^m$. Then hyperplane

$$\left\{\mathbf{y} \mid \nabla\theta(\overline{\mathbf{y}};\mathbf{y}^k)^T(\mathbf{y}-\overline{\mathbf{y}})=0\right\}$$

separates convex sets $N\cap L$ and $\left\{\mathbf{y} \mid \theta(\mathbf{y};\mathbf{y}^k) \le \theta(\overline{\mathbf{y}};\mathbf{y}^k)\right\}$. More importantly, it separates $N \cap L$ and $\mathbf{y}^k$. Hence, the constraint

$$\nabla\theta(\overline{\mathbf{y}};\mathbf{y}^k)^T(\mathbf{y}-\overline{\mathbf{y}}) = 2(\overline{\mathbf{y}}-\mathbf{y}^k)^T(\mathbf{y}-\overline{\mathbf{y}}) \ge 0 \tag{3}$$

will not cut off any points from $N \cap L$ but will make $\mathbf{y}^k$ infeasible. Linearization (3) will be added to the MILP master problem (OA-MILP$_k$).

In [26] an alternative to the constraint (3) was presented. Let $\hat{\mathcal{J}}$ be the set of indices of active constraints at point $(\overline{\mathbf{x}},\overline{\mathbf{y}})$. Then linearizations

$$\nabla g_j(\overline{\mathbf{x}},\overline{\mathbf{y}})^T \begin{pmatrix} \mathbf{x}-\overline{\mathbf{x}} \\ \mathbf{y}-\overline{\mathbf{y}} \end{pmatrix} \le 0, j \in \hat{\mathcal{J}},$$

will cut off $\mathbf{y}^k$ but points in $N \cap L$ remain feasible. However, a constraint qualification must be satisfied at $(\overline{\mathbf{x}},\overline{\mathbf{y}})$ in order to make the KKT-conditions hold. With this procedure the objective of the problem (FP-NLP($\mathbf{y}^k$)) may also be set to $\theta(\mathbf{y};\mathbf{y}^k) = \left\|\mathbf{y}-\mathbf{y}^k\right\|_1$.

The last problem in this generalization is that NLP problems have quasiconvex constraints. These problems were assumed to be solved to global optimality in [4, 26]. With some additional assumptions this can be done, for example, with the supporting hyperplane algorithm presented later.

In [22] the outer approximation method was generalized to solve the following nonsmooth MINLP problem

$$\begin{aligned} &\text{minimize} \quad f(\mathbf{x},\mathbf{y}) + h(g(\mathbf{x},\mathbf{y})) \qquad \text{(OA-MINSO)} \\ &\text{subject to} \quad (\mathbf{x},\mathbf{y}) \in L, \mathbf{x} \in \mathbb{R}^n, \mathbf{y} \in \mathbb{Z}^m. \end{aligned}$$

Here the functions $f : \mathbb{R}^{n+m} \to \mathbb{R}$ and $g : \mathbb{R}^{n+m} \to \mathbb{R}^J$ are assumed to be convex and continuously differentiable, while $h : \mathbb{R}^J \to \mathbb{R}$ is assumed to be convex but nonsmooth. The function h is also assumed to be monotone in the sense that if $a_i \le b_i$ for all $i = 1, 2, \ldots, n$ then $h(\mathbf{a}) \le h(\mathbf{b})$. This kind of restrictions are met by functions like $\max_i \{a_i\}$ and $\sum_{i=1}^n |\max\{0, a_i\}|$. The problem (OA-MINSO) is a special case of the problem (MINSO). It can be used to solve the continuously differentiable case with the exact penalty function method [22].

When solving problem (OA-MINSO) for a fixed value $\mathbf{y}^k$ there will always be a feasible solution and no feasibility problems are needed to solve. At the solution $\mathbf{z}^k = (\mathbf{x}^k, \mathbf{y}^k)$ the linearization

$$\eta \ge f(\mathbf{z}^k) + \nabla f(\mathbf{z}^k)^T \begin{pmatrix} \mathbf{x}-\mathbf{x}^k \\ \mathbf{y}-\mathbf{y}^k \end{pmatrix} + h\left(g(\mathbf{z}^k) + \nabla g(\mathbf{z}^k)^T \begin{pmatrix} \mathbf{x}-\mathbf{x}^k \\ \mathbf{y}-\mathbf{y}^k \end{pmatrix}\right)$$

is added to the MILP problem. The convergence of the algorithm was proven in [22]. Although the proof utilized subgradients of h, no subgradients are needed to calculate in the algorithm.

OA method for MINSO problem was also studied in [18]. It was proved that even if we can solve NSO problems instead of NLP problems we cannot guarantee the convergence of the OA method by simply replacing gradients by arbitrary subgradients in Algorithm 2. The trouble is that the linearizations done after (OA-NLP($\mathbf{y}^k$)) or (OA-F($\mathbf{y}^k$)) problem may not cut off the previous integer vector $\mathbf{y}^k$, thus, resulting in potential infinite loop. The reason for this is that an arbitrary subgradient does not necessarily satisfy KKT-conditions as an equality. However, if we can find a subgradient that does this, the algorithm will work as intended. This was also studied in detail in [46]. Also, in [47] it was proved that it is sufficient that the functions are continuously differentiable with respect to $\mathbf{x}$. In this case we can use arbitrary subgradients in Algorithm 2. If the integer variables are binary, the infinite loop can also be avoided by using integer cuts presented in [13, 24].

Another special case of MINSO problem that can be solved with OA method can be found in [12]. In that paper a mixed integer second order conic programming (MISOCP) problem is considered. Due to conic constraints the problem is not smooth. With help of duality the authors were able to find subgradients satisfying KKT-conditions of NLP problem, and thus, formulate an OA based algorithm with proven convergence.

Recently, the OA method was successfully generalized to handle convex MINSO problems in [11]. NSO and the feasibility problem were solved with an exact penalization proximal bundle algorithm. This provides subgradients satisfying KKT-conditions along with the KKT-coefficients. Linearizations are done only from the constraints that are active at the solution $(\mathbf{x}^k, \mathbf{y}^k)$. The components of the subgradient that corresponds to integer variables can not be chosen arbitrarily but in a way that the resulting vector is a subgradient.

A straightforward generalization of the algorithm in [11] for problems with f°-quasiconvex constraint functions is not possible. The difficulty arises in NSO problems that have f°-quasiconvex constraint functions while the exact penalization proximal bundle algorithm requires convex functions. Otherwise, since the linearizations are created only from the active constraints the algorithm should be applicable for problems with f°-quasiconvex constraints if the feasibility problems are dealt in similar manner to [26]. Of course, similar restrictions are valid as in the problem classes studied in [26], that is, the KKT-conditions must be satisfied at points where the linearizations are created.

In [11] numerical tests included an academic hybrid robust/chance-constraint problem. While the problem was convex MINSO problem it could be formulated as a quadratic smooth nonconvex MINLP problem. However, it turned out to be easier to solve the nonsmooth formulation especially when the size of the problem was increased. The OA method was also tested

against a variant of extended cutting plane method and extended level bundle method, which are presented here later. In large problem instances the OA method outperformed these two methods which are known to excel in cases where nonlinear function evaluations are time consuming. In the considered problem the nonlinear function could be evaluated fast. However, the extended cutting plane method was the fastest algorithm in almost 35% of the problems.

2.2.1 LP/NLP-based branch and bound

LP/NLP-based branch and bound [40] can be seen as an improvement of the outer approximation method where only one MILP master problem is solved. The MILP master problem of the OA algorithm is solved, but, each time an integer solution $\mathbf{y}^k$ is obtained in the tree of LP problems, an (OA-NLP($\mathbf{y}^k$)) problem is solved. Note that this is different from OA method, where MILP master problems are solved to the minimum before solving any NLP problems. After solving (OA-NLP($\mathbf{y}^k$)) or the feasibility problem (OA-F($\mathbf{y}^k$)), cuts similar to OA method are generated to all remaining LP problems. Furthermore, the problem that yielded $\mathbf{y}^k$ is solved again with the additional constraints.

The algorithm uses similar cuts as the OA method and hence the same difficulties as with the OA method arises when solving MINSO problems. In addition, a similar solution to the difficulties applies and the method presented in [11] generalizes LP/NLP-based branch and bound for MINSO problems.

2.3 Extended cutting plane method

Extended cutting plane (ECP) method was introduced in [48]. It generalizes Kelley's cutting plane method [27] for smooth mixed integer case. The method was further generalized for nonsmooth functions by replacing gradients with arbitrary subgradients in [18]. For the ECP method it is necessary to transform the nonlinear objective function to a constraint as in (OA-MILP$_k$). This is done by introducing an auxiliary variable μ that will be minimized and adding the constraint

$$f(\mathbf{z}) - \mu \leq 0 \tag{4}$$

to the problem (MINSO). Since f is assumed to be convex so is $f-\mu$. As some MILP solvers can also solve mixed integer quadratic programming (MIQP) problem, this transformation is not necessary if the objective function is quadratic. Although, this will result in solving MIQP problems instead of MILP problems.

The ECP method is similar to the OA method in the sense that the method solves MILP subproblems. However, at the MILP solution point $\mathbf{z}^k$ the most violated nonlinear constraint function is linearized instead of solving an NSO problem. For the most violated constraint j_k, the equation $G(\mathbf{z}^k) = g_{j_k}(\mathbf{z}^k)$ holds. Then the linear constraint that will be added to the MILP problem is

$$l^k(\mathbf{z}) := g_{j_k}(\mathbf{z}^k) + \boldsymbol{\xi}_{j_k}^T(\mathbf{z} - \mathbf{z}^k) \leq 0, \tag{5}$$

where $\boldsymbol{\xi}_{j_k} \in \partial g_{j_k}(\mathbf{z}^k)$ is arbitrary. Hence, at iteration k the MILP problem is

$$\begin{aligned} &\text{minimize} && \mu \\ &\text{subject to} && l^i(\mathbf{z}) \leq 0,\ i = 1, 2, \ldots, k-1 \quad \text{(ECP-MILP}_k\text{)} \\ &&& \mathbf{z} \in L \cap Z. \end{aligned}$$

The problem (ECP-MILP$_1$) corresponds to the original problem without the nonlinear constraints. The process of solving an MILP problem and adding new linear constraint is repeated until the solution of the MILP problem satisfies also the nonlinear constraints. The ECP method is described in Algorithm 3.

Algorithm 3: The ECP algorithm

Data Give the tolerance parameter $\varepsilon_g > 0$.
Step 1 Set $k = 1$.
Step 2 Solve the problem (ECP-MILP$_k$). Denote the solution by $(\mathbf{z}^k, \mu^k)$.
Step 3 If $\mathbf{z}^k$ satisfies the nonlinear constraints, that is $G(\mathbf{z}^k) \leq \varepsilon_g$, stop, the point $\mathbf{z}^k$ is a global minimizer of the original MINSO problem.
Step 4 Create a new problem (ECP-MILP$_{k+1}$) by adding in (ECP-MILP$_k$) the constraint $g_{j_k}(\mathbf{z}^k) + \boldsymbol{\xi}_{j_k}^T(\mathbf{z} - \mathbf{z}^k) \leq 0$, where $\boldsymbol{\xi}_{j_k} \in \partial g_{j_k}(\mathbf{z}^k)$ is arbitrary and $g_{j_k}(\mathbf{z}^k) = G(\mathbf{z}^k)$.
Step 5 Set $k = k + 1$ and go to Step 2.

Unlike in the OA method, the cutting plane (5) at iteration k does not cut off all solutions with integer values $\mathbf{y}^k$. However, it cuts off the previous solution $\mathbf{z}^k$. Due to the convexity of the constraint functions, the cutting planes do not cut off any points from the feasible region. Then, since the objective function is linear, if an MILP solution is feasible in the nonlinear constraint functions, it is a global minimizer of (MINSO). If this is not obtained in a finite number of iterations, Algorithm 3 generates a sequence $(\mathbf{z}^k)$ of different points. The compactness of L and the Cauchy-Weierstrass Theorem imply that there exists an accumulation point. It turns out that any accumulation point of $(\mathbf{z}^k)$ is feasible in nonlinear constraint functions, and thus a global minimizer of (MINSO).

Theorem 3. *If $\varepsilon_g = 0$, Algorithm 3 finds a global minimizer of (MINSO) in a finite number of iterations or it generates a sequence with a global minimizer as an accumulation point.*

Proof. The proof can be found in [18]. □

The advantage of ECP over NSO B&B and OA method is that no NSO problems are needed to solve. This usually results in a fewer number of nonlinear function evaluations, which is significant if the function evaluations are time consuming like in the chromatographic separation problem [17]. On the other hand, if the bottleneck of the problem is hard MILP problems, the other methods are supposedly faster.

There are two modifications that may speed up the solving process of the ECP method. Instead of one violated constraint function we may linearize more, or even all, violated constraints at the current MILP solution $\mathbf{z}^k$. While more linearizations should make algorithm solve less MILP problems, the MILP problems may become harder due to more constraints. Furthermore, we do not need to solve the MILP problems to optimality every time. Only the last MILP problem needs to be solved to optimality. We can, for example, stop MILP solving at the first integer feasible solution and create cutting planes there. The last MILP problem must be solved to optimality. To ensure this, we can gradually increase the number of integer feasible solutions a MILP solver needs to find before stopping. The strategy may prove useful if the MILP problems are hard to solve. This ploy was presented in flow chart in [49] and applied in [5] (strategy 1) for αECP algorithm that is presented in the next subsection.

2.3.1 αECP method

ECP method was generalized to deal with pseudoconvex constraint functions and objective function in [38, 49, 50]. In [19] this was further generalized to deal with nonsmooth f°-pseudoconvex functions. The difficulty in applying the Algorithm 3 to the problems with f°-pseudoconvex constraint functions is that cutting plane (5) may then cut off feasible points. To avoid this problem an α coefficient is added to the linearization (5) resulting in α-cutting plane

$$g_{j_k}(\mathbf{z}^k) + \alpha_{j_k}^k \boldsymbol{\xi}_{j_k}^T (\mathbf{z} - \mathbf{z}^k) \leq 0, \tag{6}$$

where $\boldsymbol{\xi}_{j_k} \in \partial g_{j_k}(\mathbf{z}^k)$ is arbitrary. The coefficient $\alpha_{j_k}^k$ is first set to 1. The α-cutting plane will cut off the previous solution point $\mathbf{z}^k$ from the MILP problem. As the constraint function is not convex, it may cut off some feasible points as well. However, since the constraint functions are f°-pseudoconvex and thus LLC, there exists $\hat{\alpha}$ such that (6) does not cut off any feasible point if $\alpha_{j_k}^k > \hat{\alpha}$ holds true [19]. The constant is not generally known beforehand (besides the convex case $\hat{\alpha} = 1$). In practice, we gradually increase the α co-

efficients until certain limit. If the MILP problem (ECP-MILP$_k$) is infeasible the α coefficients are increased by setting

$$\alpha_{j_k}^{k+1} = \beta \cdot \alpha_{j_k}^{k}, \tag{7}$$

where $\beta > 1$. The α coefficients are also increased if a feasible solution is found. In that case we use another parameter $\gamma > 1$ instead of β in (7). The coefficient $\alpha_{j_k}^k$ will be updated until

$$\alpha_{j_k}^{k} \geq \frac{g_{j_k}(\mathbf{z}^k)}{\varepsilon_z \left\| \boldsymbol{\xi}_{j_k} \right\|}, \tag{8}$$

where $\varepsilon_z > 0$ is a user given parameter. To interpret this inequality, consider the case that relation (8) holds as an equality. By inserting the obtained $\alpha_{j_k}^k$ to the α-cutting plane (6) we get

$$\frac{\boldsymbol{\xi}_{j_k}^T}{\left\| \boldsymbol{\xi}_{j_k} \right\|}(\mathbf{z} - \mathbf{z}^k) \leq -\varepsilon_z.$$

Thus, the points with distance less than ε_z from the hyperplane

$$H = \left\{ \mathbf{z} \in \mathbb{R}^{m+n} \mid \frac{\boldsymbol{\xi}_{j_k}^T}{\left\| \boldsymbol{\xi}_{j_k} \right\|}(\mathbf{z} - \mathbf{z}^k) = 0 \right\}$$

are cut off. The criterion (8) can be interpreted as follows. The coefficient $\alpha_{j_k}^k$ is sufficiently large if none of the feasible points with greater distance than ε_z from the hyperplane H are cut off. The smaller the parameter ε_z the more we need to update α coefficients, but the smaller the feasible region is that may be cut off.

If an f°-pseudoconvex objective function is transformed into constraint (4), the constraint may not be f°-pseudoconvex. In fact the function $f(\mathbf{z}) - \mu$ is quasiconvex only if f is convex [9]. To avoid this non-f°-pseudoconvex constraint, the f°-pseudoconvex objective function is taken into account by adding the *reduction constraint* $f(\mathbf{z}) - f_r \leq 0$ and $\mu \leq f_r$ in the original MINSO problem and minimizing μ. The constant f_r is the current upper bound, which is updated whenever we find $\mathbf{z}$ such that $G(\mathbf{z}) \leq \varepsilon_g$ and $f(\mathbf{z}) < f_r$. To make the resulting MINSO problem meaningful we need to bound μ from below with help of f. This is done with linear constraints presented next.

We will add constraints

$$f_r + \boldsymbol{\xi}^T(\mathbf{z} - \mathbf{z}^k) \leq \mu, \tag{9}$$

where $\boldsymbol{\xi} \in \partial f(\mathbf{z}^k)$, in the MINSO problem at any point $\mathbf{z}^k$ with $f(\mathbf{z}^k) = f_r$. Linearization is also added at $\mathbf{z}^k$ that is ε_g-feasible in the MINSO problem. Then, the constraint $f(\mathbf{z}) \leq f_r$ guarantees that $f(\mathbf{z}^k) \leq f_r + \varepsilon_g$. Note that

since constraints (9) are linear, they transfer to the MILP problems being solved in order to solve the MINSO problem.

Since the upper bound f_r changes whenever a new upper bound is found we need to update constraints (9). One option would be to omit the linearizations with the old f_r. This makes the convergence proof tricky requiring the additional assumption that the solution sequence has only one accumulation point [19]. Another option is to simply replace the old f_r values in (9) with the new one.

Linearizations (9) steer MILP solution away from $\mathbf{z}^k$, and thus prevent infinite loops. They are also the only constraints that bounds the minimized variable μ from below besides the box constraints. If a feasible solution $\tilde{\mathbf{z}}$ with $f(\tilde{\mathbf{z}}) < f_r$ exists, f°-pseudoconvexity implies that $\boldsymbol{\xi}^T(\tilde{\mathbf{z}}-\mathbf{z}^k) < 0$. Thus, when minimizing μ we get $\mu^k < f_r$ whenever f_r is not the global minimum value. Eventually, the algorithm stops when $f_r - \mu \leq \varepsilon_f$ is satisfied for given $\varepsilon_f > 0$. If $\varepsilon_f = 0$ it can be shown that the algorithm converges to an ε_g-feasible global minimum [19].

2.4 Extended supporting hyperplane method

Supporting hyperplane method was introduced in [45] to solve NLP and MINLP problems. The method was recently implemented and studied in more detail in [30] where the name extended supporting hyperplane (ESH) was introduced. In [20] it was generalized for MINSO problems with f°-pseudoconvex constraint functions. In [51] it was further generalized for problems with f°-pseudoconvex objective function.

The extended supporting hyperplane (ESH) method is quite similar to the ECP method. ESH method proceeds similarly to ECP method, but instead of cutting planes we create supporting hyperplanes to the nonlinear feasible set N. To find these hyperplanes we need an inner point $\mathbf{z}_{\text{int}}$ that strictly satisfies all nonlinear constraints, that is, $G(\mathbf{z}_{\text{int}}) < 0$. Furthermore, we have to do a line search between the current MILP solution point $\mathbf{z}^k_{\text{MILP}}$ and $\mathbf{z}_{\text{int}}$ to find the point $\mathbf{z}^k$ such that $G(\mathbf{z}^k) = 0$. The polyhedral approximation of N is enhanced by adding to the MILP problem the linearization

$$\boldsymbol{\xi}_{j_k}^T(\mathbf{z} - \mathbf{z}^k) \leq 0, \tag{10}$$

where $\boldsymbol{\xi}_{j_k} \in \partial g_{j_k}(\mathbf{z}^k)$ is arbitrary and $g_{j_k}(\mathbf{z}^k) = G(\mathbf{z}^k)$. Notice that this is actually a cutting plane since $g_{j_k}(\mathbf{z}^k) = 0$. It also turns out that we do not need α coefficients for f°-pseudoconvex constraint functions. The constraint functions can be f°-quasiconvex with an additional restriction that $\mathbf{0} \notin \partial g_j(\mathbf{z}^k)$ if linearization is created from g_j at $\mathbf{z}^k$ [51]. This condition is met if the Cottle constraint qualification is satisfied at $\mathbf{z}^k$.

The convergence to the global minimum can be proven similarly to the ECP method. The supporting hyperplane (10) will cut off the previous solution point $\mathbf{z}^k_{\mathrm{MILP}}$. Furthermore, it does not cut off any feasible point. The compactness of L and the Cauchy-Weierstrass Theorem will guarantee that the solution sequence $(\mathbf{z}^k_{\mathrm{MILP}})$ has an accumulation point and it can be proven to be feasible. The linearity of the objective function then implies that the accumulation point is also optimal. For more details see [20].

Above we required that $G(\mathbf{z}_{\mathrm{int}}) < 0$. Essentially this means that (MINSO) must satisfy the Slater constraint qualification. In the case some constraint functions are active at the inner point, it is possible that algorithm gets stuck to an infinite loop [20]. In practice, where we can assure feasibility only up to a tolerance $\varepsilon_g > 0$, we can relax this condition. If in the line search we find a point $\mathbf{z}^k$ such that $g_{j_k}(\mathbf{z}^k) = \frac{\varepsilon_g}{2}$ we can still guarantee to find an ε_g-feasible point. In this case it is sufficient that $G(\mathbf{z}_{\mathrm{int}}) \leq 0$. This kind of point exists if (MINSO) has a feasible solution. We will use this ploy in the subsequent consideration.

We can deal with the f°-pseudoconvex objective function similarly as we did in the αECP method. However, the inner point should satisfy $f(\mathbf{z}_{\mathrm{int}}) - f_r \leq 0$, for any upper bound f_r. Eventually, f_r will be close to the optimal value, and thus, $f(\mathbf{z}_{\mathrm{int}})$ should be less than the global minimum value. This can be achieved if we first solve the original problem with integer variables relaxed to continuous variables and let $\mathbf{z}_{\mathrm{int}}$ be the solution of this relaxed problem. Essentially this means we are solving the root node of NSO branch and bound method. Another way is to have separate inner points for the objective function and for the other constraints. The inner point for the objective function, denoted by $\mathbf{z}^f_{\mathrm{int}}$, can be updated, whenever new upper bound f_r is found.

In [51] MINLP problem with f°-pseudoconvex objective function is solved without using the reduction constraint $f(\mathbf{z}) - f_r \leq 0$. The procedure relies on line search between obtained feasible point of the MINSO problem and point $\mathbf{z}^f_{\mathrm{int}}$ such that $f(\mathbf{z}^f_{\mathrm{int}}) \leq f_r$. The point $\mathbf{z}^f_{\mathrm{int}}$ can always be set to be the point where the upper bound f_r was obtained. Furthermore, if there are several points $\mathbf{z}^1_f, \mathbf{z}^2_f, \ldots, \mathbf{z}^p_f$ satisfying $f(\mathbf{z}^i_f) \leq f_r$ we can set

$$\mathbf{z}^f_{\mathrm{int}} = \sum_{i=1}^{p} \frac{1}{p} \mathbf{z}^i_f. \tag{11}$$

The line search will find a point where the constraint (9) can be added. The line search was already studied for αECP in [38], where it ended at a point from the set $\{\mathbf{z} \mid f(\mathbf{z}) = f_r\}$. It is sufficient to find a point from the region

$$\{\mathbf{z} \mid f(\mathbf{z}) = f_r + \varepsilon_g\} \tag{12}$$

which allowed us to omit the reduction constraint, that in [38] prevented infinite loops.

Assuming that we update the upper bound f_r in constraints (9), the MILP problem solved at iteration k is

$$\begin{aligned} &\text{minimize} && \mu \\ &\text{subject to} && f_r + \boldsymbol{\xi}^T(\mathbf{z} - \mathbf{z}^k) \leq \mu,\ k \in I_f \quad \text{(ESH-MILP}_k\text{)} \\ & && \boldsymbol{\xi}_{j_k}^T(\mathbf{z} - \mathbf{z}^k) \leq 0,\ k \in I_g \\ & && \mathbf{z} \in L \cap Z. \end{aligned}$$

Here I_g represents iterations where MILP solution is not feasible in nonlinear constraints and hence a supporting hyperplane is added. Correspondingly, I_f denotes iterations where a feasible point is found and a line search is done to find a point from region (12).

Algorithm 4: The ESH algorithm

Data Give the tolerance parameters ε_g, $\varepsilon_f > 0$ and an inner point $\mathbf{z}_{\text{int}} \in N \cap L$.
Step 1 Set $f_r = \infty$ and $k = r = 1$.
Step 2 Solve the problem (ESH-MILP$_k$). Denote the solution by $(\mathbf{z}^k_{\text{MILP}}, \mu^k)$.
Step 3 If $\mu^k \geq f_r - \varepsilon_f$ then stop: f_r is the optimal value.
Step 4 If $G(\mathbf{z}^k_{\text{MILP}}) > \varepsilon_g$, do a line search between $\mathbf{z}_{\text{int}}$ and $\mathbf{z}^k_{\text{MILP}}$ to find $\mathbf{z}^k$ such that $G(\mathbf{z}^k) = \frac{\varepsilon_g}{2}$. Add to (ESH-MILP$_{k+1}$) the linear constraint $\boldsymbol{\xi}_{j_k}^T(\mathbf{z} - \mathbf{z}^k) \leq 0$, where $\boldsymbol{\xi}_{j_k} \in \partial g_j(\mathbf{z}^k)$ and $g_j(\mathbf{z}^k) = G(\mathbf{z}^k)$.
Step 5 If $G(\mathbf{z}^k_{\text{MILP}}) \leq \varepsilon_g$ then
  Step 5.1 If $f(\mathbf{z}^k_{\text{MILP}}) < f_r$, update $r = r + 1$. Set $\mathbf{z}^k = \mathbf{z}^k_{\text{MILP}}$ and $f_r = f(\mathbf{z}^k)$. Update the constraints of type (9) by using the new value f_r.
  Step 5.2 If $f_r \leq f(\mathbf{z}^k_{\text{MILP}}) \leq f_r + \varepsilon_g$, set $\mathbf{z}^k = \mathbf{z}^k_{\text{MILP}}$.
  Step 5.3 If $f(\mathbf{z}^k_{\text{MILP}}) > f_r + \varepsilon_g$, calculate $\mathbf{z}^f_{\text{int}}$ from (11). Find $\mathbf{z}^k$ such that $f(\mathbf{z}^k) = f_r + \varepsilon_g$ with a line search between $\mathbf{z}^f_{\text{int}}$ and $\mathbf{z}^k_{\text{MILP}}$.
  Step 5.4 Add to (ESH-MILP$_{k+1}$) the linear constraint $f_r + \boldsymbol{\xi}^T(\mathbf{z} - \mathbf{z}^k) \leq \mu$, where $\boldsymbol{\xi} \in \partial f(\mathbf{z}^k)$.
Step 6 Set $k = k + 1$ and go to Step 2.

Algorithm 4 will find an ε_g-feasible point after a finite number of iterations if $\varepsilon_g > 0$. Then we can add new constraint (9) and sometimes find a new upper bound. If $\varepsilon_f = 0$ the algorithm will converge to a global minimizer.

Theorem 4. *Algorithm 4 converges to an ε_g-feasible global minimizer of (MINSO) or finds one after a finite number of iterations.*

Proof. Proof can be found in [51]. □

While ESH may require more function evaluations than αECP due to the line searches, sometimes it creates tighter approximation of the feasible set. This can result in finding a minimizer in less number of iterations. Furthermore, in theory ESH handles f°-pseudoconvex functions better without

needing the α coefficients. Due to the α coefficients αECP may cut off small parts of the feasible set and, updating the coefficients, the number of MILP problems to be solved increases. In [20] there was a remarkable difference in solving times between ESH and αECP in favour of ESH in an instance of a facility layout problem with pseudoconvex constraints. This was a result of αECP generating harder MILP problems with α-cutting planes.

In [51] there was an example that showed the benefits of being able to solve MINSO problems with f°-pseudoconvex objective function rigorously. The objective function of the problem is maximum of four pseudoconvex functions while the constraints were linear. This problem can be solved with Algorithm 4. Alternatively, we can also get rid of nonsmoothness by introducing four nonlinear constraints $f_i(\mathbf{z}) - \mu \leq 0$, but then we need a nonconvex MINLP method to solve the problem to the global minimum. The BONMIN solver in GAMS [23] solved the problem fastest (30s) followed by ESH (70s) while solvers like SCIP and LINDOGLOBAL could not find the minimum in 1000 seconds.

2.5 Extended level bundle method

In [14] the extended level bundle method (ELBM) was introduced. It is based on the NSO algorithm level bundle method [32, 44]. Similarly to ECP method no NSO problems are needed to solve while solving a sequence of MILP subproblems will lead to the global minimum of the problem (MINSO).

As in the ECP method, cutting planes are created from nonlinear functions in MILP solution points. In ELBM linearizations are generated from all nonlinear functions contrary to ECP, where some linearizations are generated from some violated constraints. The objective function of the MILP problem is the stability function $\varphi(\cdot\,; \hat{\mathbf{z}}^k)$, where $\hat{\mathbf{z}}^k$ is the stability center at iteration k. The stability center can be e.g. a fixed point, the last iterate $\mathbf{z}^k$ or the incumbent iterate defined later in the algorithm pattern. In [14] stability functions

$$\varphi(\mathbf{z}; \hat{\mathbf{z}}^k) = \|\mathbf{z} - \hat{\mathbf{z}}^k\|_1 \text{ and } \varphi(\mathbf{z}; \hat{\mathbf{z}}^k) = \|\mathbf{z} - \hat{\mathbf{z}}^k\|_\infty$$

were considered in more detail. Both of these functions can be reformulated with auxiliary variables so that the objective becomes linear and the resulting problem is an MILP problem. An interesting case to notice is the stability function

$$\varphi(\mathbf{z}; \hat{\mathbf{z}}^k) = \hat{f}(\mathbf{z}) + \frac{1}{2t_k}\|\mathbf{z} - \hat{\mathbf{z}}^k\|,$$

where $t_k > 0$ is a parameter and $\hat{f}$ is the linear approximation of f so far. In this case the ELBM algorithm is straightforward generalization of the level bundle method presented in [16] to the mixed integer problems. However, this stability function results in MIQP problems instead of MILP problems.

The purpose of the stability function is to keep the solutions near the point where approximation of the functions are good. This ability is lacking for example in ECP method which may cause the consecutive solutions to be far away. Stability function also allows better utilization of good initial solution $\mathbf{z}^0$.

At iteration k we solve subproblem

$$\begin{aligned} \text{minimize} \quad & \varphi(\mathbf{z}; \hat{\mathbf{z}}^k) \\ \text{subject to} \quad & f(\mathbf{z}^i) + \boldsymbol{\xi}^T(\mathbf{z}^i)(\mathbf{z} - \mathbf{z}^i) \leq f^k_{\text{lev}}, i \in B^k \\ & g_j(\mathbf{z}^i) + \boldsymbol{\xi}_j(\mathbf{z}^i)^T(\mathbf{z} - \mathbf{z}^i) \leq 0, j \in \mathcal{J}, i \in B^k \quad \text{(ELBM-MILP}_k\text{)} \\ & \mathbf{z} \in L \cap Z, \end{aligned}$$

where $\boldsymbol{\xi}(\mathbf{z}^i) \in \partial f(\mathbf{z}^i)$ and $\boldsymbol{\xi}_j(\mathbf{z}^i) \in \partial g_j(\mathbf{z}^i)$ are arbitrary. The set B^k, called bundle, defines at which solution points the functions are linearized. The constant f^k_{lev} is defined by

$$f^k_{\text{lev}} = f^k_{\text{low}} + \gamma h^k, \tag{13}$$

where f^k_{low} is the current lower bound, $\gamma \in (0, 1)$ is a chosen parameter and h^k is the residual of optimality defined by

$$h^k = \min_{i=1,2,\ldots,k} \max_{j \in \mathcal{J}} \left\{ f(\mathbf{z}^i) - f^k_{\text{low}}, g_j(\mathbf{z}^i) \right\}. \tag{14}$$

By solving (ELBM-MILP$_k$) we get another point where linearizations are created and possibly a new value for h^k. If the MILP problem is infeasible there are no feasible points where f attains value f^k_{lev}. Thus, we obtain a new lower bound $f^k_{\text{low}} = f^k_{\text{lev}}$, which may make the MILP problem feasible. This way ELBM generates increasing sequence of lower bounds converging to the global minimum. The ELBM algorithm is presented in Algorithm 5.

The rule for bundle update is user specified. The smaller the bundle the easier the MILP problems are. However, then the linear approximations become less accurate which may result in more iterations. Bundle can be updated only if reasonable improvement of h^k has occurred and this prevents infinite loops due to too frequent bundle updates.

Theorem 5. *If $\varepsilon_g = 0$, Algorithm 5 converges to a global minimizer of (MINSO) or finds it after a finite number of iterations.*

Proof. The convergence is proven in [14]. □

The convergence can be seen as follows. For a given f^k_{lev} the problem

$$\begin{aligned} \text{minimize} \quad & \varphi(\mathbf{z}; \hat{\mathbf{z}}^k) \\ \text{subject to} \quad & f(\mathbf{z}) \leq f^k_{\text{lev}} \\ & g_j(\mathbf{z}) \leq 0, j \in \mathcal{J}, \quad \text{(ELBM-MINSO)} \\ & \mathbf{z} \in L \cap Z. \end{aligned}$$

Algorithm 5: The ELBM algorithm

Data Give the tolerance parameter $\varepsilon_g > 0$, $\gamma \in (0,1)$ and initial point $\hat{\mathbf{z}}^0 = \mathbf{z}^0 \in L \cap Z$. Give f^0_{low} or calculate it from an MILP problem based on linear approximations on $\mathbf{z}^0$.

Step 1 Set $k = 0$, $l = 0$, $k_l = 0$ and $B^0 = \{0\}$.

Step 2 Define h^k as in (14) and denote $\mathbf{z}^k_{\text{best}}$ the point at which it is attained. If $h^k \leq \varepsilon_g$ stop: the optimum is $\mathbf{z}^k_{\text{best}}$.

Step 3 If $h^k \leq (1-\gamma)h^{k_l}$, choose bundle B^k with restriction that $\{k\} \subset B^k$. A new stability center $\hat{\mathbf{z}}^k = \mathbf{z}^k_{\text{best}}$ is set. Set $k_{l+1} = k$ and $l = l + 1$.

Step 4 Calculate f^k_{lev} by equation (13) and solve the MILP problem (ELBM-MILP$_k$).

Step 5 If the MILP problem is infeasible update $f^{k+1}_{\text{low}} = f^k_{\text{lev}}$. Set $B^{k+1} = B^k$, $k_{l+1} = k$ and $l = l + 1$.

Step 6 If the MILP problem has solution $\mathbf{z}^{k+1}$, update the bundle $B^{k+1} = B^k \cup \{k+1\}$ and set $f^{k+1}_{\text{low}} = f^k_{\text{low}}$.

Step 7 Set $k = k + 1$ and go to Step 2.

is solved with a sequence of MILP problems (ELBM-MILP$_k$) as in the ECP method. Suppose that index l remains the same, that is, no new f^k_{low} is found and h^k is greater than $(1-\gamma)h^{k_l}$. Similarly as in the convergence proof of the ECP method, ELBM will generate a sequence that will converge to a feasible point of (ELBM-MINSO). At the feasible point $\overline{\mathbf{z}}$ we have, according to the equation (13),

$$f(\overline{\mathbf{z}}) - f^k_{\text{low}} \leq f^k_{\text{lev}} - f^k_{\text{low}} = \gamma h^k < h^k.$$

Thus, after a finite number of iterations h^k and f^k_{lev} will decrease. Also h^k would converge to 0 unless (ELBM-MINSO) becomes infeasible. In both cases index l will be increased after finite number of iterations.

The index l is increased if one of the conditions in Step 3 and 5 holds. In Step 5 the lower bound will be increased by amount γh^k. If h^k does not converge to 0, there can be only finitely many increases of the lower bound, otherwise, $f^k_{\text{low}} \to \infty$. In turn in Step 3 the condition $h^k \leq (1-\gamma)h^{k_l}$ holds. If this holds true infinitely many times $h^k \to 0$ since $(1-\gamma) < 1$. Thus, $h^k \to 0$ if the algorithm does not stop after a finite number of iterations.

The stability function does not play any role in the convergence proof and consequently MILP problems do not need to be solved to the optimality, feasibility being enough. ELBM method carries the same strength as ECP, namely the lack of NSO problems which usually lead to fewer number of function evaluations. Another strength that also ECP and OA share is that the functions are needed to evaluate only at point where integer variables attain integer values. In numerical comparison [14] ELBM was found out to be more effective on average than a variant of ECP method in terms of CPU and number of function evaluations when solving example problems. The test set included 39 problems from the MINLP Library [34], several instances of an academic chance constrained problem and several instances

of stochastic programming problem. The latter two problem types have hard to evaluate functions making cutting plane type methods preferable than the ones needing to solve NSO problems.

3 Illustrative example

In this section we illustrate the presented algorithms by solving an easy example problem. The problem reads

$$\begin{aligned} \text{minimize} \quad & f(x,y) := |x-4| + |y-4| \\ \text{subject to} \quad & g(x,y) := \max\left\{(y-2)^2 + x^2 - 9, x + 2y - 9\right\} \leq 0 \\ & 0 \leq x \leq 5, y \in \{0,1,2,3,4,5\}\,. \end{aligned}$$

The objective function and the constraint function are convex but nonsmooth. Stopping criteria $\varepsilon = 0.1$ has been used for all the algorithms.

3.1 NSO branch and bound method

The NSO branch and bound begins by solving the integer relaxed problem NSO(0,5) and finds $(x_1, y_1) = (1 + \frac{4}{\sqrt{5}}, 4 - \frac{2}{\sqrt{5}})$. It is not integer feasible, and thus, two problems NSO$(0,3)$ and NSO$(4,5)$ are created. Solving NSO$(0,3)$ results in the integer feasible solution $(x_1, y_1) = (2\sqrt{2}, 3)$ and no new problems need to be created. Also the integer feasible solution gives an upper bound $f(2\sqrt{2}, 3) \approx 2.17$. The solution to the problem NSO$(4,5)$ is $(x_2, y_2) = (1, 4)$ giving an upper bound $f(1,4) \approx 3$. There are no problems left to solve and the best upper bound 2.17 is the global minimum found at $(2\sqrt{2}, 3)$. The solution process of NSO B&B as well as the example problem are illustrated in Figure 1.

3.2 OA method

For the OA algorithm suppose that $y^1 = 4$. The solution of OA-NSO(4) is $x^1 = 1$ giving an upper bound $U^1 = 3$. At this point g is differentiable and its linearization is $x + 2y - 9$. The objective function is not differentiable at this point and its subdifferential is $\partial f(1,4) = \{(-1, \lambda) \mid \lambda \in [-1,1]\}$. The component corresponding to x is unique, and thus, we may choose any subgradient. Let us choose $\boldsymbol{\xi} = (-1, 0)$ resulting in linearization $2 - x$. The problem OA-MILP$_1$ then reads

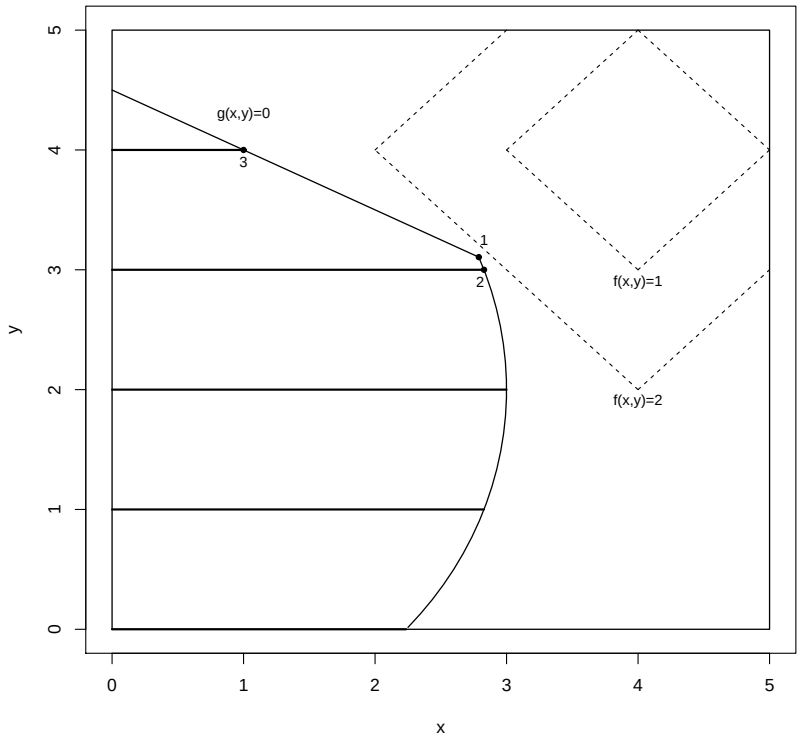

Fig. 1 The solution points found by NSO branch and bound algorithm. Thick lines correspond to feasible region of the example problem.

$$\begin{aligned} \text{minimize} \quad & \eta \\ \text{subject to} \quad & \eta \leq 3 - 0.1 \\ & 2 - x \leq \eta \\ & x + 2y - 9 \leq 0 \\ & 0 \leq x \leq 5, y \in \{0, 1, 2, 3, 4, 5\}. \end{aligned}$$

The solution to this problem is not unique with possibilities being $(5, 0)$, $(5, 1)$ and $(5, 2)$ giving η value -3. Suppose the MILP algorithm gives us $(5, 2)$. The solution point of OA-NSO(2) is $(3, 2)$ giving the same upper bound 3 as the previous integer solution. At this point both of the nonlinear functions are differentiable and OA-MILP$_2$ is

$$
\begin{aligned}
\text{minimize} \quad & \eta \\
\text{subject to} \quad & \eta \leq 3 - 0.1 \\
& 2 - x \leq \eta \\
& 8 - x - y \leq \eta \\
& x + 2y - 9 \leq 0 \\
& 6x - 18 \leq 0 \\
& 0 \leq x \leq 5, y \in \{0, 1, 2, 3, 4, 5\}\,.
\end{aligned}
$$

The unique solution to this problem is $(3, 3)$ giving $\eta = 2$. Solving OA-NSO(2) gives $(2\sqrt{2}, 3)$ with a new upper bound $U^3 = 2.17$. The next MILP problem will be infeasible stopping the algorithm. The solution process is depicted in Figure 2.

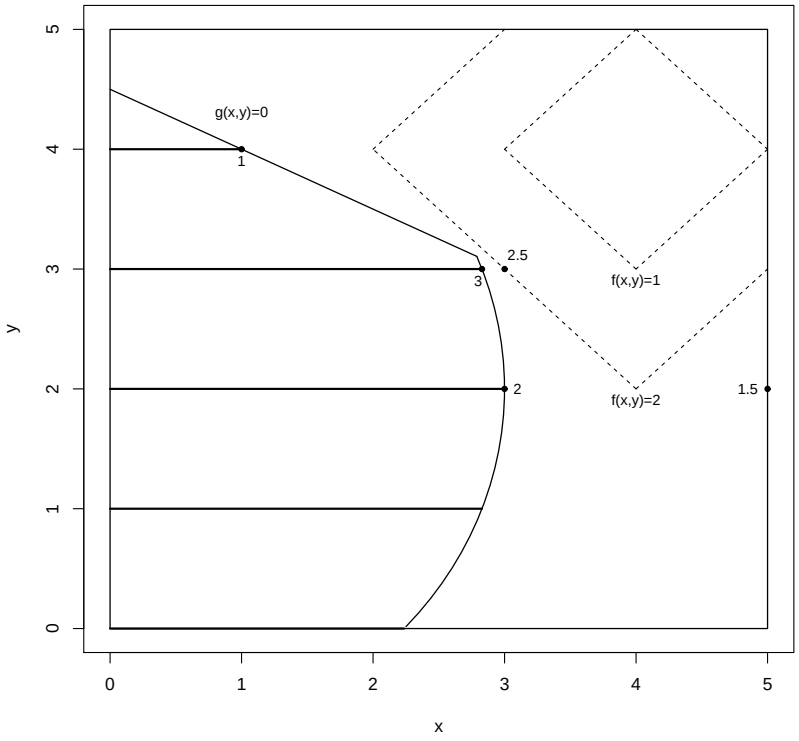

Fig. 2 The solution points found by outer approximation method. Points "1", "2" and "3" corresponds to solution points of the NSO problems while "1.5" and "2.5" are solution points of the MILP problems.

3.3 ECP method

The ECP method begins with solving MILP problem without any linearizations from the nonlinear functions. This, essentially feasibility problem, reads

$$\begin{aligned} &\text{minimize} && \mu \\ &\text{subject to} && 0 \leq x \leq 5, y \in \{0,1,2,3,4,5\}, -10 \leq \mu \leq 10, \end{aligned}$$

where bounds to the μ guarantee a finite solution. Suppose the obtained solution is $(x, y, \mu) = (5, 5, -10)$. Both nonlinear constraints $f - \mu$ and g are violated at this point. To speed up the solving process we make linearizations from all violated constraints. The problem ECP-MILP$_2$ is

$$\begin{aligned} &\text{minimize} && \mu \\ &\text{subject to} && x + y - 8 \leq \mu \\ &&& 10x + 6y - 55 \leq 0 \\ &&& 0 \leq x \leq 5, y \in \{0,1,2,3,4,5\}, -10 \leq \mu \leq 10 \end{aligned}$$

and its solution is $(0, 0, -8)$. Only $f - \mu$ is violated and a new linearization $-x - y + 8 \leq \mu$ is added to the MILP problem. The solution to this MILP problem is $(2.5, 5, 0.5)$. After adding the new linearizations from this point the problem ECP-MILP$_4$ is

$$\begin{aligned} &\text{minimize} && \mu \\ &\text{subject to} && x + y - 8 \leq \mu \\ &&& -x - y + 8 \leq \mu \\ &&& -x + y \leq \mu \\ &&& 5x + 6y - 36.25 \leq 0 \\ &&& 10x + 6y - 55 \leq 0 \\ &&& 0 \leq x \leq 5, y \in \{0,1,2,3,4,5\}, -10 \leq \mu \leq 10. \end{aligned}$$

The point $(3.65, 3, 1.35)$ will solve this problem and a new linearization $7.3x + 2y \leq 27.3$ is added to the problem. The algorithm continues similarly and stops at $(2.83, 3, 2.17)$ after 7th iteration. The solving process is depicted in Figure 3.

3.4 ESH method

For the ESH method we will use readily found inner point $(0, 0, 10)$. The first solution point $(x, y, \mu) = (5, 5, -10)$ coincides with the solution point of the ECP method. Line search between these points finds $(1, 1, 6)$, where $f - \mu$ is

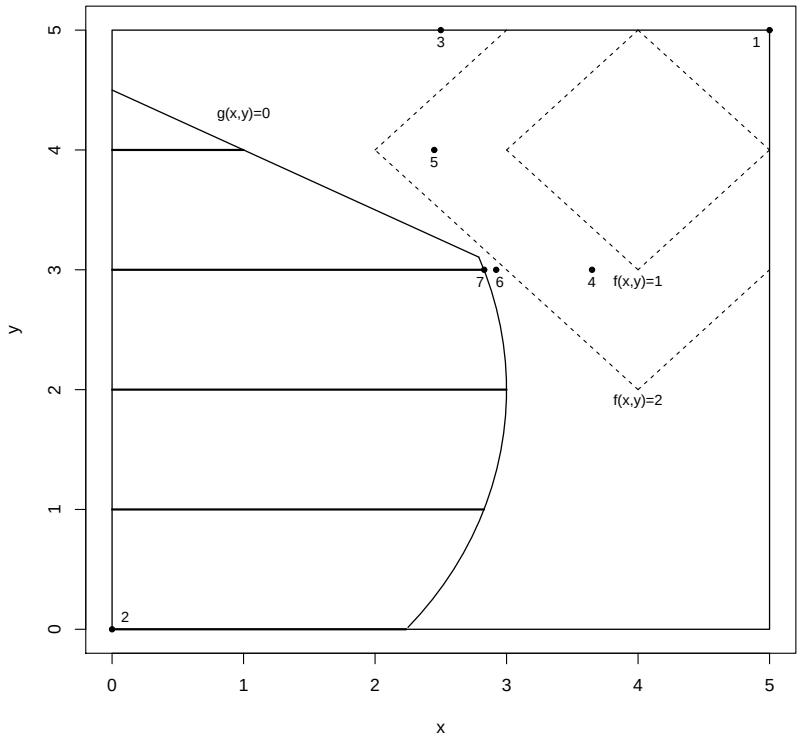

Fig. 3 The solution points found by extended cutting plane method.

active. A supporting hyperplane plane will be added to the MILP problem resulting in problem

$$\begin{aligned} \text{minimize} \quad & \mu \\ \text{subject to} \quad & -x - y + 8 \leq \mu \\ & 0 \leq x \leq 5, y \in \{0, 1, 2, 3, 4, 5\}\,. \end{aligned}$$

The solution point of this problem is $(5, 5, -2)$. The line search will find point $(2.87, 2.87, 3.11)$ where g is active. The constraint $5.74x + 1.74y - 21.5 \leq 0$ is added to the MILP problem. The next solution is $(2.22, 5, 0.77)$ and the line search will find point $(1.64, 3.68, 3.21)$. The linearization of g at this point is $x + 2y - 9$. The problem ESH-MILP$_4$ is

$$\begin{aligned} \text{minimize} \quad & \mu \\ \text{subject to} \quad & -x - y + 8 \leq \mu \\ & 5.74x + 1.74y - 21.5 \leq 0 \\ & x + 2y - 9 \leq 0 \\ & 0 \leq x \leq 5, y \in \{0, 1, 2, 3, 4, 5\}. \end{aligned}$$

The solution point is $(2.83, 3, 2.17)$. At this point the stopping criterion is met and the algorithm stops. The solving process is depicted in Figure 4.

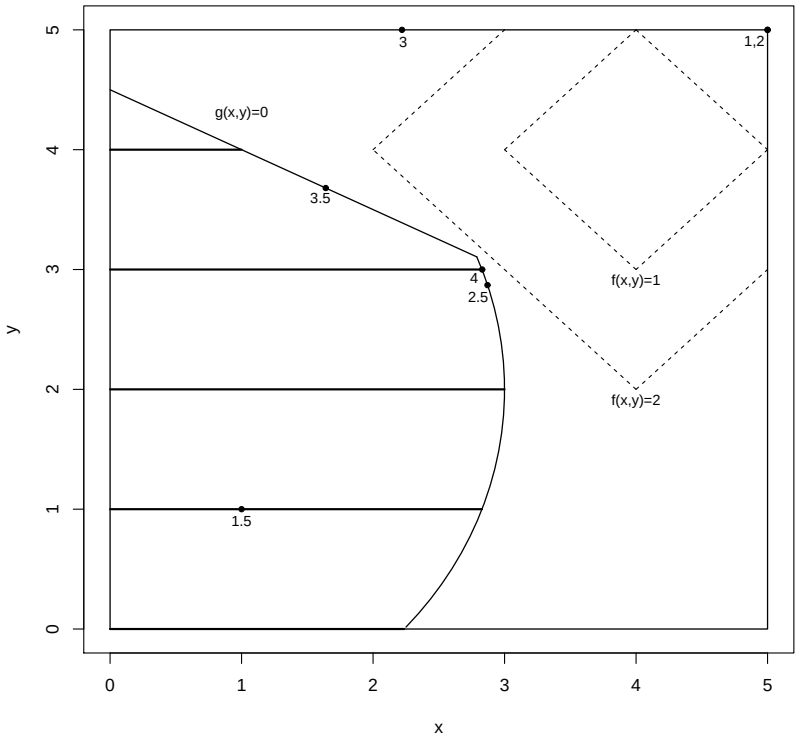

Fig. 4 The solution points found by extended supporting hyperplane method. Points "1", "2", "3", and "4" corresponds to MILP solution points. Points "1.5", "2.5" and "3.5" are found by the line search.

3.5 ELBM

For ELBM let $x^0 = y^0 = 5$, $\gamma = 0.2$ and stability function $\varphi = \|\cdot\|_1$. To get the lower bound we first linearize nonlinear functions at $\mathbf{z}^0 = (x^0, y^0) = (5, 5)$ and solve problem

$$\begin{aligned} \text{minimize} \quad & \mu \\ \text{subject to} \quad & x + y - 8 \le \mu \\ & 10x + 6y - 55 \le 0 \\ & 0 \le x \le 5, y \in \{0, 1, 2, 3, 4, 5\}. \end{aligned}$$

The solution is $(0, 0, -8)$ giving lower bound $f^0_{\text{low}} = -8$. Currently $B^0 = \{0\}$ and thus the stability center $\hat{\mathbf{z}}^0 = \mathbf{z}^0$,

$$\begin{aligned} h^0 &= \max\left\{|5-4| + |5-4| - (-8), (5-2)^2 + 5^2 - 9\right\} = 25, \\ f^0_{\text{lev}} &= f^0_{\text{low}} + \gamma h^0 = -8 + 0.2 \cdot 25 = -3. \end{aligned}$$

The next point will be found by solving ELBM-MILP$_0$

$$\begin{aligned} \text{minimize} \quad & |x - 5| + |y - 5| \\ \text{subject to} \quad & x + y - 8 \le -3 \\ & 10x + 6y - 55 \le 0 \\ & 0 \le x \le 5, y \in \{0, 1, 2, 3, 4, 5\}. \end{aligned}$$

Note that we could rewrite the objective by $\mu + \eta$ and add constraints

$$x - 5 \le \mu,\ -x + 5 \le \mu,\ y - 5 \le \eta,\ 5 - y \le \eta$$

to the problem to make it linear, and thus, an MILP problem. For simplicity, we do not do so in this presentation. The next iterate is not unique and we choose $\mathbf{z}^1 = (0, 5)$. Since

$$\max\left\{|0-4| + |5-4| - (-8), (5-2)^2 + 0^2 - 9, 0 + 2 \cdot 5 - 9\right\} = 13 < h^0,$$

we set $h^1 = 13$. Inequality $h^1 \le (1 - 0.2)h^0$ holds and we set a new stability center $\hat{\mathbf{z}}^1 = \mathbf{z}^1$. We will keep every iterate in the bundle throughout this example and, thus, neglect updating the bundle B^k. Calculating $f^1_{\text{lev}} = -8 + 0.2 \cdot 13 = -5.4$ we obtain ELBM-MILP$_1$

$$\begin{aligned} \text{minimize} \quad & |x| + |y - 5| \\ \text{subject to} \quad & x + y - 8 \leq -5.4 \\ & -x + y \leq -5.4 \\ & 10x + 6y - 55 \leq 0 \\ & x + 2y - 9 \leq 0 \\ & 0 \leq x \leq 5, y \in \{0, 1, 2, 3, 4, 5\}. \end{aligned}$$

This problem is infeasible and we update $f^2_{\text{low}} = f^1_{\text{lev}} = -5.4$. Now $h^2 = 13 - 8 + 5.4 = 10.4$ and $f^2_{\text{lev}} = -5.4 + 0.2 \cdot 10.4 = -3.32$. The next MILP problem is

$$\begin{aligned} \text{minimize} \quad & |x| + |y - 5| \\ \text{subject to} \quad & x + y - 8 \leq -3.32 \\ & -x + y \leq -3.32 \\ & 10x + 6y - 55 \leq 0 \\ & x + 2y - 9 \leq 0 \\ & 0 \leq x \leq 5, y \in \{0, 1, 2, 3, 4, 5\}. \end{aligned}$$

Again, the solution is not unique and we pick $\mathbf{z}^2 = (3.32, 0)$. The procedure continues and converges to the global optimum after 14th iteration. The solving process is reported in Table 1 and depicted in Figure 5.

Table 1 Information on iterations when solving the example problem with ELBM.

Iteration $\mathbf{z}^k$		$\hat{\mathbf{z}}^k$	$\mathbf{z}^k_{best}$	h^k	f^k_{low}	f^k_{lev}
0	$(5, 5)$	$(5, 5)$	$(5, 5)$	25	-8	-3
1	$(0, 5)$	$(0, 5)$	$(0, 5)$	13	-8	-5.4
2	Infeasible	$(0, 5)$	$(0, 5)$	10.4	-5.4	-3.32
3	$(3.32, 0)$	$(0, 5)$	$(3.32, 0)$	10.08	-5.4	-3.38
4	Infeasible	$(3.32, 0)$	$(3.32, 0)$	8.06	-3.38	-1.77
5	Infeasible	$(3.32, 0)$	$(3.32, 0)$	6.45	-1.77	-0.48
6	Infeasible	$(3.32, 0)$	$(3.32, 0)$	5.48	-0.48	0.62
7	Infeasible	$(3.32, 0)$	$(3.32, 0)$	4.38	0.62	1.49
8	Infeasible	$(3.32, 0)$	$(3.32, 0)$	3.51	1.49	2.19
9	$(3, 3)$	$(3, 3)$	$(3, 3)$	1	1.49	1.69
10	Infeasible	$(3, 3)$	$(3, 3)$	1	1.69	1.89
11	Infeasible	$(3, 3)$	$(3, 3)$	1	1.89	2.09
12	Infeasible	$(3, 3)$	$(3, 3)$	1	2.09	2.29
13	$(2.83, 3)$	$(2.83, 3)$	$(2.83, 3)$	0.07	2.09	2.11

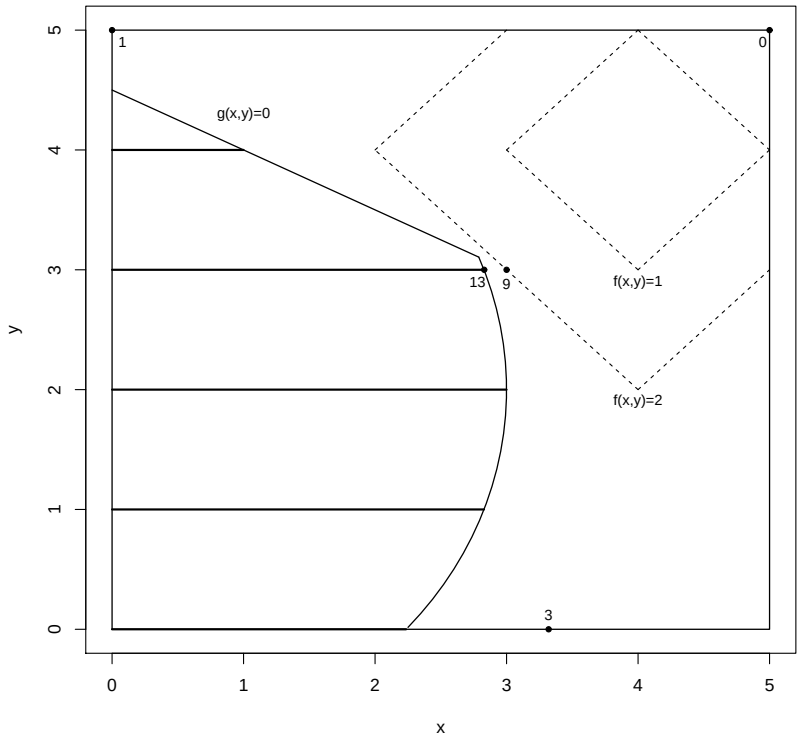

Fig. 5 The solution points found by the extended level bundle method.

4 Concluding remarks

In this chapter we considered deterministic algorithms for convex mixed integer NSO problems. Some of the algorithms can also solve problems with generalized convex functions. All of the algorithms require at least that we can calculate a subgradient from the Clarke subdifferential of the nonlinear functions at any point which algorithm visits. While most of the methods are based on MINLP algorithms for differentiable case, the ELBM method is a generalization of an NSO algorithm to the mixed integer case. By showing that this is possible, it also rises a question whether other bundle methods can be generalized to solve mixed integer problems as well.

References

1. Bagirov, A., Mäkelä, M.M., Karmitsa, N.: Introduction to Nonsmooth Optimization: Theory, Practice and Software. Springer International Publishing, Cham, Heidelberg (2014)
2. Belotti, P., Kirches, C., Leyffer, S., Linderoth, J., Luedtke, J., Mahajan A.: Mixed-integer nonlinear optimization. Acta Numerica **22**, 1–131 (2013)
3. Biegler, L.T., Grossmann, I.E., Westerberg, A.W.: Systematic Methods for Chemical Process Design. Prentice Hall PTR, (1997)
4. Bonami, P., Cornuéjols, G., Lodi, A., Margot, F.: A feasibility pump for mixed integer nonlinear programs. Mathematical Programming **119**, 331–352 (2009)
5. Castillo, I., Westerlund, J., Emet, S., Westerlund, T.: Optimization of block layout design problems with unequal areas: A comparison of MILP and MINLP optimization methods. Computers & Chemical Engineering **30**, 54–69 (2005)
6. Chaib, A.E., Bouchekara, H.R.E.H., Mehasni, R., Abido, M.A.: Optimal power flow with emission and non-smooth cost functions using backtracking search optimization algorithm. Electrical Power and Energy Systems **81**, 64–77 (2016)
7. Clarke, F. H.: Optimization and Nonsmooth Analysis. Wiley-Interscience, New York (1983)
8. CPLEX. https://www.ibm.com/analytics/cplex-optimizer
9. Crouzeix, J.-P., Lindberg, P. O.: Additively decomposed quasiconvex functions. Mathematical Programming **35**, 42–57 (1986)
10. Dakin, R.J.: A tree-search algorithm for mixed integer programming problems. Computer Journal **8**, 250–255 (1965)
11. Delfino, A., de Oliveira, W.: Outer-approximation algorithms for nonsmooth convex MINLP problems. Optimization **67**:6 797–819 (2018)
12. Drewes, S., Ulbrich, S.: Subgradient based outer approximation for mixed integer second order cone programming. In: Lee, J., Leyffer, S. (eds.) Mixed Integer Nonlinear Programming, pp. 41–59. Springer, New York (2012)
13. Duran, M. A., Grossmann, I. E.: An outer-approximation algorithm for a class of mixed-integer nonlinear programs. Mathematical Programming **36**, 307–339 (1986)
14. de Oliveira, W.: Regularized optimization methods for convex MINLP problems. TOP **24**, 665–692 (2016)
15. de Oliveira, W., Sagastizábal C.: Bundle methods in the XXIst century: A bird's eye view. Pesquisa Operacional **34**, 647–670 (2014)
16. de Oliveira, W., Solodov, M.: A doubly stabilized bundle method for nonsmooth convex optimization. Mathematical Programming **156**:1, 125–159 (2016)
17. Emet, S., Westerlund, T.: Comparisons of solving a chromatographic separation problem using MINLP methods. Computers & Chemical Engineering **28**, 673–682 (2004)
18. Eronen, V.-P., Mäkelä, M.M., Westerlund, T.: On the generalization of ECP and OA methods to nonsmooth MINLP problems. Optimization **63**:7, 1057–1073 (2014)
19. Eronen, V.-P., Mäkelä, M.M., Westerlund, T.: Extended cutting plane method for a class of nonsmooth nonconvex MINLP problems. Optimization **64**:3, 641–661 (2015)
20. Eronen, V.-P., Kronqvist, J., Westerlund, T., Mäkelä, M. M., Karmitsa, N.: Method for solving generalized convex nonsmooth mixed-integer nonlinear programming problems. Journal of Global Optimization **69**:2, 443–459 (2017)
21. Fernandes, F.P., Costa, M.F.P., Fernandes, E.M.G.P.: Branch and bound based coordinate search filter algorithm for nonsmooth nonconvex mixed-integer nonlinear programming problems. In: Murgante B. et al. (eds.) Computational Science and Its Applications - ICCSA 2014. ICCSA 2014. Lecture Notes in Computer Science, vol 8580, pp. 140–153. Springer, Cham (2014)

22. Fletcher, R., Leyffer, S.: Solving mixed integer nonlinear programs by outer approximation. Mathematical Programming **66**, 327–349 (1994)
23. GAMS. https://www.gams.com/
24. Grossmann, I. E.: Review of nonlinear mixed-integer and disjunctive programming techniques. Optimization and Engineering **3**, 227–252 (2002)
25. Gurobi. http://www.gurobi.com/
26. Hamzeei, M., Luedtke, J.: Linearization-based algorithms for mixed-integer nonlinear programs with convex continuous relaxation. Journal of Global Optimization **59**, 343–365 (2014)
27. Kelley, J. E.: The cutting plane method for solving convex programs. Journal of SIAM **8**, 703–712 (1960)
28. Khoury, G.A., Smadbeck, J., Kieslich, C.A., Floudas, C.A.: Protein folding and de novo protein design for biotechnological applications. Trends in Biotechnology **32**:2, 99–109 (2014)
29. Kravanja, S., Šilih, S., Kravanja, Z.: The multilevel MINLP optimization approach to structural synthesis: The simultaneous topology, material, standard and rounded dimension optimization. Advances in Engineering Software **36**:9, 568–583 (2005)
30. Kronqvist, J., Lundell, A., Westerlund, T.: The extended supporting hyperplane algorithm for convex mixed-integer nonlinear programming. Journal of Global Optimization **64**, 249–272 (2016)
31. Land, A.H., Doig, A.G.: An automatic method of solving discrete programming problems. Econometrica **28**:3, 497–520 (1960)
32. Lemaréchal, C., Nemirovskii, A., Nesterov, Y.: New variants of bundle methods. Mathematical Programming **69**, 111–147 (1995)
33. Liuzzi, G., Lucidi, S., Rinaldi, F.: Derivative-free methods for mixed-integer constrained optimization problems. Journal of Optimization Theory and Applications **164**:3 933–965 (2015)
34. MINLP Library. http://www.minlplib.org/instances.html
35. Mäkelä, M.M.: Survey of bundle methods for nonsmooth optimization. Optimization Methods and Software **17**:1, 1–29 (2002)
36. Müller, J., Shoemaker, C.A., Piché, R.: SO-MI: A surrogate model algorithm for computationally expensive nonlinear mixed-integer black-box global optimization problems. Computers & Operations Research **40**, 1383–1400 (2013)
37. Newby, E., Ali M.M.: A trust-region-based derivative free algorithm for mixed integer programming. Computational Optimization and Applications **60**, 199–229 (2015)
38. Pörn, R., Westerlund, T.: A cutting plane method for minimizing pseudo-convex functions in the mixed integer case. Computers & Chemical Engineering **24**, 2655–2665 (2000)
39. Pörn, R., Nissfolk, O., Jansson, F., Westerlund, T.: The Coulomb glass - modeling and computational experience with a large scale 0-1 QP problem. Computer Aided Chemical Engineering **29**, 658–662 (2011)
40. Quesada, I., Grossmann, I. E.: An LP/NLP based branch-and-bound algorithm for convex MINLP optimization problems. Computers & Chemical Engineering **16**, 937–947 (1999)
41. Rios, L.M., Sahinidis, N.V.: Derivative-free optimization: A review of algorithms and comparison of software implementations. Journal of Global Optimization **56**:3, 1247–1293 (2013)
42. Schmidt, M., Steinbach, M.C., Willert, B.M.: A primal heuristic for nonsmooth mixed integer nonlinear optimization. In: Jünger, M., Reinelt, G. (eds) Facets of Combinatorial Optimization, pp. 295–320. Springer, Berlin, Heidelberg (2013)
43. Shor, N.Z.: Minimization Methods for Non-Differentiable Functions. Springer, Berlin, (1985)

44. van Ackooij, W., de Oliveira, W.: Level bundle methods for constrained convex optimization with various oracles. Computational Optimization and Applications **57**, 555–597 (2014)
45. Veinott Jr, A.F.: The supporting hyperplane method for unimodal programming. Operations Research **15**:1, 147–152 (1967)
46. Wei, Z., Ali, M.M.: Convex mixed integer nonlinear programming problems and an outer approximation algorithm. Journal of Global Optimization **63**:2, 213–227 (2015)
47. Wei, Z., Ali, M.M.: Outer approximation algorithm for one class of convex mixed-integer nonlinear programming problems with partial differentiability. Journal of Optimization Theory and Applications **167**:2, 644–652 (2015)
48. Westerlund, T., Pettersson, F.: An extended cutting plane method for solving convex MINLP problems. Computers & Chemical Engineering **19**, Supplement 1, 131–136 (1995)
49. Westerlund, T., Pörn, R.: Solving pseudo-convex mixed integer optimization problems by cutting plane techniques. Optimization and Engineering **3**, 253–280 (2002)
50. Westerlund, T., Skrifvars, H., Harjunkoski, I., Pörn, R.: An extended cutting plane method for solving a class of non-convex MINLP problems. Computers & Chemical Engineering **22**, 357–365 (1998)
51. Westerlund, T., Eronen, V.-P., Mäkelä, M.M.: On solving generalized convex MINLP problems using supporting hyperplane techniques. Journal of Global Optimization (2018) https://doi.org/10.1007/s10898-018-0644-z
52. Zioutas, G., Chatzinakos, C., Nguyen, T.D., Pitsoulis, L.: Optimization techniques for multivariate least trimmed absolute deviation estimation. Journal of Combinatorial Optimization **34**, 781–797 (2017)