TUCS

Ville-Pekka Eronen | Jan Kronqvist | Tapio Westerlund | Marko M. Mäkelä | Napsu Karmitsa

Extended supporting hyperplane algorithm for generalized convex nonsmooth MINLP problems

Turku Centre *for* Computer Science

TUCS Technical Report
No 1179, April 2017

Extended supporting hyperplane algorithm for generalized convex nonsmooth MINLP problems

Ville-Pekka Eronen
University of Turku, Department of Mathematics and Statistics
FI-20014 Turku, Finland
vpoero@utu.fi

Jan Kronqvist
Åbo Akademi, Optimization and Systems Engineering
FI-20014 Turku, Finland
jkronqvist@abo.fi

Tapio Westerlund
Åbo Akademi, Optimization and Systems Engineering
FI-20014 Turku, Finland
twesterl@abo.fi

Marko M. Mäkelä
University of Turku, Department of Mathematics and Statistics
FI-20014 Turku, Finland
makela@utu.fi

Napsu Karmitsa
University of Turku, Department of Mathematics and Statistics
FI-20014 Turku, Finland
napsu@karmitsa.fi

TUCS Technical Report
No 1179, April 2017

Abstract

In this paper the extended supporting hyperplane algorithm is generalized for a class of nonsmooth mixed-integer nonlinear programming problems. The generalization is to use the subgradients of the Clarke subdifferential instead of gradients. Consequently, all the functions in the problem are assumed to be locally Lipschitz continuous. The algorithm is shown to converge to a global minimizer of the problem if the objective function is convex and the constraint functions are f°-pseudoconvex. Some numerical experiments are done on the parameters of the algorithm. In addition, ESH is compared against αECP.

Keywords: ESH; Nonsmooth MINLP; Convex optimization; Generalized convexity; Clarke generalized derivatives;

TUCS Laboratory
Turku Optimization Group (TOpGroup)

1 Introduction

The extended supporting hyperplane (ESH) algorithm to solve smooth convex MINLP problems was presented in [8]. It is based on the classical supporting hyperplane method derived in [12]. The numerical comparisons in [8] suggest that the ESH algorithm is efficient and on par with the current state-of-art MINLP solvers when solving MINLP problems with smooth convex objective and constraint functions.

Motivated by the promising numerical results, we will generalize the ESH algorithm to cover a certain class of nonsmooth generalized convex MINLP problems. By a convex MINLP problem we mean that the objective function is convex and the feasible set is convex when the integer variables are relaxed to continuous ones. We require that the constraint functions are f°-pseudoconvex or, with an additional assumption, f°-quasiconvex. These function classes are modifications of classical pseudo- and quasiconvexities for locally Lipschitz continuous functions. With these constraint functions the feasible set is convex if the integer variables are relaxed to continuous ones. The generalization follows similar steps to the generalization of the αECP algorithm in [7]. That is, instead of a gradient we will use a subgradient of the Clarke subdifferential. The generalization also implies that the ESH-algorithm presented in [8] is suitable for smooth pseudoconvex constraint functions too. In fact, this was essentially noted in [12], but the term pseudoconvexity was not used.

ESH and αECP methods share many similarities. In fact, supporting hyperplanes were seen as an alternative to cutting planes for the αECP algorithm in [10]. This will essentially lead to the ESH algorithm. Both methods solve a sequence of MILP subproblems. After solving an MILP subproblem they stop or add a linear constraint to the subsequent MILP subproblem. The linear constraints that ESH generates are supporting hyperplanes to the feasible set. Thus, a supporting hyperplane usually creates a tighter overestimate of the feasible set than a cutting plane would create. The downsides are that we have to know an inner point of the integer relaxed feasible set and do an additional line search at each MILP iteration. Furthermore, to find an inner point another solver than ESH is needed. In addition, it is easier to add several cutting planes in certain iterations than to add several supporting hyperplanes. Hence, specially if there are many nonlinear constraint functions αECP may cut off larger part of the infeasible set than ESH.

When dealing with f°-pseudoconvex constraint functions αECP needs to know sufficiently large α-values. Usually, this information is not given and the cutting planes may cut off small parts of the feasible region. This is not the case with the ESH method making it theoretically more sound than αECP. However, αECP can solve problems where it is not possible to evaluate nonlinear functions in points where integer variables are assigned non-integer values. ESH can not solve this kind of problems.

The ESH algorithm is presented briefly in Section 3. In that section we also prove that ESH can be generalized to solve problems with f°-pseudoconvex constraint functions. In section 4 some numerical details and example problems are considered.

2 Preliminaries

In this section we present the generalized convexities we will use as well as needed results on them. First, we give the definition of the generalization of the gradient.

DEFINITION 2.1. [4] 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 $\boldsymbol{x}$.

THEOREM 2.2. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be locally Lipschitz continuous. Then*

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

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

Proof. The proofs can be found in [4]. □

The following definition presents the main function classes we are dealing with.

DEFINITION 2.3. Function $f : \mathbb{R}^n \to \mathbb{R}$ is *f°-pseudoconvex* (*f°-quasiconvex*), if it is locally Lipschitz continuous and for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$

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

Some basic properties of these function classes can be found in e.g. [2], where the following results are presented (pp. 140-166).

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

(i) *If f is convex or pseudoconvex, then it is f°-pseudoconvex.*

(ii) *If f is f°-pseudoconvex, then it is f°-quasiconvex.*

(iii) *If f is f°-pseudoconvex, then $\mathbf{0} \in \partial f(\boldsymbol{x})$ implies that $\boldsymbol{x}$ is a global minimizer of f.*

(iv) *If f is f°-quasiconvex, then it is quasiconvex.*

(v) *If f_i, $i = 1, 2, \ldots, m$ are f°-pseudoconvex (f°-quasiconvex), then $\max_i f_i$ is f°-pseudoconvex (f°-quasiconvex).*

As can be seen from Theorem 2.4 (i) f°-pseudoconvexity is a generalization of the classical pseudoconvexity which necessitates differentiability. By definition, the level sets of a quasiconvex function are convex. Thus, Theorem 2.4 (ii) and (iv) implies that the level sets of f°-pseudoconvex or f°-quasiconvex function are also convex. The following result is a straightforward consequence of Theorem 2.4 (ii) and (iii).

COROLLARY 2.5. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be f°-pseudoconvex and $\boldsymbol{x} \in \mathbb{R}^n$. If there exists $\boldsymbol{y} \in \mathbb{R}^n$ such that $f(\boldsymbol{y}) < f(\boldsymbol{x})$, then f is f°-quasiconvex and $\mathbf{0} \notin \partial f(\boldsymbol{x})$.*

With some assumptions, f°-quasiconvexity implies f°-pseudoconvexity. For the proof we need the following lemma which is also useful in the next section.

LEMMA 2.6. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be an f°-quasiconvex, $\boldsymbol{y} \in \mathbb{R}^n$, $a > f(\boldsymbol{y})$ and $A \subset \mathbb{R}^n$. If $a \leq f(\boldsymbol{x})$ and $\mathbf{0} \notin \partial f(\boldsymbol{x})$ for all $\boldsymbol{x} \in A$, then there exists $r > 0$ such that $\boldsymbol{\xi}^T(\boldsymbol{y} - \boldsymbol{x}) \leq -r\,\|\boldsymbol{\xi}\| < 0$ for all $\boldsymbol{x} \in A$ and $\boldsymbol{\xi} \in \partial f(\boldsymbol{x})$.*

Proof. Since $f(\boldsymbol{y}) < a$ and f is continuous, there exists $r > 0$ such that $f(\boldsymbol{z}) < a$ for all $\boldsymbol{z} \in \mathbb{R}^n$ such that $\|\boldsymbol{z} - \boldsymbol{y}\| \leq r$. Let $\boldsymbol{x} \in A$ and $\boldsymbol{\xi} \in \partial f(\boldsymbol{x})$ be arbitrary. Since $\mathbf{0} \notin \partial f(\boldsymbol{x})$ we may define $\hat{\boldsymbol{y}} = \boldsymbol{y} + \frac{\boldsymbol{\xi}}{\|\boldsymbol{\xi}\|} r$. The f°-quasiconvexity and the inequalities $f(\hat{\boldsymbol{y}}) < a \leq f(\boldsymbol{x})$ imply

$$f^\circ(\boldsymbol{x}; \hat{\boldsymbol{y}} - \boldsymbol{x}) \leq 0. \tag{1}$$

By Theorem 2.2 (ii), inequality (1) implies $\boldsymbol{\xi}^T(\hat{\boldsymbol{y}} - \boldsymbol{x}) \leq 0$. Thus,

$$\boldsymbol{\xi}^T(\boldsymbol{y} - \boldsymbol{x}) = \boldsymbol{\xi}^T(\hat{\boldsymbol{y}} - \frac{\boldsymbol{\xi}}{\|\boldsymbol{\xi}\|} r - \boldsymbol{x}) = -r\,\|\boldsymbol{\xi}\| + \boldsymbol{\xi}^T(\hat{\boldsymbol{y}} - \boldsymbol{x}) \leq -r\,\|\boldsymbol{\xi}\|\,.$$

Since $\mathbf{0} \notin \partial f(\boldsymbol{x})$ we have $-r\,\|\boldsymbol{\xi}\| < 0$ proving the lemma. □

THEOREM 2.7. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be f°-quasiconvex function. If $\mathbf{0} \in \partial f(\boldsymbol{x})$ implies that $\boldsymbol{x}$ is a global minimizer, then f is f°-pseudoconvex.*

Proof. Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ be such that $f(\boldsymbol{y}) < f(\boldsymbol{x})$. By assumption we have $\mathbf{0} \notin \partial f(\boldsymbol{x})$. Choosing $a = f(\boldsymbol{x})$ and $A = \{\boldsymbol{x}\}$, Lemma 2.6 implies

$$f^\circ(\boldsymbol{x}, \boldsymbol{y} - \boldsymbol{x}) = \max_{\boldsymbol{\xi} \in \partial f(\boldsymbol{x})} \boldsymbol{\xi}^T(\boldsymbol{y} - \boldsymbol{x}) < 0.$$

Thus, f is f°-pseudoconvex. □

We will denote the closed line segment $[\boldsymbol{x}, \boldsymbol{y}] = \{\lambda \boldsymbol{x} + (1 - \lambda)\boldsymbol{y} \mid 0 \leq \lambda \leq 1\}$ and the corresponding open line segment by $(\boldsymbol{x}, \boldsymbol{y})$.

3 The ESH algorithm

Theorems for proving the ESH algorithm to converge to a global minimizer in problems with linear objective function and f°-pseudoconvex constraint functions are considered in this section. With an additional assumption, the constraint functions may be f°-quasiconvex. Note that any nonlinear convex objective function can be transformed to a linear objective function and a convex constraint. We begin by reformulating the ESH algorithm from [8] to deal with nonsmooth functions. After that we prove the convergence results for the problems with f°-pseudoconvex constraint functions.

3.1 Convex constraint functions

The considered problem is

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

where $g_m : \mathbb{R}^n \to \mathbb{R}$ are convex, $\boldsymbol{x} \in \mathbb{R}^n$ and the set L defines linear constraints. Integer variables are defined in $Y = \{\boldsymbol{x} \mid \boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{x}_i \in \mathbb{Z} \text{ if } i \in I_{\mathbb{Z}}\}$, where $I_{\mathbb{Z}} \subseteq \{1, 2, \ldots, n\}$. We assume that L is a compact set. Since L is defined by linear constraints it will also be convex. Denote $C_m = \{\boldsymbol{x} \mid g_m(\boldsymbol{x}) \leq 0\}$ and $C = \bigcap_{m=1}^{M} C_m$. Thus, the feasible set is $C \cap L \cap Y$. Denoting $F(\boldsymbol{x}) = \max_m \{g_m(\boldsymbol{x})\}$ we can also write $C = \{\boldsymbol{x} \mid F(\boldsymbol{x}) \leq 0\}$. We denote the indexes of active constraint functions on the set $\{\boldsymbol{x} \mid F(\boldsymbol{x}) = 0\}$ by $I_0(\boldsymbol{x}) = \{m \mid g_m(\boldsymbol{x}) = F(\boldsymbol{x}) = 0\}$. For the moment, we assume that the *Slater constraint qualification* holds true. In other words,

$$\text{there exists a point } \boldsymbol{x} \in L \text{ such that } F(\boldsymbol{x}) < 0. \tag{2}$$

The algorithm will need a point $\boldsymbol{x}_{\text{NLP}}$ such that $F(\boldsymbol{x}_{\text{NLP}}) < 0$. This can be obtained by solving the problem

$$\begin{aligned} \min \quad & F(\boldsymbol{x}) \qquad \text{(NLP)} \\ \text{s.t.} \quad & \boldsymbol{x} \in L. \end{aligned}$$

Note that the problem (NLP) need not to be solved to the global minimum. It suffices to find a point $\boldsymbol{x}$ such that $F(\boldsymbol{x}) < 0$. The idea of the ESH algorithm is to solve a sequence of mixed-integer linear programming problems

$$\begin{aligned} \min \quad & \boldsymbol{c}^T\boldsymbol{x} \\ \text{s.t.} \quad & l_j(\boldsymbol{x}^j) \leq 0 \quad j = 1, 2, \ldots, k-1 \qquad (\text{MILP}_k) \\ & \boldsymbol{x} \in L \cap Y, \end{aligned}$$

where $l_j(\boldsymbol{x}^j) \leq 0$ are supporting hyperplanes generated at the points $\boldsymbol{x}^j$ such that $F(\boldsymbol{x}^j) = 0$. At the first iteration, no such planes exist. Solving (MILP_1) then gives us a solution point $\boldsymbol{x}^1_{\text{MILP}}$. If $F(\boldsymbol{x}^1_{\text{MILP}}) \leq 0$, we have thus found a feasible point to the original problem (P). It will be a global minimizer since it was found through minimization on a set containing the original feasible set.

Suppose then that $F(\boldsymbol{x}^1_{\text{MILP}}) > 0$. The point $\boldsymbol{x}^1$ where the first supporting hyperplane is generated will be found through a line search between points $\boldsymbol{x}_{\text{NLP}}$ and $\boldsymbol{x}^1_{\text{MILP}}$. Since the constraint functions are continuous and $F(\boldsymbol{x}_{\text{NLP}}) < 0 < F(\boldsymbol{x}^1_{\text{MILP}})$, a point $\boldsymbol{x}^1 \in [\boldsymbol{x}_{\text{NLP}}, \boldsymbol{x}^1_{\text{MILP}}]$ such that $F(\boldsymbol{x}^1) = 0$ is guaranteed to exist. Since the constraint functions are convex so is F and $\boldsymbol{x}^1$ will be unique. A new problem (MILP$_2$) will be formed by adding to (MILP$_1$) the supporting hyperplane $l_1(\boldsymbol{x}^1) := \boldsymbol{\xi}^T(\boldsymbol{x} - \boldsymbol{x}^1) \leq 0$, where $\boldsymbol{\xi} \in \partial g_m(\boldsymbol{x}^1)$ for some $m \in I_0(\boldsymbol{x}^1)$. Our algorithm will continue solving the problems (MILP$_k$) accordingly until a stopping criterion is satisfied. The algorithm is as follows:

Algorithm 3.1 The ESH algorithm

Give a tolerance parameter $\varepsilon_g > 0$ and set $k = 1$.

1. Find $\boldsymbol{x}_{\text{NLP}}$ such that $F(\boldsymbol{x}_{\text{NLP}}) < 0$ by solving the problem (NLP).
2. Solve the problem (MILP$_k$). Denote the solution by $\boldsymbol{x}^k_{\text{MILP}}$.
3. If $F(\boldsymbol{x}^k_{\text{MILP}}) \leq \varepsilon_g$ then stop: $\boldsymbol{x}^k_{\text{MILP}}$ is the final solution.
   Otherwise, find $\boldsymbol{x}^k \in \left[\boldsymbol{x}_{\text{NLP}}, \boldsymbol{x}^k_{\text{MILP}}\right]$ such that $F(\boldsymbol{x}^k) = \frac{\varepsilon_g}{2}$ with a line search.
4. Generate (MILP$_{k+1}$) by adding to (MILP$_k$) the supporting hyperplane
   $\boldsymbol{\xi}^T(\boldsymbol{x} - \boldsymbol{x}^k) \leq 0$, where $\boldsymbol{\xi} \in \partial g_m(\boldsymbol{x}^k)$ and $m \in I_0(\boldsymbol{x}^k)$.
5. Set $k = k + 1$ and go to step 2.

Note that if $\varepsilon_g > 0$ then we will make the supporting hyperplanes on $\boldsymbol{x}^k$ such that $F(\boldsymbol{x}^k) = \frac{\varepsilon_g}{2} \neq 0$. This ploy allows us to deal with problems that do not satisfy the Slater constraint qualification. In the interim, we will consider the theoretical case $\varepsilon_g = 0$.

In [8] it was shown that the ESH algorithm will converge to a global minimizer if the constraint functions are convex and continuously differentiable (and $\varepsilon_g = 0$). In the algorithm presented in [8] an LP step was used in order to speed up the algorithm. In this step (MILP$_k$) problem is solved with the integer variables relaxed to continuous ones. This allows faster generation of supporting hyperplanes as LP problems are easier to solve than MILP problems. The LP step will stop after a certain amount of iterations. After the LP step, the algorithm starts solving MILP problems as stated but now there are supporting hyperplanes already in (MILP$_0$) giving an initial approximation of the feasible set. The use of the LP step does not effect on whether the algorithm converges to the global minimum or not.

3.2 f°-pseudoconvex constraint functions

Knowing the basics of the ESH algorithm we are now ready to prove that it can be used successfully for a larger set of problems than those with continuously differentiable and convex objective and constraint functions. We shall first consider the case $\varepsilon_g = 0$ in which case we have to require that the Slater constraint qualification holds true. We shall assume that

i) g_m is f°-quasiconvex for all $m = 1, 2, \ldots, M$

ii) $\mathbf{0} \notin \partial g_m(\boldsymbol{x})$ if $m \in I_0(\boldsymbol{x})$.

These conditions are fulfilled for f°-pseudoconvex constraint functions by Corollary 2.5 and the Slater constraint qualification (2). Since the level sets of f°-quasi- and f°-pseudoconvex functions are convex, we are dealing with a convex MINLP problem: the objective function is convex and the feasible set is a convex set, if the integer variables are relaxed to continuous ones.

The convergence proof proceeds as follows. We will first show that a supporting hyperplane does not cut off any feasible points, but it cuts off the previous solution point of (MILP$_k$) that was not within the set C. In the compact set L, this results in a solution sequence that has an accumulation point. This point will be shown to be feasible in C. Finally, this point proves to be a global minimizer to the problem (P).

We shall begin by proving that a supporting hyperplane does not cut off any feasible point.

THEOREM 3.1. *The supporting hyperplane*

$$\boldsymbol{\xi}^T(\boldsymbol{x} - \boldsymbol{x}^k) \leq 0, \quad \boldsymbol{\xi} \in \partial g_m(\boldsymbol{x}^k), \quad m \in I_0(\boldsymbol{x}^k) \tag{3}$$

does not cut off feasible points.

Proof. It is sufficient to prove that the hyperplane (3) does not cut off any points from the set $C_m \supset C$. Let $\boldsymbol{y} \in C_m$ be arbitrary. Then $g_m(\boldsymbol{y}) \leq 0 = g_m(\boldsymbol{x}^k)$ and the f°-quasiconvexity of g_m implies $g_m^\circ(\boldsymbol{x}^k; \boldsymbol{y} - \boldsymbol{x}^k) \leq 0$. By Theorem 2.2 (ii),

$$\boldsymbol{\xi}^T(\boldsymbol{y} - \boldsymbol{x}^k) \leq \max_{\boldsymbol{\zeta} \in \partial g_m(\boldsymbol{x}^k)} \boldsymbol{\zeta}^T(\boldsymbol{y} - \boldsymbol{x}^k) = g_m^\circ(\boldsymbol{x}^k; \boldsymbol{y} - \boldsymbol{x}^k) \leq 0.$$

Thus, the hyperplane (3) does not cut off $\boldsymbol{y}$ proving the theorem. □

The next theorem shows that, if the current solution $\boldsymbol{x}^k_{\text{MILP}}$ is infeasible, then it will be cut off by the supporting hyperplane generated at $\boldsymbol{x}^k$.

THEOREM 3.2. *Let* $\boldsymbol{\xi} \in \partial g_m(\boldsymbol{x}^k)$, *for any* $m \in I_0(\boldsymbol{x}^k)$. *If* $F(\boldsymbol{x}^k_{MILP}) > 0$, *then* $\boldsymbol{\xi}^T(\boldsymbol{x}^k_{MILP} - \boldsymbol{x}^k) > 0$.

Proof. We may write $\boldsymbol{x}^k = \lambda \boldsymbol{x}_{\text{NLP}} + (1-\lambda)\boldsymbol{x}^k_{\text{MILP}}$, where $\lambda \in [0,1]$. Since $F(\boldsymbol{x}^k_{\text{MILP}}) > 0$ and $F(\boldsymbol{x}_{\text{NLP}}) < 0 = F(\boldsymbol{x}^k)$, we have $\lambda \in (0,1)$. It is then straightforward to show that

$$-\frac{\lambda}{1-\lambda}(\boldsymbol{x}_{\text{NLP}} - \boldsymbol{x}^k) = \boldsymbol{x}^k_{\text{MILP}} - \boldsymbol{x}^k.$$

Since $g_m(\boldsymbol{x}_{\text{NLP}}) < g_m(\boldsymbol{x}^k)$ and $\mathbf{0} \notin \partial g_m(\boldsymbol{x}^k)$ (assumption ii) Lemma 2.6 implies (by choosing $A = \{\boldsymbol{x}^k\}$) that there exists $r > 0$ such that

$$\boldsymbol{\xi}^T(\boldsymbol{x}^k_{\text{MILP}} - \boldsymbol{x}^k) = -\frac{\lambda}{1-\lambda}\boldsymbol{\xi}^T(\boldsymbol{x}_{\text{NLP}} - \boldsymbol{x}^k) \geq -\frac{\lambda}{1-\lambda}(-r\,\|\boldsymbol{\xi}\|) > 0$$

proving the theorem. □

With Theorems 3.1 and 3.2 we may prove the uniqueness of $\boldsymbol{x}^k$.

COROLLARY 3.3. *If $F(\boldsymbol{x}^k_{MILP}) > 0$, then $\boldsymbol{x}^k$ is unique.*

Proof. Suppose there exist $\boldsymbol{y}, \boldsymbol{z} \in (\boldsymbol{x}_{\text{NLP}}, \boldsymbol{x}^k_{\text{MILP}})$ such that $F(\boldsymbol{y}) = F(\boldsymbol{z}) = 0$ and $\boldsymbol{y} \neq \boldsymbol{z}$. Without loss of generality we may assume that $\boldsymbol{z} \in (\boldsymbol{y}, \boldsymbol{x}^k_{\text{MILP}})$. By Theorem 3.1, $\boldsymbol{z}$ is not cut off by the hyperplane generated at $\boldsymbol{y}$. However, the hyperplane will then not cut off either $\boldsymbol{x}^k_{\text{MILP}}$ contradicting Theorem 3.2. □

Suppose that at some iteration k we have $F(\boldsymbol{x}^k_{\text{MILP}}) \leq 0$. By Theorem 3.1, the feasible set of the problem (MILP$_k$) includes C. Thus, $\boldsymbol{x}^k_{\text{MILP}}$ is a global minimizer of the problem (P). On the other hand, if $F(\boldsymbol{x}^k_{\text{MILP}}) > 0$ for all k, Theorem 3.2 implies that the points in the sequence $(\boldsymbol{x}^k_{\text{MILP}})$ are distinct. Since $(\boldsymbol{x}^k_{\text{MILP}}) \subset L$ and L is compact, the sequence has an accumulation point by the Bolzano-Weierstrass Theorem. Hence, there exists a converging subsequence $(\boldsymbol{x}^{k_j}_{\text{MILP}}) \subset (\boldsymbol{x}^k_{\text{MILP}})$.

Next, we will show that the subsequence $(\boldsymbol{x}^{k_j}_{\text{MILP}})$ converges to a feasible point. To prove this we need the following lemma.

LEMMA 3.4. *Let $(\boldsymbol{x}^{k_j}_{MILP})$ be a converging sequence and $\boldsymbol{\xi}_j \in \partial g_{m_j}(\boldsymbol{x}^{k_j})$, $m_j \in I_0(\boldsymbol{x}^{k_j})$. Then*

$$\lim_{j\to\infty} \frac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}^{k_j}_{MILP} - \boldsymbol{x}^{k_j}) = 0.$$

Proof. Let $\varepsilon > 0$ be arbitrary. Choose j such that $\left\|\boldsymbol{x}^{k_{j+1}}_{\text{MILP}} - \boldsymbol{x}^{k_j}_{\text{MILP}}\right\| < \varepsilon$. Then

$$\begin{aligned} &\left|\frac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}^{k_{j+1}}_{\text{MILP}} - \boldsymbol{x}^{k_j}) - \frac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}^{k_j}_{\text{MILP}} - \boldsymbol{x}^{k_j})\right| \\ \leq\ & \frac{\|\boldsymbol{\xi}_j\|}{\|\boldsymbol{\xi}_j\|}\left\|\boldsymbol{x}^{k_{j+1}}_{\text{MILP}} - \boldsymbol{x}^{k_j}_{\text{MILP}}\right\| < \varepsilon. \end{aligned} \tag{4}$$

Since $F(\boldsymbol{x}_{\text{MILP}}^{k_j}) > 0$ Theorem 3.2 implies $0 < \dfrac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}_{\text{MILP}}^{k_j} - \boldsymbol{x}^{k_j})$. Then by the feasibility of $\boldsymbol{x}_{\text{MILP}}^{k_{j+1}}$ in the problem $(\text{MILP}_{k_{j+1}})$

$$\frac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}_{\text{MILP}}^{k_{j+1}} - \boldsymbol{x}^{k_j}) \leq 0 < \frac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}_{\text{MILP}}^{k_j} - \boldsymbol{x}^{k_j})$$

for all j. This and equation (4) implies $\lim_{j\to\infty} \dfrac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}_{\text{MILP}}^{k_j} - \boldsymbol{x}^{k_j}) = 0$. $\square$

THEOREM 3.5. *An accumulation point of the sequence $(\boldsymbol{x}_{MILP}^k)$ is a feasible point.*

Proof. By Lemma 3.4 we may re-index the convergent subsequence in a way that $\dfrac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}_{\text{MILP}}^{k_j} - \boldsymbol{x}^{k_j}) - \frac{1}{j} \leq 0$ for all $j \in \mathbb{N}$, where $\boldsymbol{\xi}_j \in \partial g_{m_j}(\boldsymbol{x}^{k_j})$ and $m_j \in I_0(\boldsymbol{x}^{k_j})$. Furthermore,

$$\begin{aligned}
\frac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}_{\text{MILP}}^{k_j} - \boldsymbol{x}^{k_j}) - \frac{1}{j} &\leq 0 \\
&= \frac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}^{k_j} - \boldsymbol{x}^{k_j}) \\
&= \frac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\lambda_{k_j}\boldsymbol{x}_{\text{NLP}} + (1-\lambda_{k_j})\boldsymbol{x}_{\text{MILP}}^{k_j} - \boldsymbol{x}^{k_j}).
\end{aligned}$$

By rearranging the terms we have

$$\lambda_{k_j}\frac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}_{\text{MILP}}^{k_j} - \boldsymbol{x}^{k_j}) - \frac{1}{j} \leq \lambda_{k_j}\frac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}_{\text{NLP}} - \boldsymbol{x}^{k_j}).$$

By choosing $a = 0$ and $A_m = \{\boldsymbol{x} \in L \mid F(\boldsymbol{x}) = 0 \text{ and } g_m(\boldsymbol{x}) = 0\}$ we have $g_{m_j}(\boldsymbol{x}_{\text{NLP}}) < a$, $\boldsymbol{x}^{k_j} \in A_{m_j}$ and Lemma 2.6 implies

$$\lambda_{k_j}\frac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}_{\text{NLP}} - \boldsymbol{x}^{k_j}) \leq -\lambda_{k_j}\frac{\|\boldsymbol{\xi}_j\|}{\|\boldsymbol{\xi}_j\|}r_{m_j} = -\lambda_{k_j}r_{m_j},$$

for some $r_{m_j} > 0$. Since there are a finite number of constraints

$$-\lambda_{k_j}r_{m_j} \leq -\lambda_{k_j}r,$$

where $r = \min_{m=1,2,\ldots,M}\{r_m\} > 0$. Thus,

$$\lambda_{k_j}\frac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}_{\text{MILP}}^{k_j} - \boldsymbol{x}^{k_j}) - \frac{1}{j} \leq -\lambda_{k_j}r.$$

By solving λ_{k_j} from this inequality we obtain

$$\lambda_{k_j} \leq \frac{1}{j\left(r + \frac{\boldsymbol{\xi}_j^T}{\|\boldsymbol{\xi}_j\|}(\boldsymbol{x}_{\text{MILP}}^{k_j} - \boldsymbol{x}^{k_j})\right)} < \frac{1}{j \cdot r}.$$

Hence, $\lim_{j\to\infty} \lambda_{k_j} = 0$ implying $F(\hat{\boldsymbol{x}}) = 0$ where $\hat{\boldsymbol{x}} = \lim_{j\to\infty} \boldsymbol{x}_{\text{MILP}}^{k_j}$. □

Finally, the convergence result is given.

THEOREM 3.6. *An accumulation point of the sequence $(\boldsymbol{x}_{MILP}^k)$ is a global minimizer of the problem (P).*

Proof. The proof is similar to Lemma 4 in [8]. □

Consider the case where $\varepsilon_g > 0$. Then, the supporting hyperplanes will be generated at the points $\boldsymbol{x}^k$ such that $F(\boldsymbol{x}^k) = \frac{\varepsilon_g}{2}$. This implies that we do not need the Slater constraint qualification to hold true and it suffices that $F(\boldsymbol{x}_{\text{NLP}}) \leq 0$ as was noticed in [8]. Also, for an f°-quasiconvex constraint function we have to require $\mathbf{0} \notin \partial g_m(\boldsymbol{x})$ if a supporting hyperplane is generated from g_m at $\boldsymbol{x}$. Again, an f°-pseudoconvex constraint function meets this requirement.

When $\varepsilon_g > 0$ Theorem 3.1 is valid. In other results we have to replace 0 by $\frac{\varepsilon_g}{2}$ or ε_g when appropriate. Then it follows that if we do not find a point $\boldsymbol{x}$ such that $F(\boldsymbol{x}) < \frac{\varepsilon_g}{2}$, the sequence of MILP solutions has an accumulation point $\hat{\boldsymbol{x}}$ with $F(\hat{\boldsymbol{x}}) = \frac{\varepsilon_g}{2}$. Thus, we will find $\boldsymbol{x}$ satisfying $F(\boldsymbol{x}) \leq \varepsilon_g$ after a finite number of iterations. The final solution point $\boldsymbol{x}_{\text{MILP}}^k$ will be an ε_g-feasible global minimizer, that is, $g(\boldsymbol{x}_{\text{MILP}}^k) \leq \varepsilon_g$ and there does not exist any feasible point that gives lower value to the linear objective function than the current one. If a convex objective function is transformed to a linear objective function, there might be a feasible solution giving a lower objective function value than the one found. Nevertheless, this difference is at most ε_g.

3.3 On solving the NLP problem

To be able to solve a problem the ESH algorithm requires a point $\boldsymbol{x}_{\text{NLP}}$ satisfying all the nonlinear constraints. As previously mentioned, this can be found by solving (NLP). The compactness of L and the continuity of F guarantee that a solution exists. If the functions g_m are f°-pseudoconvex then so is F by Theorem 2.4 (v). In this case the nonsmooth problem (NLP) can be solved by e.g. the proximal bundle (PB) algorithm [11]. If functions are f°-quasiconvex then F will be f°-quasiconvex by Theorem 2.4 (v). The PB algorithm will find a stationary point, which is not guaranteed to solve the problem.

Note that if $\varepsilon_g > 0$ the point $\boldsymbol{x}_{\text{NLP}}$ need only to satisfy $F(\boldsymbol{x}_{\text{NLP}}) \leq 0$. Thus, by relaxing the integer variables of the original problem we obtain a continuous problem, whose global minimum point can be set to $\boldsymbol{x}_{\text{NLP}}$. Presumably, the objective function will steer the $\boldsymbol{x}_{\text{NLP}}$ to be close to a global minimizer. This can, in some problems, make the supporting hyperplanes to be close to the optimum, which possibly results in a fewer number of MILP problems. The relaxed problem can be solved by e.g. the αECP algorithm.

Typically some nonlinear constraints attain 0 at the relaxed global minimizer. Due to tolerances this could lead to the case $F(\boldsymbol{x}_{\text{NLP}}) > 0$. Thus, to make sure that the line search is successful the relaxed problem should be solved more accurately. More precisely, the parameter ε_g when solving the relaxed problem should be at most half of this parameter when solving the original MINLP problem.

Recall that a convex objective function f may be transformed to a constraint function $f - \mu$, where the auxiliary variable μ is the new linear objective function. The choice of μ_{NLP} affects the values of $f - \mu$ in the line search and, thus, also the frequency of computing supporting hyperplanes from $f - \mu$. The μ_{NLP} can be set to the value that the feasibility problem (NLP) gives. However, the only other constraints than $f - \mu$ that include μ are the user given box constraints. Hence, μ_{NLP} would most probably be the given upper bound being somewhat arbitrary.

The relaxed problem does not even include μ so it must be determined somehow. The choice of μ_{NLP} can be seen as a parameter of the algorithm. The condition

$$f(\boldsymbol{x}_{\text{NLP}}) - \mu_{\text{NLP}} \leq 0 \tag{5}$$

limits the choice. There are at least two systematic ways to determine μ_{NLP}. Consider values

i) $\mu_{\text{NLP}} = f(\boldsymbol{x}_{\text{NLP}})$

ii) $\mu_{\text{NLP}} = f(\boldsymbol{x}_{\text{NLP}}) - F(\boldsymbol{x}_{\text{NLP}})$.

The first one satisfies the inequality (5) as an equality. The second one makes $f - \mu$ as large as F at the inner point.

4 Numerical considerations & examples

In this section we discuss some numerical details, especially, the line search. Furthermore, we solve four problems with ESH and compare ESH to αECP. In each problem we will try out different inner points and subgradients to see what effect these have on solving process. We solve all the problems with tolerance $\varepsilon_g = 10^{-3}$. If it is required to solve an optimization problem to find an inner point it has been done with αECP.

We use the GAECP solver to solve all the problems. The solver includes both the ESH and the αECP algorithms. MILP problems were solved by CPLEX. More details on this solver including the default parameters can be found in [13]. Problems were solved using 64-bit windows 7 computer with Intel i3-2100 3.1GHz processor.

At first we motivate why the line search is made to find $\boldsymbol{x}^k$ that has been shifted $\frac{\varepsilon_g}{2}$ in function space from the set $\{\boldsymbol{x} \mid F(\boldsymbol{x}) = 0\}$. Another possibility could be a shift in the variable space.

4.1 On the line search

In the implementation and numerical examples we will have $\varepsilon_g > 0$. In Algorithm 3.1 we apply a line search to find a point $\boldsymbol{x}^k$ such that $F(\boldsymbol{x}^k) = \frac{\varepsilon_g}{2}$. This implies that we do not need the Slater constraint qualification to hold true, but we will end up to an ε_g-feasible minimizer. Comparing the point $\boldsymbol{x}^k$ to the one obtained by the algorithm with $\varepsilon_g = 0$ we will do a shift $\frac{\varepsilon_g}{2}$ in the function value space towards the point $\boldsymbol{x}^k_{\text{MILP}}$, that is, away from the point $\boldsymbol{x}_{\text{NLP}}$.

Next, we will give an example which shows that if $\varepsilon_g = 0$ or even if we do an $\frac{\varepsilon_g}{2}$ shift in the variable space we would need the Slater constraint qualification to hold true. More precisely, we will do supporting hyperplanes on points $\boldsymbol{x}^k + \delta^k$, where $F(\boldsymbol{x}^k) = 0$ and

$$\delta^k = \frac{\varepsilon_g}{2} \frac{\boldsymbol{x}^k_{\text{MILP}} - \boldsymbol{x}^k}{\left\|\boldsymbol{x}^k_{\text{MILP}} - \boldsymbol{x}^k\right\|}. \tag{6}$$

Consider problem

$$\begin{aligned} \min \quad & -x_1 - x_2 \\ \text{s.t.} \quad & g(\boldsymbol{x}) \le 0 \\ & -2 \le x_1 \le 2,\ -2 \le x_2 \le 2, \end{aligned}$$

where $g(\boldsymbol{x}) = \max\{0, g_1(\boldsymbol{x})\}$ and

$$g_1(\boldsymbol{x}) = \begin{cases} (x_2-1)^2 + x_1^2 - 1, & x_2 > 1 \\ x_1^2 - 1, & -1 \le x_2 \le 1 \\ (x_2+1)^2 + x_1^2 - 1, & x_2 < -1 \end{cases}.$$

The constraint function g is convex but does not satisfy the Slater constraint qualification.

Suppose that we have $\boldsymbol{x}_{\text{NLP}} = (1, 0)$. Note that $(1, 0)$ is on the boundary. When trying to solve this problem with the ESH algorithm with the shifts (6) and $0 < \frac{\varepsilon_g}{2} < 1$, the first solution will be $\boldsymbol{x}^1_{\text{MILP}} = (2, 2)$. The line search will find $\boldsymbol{x}^k = \boldsymbol{x}_{\text{NLP}} = (1, 0)$ and the supporting hyperplane $x_1 \le a$, $(a > 1)$ will be added to the problem. More precisely $a = 1 + \frac{\varepsilon_g}{2\sqrt{5}}$ according to (6), but $a > 1$ is the property we are interested in. The next solution will be $\boldsymbol{x}^2_{\text{MILP}} = (a, 2)$. The solving process generates sequence $(\boldsymbol{x}^k_{\text{MILP}})$ converging to the point $(1, 2)$. However, this point is not even feasible. The ESH algorithm without shift (6) and $\varepsilon_g = 0$ will find $\boldsymbol{x}^2_{\text{MILP}} = (1, 2)$ and get stuck there. This is due to the fact that a supporting hyperplane at $\boldsymbol{x}_{\text{NLP}}$ does not cut off the point $(1, 2)$. Note that the Algorithm 3.1 can solve the problem when $\varepsilon_g > 0$.

In the forthcoming examples we use a line search based on the bisection method. This guarantees that at each iteration of the line search, we find an interval $[\boldsymbol{x}_L, \boldsymbol{x}_U]$ such that $F(\boldsymbol{x}_L) \le \frac{\varepsilon_g}{2}$ and $\frac{\varepsilon_g}{2} < F(\boldsymbol{x}_U) \le F(\boldsymbol{x}^k_{\text{MILP}})$. We will stop the line search when $\boldsymbol{x}$ satisfying $\frac{\varepsilon_g}{4} \le F(\boldsymbol{x}) < \varepsilon_g$ has been found. The supporting hyperplane will be generated at this point.

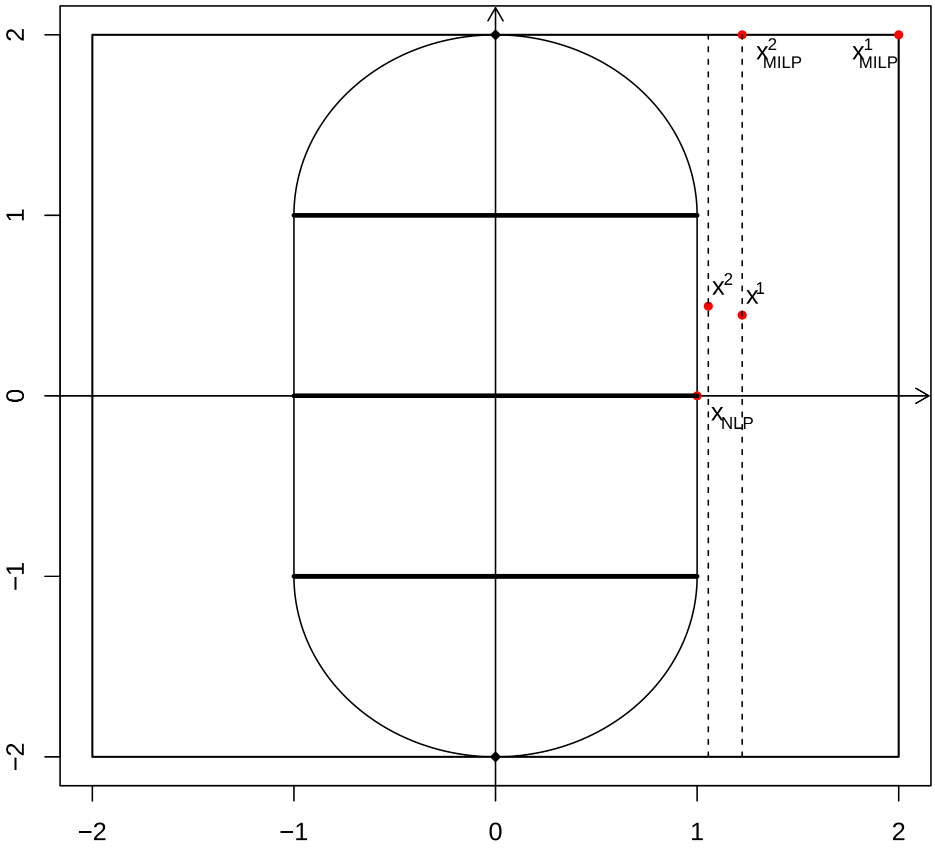

Figure 1: The feasible set and solving process of the given example when $\delta = \frac{1}{2}$. The dashed lines represents the first two hyperplanes and bolded region is the feasible set.

In the line search we do not need to calculate the values of all constraint functions in each step. If $g_m(\boldsymbol{x}^k_{\text{MILP}}) < \varepsilon_g$ then, by quasiconvexity (Theorem 2.4 (iv)), $g_m(\boldsymbol{x}) < \varepsilon_g$ for all $\boldsymbol{x} \in \left[\boldsymbol{x}_{\text{NLP}}, \boldsymbol{x}^k_{\text{MILP}}\right]$. Hence, in the line search on the line segment $\left[\boldsymbol{x}_{\text{NLP}}, \boldsymbol{x}^k_{\text{MILP}}\right]$, it is enough to apply the line search on function

$$F_k(\boldsymbol{x}) = \max_{m \in M_k} g_m(\boldsymbol{x}), \text{ where } M_k = \left\{m \mid g_m(\boldsymbol{x}^k_{\text{MILP}}) \geq \varepsilon_g\right\}.$$

4.2 Example problem 1

The problem 1 is a modification of the problem CB3 from [2, p. 252]. The problem is modified by adding box constraints and by making one variable integer valued. The problem is

$$\begin{array}{ll} \min & \max\left\{x_1^4 + x_2^2, (2-x_1)^2 + (2-x_1)^2, 2e^{x_2 - x_1}\right\} \quad (P1) \\ \text{s.t.} & 0 \le x_1 \le 5, 0 \le x_2 \le 5, x_2 \in \mathbb{Z}. \end{array}$$

The objective function is nonsmooth and convex. The unique optimal point is $(1, 1)$ and the objective function attains value 2 at it. When the objective function is transformed to the constraint $f - \mu \le 0$, a new variable μ is introduced. The upper bound of the objective function is $f(-5, 5) \approx 44\,100$. Thus, we add the box constraint $-50\,000 \le \mu \le 50\,000$ to the transformed problem.

The subgradient of the objective function can be set to be the gradient of an active function. This rule will also be applied in the other problems. The rule does not define a subgradient uniquely at the points where more than one function is active. Then, one of the possible values is used. In this problem we prefer the gradient of the first function over the others and the second function over the third.

First we tried out different inner points. As was discussed in Subsection 3.3, there are two systematic choices of μ_{NLP}. Since there are no nonlinear constraint functions in the problem P1, these two coincides. In addition to this choice we try μ_{NLP} to be the upper bound $50\,000$. The results are given in Table 1.

Table 1: The results on solving the problem P1 with different inner points. Function evaluations include evaluations in the line search and analytic subgradient evaluations.

Ip:(x_1, x_2)	μ_{NLP}	Function eval.	# MILP problems
(1,1)	2	67	4
	50 000	387	14
(0,1)	5.436564	194	13
	50 000	348	13
(1,0)	5	197	13
	50 000	352	13
(1,2)	5.436564	219	15
	50 000	327	12
(2,1)	17	281	17
	50 000	324	12
(0,3)	40.17	208	9
	50 000	323	12
(3,3)	90	351	19
	50 000	267	10
(3,0)	81	353	20
	50 000	285	11
(5,5)	650	461	22
	50 000	327	12

Each run took only a few seconds and the optimum was found every time. Usually, by having $\mu_{\text{NLP}} = 50\,000$ resulted in less or equal number of MILP problems than having $\mu_{\text{NLP}} = f(\boldsymbol{x}_{\text{NLP}})$. There are two exceptions: $\boldsymbol{x}_{\text{NLP}} = (1, 1)$ which is the optimal point and $\boldsymbol{x}_{\text{NLP}} = (0, 3)$. Without the exception $(0, 3)$, it seems that the farther the point $\boldsymbol{x}_{\text{NLP}}$ is from the minimizer the better the choice $\mu_{\text{NLP}} = 50\,000$ is compared to the systematic choice. When the number of MILP problems were roughly the same the choice $\mu_{\text{NLP}} = 50\,000$ resulted in more function evaluations. This is due to the line search. The greater the μ_{NLP} the larger the interval where the line search is done. Hence, more iterations are usually needed to find a point with given accuracy.

In three dimensions it is possible to visualize how μ_{NLP} effects on the supporting hyperplanes. In $x_1x_2\mu$-coordinate system the box constraints represent a rectangular prism. Equation $f(x_1, x_2) - \mu = 0$ represents a surface and the half of the prism where $f(x_1, x_2) - \mu \leq 0$ is the feasible set. Any point there can be an inner point. Suppose we are at one corner which is not feasible. The point, where the line between this corner point and the inner point crosses the surface $f(x_1, x_2) - \mu = 0$, is approximately the point where the supporting hyperplane is created. When the inner point is changed the point where the hyperplane is created is changed too. With this construction in mind the above results sound reasonable. When the inner point is close to a minimizer the linearizations will be close to the optimum as well and we do not need to make many eventually unnecessary linearizations.

In the further considerations we will use both the best inner point $(1, 1)$ and the worst inner point $(5, 5)$. Both of the tried μ_{NLP} values will be used. Next we alter the subgradient. Instead of the previously mentioned rule, we set the subgradient to be the mean of the gradients of the active functions. Since the mean is a convex combination the resulting vector belongs to the subdifferential. In addition to the analytical subgradients, we try vectors generated by the numerical finite difference method with the step size $\delta = 10^{-3}$. This step size will be used in every forthcoming example problem if not told otherwise. It is good to note that the finite difference method may lead to an erroneous subgradient in the case of a nonsmooth function, and thus, does not have any guarantee to be successful.

The choice of a correct analytical subgradient did not have any effect on solving process. With the finite difference some changes are bound to happen. The changes were generally minor. The number of MILP problems and the number of function evaluations were almost unchanged. However, with the inner point $(\boldsymbol{x}_{\text{NLP}}, \mu_{\text{NLP}}) = (1, 1, 2)$ the problem could not be solved with the numerical gradients (with the given step size). The objective function is not differentiable at this point. The line search found a point close to this point and the subgradient was approximated wrongly. This resulted in a supporting hyperplane that is not in accordance with theory. The current MILP solution was not cut off and the algorithm became fixed there. If the step size of the finite difference method is decreased to $\delta = 10^{-4}$ the problem can be solved quite similarly than with the analytical subgradients.

In Section 3 we assumed that each MILP problem is solved to the optimum. Another way is to set $\boldsymbol{x}^k_{\text{MILP}}$ the lth feasible solution when solving MILP problem. This reduces time needed to find $\boldsymbol{x}^k_{\text{MILP}}$ but usually leads to more MILP problems all of which are not solved to the optimum. The last MILP problem must be solved to an optimal point to guarantee the optimality of the original MINLP problem. When the algorithm encounters $\boldsymbol{x}^k_{\text{MILP}}$ that is not an optimal point of the MILP problem but is feasible in the original problem, the value of l is increased by one. This process eventually leads to $\boldsymbol{x}^k_{\text{MILP}}$ being an optimal point in both MILP and MINLP problem. The parameter l is also known as the MIP solution limit. We tried out initial values $l = 1, 10, 100$. The results are in Table 2.

Table 2: The results for the problem 1 when altering the "MIP solution limit"-parameter. The column # MILP problems is of form # MILP problems solved to optimum/# MILP problems totally.

Inner point	μ_{NLP}	l	Function eval.	# MILP problems
(1,1)	2	10	67	4/4
	2	1	56	2/5
	50 000	10	387	14/14
	50 000	1	383	5/16
(5,5)	650	10	461	22/22
	650	1	496	16/27
	50 000	10	327	12/12
	50 000	1	376	5/16

Values $l = 10$ and $l = 100$ gave identical solving processes. This is due to the fact that the MILP problems can be solved to the optimum when $l = 10$. When $l = 1$ more MILP problems have to be solved. Solving times were roughly the same.

The best way to solve the problem was to set the inner point to be the optimal point of the (relaxed) problem and give μ_{NLP} such value that $f - \mu = 0$. MIP solution limit could be set to 100.

4.3 Example problem 2

The second problem is another modified problem from [2]. There it is named Wolfe and it can be found on page 257. Again we set the box constraints and define one variable to be integer. The problem is

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) && (P2) \\ \text{s.t.} \quad & -5 \le x_1 \le 5,\ -5 \le x_2 \le 5,\ x_2 \in \mathbb{Z}, \end{aligned}$$

where

$$f(\boldsymbol{x}) = \begin{cases} 5\sqrt{9x_1^2 + 16x_2^2} & x_1 \ge |x_2| \\ 9x_1 + 16|x_2| & 0 < x_1 \le |x_2| \\ 9x_1 + 16|x_2| - x_1^9 & x_1 \le 0 \end{cases}.$$

The objective function is convex and the unique optimal point is $(-1, 0)$. The objective function value at that point is -8. The upper bound for the objective function is $f(-5, 5) = 1\,953\,160$ and bounds $-2\,000\,000 \leq \mu \leq 2\,000\,000$ are added to the transformed problem. First we tried out different inner points. The results are given in Table 3.

Table 3: The results on solving the problem P2 when trying different inner points.

Ip:(x_1, x_2)	μ_{NLP}	Function eval.	# MILP problems
(-1,0)	-8	58	3
	2 000 000	526	16
(0,0)	0	281	17
	2 000 000	379	12
(-2,0)	494	615	28
	2 000 000	582	18
(-1,1)	8	71	4
	2 000 000	522	16
(-1,-1)	8	62	4
	2 000 000	521	16
(-5,-5)	1 953 160	1058	36
	2 000 000	720	22
(-5,5)	1 953 160	1058	36
	2 000 000	723	22
(5,-5)	125	228	12
	2 000 000	551	17
(5,5)	125	141	7
	2 000 000	552	17

The optimum was reached every time. Here the best inner point was $(-1, 0)$ with $\mu_{\text{NLP}} = f(\boldsymbol{x}_{\text{NLP}}) = -8$. The point $(-1, 0)$ is the optimal point of both the original and the relaxed problem. Otherwise the results were mixed. Sometimes when μ_{NLP} is at the upper limit less MILP problems need to be solved than when $\mu_{\text{NLP}} = f(\boldsymbol{x}_{\text{NLP}})$. We will investigate further the best and the worst case, that is, $(-1, 0)$ and $(-5, -5)$. With each of these points both choices of μ_{NLP} will be tested.

Next we tried out different subgradients. We took the extremal value and the middle point of the subdifferential as a subgradient exactly like in the previous example problem. It turns out that the choice of a correct analytical subgradient did not have any effect. This is natural as $x_1 < 0$ for all points where supporting hyperplanes were created. At these points the function is continuously differentiable. We also tried to calculate the subgradients with the numerical finite difference. As with the previous example, the differences between results when using numerical and analytical subgradients were usually minor.

The finite difference did not find the optimum when the inner point was the optimal point and $\mu_{\text{NLP}} = f(\boldsymbol{x}_{\text{NLP}})$. The reason for this is that the line search finds a point close to the optimal one and the step size in the finite difference is larger than the distance between these points. Due to this the approximated partial derivative $\frac{\partial f}{\partial x_1}$ has different sign than the correct value. Therefore, the supporting hyperplane does not cut off the previous MILP point and the algorithm gets stuck. This problem could be avoided by decreasing the stepsize to $\delta = 10^{-5}$. It is interesting to notice that the bad behaviour of the finite difference was not due to nonsmoothness: the objective function is differentiable at $(-1, 0)$. The results on altering the MIP solution limit parameter were similar to those with the problem P1.

The best way to solve this problem was to use the inner point obtained from the relaxed problem and set $\mu_{\text{NLP}} = f(\boldsymbol{x}_{\text{NLP}})$. This result was also obtained in the previous problem probably due to the problems being quite similar. Neither of the problems have nonlinear constraints and the objective functions are convex. Furthermore, in the both cases the solution of the relaxed problem is also the solution of the original problem.

4.4 Example problem 3

The third example problem is a modified version of the problem 1 in [14]. It is

$$
\begin{aligned}
\min \quad & \max\left\{(x_1-2)^2, (x_2-4)^2\right\} \\
\text{s.t.} \quad & \frac{|x_1-3| - 10x_1}{3x_1 + x_2 + 1} + 2 \le 0 \\
& (x_1-7)^2 - 5x_2 \le 0 \\
& 1 \le x_1 \le 8 \\
& 1 \le x_2 \le 8, \quad x_2 \in \mathbb{Z}.
\end{aligned} \tag{$P3$}
$$

The objective function is convex and nonsmooth. The first constraint function is f°-pseudoconvex and the second constraint function is convex. The unique optimal point is $(2.6, 4)$ giving value 0.36 to the objective function. Bounds $-100 \le \mu \le 100$ is added to the problem where the objective function is transformed to a constraint function.

Again we tried out some different inner points. The results are presented in Table 4. The first inner point corresponds to the optimum point of the relaxed problem. Since there are other constraints than the objective function constraint we may have $f(\boldsymbol{x}_{\text{MILP}}) \neq f(\boldsymbol{x}_{\text{MILP}}) - F(\boldsymbol{x}_{\text{MILP}})$. Thus, we may try three different μ_{MILP} values. The upper bound 100 was used although $(8-2)^2 = 36$ would have been sufficient.

Again, the optimum was found every time. When comparing different values of μ_{NLP}, we can see from Table 4 that using μ_{NLP} such that $f(\boldsymbol{x}_{\text{NLP}}) = \mu_{\text{NLP}}$ resulted in the largest number of MILP problems with the exception of $\boldsymbol{x}_{\text{NLP}}$ being the optimal point of the relaxed problem. By using simply $\mu_{\text{NLP}} = 100$ usually needed the least number of MILP problems. It is also interesting to note that the supporting hyperplanes were done from the original constraints only 2-3 times totally in the problem with each of the inner points. The other hyperplanes were created from the objective function constraint.

Table 4: The results when solving the problem P3 with different inner points. The first μ_{NLP} corresponds to $f(\boldsymbol{x}_{\text{NLP}})$, the second is $f(\boldsymbol{x}_{\text{NLP}}) - F(\boldsymbol{x}_{\text{NLP}})$ and the third one is the upper bound. The row for the second value is omitted if it coincides with the first one.

Ip:(x_1, x_2)	μ_{NLP}	Function eval.	# MILP problems
(2.57,3.925)	0.325	97	4
	100	108	6
(3,4)	1	163	10
	1.143	96	6
	100	86	5
(4,2.5)	4	207	12
	4.516	142	8
	100	92	5
(6,8)	16	317	16
	16.111	244	12
	100	141	7
(5,5)	9	273	14
	9.286	205	10
	100	148	7
($\frac{13}{3}$,7)	9	237	13
	100	113	6

The best and the worst inner points $(2.57, 3.925), (6, 8)$ will be studied further with all tested μ_{NLP} values.

The choice of different analytic subgradients did not affect the solving process. When calculating subgradients with the finite difference method and step size $\delta = 10^{-3}$ the algorithm could not solve the problem with the inner point $(6, 8, 16)$. The reason is that the objective function is not differentiable at that point. The first line search will ended up close to this point and calculated a wrong partial derivative due to the step size. The problem can be solved by simply setting $\delta = 10^{-5}$. Otherwise, the results were quite similar to those obtained by using analytic subgradients.

Since there are more than one constraint, we may try to add more than one supporting hyperplane at $\boldsymbol{x}^k$. This can be done to any constraint function g that satisfies $\frac{\varepsilon}{4} < g(\boldsymbol{x}^k) < \varepsilon$. This did not have any effect on the solution sequences. When the MIP solution limit parameter was altered the results were in line with the previous results. The choice MIP solution limit $= 1$ resulted in more MILP problems than the choice MIP solution limit $= 10$.

Again, the best way to solve the problem was to set the inner point to be the optimum of the relaxed problem and set $\mu_{\text{NLP}} = f(\boldsymbol{x}_{\text{NLP}})$.

4.5 Example problem 4

The fourth problem P4 is the facility layout problem presented in [3] and [9]. There are several formulations of the problem. The version that we solve is presented in the appendix. The objective function is a sum of l_1-norms and it is nonsmooth and convex. In addition, there are 7 pseudoconvex constraints and 114 linear constraints. There are 42 binary variables and 28 continuous variables. The objective function will be transformed to 6 nonsmooth convex constraints. The use of only one objective function constraint would result in a complex constraint function. Since the objective function is a sum of 12 l_1-norms of linear functions, the transformation to only one single constraint would have 2^{12} different gradients on its domain of definition. Thus, to make the perfect approximation of the objective function, 2^{12} linearizations would be needed. With 6 constraints each will have only 4 different gradients. By having 6 objective function constraints we also need 6 auxiliary variables. Lower and upper bounds for these variables are set to 0 and 100, respectively.

It is hard to manually find inner points to this problem since the feasible set is relatively small and there are many variables. We tested 2 different inner points. One is obtained by solving the relaxed problem and the other one by solving the regular feasibility problem. Three different values for μ_{NLP} was tested. At the same time we tested if it has an effect to create all possible supporting hyperplanes.

Table 5: The result when solving the facility layout problem with different inner points and different number of linearisations (# Lin). Column "lin." denotes how many supporting hyperplanes were made.

# Lin.	inner point	μ_{NLP}	F. eval.	# MILP	lin.	CPU (s)
one	relaxed	f	6 991	108	107	226
	relaxed	100	6 555	111	110	117
	regular	f	3 393	57	56	131
	regular	$f-F$	3 570	57	56	127
	regular	100	4 223	58	57	103
all possible	relaxed	f	5 063	80	109	201
	relaxed	100	5 368	90	110	113
	regular	f	2 278	37	56	74
	regular	$f-F$	2 621	40	56	81
	regular	100	3 266	43	57	79

As can be seen from Table 5, using as many linearisations as possible was useful. In the further considerations we will use all possible supporting hyperplanes. The optimal value 20.73 was found every time. In terms of CPU time the best inner point was from the regular feasibility problem with $\mu_{\text{NLP}} = f(\boldsymbol{x}_{\text{NLP}})$ (74s) and the worst was the inner point from relaxed problem with $\mu_{\text{NLP}} = f(\boldsymbol{x}_{\text{NLP}})$ (201s). This result is different from that of the previous problems, where the relaxed optimal point was the best choice. In those problems the relaxed optimal point was close to the actual optimal point. This motivated us to try the inner point that is the correct minimizer. In terms of number of MILP problems this choice was worse than the minimizer of the relaxed problem. However, the solutions were obtained faster, although, not as fast as with the inner point from the regular feasibility problem.

Next we tried out different subgradients for the best and the worst inner points. The nonsmooth functions are of form $|x_{i+1} - x_i|$. When $x_{i+1} = x_i$ the subgradient is not unique. At these points an arbitrary subgradient of this function can be written

$$\lambda(-1, 1) + (1 - \lambda)(1, -1), \text{ where } \lambda \in [0, 1] .$$

This λ was altered to test different subgradients. The objective function is a sum of 24 l_1-norms and we used the same λ for all of these. The results are in Table 6.

Table 6: The results when solving the facility layout problem with different subgradients. The row "num" indicates that subgradients were calculated numerically. In the row "numd" the step size has been decreased to make it possible to solve the problem.

case	λ	F. eval.	# MILP	lin.	CPU (s)
best	1	2 278	37	56	74
	0.75	2 308	37	58	73
	0.5	2 274	38	58	75
	0.25	2 202	35	57	73
	0	2 253	36	56	79
	num	-	-	-	8
	numd	2 193	35	54	45
worst	1	5 063	80	109	201
	num	-	-	-	5
	numd	5 063	80	109	202

Again the choice of the analytical subgradient did not have marked effect. In fact, with the worst parameter combination it did not have any effect. The problem could not be solved if the step size $\delta = 10^{-3}$ was used in the finite difference method. Yet again the reason is that the nonsmooth function is not differentiable at the inner point. The problem can be solved to the optimum when the step size of the finite difference is decreased. The sufficiently small step sizes are $\delta = 10^{-7}$ for the best interior point and $\delta = 10^{-4}$ for the worst interior point.

Finally, we altered the parameter MIP solution limit. The results are in Table 7. In this problem choosing MIP solution limit $= 1$ was useful. Compared to MIP solution limit $= 10$ it resulted in more MILP problems like in the previous examples, but solving times were reduced significantly. The MILP problems were hard and it paid off to not solve every MILP problem to optimum. With the choice 1 the algorithm could solve the problem faster with the worst parameter combination than with the best parameter combination and choice 10.

Table 7: The results when solving the problem P4 with different values of MIP solution limit.

case	MIP. sol.	F. eval.	# MILP	lin.	CPU (s)
best	1	8 432	2/136	160	40
	10	2 524	28/40	65	62
	100	2 202	35/35	57	73
worst	1	10 401	4/197	231	45
	10	5 105	32/81	116	127
	100	5 063	281/281	109	201

4.6 Comparison to αECP

Here we present some theoretical differences between the αECP and ESH algorithms. In addition, we solve the previously solved problems with αECP method and compare it against ESH with best parameter combination. For details on the αECP algorithm we refer to [7, 14].

The principal difference between the αECP and ESH algorithms is the type of cutting planes that algorithms use. ESH creates supporting hyperplanes explained in the previous section while αECP creates α-cutting planes. For the constraint function g_m at the point $\boldsymbol{x}^k_{\text{MILP}}$ the α-cutting plane is

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

where $\boldsymbol{\xi} \in \partial g_m(\boldsymbol{x}^k_{\text{MILP}})$ and $\alpha \geq 1$ is sufficiently large. For a convex function $\alpha = 1$ suffices but for an f°-pseudoconvex function sufficiently large α value is usually not known. In practice α is first set to 1 but it is updated until the distance between the α-cutting plane and $\boldsymbol{x}^k_{\text{MILP}}$ is less than a given parameter $\varepsilon_z > 0$. Nevertheless, for the given ε_z it can not be guaranteed that an updated α-cutting plane does not cut off any feasible point. The ESH does not cut off any points from the feasible set even if the constraint functions are f°-pseudoconvex. This makes ESH more appealing than αECP to solve problems involving f°-pseudoconvex functions. With ESH the nonlinear functions have to be evaluated also at points where integer variables attain non-integer values due to the line search, while with αECP the evaluations are needed only at the points where the integer variables attains integer values.

Presumably, a hyperplane that ESH creates usually cuts off more of the infeasible region than an updated α-cutting plane. This could, in principle, lead to a solution process with less number of MILP problems. If there are more than one constraint function the approximation of the feasible set may be enhanced by adding more than one α-cutting plane or supporting hyperplane per iteration. Naturally, creating more linear constraints leads to larger MILP problems. It is straightforward to create an α-cutting plane for any violating constraint from the solution point $\boldsymbol{x}^k_{\text{MILP}}$. A supporting hyperplane may be created from a constraint function that is active at the point $\boldsymbol{x}^k$ in which the line search ends up to. The quasiconvexity of constraint functions implies that there are at least as many violating constraints at point $\boldsymbol{x}^k_{\text{MILP}}$ as there are active constraints at $\boldsymbol{x}^k$. In other words, if there are many constraint functions it is possible to make at least as many α-cutting planes than supporting hyperplanes per iteration. Thus, the use of more than one cutting plane per iteration benefits more αECP than ESH. Indeed, in the problem P3 ESH could do only one supporting hyperplane per iteration although there were 3 constraints.

For the ESH method, the cost of solving less MILP problems is more function evaluations per iteration due to the line search. This will most probably lead to a greater total number of nonlinear function evaluations. Nevertheless, MILP problems are usually difficult to solve and more time consuming than multiple nonlinear function evaluations. However, this is not always the case. In a chromatographic separation problem [6] some function evaluations require solving partial differential equations and are more time consuming than solving an MILP problem. Another type of time consuming constraint functions are probabilistic constraints which are considered, for example, in [1, 5].

When solving the example problems with αECP all possible linearisations are used and MIP solution limit $= 10$ for all but the fourth problem where it is 1. Otherwise parameters have default values including $\varepsilon_z = 0.1$ and $\varepsilon_g = 0.001$. Subgradients are given analytically in a similar way they were given for ESH.

In addition, we solve different reformulations of the fourth problem. In one of these the pseudoconvex constraints of form $-h_i w_i + a_i < 0$ are transformed to two convex constraints

$$-h_i + \frac{a_i}{w_i} < 0 \text{ and } -w_i + \frac{a_i}{h_i} < 0. \tag{7}$$

Note that the constraints are convex in the feasible set where $w_i, h_i \geq 0$. This problem is denoted P5. Another transformation is to replace the nonsmooth convex constraints

$$|x_{i+1} - x_i| + |y_{i+1} - y_i| - d_{i,i+1} \leq 0,$$

created from the objective function, by linear constraints

$$\begin{aligned} x_{i+1} - x_i &\leq dx_{i,i+1}, \qquad & x_i - x_{i+1} &\leq dx_{i,i+1}, \\ y_{i+1} - y_i &\leq dy_{i,i+1}, \qquad & y_i - y_{i+1} &\leq dy_{i,i+1}. \end{aligned}$$

This problem is denoted P6. Finally, if both of these transformations are done the problem is smooth and convex. We denote this problem P7.

The supposed strength of the ESH method compared to the αECP method is the need to solve less MILP problems due to the tighter overestimate of the feasible set. From Table 8 we see that this was true for the problems P1, P2 and P3. In these problems ESH was also faster than αECP. However, αECP managed to solve all the problems in a fewer number of function evaluations. This holds true even without taking account of the effort needed to solve the NLP problem to find the inner point. The result is not surprising since ESH will do a line search at every iteration resulting in more functions evaluations.

Table 8: Numerical results when solving the example problems with αECP and ESH with the best found parameter combinations.

Problem Name	Method	function evaluations	# MILP	# lin.	CPU time (s)
P1	ESH	67	4/4	3	0.96
	αECP	53	14/14	13	2.9
P2	ESH	58	3/3	2	0.79
	αECP	81	21/21	20	2.4
P3	ESH	97	4/4	3	0.97
	αECP	79	11/11	18	2.7
P4	ESH	8 432	2/136	160	40
	αECP	5 029	2/98	1056	126
P5	ESH	11 617	2/119	178	36
	αECP	2 664	2/39	654	29
P6	ESH	8 235	1/146	142	36
	αECP	831	1/56	223	92
P7	ESH	8 405	1/90	153	26
	αECP	1 568	1/40	504	18

In problems P4-P7 ESH created more MILP problems than αECP. However, it managed to solve the problems P4 and P6 faster than αECP. In these problems there are pseudoconvex constraint functions whereas in the problems P5 and P7 there are not. Thus, it seems that αECP struggled with the pseudoconvex constraints. In these problems it took longer time to solve the MILP problems that αECP created. Actually, since MIP solution limit was set to 1 it was hard to find a feasible point for the MILP problems.

αECP handles pseudoconvex constraints by creating a cutting plane that may cut off feasible points. It is updated in later iterations and finally only small parts of the feasible region is cut off if any. The second MILP problem that αECP created in the problems P4 and P6 had no feasible points. Consequently, the subsequent MILP problems may have had a small feasible set. In the problems P5 and P7 the convex constraint (7) are of the form $l(x) + h(y)$, where l is linear. In these type of constraints a cutting plane is already a supporting hyperplane to the feasible set defined by this constraint. Thus, the cutting planes makes tight overestimate of the feasible set without needing the line search.

It should be noted that parameters were not tuned for the problems P5, P6 and P7. When changing MIP solution limit to 100 αECP could solve the Problems P5 and P7 in 17 and 13 seconds respectively. Otherwise, setting MIP solution limit to 1 was better choice. The fastest way to solve the facility layout problem is to solve the formulation in the problem P7 with the αECP method. This is the same formulation that can be found in MINLP Library2 (http://www.gamsworld.org/minlp/minlplib2/html/) as problem fo7.

5 Concluding remarks

In this paper, it was shown that the ESH method can be applied to nonsmooth locally Lipschitz continuous functions. The only change to the original method is to use Clarke subgradients instead of gradients. For a convex objective function and f°-pseudoconvex constraints the algorithm was shown to converge to a global minimizer. This result requires that the Slater constraint qualification holds true. If it does not, we can still solve the problem but the obtained solution might be only ε_g-feasible. If the subdifferentials of the constraint functions do not contain zero at the points where supporting hyperplanes are created, the convergence theorems are also valid for f°-quasiconvex constraint functions.

The ESH algorithm could solve all of the considered example problems to the best known solutions. Comparison to the αECP algorithm showed that even if ESH managed to solve a problem with less number of MILP problems the price we have to pay for it, namely, doing the line searches and solving one NLP problem, may lead to a larger number of nonlinear function evaluations. Also, it is not guaranteed that the ESH algorithm solves a problem with a fewer number of MILP problems but it may still solve the problem faster than αECP. ESH may have en edge over αECP when there are several f°-pseudoconvex constraints. ESH solved most of the problems faster but an appropriate inner point was searched before solving the problems. In this sense the results were skewed to favour the ESH method. However, solving the feasibility problem to find the inner point took merely few seconds in all of the problems.

There does not seem to be a systematic way to find the best inner point easily. The results on the first three problems suggested to solve relaxed problem and use this solution as an inner point. Results on the fourth problem proved this method to be somewhat suboptimal. The choice of a correct analytic subgradient had mostly only minor effect on solving process. However, when the numerical finite difference with step size 10^{-3} was used to calculate subgradients ESH failed surprisingly often. This problem occurs specially if a constraint function is not differentiable at the inner point.

Acknowledgements

This research was supported by the grant no. $289\,500$ of the Academy of Finland. Jan Kronqvist thanks the graduate school in chemical engineering.

References

[1] Arnold, T., Henrion, R., Moller, A., and Vigerske, S.: A mixed-integer stochastic nonlinear optimization problem with joint probabilistic constraints. Pacific Journal

of Optimization 10, 5–20 (2014).

[2] BAGIROV, A., MÄKELÄ, M.M., AND KARMITSA, N. Introduction to Nonsmooth Optimization: Theory, Practice and Software. Springer International Publishing, Cham, Heidelberg (2014).

[3] 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).

[4] CLARKE, F. H. Optimization and Nonsmooth Analysis. Wiley-Interscience, New York (1983).

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

[6] Emet S., and Westerlund, T.: Comparisons of solving a chromatographic separation problem using MINLP methods. Computers & Chemical Engineering 28, 673–682 (2004).

[7] 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).

[8] KRONQVIST, J., LUNDELL, A., AND WESTERLUND, T. The ESH algorithm for convex mixed-integer nonlinear programming. *Journal of Global Optimization* (to appear) doi: 10.1007/s10898-015-0322-3

[9] MELLER, R. D., NARAYANAN, V., AND VANCE, P. H. Optimal facility layout design. *Operations Research Letters 23*, 117-127 (1999).

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

[11] SCHRAMM, H. AND ZOWE, J. A version of the bundle idea for minimizing a nonsmooth function: conceptual idea, convergence analysis, numerical results. *SIAM Journal on Optimization 2:1*, 121-152 (1992).

[12] VEINOTT JR, A.F. The supporting hyperplane method for unimodal programming. *Operations Research 15:1*, 147-152 (1967).

[13] WESTERLUND, T. User's Guide for GAECP, Version 5.533, An Interactive Solver for Generalized Convex MINLP-Problems Using Cutting Plane and Supporting Hyperplane Techniques. Technical Report, Åbo Akademi University (2016). URL: http://www.abo.fi/ twesterl/GAECPDocumentation.pdf

[14] WESTERLUND, T., AND PÖRN, R. Solving pseudo-convex mixed integer optimization problems by cutting plane techniques. *Optimization and Engineering 3*, 253-280 (2002).

A The facility layout problem: P3

In this problem, 7 departments should be placed in a facility. The width and height of the facility are $w_F = 8.54$ and $h_F = 13$ respectively. Decision variables, indices and the problem are:
$i, j =$ index of a department: $1, 2, \dots, 7$.
x_i, y_i= coordinates of the center of department i.
w_i = width of department i.
h_i = height of department i.
X_{ij}, Y_{ij} = auxiliary variables.

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{6} |x_i - x_{i+1}| + |y_i - y_{i+1}| && (8)\\
\text{s.t.} \quad & h_i w_i \geq a_i,\ i = 1, 2, \dots, 7 && (9)\\
& x_i + \frac{1}{2} w_i \leq w_F,\ i = 1, 2, \dots, 7 && (10)\\
& -x_i + \frac{1}{2} w_i \leq 0,\ i = 1, 2, \dots, 7 && (11)\\
& y_i + \frac{1}{2} h_i \leq h_F,\ i = 1, 2, \dots, 7 && (12)\\
& -y_i + \frac{1}{2} h_i \leq 0,\ i = 1, 2, \dots, 7 && (13)\\
& \frac{1}{2}(w_i + w_j) - (x_i - x_j) \leq w_F(X_{ij} + Y_{ij}),\ 1 \leq i < j \leq 7 && (14)\\
& \frac{1}{2}(w_i + w_j) - (x_j - x_i) \leq w_F(1 + X_{ij} - Y_{ij}),\ 1 \leq i < j \leq 7 && (15)\\
& \frac{1}{2}(h_i + h_j) - (y_i - y_j) \leq h_F(1 - X_{ij} + Y_{ij}),\ 1 \leq i < j \leq 7 && (16)\\
& \frac{1}{2}(h_i + h_j) - (y_j - y_i) \leq h_F(2 - X_{ij} - Y_{ij}),\ 1 \leq i < j \leq 7 && (17)\\
& x_1 - x_2 \leq 0 && (18)\\
& y_1 - y_2 \leq 0 && (19)\\
& w_i^{low} \leq w_i \leq w_i^{up},\ i = 1, 2, \dots, 7 && (20)\\
& h_i^{low} \leq h_i \leq h_i^{up},\ i = 1, 2, \dots, 7 && (21)\\
& X_{ij} \in \{0, 1\},\ 1 \leq i < j \leq 7 && (22)\\
& Y_{ij} \in \{0, 1\},\ 1 \leq i < j \leq 7 && (23)
\end{aligned}
$$

Constraints (8) define the minimum area of the departments. Constraints (9)-(12) make sure the departments are located inside the facility. Constraints (13)-(16) prevent overlapping of the departments. Constraints (17)-(18) erase symmetric solutions. The following parameters were used.

Table 9: Parameters of problem P3

i	1	2	3	4	5	6	7
a_i	16	16	16	36	9	9	9
w_i^{low}	2	2	2	3	1.5	1.5	1.5
w_i^{up}	8	8	8	8.54	6	6	6
h_i^{low}	2	2	2	4.2155	1.5	1.5	1.5
h_i^{up}	8	8	8	12	6	6	6

The feasible set of the problem is quite small. The total area of the facility is

$$w_F \cdot h_F = 8.54 \cdot 13 = 111.02.$$

The sum of the areas of the departments is at least $\sum_{i=1}^{7} a_i = 111$. Since the departments do not overlap the pseudoconvex constraints (7) are almost active at any feasible point.

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 978-952-12-3537-5
ISSN 1239-1891