Tapio Westerlund | Ville-Pekka Eronen | Marko M. Mäkelä

Using Supporting Hyperplane Techniques in Solving Generalized Convex MINLP Problems

TURKU CENTRE *for* COMPUTER SCIENCE

TUCS Technical Report
No 1186, April 2017

Using Supporting Hyperplane Techniques in Solving Generalized Convex MINLP Problems

Tapio Westerlund
Åbo Akademi, Optimization and Systems Engineering
FI-20014 Turku, Finland
twesterl@abo.fi
Ville-Pekka Eronen
University of Turku, Department of Mathematics and Statistics
FI-20014 Turku, Finland
vpoero@utu.fi
Marko M. Mäkelä
University of Turku, Department of Mathematics and Statistics
FI-20014 Turku, Finland
makela@utu.fi

TUCS Technical Report
No 1186, April 2017

Abstract

Solution methods for convex mixed integer nonlinear programming (MINLP) problems have, usually, proven convergence properties if the functions involved are differentiable and convex. For other classes of convex MINLP problems fewer results have been given. Classical differential calculus can, though, be generalized to more general classes of functions than differentiable, via subdifferentials and subgradients. In addition, more general than convex functions can be included in a convex problem if the functions involved are defined from convex level sets, instead of being defined as convex functions only. The notion *generalized convex*, used in the heading of this paper, refers to such additional properties.

The generalization for the differentiability is made by using subgradients of Clarke's subdifferential. Thus, all the functions in the problem are assumed to be locally Lipschitz continuous. The generalization of the functions is done by considering quasiconvex functions. Thus, instead of differentiable convex functions, nondifferentiable quasiconvex functions can be included in the actual problem formulation and a combined supporting hyperplane and cutting plane approach is given for the solution of the considered MINLP problem. Convergence to a global minimum is proved for the algorithm, when minimizing an f°-pseudoconvex function, subject to f°-pseudoconvex constraints. With some additional conditions, the proof is also valid for quasiconvex constraint functions, which sums up the properties of the method, treated in the paper.

Keywords: Nonsmooth optimization; MINLP; Generalized convexities; Clarke generalized derivatives; Cutting planes; Supporting hyperplanes

TUCS Laboratory
Turku Optimization Group (TOpGroup)

1 Introduction

Mixed-integer problems are generally nonconvex, because of the inherent nature of the integer variables. Classifying mixed integer problems into *linear*, *convex* or *nonconvex* is, therefore, somewhat confusing. However, the classification is done on the integer relaxed problem. This is quite convenient since the integer requirements are, in all state of the art mixed-integer nonlinear programming (MINLP) solvers, handled separately by a branch-and-bound (or corresponding) procedure, while solving relaxed subproblems.

Several algorithms to solve smooth (continuously differentiable) convex MINLP problems, have been published over the last few decades. The methods behind the solvers are often divided into branch-and-bound (BB), decomposition, cutting plane and outer approximation methods. In direct BB methods [14, 20, 26] and decomposition methods [15], integer relaxed convex subproblems are solved in each node of a BB tree. In cutting plane [32, 33] and outer approximation methods [4, 9, 13, 19], the original MINLP problem is relaxed into a series of mixed integer linear programming (MILP) problems. The linearly relaxed subproblems are built up of cutting planes and/or supporting hyperplanes and sequentially solved as a series of subproblems, which finally give the solution to the original convex MINLP problem. In the outer approximation methods [3, 9, 13], NLP problems are additionally solved in order to obtain the solution points where the supporting hyperplanes are generated. In the extended supporting hyperplane (ESH) methods [12, 19] a line search procedure is used to obtain these points. In the cutting plane methods [31, 32, 33] no NLP problems are solved, since the cutting planes are already generated at the solution points obtained from the subproblems. Usually, MILP subproblems are solved but if the objective function is quadratic, the MIQP subproblems often result in a more efficient procedure. As shown in [32], all linearly relaxed subproblems need not be solved to optimality, but in order to finally guarantee the optimality of the MINLP solution at least, the last subproblem must be solved to optimality. In a comparison of solving smooth convex MINLP problems in [6], it was found in many instances, that only one MILP subproblem needed to be solved to optimality, thus, resulting in a very efficient procedure.

Many smooth convex algorithms are already in commercial use in different solution packages, such as GAMS, AIMM, AMPL and LINDO. Reviews of several solution approaches can be found in [3, 4, 16]. Comparisons of the efficiency and performance of the solvers on smooth convex problems, are additionally given, for example, in [6, 19].

Despite a large number of solvers, with proven convergence properties for differentiable convex problems, the development of new algorithms for solving convex MINLP problems is still an important activity. This is not only true because of the large number of applied problems that can be formulated in a general convex context, but especially because convexity induces several fundamental properties, which have to be taken into account, in order to be able to rigorously solve generalized convex MINLP problems. In addition, new algorithms for solving nonconvex problems need solve sequences of convex problems [1, 21, 29], forcing additional requirements to be handled, by the convex subsolver. Since convexity for functions and sets does not induce exactly the same reverse properties, the development of generalized algorithms is more demanding than it turns out to be at first glance.

Today, the majority of the state of the art solvers, for convex MINLP problems, have proven convergence properties for problems including differentiable convex functions. However, for example, replacing gradients with subgradients, in such an algorithm does not automatically ensure that the same convergence properties are fulfilled, even if the constraints are convex, but nonsmooth. For example, an endless cycling behavior between the solution points from the NLP problem and the MILP masters' problem was obtained in the linear outer approximation (LOA) algorithm [13] by such a replacement in [10]. This was the case, despite the fact that the convergence properties of the LOA algorithm for smooth convex functions were still ensured. Therefore, it is important to note that nonsmooth techniques can successfully be applied to smooth problems, but not always vice versa.

Nonsmooth convex functions, such as *abs*-functions and *max*-functions are simple examples of nondifferentiable functions, frequently appearing in a wide variety of problems. Nondifferentiable functions are commonly used in optimal control problems, in mechanics, economics, data mining, machine learning, medical diagnosis etc., showing the importance of being able to handle such functions rigorously, in a solver. Generalized convex functions, such as fractional functions composed of a convex nominator and a positive linear denominator, typically appear as the objective function in cyclic problems. Such fractional functions, give rise to convex level sets, are quasiconvex but not necessarily convex. Nonsmooth convex spline functions, used for tightening the underestimation and improving the efficiency of certain global optimization solvers [21, 22], exemplify the importance of also being able to solve nonsmooth convex subproblems in global optimization algorithms.

In order to solve such problems rigorously, we introduce in this paper an algorithm for solving generalized convex MINLP problems. The notation *generalized convex*, used in the heading of the paper refers to the additional convex properties that are taken into account in the algorithm. In the method considered, we assume all functions to be at least locally Lipschitz continuous and f°-pseudoconvex. With some additional assumptions the functions may be quasiconvex. Thus, in addition to differentiable convex functions, nondifferentiable, pseudo and quasiconvex functions can be handled with this actual method.

The solution approach, studied in the paper, has its origin in the cutting plane method [18] and the supporting hyperplane method [30], which were introduced for solving differentiable convex NLP problems. The cutting plane approach was extended to smooth convex MINLP problems in [33] and further extended to handle smooth pseudoconvex functions both in the objective function and the constraints in [31, 32]. In [10, 11] the cutting plane approach was generalized to be able to handle nonsmooth f°-pseudoconvex functions and a regularized cutting plane method for solving nonsmooth convex MINLP problems has been given in [8]. In [25] supporting hyperplanes were introduced as alternatives to cutting planes when solving differentiable convex MINLP problems and in [19] a convergence proof for the differentiable convex case has been given and the method was named the extended supporting hyperplane method. The convergence proof, for the supporting hyperplane approach, was extended to cover problems including nonsmooth f°-pseudoconvex constraints, in [12]. In this paper, we finally generalize the supporting hyperplane approach to MINLP problems including f°-pseudoconvex functions both in the objective function as well as in the constraints. The proof is also valid for quasiconvex constraint functions, with the restriction, that the supports, then need to be generated at points where the subgradients are nonzero.

In the method considered in this paper, the supporting hyperplanes are generated at solution points obtained from one or two different line searches: one for the objective function and one for the constraints. Supports to the constraints are generated on the boundary of the feasible region and supports to the objective function on boundaries of decreasing level sets of the objective function. The supports to the objective function thus form convex cones, used in the solution approach. Nevertheless, interior points for the line searchers are needed. If an optimal integer relaxed solution point of the problem is given or can be calculated, this point can be used as an interior point in both line searches. However, an interior point obtained from solving an integer relaxed feasibility problem is preferable, since such a problem can easily be solved, for example, with a linear programming (LP) based hyperplane approach, such as the one given in the paper. In a case where the objective function is convex, the point obtained from an integer relaxed feasibility problem is needed only in the line search for the constraints, but is usable for the objective function line search as well. However, if the objective function is f°-pseudoconvex an optimal integer relaxed solution point is, in principle, needed for the objective function. Such an NLP point can be calculated using the approaches presented in [24] or in [11, 32]. However, in the paper we will show that the given hyperplane method is able to solve a corresponding NLP problem, by itself, and thus any other feasible interior point for the objective function as well. With a modified approach to that presented in [32], a sequentially improved interior point for the objective function can also be obtained and thus it is not necessary to give a presolved interior point for the objective function in the given solution approach.

In the next sections, we will prove that the method converges to an ε-global optimal value, when solving problems with an f°-pseudoconvex objective function and f°-pseudoconvex constraints. The bisection method is used in all the line searches to ensure successful solutions. In the considered numerical examples we compare the supporting hyperplane approach with the cutting plane approach in [11, 32]. Furthermore, the use of different solution strategies are illustrated. The solver in [34] has been used in all the computations.

To make the paper easier to read, some notations and basic information on generalized convexity and nonsmooth optimization has been provided in the first chapter. More information on generalized convexity can be found, for example, in the textbooks [27, 28] and on nonsmooth optimization, in the references [2, 7, 23]. In addition, the textbook [5] can be recommended as related background material when considering solution approaches for solving generalized convex NLP problems.

2 Preliminaries

In this section some basic definitions and results are given on the function classes we consider.

DEFINITION 2.1. A function $f : \mathbb{R}^n \to \mathbb{R}$ is *locally Lipschitz continuous* at a point $\boldsymbol{x} \in \mathbb{R}^n$ if there exist scalars $K > 0$ and $\delta > 0$ such that

$$|f(\boldsymbol{y}) - f(\boldsymbol{z})| \leq K\|\boldsymbol{y} - \boldsymbol{z}\| \quad \text{for all } \boldsymbol{y}, \boldsymbol{z} \in B(\boldsymbol{x}; \delta),$$

where $B(\boldsymbol{x}; \delta) \subset \mathbb{R}^n$ is an open ball with center $\boldsymbol{x}$ and radius δ.

For a locally Lipschitz continuous function a gradient may not exist everywhere. However, a *Clarke subgradient* can be defined at any point.

DEFINITION 2.2. [7] Let $f : \mathbb{R}^n \to \mathbb{R}$ be locally Lipschitz continuous at $\boldsymbol{x} \in \mathbb{R}^n$. The *Clarke generalized directional derivative* of f at $\boldsymbol{x}$ in the direction $\boldsymbol{d} \in \mathbb{R}^n$ is defined by

$$f^\circ(\boldsymbol{x}; \boldsymbol{d}) := \limsup_{\substack{\boldsymbol{y} \to \boldsymbol{x} \\ t \downarrow 0}} \frac{f(\boldsymbol{y} + t\boldsymbol{d}) - f(\boldsymbol{y})}{t}$$

and the *Clarke subdifferential* of f at $\boldsymbol{x}$ by

$$\partial f(\boldsymbol{x}) := \{\boldsymbol{\xi} \in \mathbb{R}^n \mid f^\circ(\boldsymbol{x}; \boldsymbol{d}) \geq \boldsymbol{\xi}^T \boldsymbol{d} \text{ for all } \boldsymbol{d} \in \mathbb{R}^n\}.$$

Each element $\boldsymbol{\xi} \in \partial f(\boldsymbol{x})$ is called a *subgradient* of f at $\boldsymbol{x}$.

Note that for a smooth (i.e. continuously differentiable) function $f : \mathbb{R}^n \to \mathbb{R}$ we have $\partial f(\boldsymbol{x}) = \{\nabla f(\boldsymbol{x})\}$ for any $\boldsymbol{x} \in \mathbb{R}^n$.

THEOREM 2.3. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be locally Lipschitz continuous at $\boldsymbol{x} \in \mathbb{R}^n$. Then*

(i) $\partial f(\boldsymbol{x})$ *is a nonempty, convex and compact set.*

(ii) $\partial f(\boldsymbol{x}) \subset B(\boldsymbol{0}; K)$, *where K is a Lipschitz constant of f at $\boldsymbol{x}$.*

(iii) $f^\circ(\boldsymbol{x}; \boldsymbol{d}) = \max\{\boldsymbol{\xi}^T \boldsymbol{d} \mid \boldsymbol{\xi} \in \partial f(\boldsymbol{x})\}$ *for all $\boldsymbol{d} \in \mathbb{R}^n$.*

(iv) $f^\circ(\boldsymbol{x}; \boldsymbol{d})$ *is an upper semicontinuous function of $(\boldsymbol{x}, \boldsymbol{d})$.*

Proof. The proofs can be found in [7, pp. 26–27]. □

The following fundamental theorem presents an easy way to determine the subdifferentials of a function.

THEOREM 2.4. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be locally Lipschitz continuous at $\boldsymbol{x} \in \mathbb{R}^n$. Then*

$$\partial f(\boldsymbol{x}) = \operatorname{conv}\{\boldsymbol{\xi} \in \mathbb{R}^n \mid \exists (\boldsymbol{x}_i) \subset \mathbb{R}^n \backslash \Omega_f \text{ s.t. } \boldsymbol{x}_i \to \boldsymbol{x} \text{ and } \nabla f(\boldsymbol{x}_i) \to \boldsymbol{\xi}\},$$

where conv *denotes the convex hull of a set and Ω_f is the set of points on which function f is not differentiable.*

Proof. The proof can be found in [7, pp. 63]. □

The function classes being considered can now be defined stating with a recall of the definition of the classical pseudoconvexity.

DEFINITION 2.5. A function $f : \mathbb{R}^n \to \mathbb{R}$ is *pseudoconvex*, if it is smooth and for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$

$$f(\boldsymbol{y}) < f(\boldsymbol{x}) \qquad \text{implies} \qquad \nabla f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x}) < 0.$$

In Definition 2.5, it can also be written $\nabla f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x}) = f'(\boldsymbol{x}; \boldsymbol{y} - \boldsymbol{x})$, where f' is the classical notation of the directional derivative. This will make the definition analogous to the following generalization.

DEFINITION 2.6. A locally Lipschitz continuous function $f : \mathbb{R}^n \to \mathbb{R}$ is *f°-pseudoconvex* (*f°-quasiconvex*) if for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$

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

It is known that a convex or pseudoconvex function is f°-pseudoconvex. Furthermore, an f°-pseudoconvex function is f°-quasiconvex. The level sets of all these function classes are convex. These results can be found in [2].

We say that function is *l-quasiconvex* if it is locally Lipschitz continuous and quasiconvex. We can also define it in a similar way to f°-quasiconvexity (see e.g. [2]).

DEFINITION 2.7. A locally Lipschitz continuous function $f : \mathbb{R}^n \to \mathbb{R}$ is l-quasiconvex if for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$

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

An f°-quasiconvex function is l-quasiconvex, but the reverse is not always true [2]. Thus, the results that we will formulate for l-quasiconvex functions holds also for f°-quasiconvex functions.

With the following lemma it can be seen that in Definition 2.6 we could use appropriate compact sets instead of points $\boldsymbol{x}$ and $\boldsymbol{y}$.

LEMMA 2.8. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be f°-pseudoconvex. Let $A, C \subset \mathbb{R}^n$ be nonempty compact sets such that there exists $a \in \mathbb{R}$ such that $f(\boldsymbol{y}) < a \leq f(\boldsymbol{x})$ for all $\boldsymbol{x} \in A$ and $\boldsymbol{y} \in C$. Then, there exists $\delta > 0$ such that*

$$\sup_{\substack{\boldsymbol{x} \in A \\ \boldsymbol{\xi} \in \partial f(\boldsymbol{x}) \\ \boldsymbol{y} \in C}} \boldsymbol{\xi}^T(\boldsymbol{y} - \boldsymbol{x}) = -\delta.$$

Proof. The proof can be found in [11] Lemma 2.10. □

We need an additional assumption to prove the corresponding result for l-quasiconvex function. In addition, another lemma is first needed.

LEMMA 2.9. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be l-quasiconvex and $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$. If $f(\boldsymbol{y}) < f(\boldsymbol{x})$ and $\boldsymbol{0} \notin \partial f(\boldsymbol{x})$, then $f^\circ(\boldsymbol{x}; \boldsymbol{y} - \boldsymbol{x}) < 0$.*

Proof. The proof is similar to that of Lemma 1 in [12]. □

With Lemma 2.9 we can prove that if $\mathbf{0} \in \partial f(\boldsymbol{x})$ implies that $\boldsymbol{x}$ is a global minimum of f, then the l-quasiconvex function f is f°-pseudoconvex. This result is proven in e.g. [12] for f°-quasiconvex functions but the proof holds true for l-quasiconvex functions as well.

LEMMA 2.10. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be l-quasiconvex. Let $A, C \subset \mathbb{R}^n$ be nonempty compact sets such that there exists $a \in \mathbb{R}$ such that $f(\boldsymbol{y}) < a \le f(\boldsymbol{x})$ for all $\boldsymbol{x} \in A$ and $\boldsymbol{y} \in C$. Suppose that $\mathbf{0} \notin \partial f(\boldsymbol{x})$ for all $\boldsymbol{x} \in A$. Then, there exists $\delta > 0$ such that*

$$\sup_{\substack{\boldsymbol{x} \in A \\ \boldsymbol{\xi} \in \partial f(\boldsymbol{x}) \\ \boldsymbol{y} \in C}} \boldsymbol{\xi}^T(\boldsymbol{y} - \boldsymbol{x}) = -\delta.$$

Proof. We can write

$$\sup_{\substack{\boldsymbol{x} \in A \\ \boldsymbol{\xi} \in \partial f(\boldsymbol{x}) \\ \boldsymbol{y} \in C}} \boldsymbol{\xi}^T(\boldsymbol{y} - \boldsymbol{x}) = \sup_{\substack{\boldsymbol{x} \in A \\ \boldsymbol{y} \in C}} \sup_{\boldsymbol{\xi} \in \partial f(\boldsymbol{x})} \boldsymbol{\xi}^T(\boldsymbol{y} - \boldsymbol{x}).$$

By Lemma 2.3 (ii)

$$\sup_{\boldsymbol{\xi} \in \partial f(\boldsymbol{x})} \boldsymbol{\xi}^T(\boldsymbol{y} - \boldsymbol{x}) = f^\circ(\boldsymbol{x}; \boldsymbol{y} - \boldsymbol{x}).$$

Recall that an upper semicontinuous function attains its maximum value on a compact set. Thus, there exists $\hat{\boldsymbol{x}} \in A$ and $\hat{\boldsymbol{y}} \in C$ such that

$$\sup_{\substack{\boldsymbol{x} \in A \\ \boldsymbol{y} \in C}} f^\circ(\boldsymbol{x}; \boldsymbol{y} - \boldsymbol{x}) = f^\circ(\hat{\boldsymbol{x}}; \hat{\boldsymbol{y}} - \hat{\boldsymbol{x}}).$$

By Lemma 2.9 $f^\circ(\hat{\boldsymbol{x}}; \hat{\boldsymbol{y}} - \hat{\boldsymbol{x}}) < 0$, which completes the proof. □

The following result allows us to treat locally Lipschitz continuous functions as Lipschitz continuous ones.

LEMMA 2.11. *If $f : \mathbb{R}^n \to \mathbb{R}$ is locally Lipschitz continuous on a compact set L, then it is Lipschitz continuous on the set L.*

3 ESH for the problem with an f°-pseudoconvex objective function

In this section, the extended supporting hyperplane method [12, 19] is generalized to solve a problem with an f°-pseudoconvex objective function and f°-pseudoconvex constraint functions. Unlike with a convex objective function, we can not transform the f°-pseudoconvex objective function f to the constraint function $f - \mu \le 0$ and minimize μ, since generally $f - \mu$ may not be f°-pseudoconvex even if f is. Consider the problem:

$$\begin{array}{lll} \min & f(\boldsymbol{x}) & \\ \text{s.t.} & g_m(\boldsymbol{x}) \leq 0, \quad \forall m = 1, \ldots, M & \text{(P)} \\ & \boldsymbol{x} \in L \cap Y, & \end{array}$$

where f and g_m are f°-pseudoconvex functions and $L \subset \mathbb{R}^n$ is a convex compact polytope defined by linear constraints. Integer variables are defined by the index set $I_{\mathbb{Z}} \subseteq \{1, 2, \ldots, n\}$ and the set $Y = \{\boldsymbol{x} \mid \boldsymbol{x} \in \mathbb{R}^n, x_i \in \mathbb{Z} \text{ if } i \in I_{\mathbb{Z}}\}$. Naturally, all the functions are locally Lipschitz continuous. Denote

$$\begin{aligned} &F(\boldsymbol{x}) = \max_{m=1,\ldots,M} \{g_m(\boldsymbol{x})\}, \\ &N = \{\boldsymbol{x} \in \mathbb{R}^n \mid F(\boldsymbol{x}) \leq 0\} \text{ and} \\ &I_0(\boldsymbol{x}) = \{m \mid g_m(\boldsymbol{x}) = F(\boldsymbol{x}) = 0\}. \end{aligned}$$

The key idea of the ESH method is to approximate the nonlinear feasible set by supporting hyperplanes. The point at which a hyperplane is created is found through a line search. The one end point of the line search is the obtained solution point of an MILP subproblem. The other end point denoted by $\boldsymbol{x}_{\text{NLP}}$ is any point from the set $N \cap L$ which must be given to or initially solved by the algorithm. This point is also called the feasible point and it can be found e.g. by the algorithm presented in Section 6. It should be noted that $\boldsymbol{x}_{\text{NLP}}$ is a feasible point of (P) when the integer variables are relaxed to continuous ones. For simplicity we do not mention the integer relaxation explicitly in the next two sections.

A line search may be done on the objective function as well. Let f_r be the current upper bound for the objective function and $I_r^p = \{\boldsymbol{x}_r^1, \boldsymbol{x}_r^2, \ldots, \boldsymbol{x}_r^p\}$ be the set of the found points $\boldsymbol{x}_r^j$ that satisfies $f(\boldsymbol{x}_r^j) = f_r$. On the one end point of the line search f should attain a value that is lower than or equal to f_r. Define

$$\boldsymbol{x}_{\text{NLP}}^{fr} = \begin{cases} \boldsymbol{x}_{\text{NLP}} & \text{, if } f(\boldsymbol{x}_{\text{NLP}}) < f_r \\ \overline{\boldsymbol{x}}_r^p := \frac{1}{p} \sum_{j=1}^{p} \boldsymbol{x}_r^j & \text{, if } f(\boldsymbol{x}_{\text{NLP}}) \geq f_r. \end{cases} \tag{1}$$

The line search for the objective function is done between the solution point of an MILP subproblem and $\boldsymbol{x}_{\text{NLP}}^{fr}$. Note that if $\boldsymbol{x}_{\text{NLP}}$ is an optimum of the integer relaxed version of (P), then $\boldsymbol{x}_{\text{NLP}}^{fr} = \boldsymbol{x}_{\text{NLP}}$ for all r. Any $\boldsymbol{x}_{\text{NLP}} \in N \cap L$ can be used as $\boldsymbol{x}_{\text{NLP}}^{fr}$ as long as $f(\boldsymbol{x}_{\text{NLP}}) < f_r$. When $f(\boldsymbol{x}_{\text{NLP}}) \geq f_r$ the point $\overline{\boldsymbol{x}}_r^p$ is used in the line search. Since the level sets of the f°-pseudoconvex function f are convex, we have $f(\overline{\boldsymbol{x}}_r^p) \leq f_r$. In theory we could use $\boldsymbol{x}_{\text{NLP}}$ as the inner point even when $f(\boldsymbol{x}_{\text{NLP}}) = f_r$, but we choose the follow the equation (1).

The problem (P) is solved as a sequence of MILP problems. At iteration k we solve the problem

$$\begin{array}{lll} \min & \mu & \\ \text{s.t.} & f_r + \boldsymbol{\xi}_i^T(\boldsymbol{x} - \boldsymbol{x}_f^i) \leq \mu, \quad i \in I_f^k & \\ & \boldsymbol{\xi}_i^T(\boldsymbol{x} - \boldsymbol{x}_g^i) \leq 0, \quad i \in I_g^k & (\text{MILP}_k) \\ & \boldsymbol{x} \in L \cap Y, \mu \in [\mu_{\text{L}}, \mu_{\text{U}}], & \end{array} \tag{2}$$

where f_r is the current upper bound, $\mu_{\text{L}}, \mu_{\text{U}}$ are user given bounds, $\boldsymbol{\xi}_i \in \partial f(\boldsymbol{x}_f^i)$ if $i \in I_f^k$ and $\boldsymbol{\xi}_i \in \partial g_{m_i}(\boldsymbol{x}_g^i)$, where $m_i \in I_0(\boldsymbol{x}_g^i)$, if $i \in I_g^k$. Furthermore,

$$I_f^k = \left\{ i < k \mid F(\boldsymbol{x}_{\text{MILP}}^i) \leq \varepsilon_g \right\} \text{ and } I_g^k = \left\{ i < k \mid F(\boldsymbol{x}_{\text{MILP}}^i) > \varepsilon_g \right\},$$

where $\varepsilon_g > 0$ is a tolerance parameter given by the user and $(\boldsymbol{x}_{\text{MILP}}^i, \mu^i)$ is the solution point of (MILP_i). Points $\boldsymbol{x}_g^i$ are found through a line search between points $\boldsymbol{x}_{\text{MILP}}^i$ and $\boldsymbol{x}_{\text{NLP}}$. Points $\boldsymbol{x}_f^i$ are solutions points $\boldsymbol{x}_{\text{MILP}}^i$ or, if necessary, they are found through a line search between points $\boldsymbol{x}_{\text{MILP}}^i$ and $\boldsymbol{x}_{\text{NLP}}^{fr}$. At first $I_f^1 = I_g^1 = \emptyset$ but after the first iteration $I_f^k \cup I_g^k = \{1, 2, \ldots, k-1\}$ and $I_f^k \cap I_g^k = \emptyset$.

Algorithm 3.1, presented on the next page, handles constraints in the same manner as the ESH algorithm in [12]. The f°-pseudoconvex objective function is handled in a closely related way to how the αECP method handles it in [32]. The point $\boldsymbol{x}_{\text{NLP}}^{fr}$ guarantees that in step 4.4 we can always find a point on the contour $\{\boldsymbol{x} \in \mathbb{R}^n \mid f(\boldsymbol{x}) = f_r + \varepsilon_g\}$. This implies that we can add a constraint of type (2) whenever $F(\boldsymbol{x}_{\text{MILP}}^k) \leq \varepsilon_g$. However, we do not need to use the constraint $f(\boldsymbol{x}) - f_r \leq 0$ that was used in [32].

Algorithm 3.1 produces two sequences of values of the objective function. The sequence $(f_{r(k)})$ corresponds to objective function values of ε_g-feasible solutions of the primal problem (P), while the sequence (μ^k) corresponds to the objective function values of the linearly relaxed problems (MILP_k). The ε_g-feasibility will be satisfied in step 4 of the algorithm, while the final termination will occur in step 2, when the gap between f_r and μ is less than or equal to ε_f, i.e $f_r - \mu^k \leq \varepsilon_f$. In Algorithm 3.1 and throughout the text we have used ε_g and ε_f as absolute tolerances for feasibility and for the gap between f_r and μ^k, respectively. In a numerical algorithm these can be represented by relative tolerances.

4 The convergence proof

In what follows, we show that if $\varepsilon_g > 0$ and $\varepsilon_f = 0$ Algorithm 3.1 converges to an ε_g-feasible global minimum value. A point $\boldsymbol{x} \in L \cap Y$ is an ε_g-feasible global minimum if $F(\boldsymbol{x}) \leq \varepsilon_g$ and there does not exist $\boldsymbol{y} \in N \cap L \cap Y$ such that $f(\boldsymbol{y}) < f(\boldsymbol{x})$. Then $f(\boldsymbol{x})$ is an ε_g-feasible global minimum value.

When considering the convergence of Algorithm 3.1 there is a useful result that has already been proven in [12].

Algorithm 3.1 The ESH algorithm

Give the tolerance parameters ε_g, $\varepsilon_f > 0$, give $\boldsymbol{x}_{\text{NLP}} \in N \cap L$ (can be found by e. g. the algorithm in Section 6), set $I_g^k = I_f^k = \emptyset$ and $k = r = p = 1$. Set $f_r = \infty$ or if an integer feasible point $\boldsymbol{x}_{\text{MILP}}^0$ is known let $f_r = f(\boldsymbol{x}_{\text{MILP}}^0)$, $I_f^k = \{0\}$, $I_r^p = \{\boldsymbol{x}_{\text{MILP}}^0\}$ and add $f_r + \boldsymbol{\xi}^T(\boldsymbol{x} - \boldsymbol{x}_{\text{MILP}}^0) \leq \mu$, where $\boldsymbol{\xi} \in \partial f(\boldsymbol{x}_{\text{MILP}}^0)$, to (MILP_1).

1. Solve the problem (MILP_k). Denote the solution by $(\boldsymbol{x}_{\text{MILP}}^k, \mu^k)$.

2. If $\mu^k \geq f_r - \varepsilon_f$ then stop: f_r is the optimal value and the first element of I_r^p is the final solution.

3. If $F(\boldsymbol{x}_{\text{MILP}}^k) > \varepsilon_g$, do a line search between $\boldsymbol{x}_{\text{NLP}}$ and $\boldsymbol{x}_{\text{MILP}}^k$ to find $\boldsymbol{x}_g^k$ such that $F(\boldsymbol{x}_g^k) = \frac{\varepsilon_g}{2}$. Add to the problem (MILP_{k+1}) the linear constraint $\boldsymbol{\xi}^T(\boldsymbol{x} - \boldsymbol{x}_g^k) \leq 0$, where $\boldsymbol{\xi} \in \partial g_m(\boldsymbol{x}_g^k)$ and $g_m(\boldsymbol{x}_g^k) = F(\boldsymbol{x}_g^k)$. Update $I_g^{k+1} = I_g^k \cup \{k\}$ and $I_f^{k+1} = I_f^k$.

4. If $F(\boldsymbol{x}_{\text{MILP}}^k) \leq \varepsilon_g$ then

   4.1 If $f(\boldsymbol{x}_{\text{MILP}}^k) < f_r$, update $r = r + 1$ and $p = 1$. Set $\boldsymbol{x}_f^k = \boldsymbol{x}_{\text{MILP}}^k$, $f_r = f(\boldsymbol{x}_f^k)$ and $I_r^p = \left\{\boldsymbol{x}_f^k\right\}$. Update the constraints of type (2) by using the new value f_r.

   4.2 If $f(\boldsymbol{x}_{\text{MILP}}^k) = f_r$, then set $\boldsymbol{x}_f^k = \boldsymbol{x}_{\text{MILP}}^k$, $I_r^{p+1} = I_r^p \cup \left\{\boldsymbol{x}_f^k\right\}$ and $p = p + 1$.

   4.3 If $f_r < f(\boldsymbol{x}_{\text{MILP}}^k) \leq f_r + \varepsilon_g$, set $\boldsymbol{x}_f^k = \boldsymbol{x}_{\text{MILP}}^k$.

   4.4 If $f(\boldsymbol{x}_{\text{MILP}}^k) > f_r + \varepsilon_g$, calculate $\boldsymbol{x}_{\text{NLP}}^{fr}$ from (1). Find $\boldsymbol{x}_f^k$ such that $f(\boldsymbol{x}_f^k) = f_r + \varepsilon_g$ with a line search between $\boldsymbol{x}_{\text{NLP}}^{fr}$ and $\boldsymbol{x}_{\text{MILP}}^k$.

   4.5 Add to the problem (MILP_{k+1}) the linear constraint $f_r + \boldsymbol{\xi}^T(\boldsymbol{x} - \boldsymbol{x}_f^k) \leq \mu$, where $\boldsymbol{\xi} \in \partial f(\boldsymbol{x}_f^k)$. Update $I_f^{k+1} = I_f^k \cup \{k\}$ and $I_g^{k+1} = I_g^k$.

5. Set $k = k + 1$ and go to step 1.

LEMMA 4.1. *If $\varepsilon_g > 0$, then the algorithm will find a point $\boldsymbol{x}^k_{MILP}$ such that $F(\boldsymbol{x}^k_{MILP}) \leq \varepsilon_g$ after a finite number of iterations.*

Proof. This is stated in [12] after Theorem 3.6. □

The algorithm in [12] assumes a convex objective function and, thus, it is different from Algorithm 3.1. Despite this, Lemma 4.1 is valid also when having an f°-pseudoconvex objective function and the proof is similar to that in [12].

The convergence proof of Algorithm 3.1 proceeds as follows. If the algorithm stops it is shown that the current upper bound f_r is an ε_g-feasible minimum value. If the algorithm does not stop after a finite number of iterations it is shown that the sequence $(\mu^k - f_r)$ converges to zero. Furthermore, this will imply that f_r converges to an ε_g-feasible minimum value.

We begin by proving a technical lemma.

LEMMA 4.2. *Let $i \in I^k_f$. Then $\boldsymbol{\xi}_i^T(\boldsymbol{x}^i_{MILP} - \boldsymbol{x}^i_f) \geq 0$, where $\boldsymbol{\xi}_i \in \partial f(\boldsymbol{x}^i_f)$.*

Proof. If $\boldsymbol{x}^i_f = \boldsymbol{x}^i_{\text{MILP}}$ the result is true. Otherwise, $\boldsymbol{x}^i_f$ is found through the line search between $\boldsymbol{x}^i_{\text{MILP}}$ and $\boldsymbol{x}^{fr}_{\text{NLP}}$. Since $f(\boldsymbol{x}^{fr}_{\text{NLP}}) < f(\boldsymbol{x}^i_f)$ the pseudoconvexity implies $\boldsymbol{\xi}_i^T(\boldsymbol{x}^{fr}_{\text{NLP}} - \boldsymbol{x}^i_f) < 0$. Due to line search $\boldsymbol{x}^i_f = \lambda \boldsymbol{x}^i_{\text{MILP}} + (1-\lambda)\boldsymbol{x}^{fr}_{\text{NLP}}$, for some $\lambda \in (0,1)$. Thus,

$$\boldsymbol{\xi}_i^T(\boldsymbol{x}^i_{\text{MILP}} - \boldsymbol{x}^i_f) = -\frac{1-\lambda}{\lambda}\boldsymbol{\xi}_i^T(\boldsymbol{x}^{fr}_{\text{NLP}} - \boldsymbol{x}^i_f) > 0,$$

completing the proof. □

Notice that the index r is a function of the index k by Algorithm 3.1. For simplicity, we will write r instead of $r(k)$. Let $\overline{\boldsymbol{x}} \in L$ and $k \in \mathbb{N}$ be arbitrary. Denote

$$\mu^k_{\overline{\boldsymbol{x}}} = f_r + \max_{i<k} \left\{ \boldsymbol{\xi}_i^T(\overline{\boldsymbol{x}} - \boldsymbol{x}^i_f) \right\}, \tag{3}$$

where $\boldsymbol{\xi}_i \in \partial f(\boldsymbol{x}^i_f)$ is used in (MILP$_k$). Equivalently, $\mu^k_{\overline{\boldsymbol{x}}}$ is the minimum of the problem (MILP$_k$) with added constraint $\boldsymbol{x} = \overline{\boldsymbol{x}}$. Clearly, if $\overline{\boldsymbol{x}}$ is feasible in (MILP$_k$) then $\mu^k_{\overline{\boldsymbol{x}}} \geq \mu^k$ since μ is minimized in (MILP$_k$).

The following theorem justifies the stopping criterion of Algorithm 3.1 when $\varepsilon_f = 0$.

THEOREM 4.3. *If $\mu^k \geq f_r$, then the current upper bound f_r is an ε_g-feasible global minimum value.*

Proof. On the contrary, suppose there exists $\hat{\boldsymbol{x}} \in N \cap L \cap Y$ such that $f(\hat{\boldsymbol{x}}) < f_r$. Let $C = \{\hat{\boldsymbol{x}}\}$, $A = \{\boldsymbol{x} \in \mathbb{R}^n \mid f(\boldsymbol{x}) \geq f_r\} \cap L$ and $a = f_r$. By Lemma 2.8 there exists $\delta > 0$ such that

$$\mu^k_{\hat{\boldsymbol{x}}} = f_r + \max_{i \in I^k_f} \left\{ \boldsymbol{\xi}_i^T(\hat{\boldsymbol{x}} - \boldsymbol{x}^i_f) \right\} \leq f_r + \sup_{\substack{\boldsymbol{z} \in A \\ \boldsymbol{\xi} \in \partial f(\boldsymbol{z})}} \left\{ \boldsymbol{\xi}^T(\hat{\boldsymbol{x}} - \boldsymbol{z}) \right\} = f_r - \delta.$$

Since $\hat{\boldsymbol{x}}$ is feasible this implies $\mu^k \leq \mu^k_{\hat{\boldsymbol{x}}} < f_r$ contradicting the assumption. □

Theorem 4.3 proves the convergence if the algorithm stops after a finite number of iterations. The next step of the proof of the convergence is to consider the case when the algorithm does not stop after a finite number of iterations.

In the following lemma we explicitly write $r = r(k)$ to make the proof easier to understand.

LEMMA 4.4. *The sequence* $(\mu^k - f_{r(k)})$ *is increasing.*

Proof. Let $\boldsymbol{x}^k_{\text{MILP}}$ be the solution to the problem (MILP_k). Then

$$\mu^k - f_{r(k)} = \max_{i \in I_f^k} \left\{ \boldsymbol{\xi}_i^T (\boldsymbol{x}^k_{\text{MILP}} - \boldsymbol{x}^i_f) \right\} \geq \max_{i \in I_f^{k-1}} \left\{ \boldsymbol{\xi}_i^T (\boldsymbol{x}^k_{\text{MILP}} - \boldsymbol{x}^i_f) \right\},$$

where $\boldsymbol{\xi}_i \in \partial f(\boldsymbol{x}^i_f)$. Furthermore, by (3)

$$\max_{i \in I_f^{k-1}} \left\{ \boldsymbol{\xi}_i^T (\boldsymbol{x}^k_{\text{MILP}} - \boldsymbol{x}^i_f) \right\} = \mu^{k-1}_{\boldsymbol{x}^k_{\text{MILP}}} - f_{r(k-1)} \geq \mu^{k-1} - f_{r(k-1)}.$$

Thus, $\mu^k - f_{r(k)} \geq \mu^{k-1} - f_{r(k-1)}$ for all $k \in \mathbb{N}$. □

By the stopping criterion of Algorithm 3.1, the sequence $(\mu^k - f_{r(k)})$ is bounded above by 0. This implies with Lemma 4.4 that the sequence converges. The following lemma proves that it converges to 0. Denote $I_f := \left\{ i \mid i \in I_f^k \text{ for some } k \right\} = \bigcup_{k=1}^{\infty} I_f^k$.

LEMMA 4.5. *If the algorithm does not stop after a finite number of iterations, then* $\mu^k - f_r \to 0$.

Proof. Since $\varepsilon_g > 0$ the algorithm will find an ε_g-feasible point after a finite number of iterations by Lemma 4.1. Then a new index is added to the set I_f. Since the algorithm does not stop, the sequence $(\boldsymbol{x}^k_{\text{MILP}})_{k \in I_f}$ must be infinite.

By the Bolzano-Weierstrass Theorem, the sequence $(\boldsymbol{x}^k_{\text{MILP}})_{k \in I_f}$ has an accumulation point in the compact set L. Furthermore, there is a convergent subsequence which is a Cauchy sequence. Then, for given $\varepsilon > 0$ there exists $j > i$ such that $i, j \in I_f$ and $\boldsymbol{x}^j_{\text{MILP}} \in B(\boldsymbol{x}^i_{\text{MILP}}; \frac{\varepsilon}{K})$, where K is a Lipschitz constant of f on L. Thus, for any $\boldsymbol{\xi} \in \partial f(\boldsymbol{x}^i_f)$

$$\begin{aligned} \mu^j - f_r &\geq \boldsymbol{\xi}^T (\boldsymbol{x}^j_{\text{MILP}} - \boldsymbol{x}^i_f) = \boldsymbol{\xi}^T (\boldsymbol{x}^j_{\text{MILP}} - \boldsymbol{x}^i_{\text{MILP}}) + \boldsymbol{\xi}^T (\boldsymbol{x}^i_{\text{MILP}} - \boldsymbol{x}^i_f) \\ &\geq -\left\| \boldsymbol{\xi}^T \right\| \left\| \boldsymbol{x}^j_{\text{MILP}} - \boldsymbol{x}^i_{\text{MILP}} \right\| + 0 > -K \frac{\varepsilon}{K} = -\varepsilon, \end{aligned}$$

where inequality $\|\boldsymbol{\xi}\| \leq K$ is obtained from Theorem 2.3 (ii) and inequality $\boldsymbol{\xi}^T (\boldsymbol{x}^i_{\text{MILP}} - \boldsymbol{x}^i_f) \geq 0$ is proved in Lemma 4.2. Hence, the sequence $(\mu^k - f_r)$ has a convergent subsequence which converges to 0. Since the sequence is increasing and bounded above it converges. Thus, $\mu^k - f_r \to 0$. □

THEOREM 4.6. *If* $\mu^k - f_r \to 0$, *then* (f_r) *converges to an* ε_g*-feasible global minimum value.*

Proof. Since f has a lower bound on the compact set L, and (f_r) is decreasing, (f_r) converges to, say, at $\hat{f}$. On the contrary, suppose that this is not an ε_g-feasible global minimum value. Thus, there exists $\hat{\boldsymbol{x}} \in N \cap L \cap Y$ such that $f(\hat{\boldsymbol{x}}) < \hat{f}$. In Lemma 2.8 choose $C = \{\hat{\boldsymbol{x}}\}$, $A = \left\{\boldsymbol{x} \in \mathbb{R}^n \mid f(\boldsymbol{x}) \geq \hat{f}\right\} \cap L$ and $a = \hat{f}$. Then for some $\delta > 0$,

$$\mu^k - f_r \leq \mu^k_{\hat{\boldsymbol{x}}} - f_r \leq \sup_{\substack{\boldsymbol{z} \in A \\ \boldsymbol{\xi} \in \partial f(\boldsymbol{z})}} \left\{\boldsymbol{\xi}^T(\hat{\boldsymbol{x}} - \boldsymbol{z})\right\} = -\delta$$

for all $k \in \mathbb{N}$. This contradicts with the assumption $\mu^k - f_r \to 0$, which proves the theorem. $\square$

Finally, the theorem of convergence, that sums up the previous results, is given.

THEOREM 4.7. *If the nonlinear constraint functions are f°-pseudoconvex, Algorithm 3.1 converges to an ε_g-feasible global minimum value.*

Proof. If $\mu^k \geq f_r$ for some $k \in \mathbb{N}$, then the minimum is obtained by Theorem 4.3. On the other hand, if $\mu^k < f_r$ for all $k \in \mathbb{N}$ then the algorithm does not stop after a finite number of iterations. By Lemma 4.5 $(\mu^k - f_r)$ converges to 0. By Theorem 4.6, this implies that the algorithm converges to an ε_g-feasible global minimum value. $\square$

In Algorithm 3.1 step 4.2 one could also leave out the old linearizations instead of updating them. However, in this case and if f_r is updated infinitely many times we need to additionally require that the solution sequence has an unique accumulation point as in [11]. When the linearizations are updated this is not needed.

Algorithm 3.1 can also solve problems with l-quasiconvex constraint functions if an additional condition holds true. The only proof that considers constraint functions is that of Lemma 4.1. In [12] it was noted that the lemma is true for f°-quasiconvex functions g_m, if $\boldsymbol{0} \notin \partial g_m(\boldsymbol{x})$ when a supporting hyperplane is created from g_m at $\boldsymbol{x}$. This can be proven also for l-quasiconvex functions. The proof goes similarly to that in [12], but we need to reformulate one theorem. This is done in the appendix.
The condition

$$\boldsymbol{0} \notin \partial g_m(\boldsymbol{x}) \text{ for all } \boldsymbol{x} \in L \cap \left\{\boldsymbol{y} \in \mathbb{R}^n \mid g_m(\boldsymbol{y}) = \frac{\varepsilon_g}{2}\right\} \cap \left\{\boldsymbol{y} \in \mathbb{R}^n \mid F(\boldsymbol{y}) = \frac{\varepsilon_g}{2}\right\} \quad (4)$$

for all $m = 1, \ldots, M$ guarantees that l-quasiconvex constraint functions can be used.

We can also consider problems with an l-quasiconvex objective function with the help of Lemmas 2.9 and 2.10. These imply that convergence proofs are valid for the l-quasiconvex objective function if $\boldsymbol{0} \notin \partial f(\boldsymbol{x}^k_f)$ for any point $\boldsymbol{x}^k_f$ where linearization of type (2) is created.

5 Feasibility problem

In this section we consider finding an integer relaxed feasible point for (P) needed in Algorithm 3.1. That is, the point $\boldsymbol{x}_{\text{NLP}} \in N \cap L$. For simplicity, we will write 'feasible' instead of 'integer relaxed feasible' as in the title of current section. Since this is the only type of feasibility we consider, this should not create any confusion. Due to tolerances, we will find only an ε_F-feasible point. In order to guarantee that it is applicable for Algorithm 3.1 the ε_F should be smaller than the given tolerance ε_g for Algorithm 3.1. The integer relaxed feasibility problem of (P) is:

$$\begin{aligned} \min \quad & \mu \\ \text{s.t.} \quad & g_m(\boldsymbol{x}) \leq \mu, \quad \forall\, m = 1, \ldots, M \quad \text{(FP)} \\ & \boldsymbol{x} \in L, \mu \in [\mu_L, \mu_U], \end{aligned}$$

where μ_L and μ_U are given bounds on μ. If the final solution of (FP) results in $\mu > \varepsilon_F$, the problem (P) does not have any ε_F-feasible point. On the other hand, if in solving (FP) we encounter a point $(\boldsymbol{x}, \mu)$ satisfying $F(\boldsymbol{x}) \leq \varepsilon_F$, then (P) has an ε_F-feasible solution. In this case, we may stop solving (FP) and declare the point $\boldsymbol{x}$ to be a feasible point.

Denote $I(\boldsymbol{x}) = \{m \mid g_m(\boldsymbol{x}) = F(\boldsymbol{x})\}$. We obtain the solution of the feasibility problem (FP) by solving a sequence of LP problems

$$\begin{aligned} \min \quad & \mu \\ \text{s.t.} \quad & \boldsymbol{\xi}_i^T(\boldsymbol{x} - \boldsymbol{x}^i) \leq \mu, \quad i < k \quad (\text{LP}_k) \\ & \boldsymbol{x} \in L, \end{aligned}$$

where $\boldsymbol{\xi}_i \in \partial g_{m_i}(\boldsymbol{x}^i)$ is arbitrary and $m_i \in I(\boldsymbol{x}^i)$. Throughout this section $\boldsymbol{\xi}_i$ is the subgradient that was chosen in the ith iteration. The first problem (LP_1) is simply (FP) without the nonlinear constraints. The algorithm to find a feasible point is presented next.

Algorithm 5.1 The feasibility algorithm

Give a tolerance parameter $\varepsilon_F \geq 0$ and set $k = 1$.

1. Solve the problem (LP_k). Denote the solution by $(\boldsymbol{x}^k, \mu^k)$.
2. If $F(\boldsymbol{x}^k) \leq \varepsilon_F$ then stop: $\boldsymbol{x}^k$ is the ε_F-feasible point.
3. Let $m_k \in I(\boldsymbol{x}^k)$ and $\boldsymbol{\xi}_k^T \in \partial g_{m_k}(\boldsymbol{x}^k)$. Create a new problem (LP_{k+1}) by adding the linear constraint $\boldsymbol{\xi}_k^T(\boldsymbol{x} - \boldsymbol{x}^k) \leq \mu$ to the problem (LP_k).
4. Set $k = k + 1$ and go to step 1.

There are three distinct cases of problem types:

1. $F(\boldsymbol{x}) > \varepsilon_F$ for all $\boldsymbol{x} \in L$. The original problem (P) has no ε_F-feasible solution.
2. There does not exist a point $\boldsymbol{x} \in L$ such that $F(\boldsymbol{x}) < \varepsilon_F$, but there exists $\boldsymbol{y} \in L$ such that $F(\boldsymbol{y}) = \varepsilon_F$.

3. There exists $\boldsymbol{x} \in L$ such that $F(\boldsymbol{x}) < \varepsilon_F$.

In the convergence proofs we will assume that $\varepsilon_F = 0$. Then, it is clear that in case 1 Algorithm 5.1 will not stop. In case 2 the algorithm may not stop, but it will converge to a feasible point. In case 3 the algorithm finds a feasible point after a finite number of iterations. Case 3 (with $\varepsilon_F = 0$) can be restated so that the problem (P) satisfies the *Slater constraint qualification*. We continue analysing the cases 2 and 3. Hence, from now on we assume that a feasible point exists.

In the convergence analysis, it is first proved that the optimal values μ^k of (LP_k) are always negative. If the algorithm does not stop after a finite number of iterations, the sequence (μ^k) converges to zero. This implies that any accumulation point of the sequence $(\boldsymbol{x}^k)$ is a feasible point.

Clearly, the sequence (μ^k) is increasing since for the feasible sets Ω_k of problem (LP_k) we have $\Omega_{k+1} \subseteq \Omega_k$ for all $k \in \mathbb{N}$. In a similar manner to equation (3), denote

$$\mu^k_{\overline{\boldsymbol{x}}} = \max_{i<k} \left\{ \boldsymbol{\xi}_i^T (\overline{\boldsymbol{x}} - \boldsymbol{x}^i) \right\},$$

where $\overline{\boldsymbol{x}} \in L$. Then problem (LP_k) can also be written as

$$\begin{array}{ll} \min & \mu^k_{\boldsymbol{x}} \\ \text{s.t.} & \boldsymbol{x} \in L. \end{array}$$

Consequently, $\mu^k_{\boldsymbol{x}} \geq \mu^k$ for any $\boldsymbol{x} \in L$.

The following two lemmas sum up the results needed for the convergence in cases 2 and 3.

LEMMA 5.1. *Consider Algorithm 5.1. We have for all* $k \in \mathbb{N}$

1. $\mu^k < 0$

2. $\mu^k \leq \mu_0$ *for some* $\mu_0 < 0$*, if the Slater constraint qualification holds true.*

Proof. Let $\boldsymbol{x} \in L$ and $F(\boldsymbol{x}) \leq 0$. A linearization in Algorithm 5.1 in step 3 is made from the constraint function g_{m_i} only at $\boldsymbol{x}^i$ where $g_{m_i}(\boldsymbol{x}^i) > 0 \geq g_{m_i}(\boldsymbol{x})$. The f°-pseudoconvexity of the constraint functions implies $\boldsymbol{\xi}_i^T(\boldsymbol{x} - \boldsymbol{x}^i) < 0$ for all $i < k$. Thus,

$$\mu^k \leq \mu^k_{\boldsymbol{x}} = \max_{i<k} \left\{ \boldsymbol{\xi}_i^T (\boldsymbol{x} - \boldsymbol{x}^i) \right\} < 0,$$

proving the first part of the lemma.

Suppose then that there exists $\boldsymbol{x} \in L$ such that $F(\boldsymbol{x}) < 0$. By choosing $a = 0$, $A = \{\boldsymbol{y} \in \mathbb{R}^n \mid g_m(\boldsymbol{y}) \geq 0\} \cap L$ and $C = \{\boldsymbol{x}\}$ in Lemma 2.8 we get for every $m \in \{1, \ldots, M\}$ a constant $\delta_m > 0$ such that

$$\sup_{\substack{\boldsymbol{z} \in A \\ \boldsymbol{\xi} \in \partial g_m(\boldsymbol{z})}} \left\{ \boldsymbol{\xi}^T (\boldsymbol{x} - \boldsymbol{z}) \right\} = -\delta_m.$$

Thus, for any $k \in \mathbb{N}$

$$\mu^k \leq \mu^k_{\boldsymbol{x}} \leq \max_m \{-\delta_m\} < 0$$

and we may choose $\mu_0 = \max_m \{-\delta_m\}$. □

LEMMA 5.2. *If Algorithm 5.1 does not stop after a finite number of iterations then the sequence (μ^k) converges to zero.*

Proof. By Lemma 5.1, we have $\mu^k < 0$ for all $k \in \mathbb{N}$. Thus, (μ^k) has an upper bound 0. Since the sequence (μ^k) is increasing and bounded above, it converges.

The infinite sequence $(\boldsymbol{x}^k)$ has an accumulation point $\hat{\boldsymbol{x}}$ on the compact set L by the Bolzano-Weierstrass Theorem. Let $\varepsilon > 0$ be arbitrary and $\boldsymbol{x}^i, \boldsymbol{x}^j \in B(\hat{\boldsymbol{x}}, \frac{\varepsilon}{2K})$, $j > i$, where K is a Lipschitz constant of F on L. Then by Theorem 2.3 (ii)

$$\mu^j \geq \boldsymbol{\xi}_i^T(\boldsymbol{x}^j - \boldsymbol{x}^i) \geq -\|\boldsymbol{\xi}_i\| \left\|\boldsymbol{x}^j - \boldsymbol{x}^i\right\| \geq -K \cdot 2\frac{\varepsilon}{2K} = -\varepsilon.$$

Hence, (μ^k) converges to zero. □

The proof of convergence of case 2 is given below.

THEOREM 5.3. *Suppose Algorithm 5.1 does not stop after a finite number of iterations. Then any accumulation point of the sequence $(\boldsymbol{x}^k)$ is feasible in the problem (P).*

Proof. First, we prove that the sequence $(F(\boldsymbol{x}^k))$ converges to 0. On the contrary, we suppose there exist $\varepsilon > 0$ and subsequence $(\boldsymbol{x}^{k_j})$ such that for all $j \in \mathbb{N}$ we have $F(\boldsymbol{x}^{k_j}) \geq \varepsilon$. Let $m \in \{1, 2, \ldots, M\}$. By choosing

$$\begin{aligned} A_m &= \{\boldsymbol{x} \in \mathbb{R}^n \mid g_m(\boldsymbol{x}) \geq \varepsilon\} \cap L \subseteq \{\boldsymbol{x} \in \mathbb{R}^n \mid F(\boldsymbol{x}) \geq \varepsilon\} \cap L \text{ and} \\ C &= \left\{\boldsymbol{x} \in \mathbb{R}^n \mid F(\boldsymbol{x}) \leq \frac{\varepsilon}{2}\right\} \cap L \end{aligned}$$

in Lemma 2.8 we obtain

$$\sup_{\substack{\boldsymbol{y} \in A_m \\ \boldsymbol{\xi} \in \partial f(\boldsymbol{y}) \\ \boldsymbol{x} \in C}} \boldsymbol{\xi}^T(\boldsymbol{x} - \boldsymbol{y}) = -\delta_m < 0$$

for some $\delta_m > 0$. Denote $-\delta = \max_m \{-\delta_m\}$. We deduce that for any $j \in \mathbb{N}$ and $\boldsymbol{x} \in C$ inequality $\mu_{\boldsymbol{x}}^{k_j} \leq -\delta$ holds. Hence, $\mu^{k_j} \leq \mu_{\boldsymbol{x}}^{k_j} \leq -\delta$ for all $j \in \mathbb{N}$ contradicting Lemma 5.2.

Let $\overline{\boldsymbol{x}}$ be an accumulation point of the sequence $(\boldsymbol{x}^k)$. Then there exists a subsequence $(\boldsymbol{x}^{k_i})$ such that $\lim_{i\to\infty} \boldsymbol{x}^{k_i} = \overline{\boldsymbol{x}}$. By continuity of F we have $F(\overline{\boldsymbol{x}}) = \lim_{i\to\infty} F(\boldsymbol{x}^{k_i}) = 0$. □

Finally, we give the proof that a feasible point is found in a finite number of steps if the Slater constraint qualification holds true.

THEOREM 5.4. *If the problem (P) satisfies the Slater constraint qualification, Algorithm 5.1 finds a feasible point after a finite number of iterations.*

Proof. Suppose that Algorithm 5.1 does not converge after a finite number of iterations. By Lemma 5.1 there exists $\mu_0 < 0$ such that

$$\mu^k \leq \mu_0 < 0 \text{ for all } k \in \mathbb{N}.$$

This contradicts with Lemma 5.2. □

If the constraint functions are l-quasiconvex we need an additional assumption. The assumption is that

$$\mathbf{0} \notin \partial g_m(\boldsymbol{x}) \quad \text{if } m \in I(\boldsymbol{x}) \text{ and } \boldsymbol{x} \in L \cap \{\boldsymbol{y} \in \mathbb{R}^n \mid g_m(\boldsymbol{y}) \geq 0\} \tag{5}$$

for all $m = 1, \ldots, M$. Note that this is a more strict condition than (4), which was needed to guarantee the global convergence of ESH for the problems with l-quasiconvex constraint functions. When the condition (5) holds, we may use Lemma 2.10 instead of Lemma 2.8 in the previous proofs. Furthermore, Lemma 2.10 is valid with the choices $A = \{\boldsymbol{x}\}$, where $F(\boldsymbol{x}) > 0$, and $B = \{\boldsymbol{y}\}$, where $\boldsymbol{y} \in L \cap \{\boldsymbol{z} \in \mathbb{R}^n \mid F(\boldsymbol{z}) \leq 0\}$. Then, $f^\circ(\boldsymbol{x}; \boldsymbol{y} - \boldsymbol{x}) < 0$ and with this Lemma 5.1 can be proven for l-quasiconvex functions. Hence, we could prove the convergence of Algorithm 5.1 in the same way we did with the f°-pseudoconvex constraint functions.

Another way to deal with convex constraint functions would be to add the cutting plane

$$g(\boldsymbol{x}^k) + \boldsymbol{\xi}_k^T(\boldsymbol{x} - \boldsymbol{x}_k) \leq \mu$$

instead of the linear constraint in step 3 in Algorithm 5.1. However, if there is additionally an f°-pseudoconvex constraint the algorithm may fail to find a feasible point. To show this consider the feasibility problem

$$\begin{array}{ll} \min & \mu \\ \text{s.t.} & |x_1| \leq \mu \\ & \arctan(|x_2| - 2) \leq \mu \\ & -2 \leq x_1 \leq 2, -2 \leq x_2 \leq 2. \end{array}$$

Algorithm 5.1 finds the first point from one of the corners. At this point the convex constraint $|x_1| \leq \mu$ is active and a cutting plane is made from that point. The second point will be on the opposite side in terms of variable x_1. A cutting plane is made from that point. Now the convex constraint is perfectly approximated and μ can not have lower value than 0. The next point can be any point from the line segment $[(0, -2), (0, 2)]$. Suppose the next point is $(0, 2)$. The linearization done from this point is

$$x_2 - 2 \leq \mu.$$

It does not have any effect on subsequent iterations since it allows μ to be 0 at the line segment $[(0, -2), (0, 2)]$. Hence, the next point can be any point from the line segment and the algorithm may form an infinite loop. Notice that this problem does not satisfy the Slater constraint qualification. If it is satisfied, the algorithm will find a feasible point as shown in the appendix.

If all constraint functions are convex the use of cutting planes is equal to solving the feasibility problem with the nonsmooth ECP method [10]. Hence, the algorithm finds a minimizer and it is easy to deduce convergence properties from this fact. If none feasible points exist the algorithm does not stop and it finds $\mu^k > 0$. If the Slater constraint qualification does not hold but a feasible point exists, the algorithm converges to a feasible point. If the Slater constraint qualification holds a feasible point will be found after a finite number of iterations.

6 Numerical examples

In this section, we solve some problems having f°-pseudoconvex objective function with Algorithm 3.1 and the αECP algorithm [11, 32]. We also find a feasible point with Algorithm 5.1 to a facility layout problem. In order to understand the solution approach of the αECP algorithm, we revise briefly its key features.

6.1 On the αECP algorithm

The αECP algorithm takes an f°-pseudoconvex objective function f into account by adding to the MINLP problem (P) the f°-pseudoconvex constraint

$$f(\boldsymbol{x}) - f_r \leq 0 \tag{6}$$

and using linearizations (2). The constraint (6) guarantees that we will eventually, by solving a sequence of MILP subproblems, find a point where a linearization of type (2) can be done. Additional linearizations can be generated at the points that are found through a line search between MILP solutions and the previously defined $\overline{\boldsymbol{x}}_r^p$ (equation (1)). Due to the use of the constraint (6), the line search is optional in αECP, contrary to the case in the ESH algorithm. This gives a certain benefit to αECP, since the objective function may also be restricted to be evaluated at integer points on the integer variables only. On the other hand, if the objective function is allowed to be evaluated at relaxed values of the integer variables, then the line search procedure makes the algorithm more efficient.

When a new upper bound f_r is found, the old constraints of type (2) are omitted and a new one is added. Furthermore, constraint (6) is updated as well as the α-cutting planes (defined below) generated from it.

The f°-pseudoconvex constraint functions are handled by creating α-cutting planes

$$g_m(\boldsymbol{x}_{\text{MILP}}^k) + \alpha_k \cdot \boldsymbol{\xi}^T(\boldsymbol{x} - \boldsymbol{x}_{\text{MILP}}^k) \leq 0,$$

instead of traditional cutting planes. The constant α_k is at first set to 1 and $\boldsymbol{\xi} \in \partial g_m(\boldsymbol{x}_{\text{MILP}}^k)$. An α-cutting plane may cut off parts of the feasible region and this problem is resolved by updating the α_k values. The updating is no longer needed if α_k satisfies inequality

$$\alpha_k \geq \frac{g_m(\boldsymbol{x}_{\text{MILP}}^k)}{\|\boldsymbol{\xi}\|\, \varepsilon_z}, \tag{7}$$

where $\varepsilon_z > 0$ is a user specified parameter. The constants α_k that do not satisfy inequality (7) are multiplied by a factor greater than 1 whenever the feasible region of an MILP subproblem is empty or a feasible solution to the MINLP problem is found. More details on the αECP algorithm can be found in [11, 32].

6.2 Example problems

All of the computational results are performed by the solver described in [34]. The MILP and LP problems are solved by using CPLEX version 12.6.1 (https://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/) with default parameters. Problems are solved by using 64-bit windows 7 computer with Intel i3-2100 3.1GHz processor. In the ESH and αECP algorithms we used the value 10^{-3} for the tolerances ε_g and ε_f if not otherwise stated.

To illustrate the methods, we solve two simple problems. The first problem is

$$\begin{array}{ll} \min & \dfrac{|x_1-3|-10x_1}{3x_1+x_2+1} \\ \text{s.t.} & (x_1-7)^2-5x_2 \le 0 \\ & x_1-1.8x_2 \le 0 \qquad\qquad (P1) \\ & 1 \le x_1, x_2 \le 8, \quad x_2 \in \mathbb{Z}^+. \end{array}$$

This problem was already solved with αECP in [11]. The objective function is f°-pseudoconvex and its subdifferential is

$$\partial f(x_1,x_2) = \begin{cases} \left\{\frac{1}{(3x_1+x_2+1)^2}(-11x_2-20, 11x_1-3)\right\} = \{\boldsymbol{a}_1(x_1,x_2)\} & ,x_1<3 \\ \left\{\frac{1}{(3x_1+x_2+1)^2}(-9x_2, 9x_1+3)\right\} = \{\boldsymbol{a}_2(x_1,x_2)\} & ,x_1>3 \\ \{\lambda\cdot\boldsymbol{a}_1(x_1,x_2)+(1-\lambda)\cdot\boldsymbol{a}_2(x_1,x_2) \mid \lambda\in[0,1]\} & ,x_1=3 \end{cases}.$$

Basically, when $x_1 \neq 3$ the subdifferential consists of the gradient and when $x_1 = 3$ it is the convex combination of limiting gradients as stated in Theorem 2.4. When solving the problem with the algorithms we choose $\lambda = 1$.

For the ESH algorithm we used $\boldsymbol{x}_{\text{NLP}} = (1,8)$. The numbered MILP solutions and the feasible set are illustrated in Figure 1.

Table 1: Information on iterations when solving the first example problem with ESH.

iteration	1	2	3	4	5	6	7
x_1	1.000	1.000	8.000	7.200	3.600	5.400	5.400
x_2	1.000	8.000	5.000	4.000	2.000	3.000	3.000
$f(x_1,x_2)$	-1.600	-0.667	-2.500	-2.549	-2.565	-2.554	-2.554
μ	-100	-100	-6.083	-2.543	-2.557	-2.553	-2.554
f_r	∞	∞	-0.667	-2.500	-2.549	-2.549	-2.554

At the first point, the only nonlinear constraint is violated and a supporting hyperplane is done at it. At points 2,3 and 4 the upper bound f_r is improved and linearizations from the objective function are added to the MILP problem. At the fifth point the constraint is violated again and a supporting hyperplane is done at it. The optimal solution

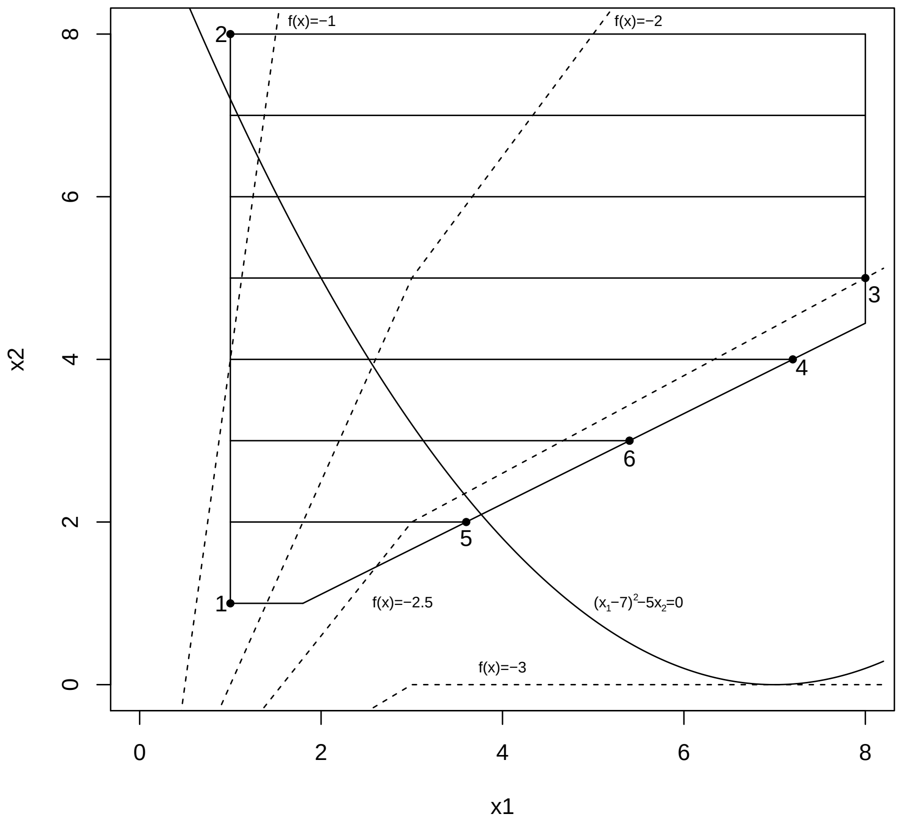

Figure 1: The feasible set of the first example. The dashed lines represent level curves of the objective function. The dots represent MILP solution points when solving the problem by ESH or αECP

is found at the sixth point but the stopping criteria is satisfied first at the seventh iteration. Information on iterations are summarized in Table 1. Note that the line search for the objective function was not needed. Every time a feasible point was found, the objective function attained a new upper bound on it. Note also that the algorithm visits only at points where the nonsmooth objective function is continuously differentiable. Hence, traditional gradients could also have been used in this example.

Surprisingly, the αECP algorithm proceeds exactly as the ESH algorithm as far as MILP solutions are concerned. Note that the nonlinear constraint function is convex and $\alpha = 1$ does not require updating. At the first iteration, the generated cutting plane is the same as the supporting hyperplane. Actually, the constraint function is of the form $f(x_1) - x_2$ where f is convex. Cutting planes generated from this kind of constraint function are also supporting hyperplanes, as will be proven later. The cutting plane will be a supporting hyperplane at the point $(1, 7.2)$. The ESH algorithm creates a supporting hyperplane near this point since the line search is done between $(1, 1)$ and $(1, 8)$. Since a new upper bound is found at each iteration $2 - 4$, αECP proceeds similarly to ESH. At the 5th iteration the cutting plane and the supporting hyperplane are not the same but similar enough to end the algorithms at the same point.

Next we will prove that in a special case a cutting plane is also a supporting hyperplane.

THEOREM 6.1. *Let a constraint function $g : \mathbb{R}^{n+1} \to \mathbb{R}$ be of the form $g(\boldsymbol{x}, y) = f(\boldsymbol{x}) - y$, where $f : \mathbb{R}^n \to \mathbb{R}$ is convex, $\boldsymbol{x} \in \mathbb{R}^n$ and $y \in \mathbb{R}$. Then a cutting plane is a supporting hyperplane to the level set $\{(\boldsymbol{x}, y) \in \mathbb{R}^{n+1} \mid g(\boldsymbol{x}, y) \leq 0\}$.*

Proof. A cutting plane at $(\boldsymbol{x}^1, y^1)$ is

$$f(\boldsymbol{x}^1) - y^1 + (\nabla f(\boldsymbol{x}^1), -1)(\boldsymbol{x} - \boldsymbol{x}^1, y - y^1)^T \leq 0.$$

By rearranging the terms we obtain

$$(\nabla f(\boldsymbol{x}^1), -1)(\boldsymbol{x} - \boldsymbol{x}^1, y - f(\boldsymbol{x}^1))^T \leq 0$$

being a supporting hyperplane to the level set at point $(\boldsymbol{x}^1, f(\boldsymbol{x}^1))$. $\square$

The second illustrative example is selected such that subgradients are needed. The second problem is:

$$\begin{array}{ll} \min & \max\left\{\sqrt{1+|x_1|}, \sqrt{1+|x_2|}\right\} \quad (P2) \\ \text{s.t.} & -5 \leq x_1 \leq 5, -5 \leq x_2 \leq 5. \end{array}$$

The objective function is f°-pseudoconvex and it is not differentiable at lines $|x_1| = |x_2|$. The subdifferential is

$$\partial f(x_1, x_2) = \begin{cases} \left\{\left(\frac{x_1}{2|x_1|\sqrt{1+|x_1|}}, 0\right)\right\} & , |x_1| > |x_2| \\ \left\{\left(0, \frac{x_2}{2|x_2|\sqrt{1+|x_2|}}\right)\right\} & , |x_1| < |x_2| \\ \left\{\left(\frac{\lambda x_1}{2|x_1|\sqrt{1+|x_1|}}, \frac{(1-\lambda) x_2}{2|x_2|\sqrt{1+|x_2|}}\right) \mid \lambda \in [0,1]\right\} & , |x_1| = |x_2| \neq 0 \\ \left\{\left(\frac{\lambda_1 - \lambda_2}{2}, \frac{\lambda_3 - \lambda_4}{2}\right) \mid \sum_{i=1}^4 \lambda_i = 1, \lambda_i \geq 0\right\} & , x_1 = x_2 = 0. \end{cases}$$

If $|x_1| = |x_2| \neq 0$ we choose the subgradient with $\lambda = 0$, that is, the gradient of $\sqrt{1+|x_2|}$. If $x_1 = x_2 = 0$ we choose the subgradient $(0, \frac{1}{2})$. The progression of the ESH algorithm is illustrated in Figure 2. We start with the feasible point $\boldsymbol{x}_{\text{NLP}} = (1, 0)$.

At the first iteration point $(-5, -5)$ a new upper bound $f_r = \sqrt{6}$ is found and the linearization

$$\sqrt{6} + (0, -\frac{1}{2\sqrt{6}})^T((x_1, x_2) - (-5, -5)) \leq \mu \Leftrightarrow -\frac{1}{2\sqrt{6}}x_2 + \frac{7}{12}\sqrt{6} \leq \mu$$

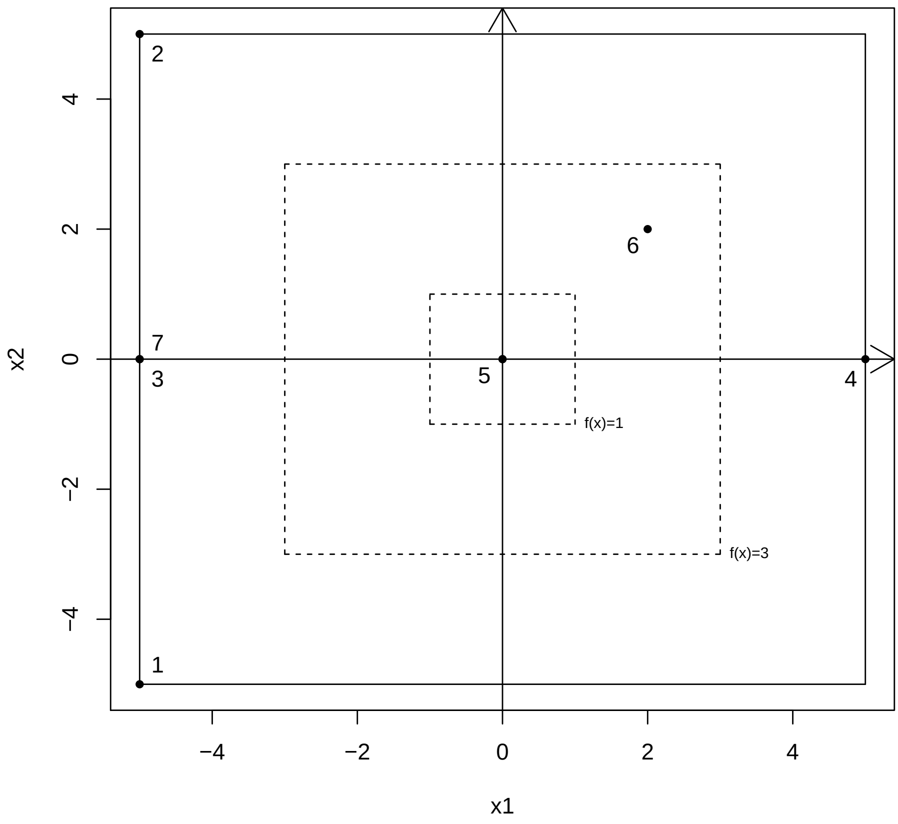

Figure 2: The integer relaxed feasible set of the second example. The dashed lines represent level curves of the objective function. The dots represent MILP solution points when solving the problem by ESH

is added to the MILP subproblem. The next three iteration points $(-5,5),(-5,0)$ and $(5,0)$ will be at the same contour and linearizations will be added from these points. The fifth iteration point $(0,0)$ is the global minimum point, but the algorithm needs to verify it. At that point a new upper bound $f_r = 1$ is found and all of the previous linearizations are updated by adding $1-\sqrt{6}$ on the left hand side. Furthermore, the point $\boldsymbol{x}_{\text{NLP}}^{fr}$ is updated to $(0,0)$. The sixth iteration point is $(2,2)$. Since $f(2,2) = \sqrt{3} > 1$ a line search is done and it ends to a point close to $(0,0)$. A linearization is done there. The seventh iteration point is close to the third point. The value of x_1 does not affect the optimum of MILP and CPLEX chose -5 for x_1. The stopping criteria is satisfied at the seventh iteration and algorithm stops. Some information on iterations are presented in Table 2. Linearizations generated at each iteration are presented in Table 3.

Note that in this problem the solution process is not affected by the given feasible

Table 2: Information on iterations when solving the first example problem with ESH. Observe that the optimal solution is found at iteration 5 and the termination criteria is satisfied at iteration 7.

iteration	1	2	3	4	5	6	7
x_1	-5.000	-5.000	-5.000	5.000	0.000	2.000	-5.000
x_2	-5.000	5.000	0.000	0.000	0.000	2.000	$2.0 \cdot 10^{-8}$
$f(x_1, x_2)$	2.449	2.449	2.449	2.449	1.000	1.732	2.449
μ	-100.0	0.408	1.429	1.429	1.429	0.388	1.000
f_r	∞	2.449	2.449	2.449	2.449	1.000	1.000

Table 3: Linearizations generated by ESH in the second example problem. Linearizations are of the form $\beta_1 \cdot x_1 + \beta_2 \cdot x_2 - \mu \leq \text{rhs}_r$. At the fifth iteration a new upper bound $f_r = 1.0$ is found, r is updated to 3 and the previously generated linearizations are updated.

order	β_1	β_2	rhs_2	rhs_3
1.	0.000	-0.204	-1.429	0.0206
2.	0.000	0.204	-1.429	0.0206
3.	-0.204	0.000	-1.429	0.0206
4.	0.204	0.000	-1.429	0.0206
5.	0.000	-0.500	$-$	-1.000
6.	0.000	-0.500	$-$	-1.000

point $\boldsymbol{x}_{\text{NLP}}$. The first 4 points will be on the same contour and the line search is not needed according to Algorithm 3.1. The fifth point is the global minimum and hence will replace any given feasible point by equation (1). The solution process would be affected only if f_r lower than $\sqrt{6}$ would be given at the start. In which case, the line search for the objective function could be done at the first iteration point.

The solution process of αECP is depicted in Figure 3. The first five points will be the same as with ESH. At the fifth iteration the old linearizations of type (2) is removed and the one generated at $(0, 0)$ is added. At the sixth point $(-5, 5)$ the constraint $f - f_r \leq 0$ is violated and an α-cutting plane is added. An α-cutting plane is also added at iterations 7 and 8. At the ninth iteration the MILP problem is infeasible and coefficients α are updated. This kind of behavior continues, i.e., an α-cutting plane is created every time when the MILP problem is feasible and the coefficients α are updated when it is not. The 17th MILP solution is an ε_g-feasible solution and in subsequent iterations α-coefficients are updated until they satisfy the criterion (7). Points after the 13th iteration are not shown in Figure 3 since they all are close to $(0, 0)$.

The other problems considered are the cyclic scheduling problem from [17] and its modification solved in [11]. Table 4 summarizes some basic properties of the problems.

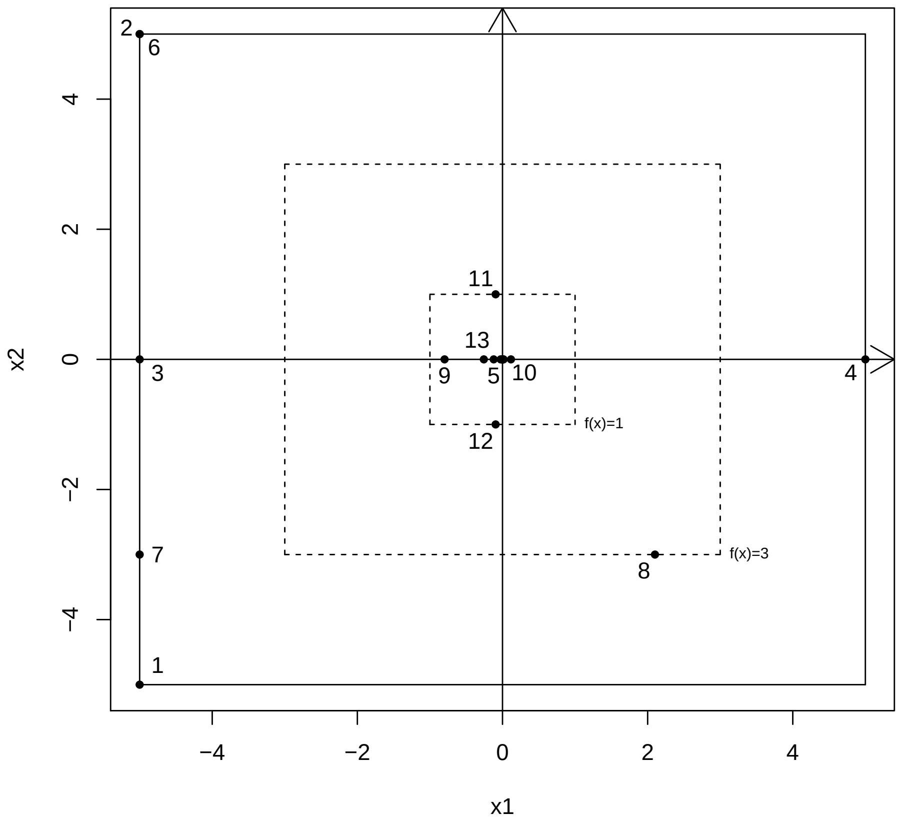

Figure 3: The feasible set of the second example. The dashed lines represent level curves of the objective function. The dots represent MILP solution points when solving the problem by αECP

While the problems P1 and P2 are simple examples with two variables, the problem P3 is a more complicated cyclic scheduling problem [17] with 233 variables and 137 constraints. Problem P4 is otherwise similar to P3, but the objective function is modified to a nonsmooth form. Instead of summing the four pseudoconvex functions as in P3, the maximum of the functions is calculated. This leads to an f°-pseudoconvex function. The magnitude of the objective function in P3 and P4 is 10^4 so $\varepsilon_f = 0.1$ was used instead of 10^{-3}. In P3 and P4 the inner point was found by solving the feasibility problem (FP). Since there are no nonlinear constraints it will, in this case, be the first feasible point of the LP problem. The results are summarized in Table 5.

Algorithm 5.1 to solve the feasibility problem can easily be integrated within the ESH algorithm 3.1. Then also an inner point can initially be solved with the integrated algorithm. This is, in fact, done in the solver [34], where an inner point can be specified to be initially given or solved.

Table 4: Basic information on example problems. Here cont=continuous, int=integers and bin=binary.

problem	objective	constraints		variables		
		linear	convex	cont	int	bin
P1	f°-pseudo	-	-	1	1	-
P2	f°-pseudo	1	1	1	1	-
P3	pseudo	137	-	60	28	145
P4	f°-pseudo	137	-	60	28	145

Table 5: Numerical results. The column "f.eval." takes into account function evaluations, partial derivative evaluations and function evaluations used in the line searches.

problem	method	optimal value	f. eval.	# MILP	CPU-time(s)
P1	ESH	-2.55	59	7	2.76
	αECP	-2.55	28	7	2.20
P2	ESH	1.00	34	7	1.85
	αECP	1.00	125	26	2.97
P3	ESH	-165399	8026	119	47.9
	αECP	-165399	28914	402	63.7
P4	ESH	-39071	10289	152	75.2
	αECP	-39071	33770	463	80.3

An optimum or the best known objective function value was obtained in each case. In problems P2, P3, and P4 the ESH algorithm needed fewer MILP subproblems, fewer function evaluations and spent less time than αECP solving the problem. Note that there were no nonlinear constraints in these problems and these results suggest that the ESH handles the pseudoconvex objective function more effectively than αECP. In the problem P1, αECP was faster and needed fewer function evaluations than ESH. As discussed previously, the algorithms proceeded very similarly in this problem. Hence the ESH is less effective since a few times it needed to use a line search.

We also solved the problems by using the optimal point of the relaxed problem as the inner point. These results are presented in Table 6. The relaxed problems were solved by αECP. Generally, finding the minimum of the relaxed problem is more time consuming than finding a feasible point. In P3 and P4 there are no nonlinear constraints and solving the feasibility problem (FP) takes less than a second. For P3 finding the relaxed minimum takes about 10 seconds, whereas for P4 it takes about 50 seconds.

For large problems it is sometimes beneficial to assign $\boldsymbol{x}^k_{\text{MILP}}$ the first feasible MILP point instead of the optimal MILP point. This may reduce the time needed to solve

MILP problems to the optimum. Eventually, $\boldsymbol{x}^k_{\text{MILP}}$ has to be the optimum of the MILP subproblem to guarantee the optimality of the MINLP problem. Hence, the rule to choose $\boldsymbol{x}^k_{\text{MILP}}$ is updated as the algorithm proceeds. Details on this procedure can be found for example in [32]. Results on testing this strategy ("MIP sol"=1) can also be found in Table 6. Having "MIP sol"=1 resulted in a faster solving time when solving P4 with ESH and P3 with αECP. Otherwise, the changes did not accelerate the solution process.

Table 6: Numerical results on the problems P3 and P4 when trying relaxed optimum as the inner point or "MIP sol =1"-strategy. The column "f.eval." takes into account function evaluations, partial derivative evaluations and function evaluations used in the line searches.

problem	method	optimal value	f. eval.	# MILP	CPU-time(s)
P3	ESH (rel)	-165399	10463	153	81.4
	ESH (MIP=1)	-165399	9896	145	72.5
	αECP (MIP=1)	-165399	20955	289	44.9
P4	ESH (rel)	-39071	12831	191	107
	ESH (MIP=1)	-39071	9338	138	65.4
	αECP (MIP=1)	-39071	39824	548	88.8

Finally, we find a feasible point to a facility layout problem from [6]. In [12] the instance of the problem, that we consider here, was solved by ESH. Here we find a feasible point with Algorithm 5.1 and also the relaxed optimum by αECP. The results are given in Table 7.

Table 7: Numerical results when finding a feasible point of the facility layout problem. The value of the total constraint function F at the final point is in the column F. Interpretation of the other columns are similar to the previous tables.

method	F	f. eval.	# LP	CPU-time(s)
FPA	-5.02	160	10	1.35
αECP	0.00	1468	95	2.66

As expected, it is easier to find a feasible point than the optimal point. Next we solve the facility layout problem with ESH algorithm using both obtained feasible points. The results are in Table 8.

The ESH algorithm performed better when the feasible point found by FPA was used as the inner point. This is surprising, at first, since the minimum of the relaxed problem is presumably close to the true minimum. However, this is not the case with this problem as the minimum value of the relaxed optimum is 0 while the true optimal value is 20.73.

Table 8: Numerical results when solving the facility layout problem. The problem was solved with ESH method and linearizations were done from all possible constraints. The used method to find the feasible point is listed in column ipm. Interpretation of the other columns are similar to the previous tables.

ipm	optimal value	f. eval.	# MILP	CPU-time(s)
FPA	20.73	3490	48	94
αECP	20.73	5063	80	204

7 Conclusions

In this paper, the ESH algorithm ([12, 19, 30]) was generalized to handle MINLP problems with an f°-pseudoconvex objective function. In addition, if the constraint functions of the problem are f°-pseudoconvex the algorithm was shown to converge to an ε_g-global minimum value. The solution procedure was illustrated by solving some numerical examples.

The key technique of this generalization is to use linearizations of type (2). Similar types of linearizations were also used to generalize αECP in order to handle pseudoconvex and f°-pseudoconvex objective functions in [11, 32]. However, in αECP an additional pseudoconvex objective function constraint was also used. In the current paper, it was further shown that a feasibility problem can be solved with similar kinds of linearizations as in the given ESH algorithm. Such an algorithm can be used to find the necessary integer relaxed feasible point needed in the ESH algorithm and, as earlier mentioned, can also be used in an initial step of an integrated algorithm.

Acknowledgements

This research was supported by the grant no. 294002 of the Academy of Finland.

References

[1] Androulakis I., Maranas C., and Floudas C.A.: αBB: A global optimization method for general constrained nonconvex problems. Journal of Global Optimization 7, 337–363 (1995)

[2] Bagirov, A., Mäkelä, M.M., and Karmitsa, N.: Introduction to Nonsmooth Optimization: Theory, Practice and Software. Springer International Publishing, Cham, Heidelberg (2014)

[3] Bonami P., Kilinc M. and Linderoth J.: Algorithms and software for convex mixed-integer nonlinear programs. In: Lee J. and Leyffer. S. (eds.). Mixed Integer Pro-

gramming, The IMA Volumes in Mathematics and Its Applications 154, 1–39 (2012) Springer, New York.

[4] Bussieck M. R. and Vigerske S.: MINLP Solver Software. Wiley encyclopedia of operations research and management science. John Wiley & Sons, Inc. (2011) DOI: 10.1002/9780470400531.eorms0527

[5] Cambini A. and Martein L.: Generalized Convexity and Optimization – Theory and Applications. Lecture Notes in Economics and Mathematical Systems. Springer-Verlag Berlin Heidelberg. (2009)

[6] Castillo, I., Westerlund, J., Emet, S., and 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)

[7] Clarke, F. H.: Optimization and Nonsmooth Analysis.: Wiley-Interscience, New York (1983)

[8] de Oliveira, W.: Regularized optimization methods for convex MINLP problems. TOP 24, 665–692 (2016)

[9] Duran M. A. and Grossmann I. E.: An outer-approximation algorithm for a class of mixed-integer nonlinear programs. Mathematical Programming 36, 307–339 (1986)

[10] Eronen, V-P., Mäkelä, M.M., and Westerlund, T.: On the generalization of ECP and OA methods to nonsmooth MINLP problems. Optimization 63:7, 1057–1073 (2014)

[11] Eronen, V-P., Mäkelä, M.M., and Westerlund, T.: Extended cutting plane method for a class of nonsmooth nonconvex MINLP problems. Optimization 64:3, 641–661 (2015)

[12] Eronen, V-P., Kronqvist, J., Westerlund, T., Mäkelä, M. M., and Karmitsa, N.: Method for solving generalized convex nonsmooth mixed-integer nonlinear programming problems. Journal of Global Optimization, Published online first, doi:10.1007/s10898-017-0528-7

[13] Fletcher, R., and Leyffer, S.: Solving mixed integer nonlinear programs by outer approximation. Mathematical Programming 66, 327–349 (1994)

[14] Fletcher, R., and Leyffer, S.: Numerical experience with lower bounds for MIQP branch-and-bound. SIAM Journal on Optimization 8, 604–616 (1998)

[15] Geoffrion, A. M.: Generalized Benders decomposition. Journal of Optimization Theory and Applications 10, 237–260 (1973)

[16] Grossmann, I. E.: Review of nonlinear mixed-integer and disjunctive programming techniques. Optimization and Engineering 3, 227–252 (2002)

[17] Jain, V., and Grossmann, I. Cyclic scheduling of continuous parallel-process units with decaying performance. AIChE Journal 44, 1623–1636 (1999)

[18] Kelley, J. E.: The cutting plane method for solving convex programs. Journal of SIAM 8, 703–712 (1960)

[19] Kronqvist, J., Lundell, A., and Westerlund, T.: The extended supporting hyperplane algorithm for convex mixed-integer nonlinear programming. Journal of Global Optimization 64, 249–272 (2016)

[20] Leyffer, S.: Integrating SQP and branch-and-bound for mixed integer nonlinear programming. Computational optimization and applications 18, 295–309 (2001)

[21] Lundell, A., Skjäl, A. and Westerlund, T.: A reformulation framework for global optimization. Journal of Global Optimization 57, 115–141 (2013)

[22] Meyer, C. A., and Floudas, C. A.: Convex underestimation of twice continuously differential functions by piecewise quadratic perturbations: Spline αBB underestimators. Journal of Global Optimization 32, 221–258 (2005)

[23] Mäkelä, M. M. and Neittaanmäki, P.: Nonsmooth Optimization: Analysis and Algorithms with Applications to Optimal Control. World Scientific Publishing Co., Singapore (1992)

[24] Nestorov, Y., and Nemirowskii: Interior-Point Polynomial Algorithms in Convex Programming. SIAM Studies in Applied Mathematics 13, Philadelphia (1994)

[25] Pörn, R.: Mixed-Integer Non-Linear Programming: Convexification Techniques and Algorithm Development. Ph.D. Thesis, Åbo Akademi University (2000)

[26] Quesada, I., and Grossmann, I. E.: An LP/NLP based branch-and-bound algorithm for convex MINLP optimization problems. Computers & Chemical Engineering 16, 937–947 (1999)

[27] Roberts, A. W., and Varberg, D. E.: Convex functions. Academic Press, New York and London (1973)

[28] Rockafellar, R.T.: Convex Analysis. Princeton Landmarks in Mathematics and Physics. Princeton University Press (1997)

[29] Ryoo H.S. and Sahinidis N.V.: A branch-and-reduce approach to global optimization. Journal of Global Optimization 8, 107–138 (1996)

[30] Veinott Jr, A.F.: The supporting hyperplane method for unimodal programming. Operations Research 15:1, 147–152 (1967)

[31] Westerlund T., Skrifvars, H., Harjunkoski, I. and Pörn, R. An extended cutting plane method for solving a class of non-convex MINLP problems. Computers & Chemical Engineering 22, 357–365 (1998)

[32] Westerlund, T., and Pörn, R.: Solving pseudo-convex mixed integer optimization problems by cutting plane techniques. Optimization and Engineering 3, 253–280 (2002)

[33] Westerlund, T., and Pettersson F.: An extended cutting plane method for solving convex MINLP problems. Computers & Chemical Engineering Sup. 19, 131–136 (1995)

[34] Westerlund, T.: User's Guide for GAECP, Version 5.537, An Interactive Solver for Generalized Convex MINLP-Problems using Cutting Plane and Supporting Hyperplane techniques. Åbo Akademi University. www.abo.fi/ twesterl/GAECPDocumentation.pdf. (2017)

A A theorem reformulated for l-quasiconvex constraint functions.

The following theorem generalizes Theorem 4 in [12] for l-quasiconvex constraint functions. We assume that the Slater constraint qualification holds true. We consider the case $\varepsilon_g = 0$ implying that the supporting hyperplanes are generated at level curve $\{\boldsymbol{x} \mid F(\boldsymbol{x}) = 0\}$.

THEOREM A.1. *Let the constraint function g be l-quasiconvex and $\mathbf{0} \notin \partial g(\boldsymbol{x}_g^i)$. Then, the supporting hyperplane $\boldsymbol{\xi}_i^T(\boldsymbol{x} - \boldsymbol{x}_g^i) \leq 0$ does not cut off any feasible point.*

Proof. On the contrary, suppose $\boldsymbol{y}$ is feasible and it is cut off. By Theorem 2.3 (iii) $g^\circ(\boldsymbol{x}_g^i; \boldsymbol{y} - \boldsymbol{x}_g^i) > 0$. Then, by Definition 2.7 we have $g(\boldsymbol{y}) \geq g(\boldsymbol{x}_g^i)$. By the feasibility of $\boldsymbol{y}$ we have $g(\boldsymbol{y}) \leq 0$ implying $g(\boldsymbol{y}) = g(\boldsymbol{x}_g^i) = 0$.

Since the Slater constraint qualification holds true, there exists $\boldsymbol{z}$ such that $g(\boldsymbol{z}) < 0 = g(\boldsymbol{y})$. By continuity of g there exists $\varepsilon > 0$ such that $g(\overline{\boldsymbol{z}}) < 0$ for all $\overline{\boldsymbol{z}} \in B(\boldsymbol{z}; \varepsilon)$. Let $A \neq \emptyset$ be an open set such that

$$A \subset \operatorname{conv}\{B(\boldsymbol{z};\varepsilon), \{\boldsymbol{y}\}\} \cap \left\{\boldsymbol{x} \mid \boldsymbol{\xi}_i^T(\boldsymbol{x} - \boldsymbol{x}_g^i) > 0\right\}.$$

Let $\boldsymbol{a} \in A$ be arbitrary. Since $\boldsymbol{a}$ is cut off by the hyperplane we have $g^\circ(\boldsymbol{x}_g^i; \boldsymbol{a} - \boldsymbol{x}_g^i) > 0$ and l-quasiconvexity implies $g(\boldsymbol{a}) \geq 0$. Since $\boldsymbol{a} \in \operatorname{conv}\{B(\boldsymbol{z};\varepsilon), \{\boldsymbol{y}\}\}$, the quasiconvexity of g implies $g(\boldsymbol{a}) \leq 0$. Thus, $g(\boldsymbol{a}) = 0$ for all $\boldsymbol{a} \in A$. Consider the set $B = \operatorname{conv}\left\{A, \left\{\boldsymbol{x}_g^i\right\}\right\} \setminus \left\{\boldsymbol{x}_g^i\right\}$. By similar deductions used for the set A we have that $g(\boldsymbol{b}) = 0$ for all $\boldsymbol{b} \in B$. Furthermore, the set B is open. Hence, $\nabla g(\boldsymbol{b}) = \mathbf{0}$ for all $\boldsymbol{b} \in B$. There exists a sequence $(\boldsymbol{b}_i) \subset B$ such that $\lim_{i\to\infty} \boldsymbol{b}_i = \boldsymbol{x}_g^i$. Then, by Theorem 2.4 we have $\mathbf{0} \in \partial g(\boldsymbol{x}_g^i)$, which contradicts the assumption. □

B Feasibility problem: cutting planes from the convex constraints

Here we prove that if the Slater constraint qualifications holds the feasiblity algorithm finds a feasible point even if cutting planes are created from convex constraint functions. Recall that the problem is

$$\begin{aligned} \min \quad & \mu \\ \text{s.t.} \quad & g_m(\boldsymbol{x}) \leq \mu, \quad \forall\, m = 1,\dots,M \qquad \text{(FP)} \\ & \boldsymbol{x} \in L, \mu \in [\mu_L, \mu_U], \end{aligned}$$

We assume that the problem has both convex and f°-pseudoconvex constraint functions. Denote index sets

$$M_P = \{m \in \mathbb{N} \mid g_m \text{ is not convex but } f^\circ - \text{pseudoconvex}\}$$

and

$$\begin{aligned} K_C &= \left\{k \in \mathbb{N} \mid k < K, \text{ the most violating constraint at } \boldsymbol{x}^k \text{ is convex}\right\} \\ K_P &= \{k \in \mathbb{N} \mid k < K, k \notin K_C\}. \end{aligned}$$

Thus, if $k \in K_C$ a cutting plane will be added to the LP problem. The problem LP_K is

$$\begin{array}{ll} \min & \mu \\ \text{s.t.} & g_{m_k}(\boldsymbol{x}^k) + \boldsymbol{\xi}_{m_k}^T(\boldsymbol{x} - \boldsymbol{x}^k) \le \mu, \quad k \in K_C \quad (\mathrm{LP}_K) \\ & \boldsymbol{\xi}_{m_k}^T(\boldsymbol{x} - \boldsymbol{x}^k) \le \mu, \quad k \in K_P \\ & \boldsymbol{x} \in L, \end{array}$$

where g_{m_k} is the most violating constraint function at iteration k and $\boldsymbol{\xi}_{m_k} \in \partial g_{m_k}(\boldsymbol{x}^k)$. Clearly, the sequence (μ^k) is increasing as we minimize μ and added linearizations will not add new feasible points to the problem. Denote

$$\mu_{\overline{\boldsymbol{x}}}^K = \max\left\{\max_{k \in K_C}\left\{g_{m_k}(\boldsymbol{x}^k) + \boldsymbol{\xi}_{m_k}^T(\overline{\boldsymbol{x}} - \boldsymbol{x}^k)\right\}, \max_{k \in K_P}\left\{\boldsymbol{\xi}_{m_k}^T(\overline{\boldsymbol{x}} - \boldsymbol{x}^k)\right\}\right\},$$

where $\overline{\boldsymbol{x}} \in L$. Then, the problem ($\mathrm{LP}_K$) can be written

$$\begin{array}{ll} \min & \mu_{\boldsymbol{x}}^K \\ \text{s.t.} & \boldsymbol{x} \in L. \end{array}$$

The convergence proof is quite similar to that in Section 5. The following Lemma corresponds to Lemma 5.1.

LEMMA B.1. *Suppose problem (FP) has a feasible solution and it satisfies the Slater constraint qualification. Then there exists $\mu_0 < 0$ such that $\mu^k \le \mu_0$ for all $k \in \mathbb{N}$.*

Proof. Since the Slater constraint qualification holds true there exists $\boldsymbol{x} \in L$ such that $F(\boldsymbol{x}) < 0$. By convexity we have

$$g_{m_k}(\boldsymbol{x}^k) + \boldsymbol{\xi}_{m_k}^T(\boldsymbol{x} - \boldsymbol{x}^k) \le g_{m_k}(\boldsymbol{x})$$

for any $k \in K_C$. Denote $A_m = L \cap \{\boldsymbol{y} \mid g_m(\boldsymbol{y}) \ge 0\}$. Then, we can write

$$\begin{aligned} \mu_{\boldsymbol{x}}^K &\le \max\left\{\max_{k \in K_C}\{g_{m_k}(\boldsymbol{x})\}, \max_{k \in K_P}\left\{\boldsymbol{\xi}_{m_k}^T(\boldsymbol{x} - \boldsymbol{x}^k)\right\}\right\} \\ &\le \max\left\{F(\boldsymbol{x}), \max_{k \in K_P} \sup_{\substack{\boldsymbol{z} \in A_{m_k} \\ \boldsymbol{\xi} \in \partial g_{m_k}(\boldsymbol{z})}}\left\{\boldsymbol{\xi}^T(\boldsymbol{x} - \boldsymbol{z})\right\}\right\}. \end{aligned}$$

By choosing $a = 0$, $A = A_{m_k}$ and $C = \{\boldsymbol{x}\}$ in Lemma 2.8 we get for every $m_k \in M_P$ a positive $\delta_{m_k} > 0$ such that

$$\sup_{\substack{\boldsymbol{z} \in A_{m_k} \\ \boldsymbol{\xi} \in \partial g_{m_k}(\boldsymbol{z})}}\left\{\boldsymbol{\xi}^T(\boldsymbol{x} - \boldsymbol{z})\right\} < -\delta_{m_k}.$$

Thus,

$$\mu^K \leq \mu_{\boldsymbol{x}}^K \leq \max\left\{F(\boldsymbol{x}), \max_{m\in M_P}\{-\delta_m\}\right\} < 0$$

and we may choose $\mu_0 = \max\{F(\boldsymbol{x}), \max_{m\in M_P}\{-\delta_m\}\}$. □

The convergence proof is given below.

THEOREM B.2. *If problem (FP) satisfies Slater constraint qualification, a feasible point is found after a finite number of iterations.*

Proof. Suppose the algorithm does not converge after a finite number of iterations. By Lemma B.1 there exists $\mu_0 < 0$ such that

$$\mu^k \leq \mu_0 < 0 \text{ for all } k \in \mathbb{N}. \tag{8}$$

Furthermore, the sequence $(\boldsymbol{x}^k)$ has a converging subsequence $(\boldsymbol{x}^{k_l})$ by the Bolzano-Weierstrass Theorem. Let $\boldsymbol{x}^{k_i}, \boldsymbol{x}^{k_j} \in (\boldsymbol{x}^{k_l})$ be such that $j > i$ and $\boldsymbol{x}^{k_j} \in B(\boldsymbol{x}^{k_i}; \frac{-\mu_0}{2K_F})$, where K_F is a Lipschitz constant of F. Let $\boldsymbol{\xi}_{m_i} \in \partial g_{m_i}(\boldsymbol{x}^{k_i})$, where g_{m_i} is active at $\boldsymbol{x}^{k_i}$. Then, independently on whether $i \in K_C$ or $i \in K_P$,

$$\mu^j \geq \boldsymbol{\xi}_{m_i}^T(\boldsymbol{x}^{k_j} - \boldsymbol{x}^{k_i}) \geq -\left\|\boldsymbol{\xi}_{m_i}\right\|\left\|\boldsymbol{x}^{k_j} - \boldsymbol{x}^{k_i}\right\| \geq -K_F\frac{-\mu_0}{2K_F} = \frac{\mu_0}{2} > \mu_0.$$

This contradicts with inequality (8) proving the theorem. □

TURKU

CENTRE *for*

COMPUTER

SCIENCE

Joukahaisenkatu 3-5 A, 20520 TURKU, Finland | www.tucs.fi

University of Turku

Faculty of Mathematics and Natural Sciences

- Department of Information Technology
- Department of Mathematics and Statistics

Turku School of Economics

- Institute of Information Systems Sciences

Åbo Akademi University

- Computer Science
- Computer Engineering

ISBN 978-952-12-3578-8
ISSN 1239-1891