An extended supporting hyperplane algorithm for convex MINLP problems

Jan Kronqvist, Andreas Lundell and Tapio Westerlund

Center of Excellence in
Optimization and Systems Engineering
Åbo Akademi University, Finland

OSE annual seminar
Turku
November 14, 2014

Åbo Akademi University

ose
OPTIMIZATION AND
SYSTEMS ENGINEERING

Contents of the talk

- The extended cutting plane (ECP) algorithm is briefly introduced.

Contents of the talk

- The extended cutting plane (ECP) algorithm is briefly introduced.
- The extended supporting hyperplane (ESH) algorithm, a new algorithm for solving convex MINLP problems to global optimality, is introduced
  - Cutting planes are replaced with supporting hyperplanes using a line search procedure.
  - Two LP preprocessing steps are utilized to quickly get a tight linear relaxation of the part of the feasible region defined by the convex/quasiconvex constraints.

The ECP algorithm

The extended cutting plane algorithm

- The ECP algorithm is a solver for generally convex mixed-integer nonlinear programming (MINLP) problems.

The extended cutting plane algorithm

- The ECP algorithm is a solver for generally convex mixed-integer nonlinear programming (MINLP) problems.
- Solves MILP relaxations of the MINLP problem where the nonlinear constraints are approximated using cutting planes.

The extended cutting plane algorithm

- The ECP algorithm is a solver for generally convex mixed-integer nonlinear programming (MINLP) problems.
- Solves MILP relaxations of the MINLP problem where the nonlinear constraints are approximated using cutting planes.
- Implemented, *e.g.,* in the AlphaECP solver in GAMS and available on the NEOS server.

The extended cutting plane algorithm

- First presented in internal report in 1992
- Inspired by Kelley's cutting plane method
  - Convex NLP problems Kelley Jr. J., The cutting-plane method for solving convex programs, Journal of the SIAM, vol. 8(4), pp. 703–712, 1960.
- Published in Computers & Chemical Engineering in 1995
  - Westerlund T. and Pettersson F., An extended cutting plane method for solving convex MINLP problems, Computers & Chemical Engineering 19, pp. 131–136, 1995.

Extensions

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

Extensions

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

Pseudoconvex objective function and constraints Westerlund T. and Pörn R. Solving Pseudo-Convex Mixed Integer Optimization Problems by Cutting Plane techniques. Optimization and Engineering, 3, 253–280, 2002.

Extensions

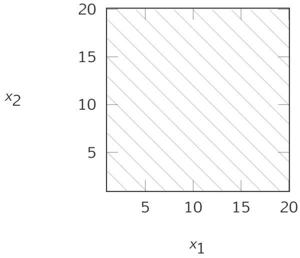
Pseudoconvex constraints Westerlund T., Skrifvars H., Harjunkoski I. and Pörn R. An extended cutting plane method for solving a class of non-convex MINLP problems. Computers and Chemical Engineering, 22, 357–365, 1998.

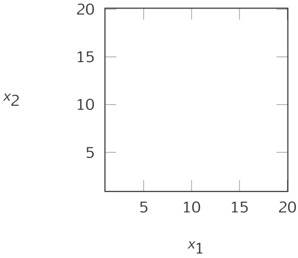
Pseudoconvex objective function and constraints Westerlund T. and Pörn R. Solving Pseudo-Convex Mixed Integer Optimization Problems by Cutting Plane techniques. Optimization and Engineering, 3, 253–280, 2002.

Nonsmooth constraints Eronen V.-P., Mäkelä M. M. and Westerlund T. On the generalization of ECP and OA methods to nonsmooth convex MINLP problems, Taylor and Francis, 2014.

An example

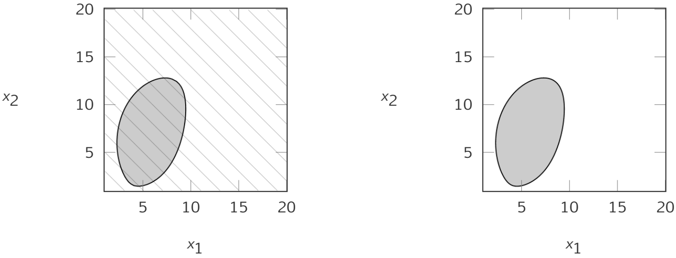
$$
\begin{aligned}
\text{minimize} \quad & c^T x = -x_1 - x_2 \\
\text{subject to} \quad & g_1(x_1, x_2) = 0.15(x_1 - 8)^2 + 0.1(x_2 - 6)^2 + 0.025 e^{x_1} x_2^{-2} - 5 \le 0 \\
& g_2(x_1, x_2) = 1/x_1 + 1/x2 - x_1^{0.5} x_2^{0.5} + 4 \le 0 \\
& 2x_1 - 3x_2 - 2 \le 0 \\
& 1 \le x_1 \le 20, \quad 1 \le x_2 \le 20, \quad x_1 \in \mathbb{R}, \quad x_2 \in \mathbb{Z}.
\end{aligned}
$$





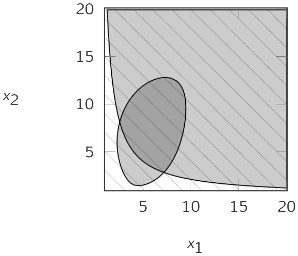
An example

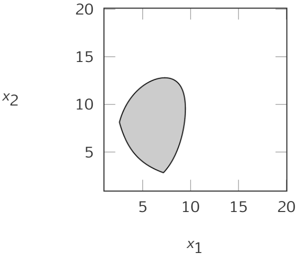
$$
\begin{aligned}
\text{minimize} \quad & c^T x = -x_1 - x_2 \\
\text{subject to} \quad & g_1(x_1, x_2) = 0.15(x_1 - 8)^2 + 0.1(x_2 - 6)^2 + 0.025 e^{x_1} x_2^{-2} - 5 \leq 0 \\
& g_2(x_1, x_2) = 1/x_1 + 1/x2 - x_1^{0.5} x_2^{0.5} + 4 \leq 0 \\
& 2x_1 - 3x_2 - 2 \leq 0 \\
& 1 \leq x_1 \leq 20, \quad 1 \leq x_2 \leq 20, \quad x_1 \in \mathbb{R}, \quad x_2 \in \mathbb{Z}.
\end{aligned}
$$



An example

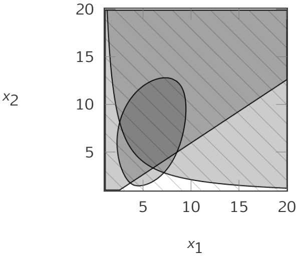
$$
\begin{array}{ll}
\text{minimize} & c^T x = -x_1 - x_2 \\
\text{subject to} & g_1(x_1, x_2) = 0.15(x_1 - 8)^2 + 0.1(x_2 - 6)^2 + 0.025 e^{x_1} x_2^{-2} - 5 \le 0 \\
& g_2(x_1, x_2) = 1/x_1 + 1/x2 - x_1^{0.5} x_2^{0.5} + 4 \le 0 \\
& 2x_1 - 3x_2 - 2 \le 0 \\
& 1 \le x_1 \le 20, \quad 1 \le x_2 \le 20, \quad x_1 \in \mathbb{R}, \quad x_2 \in \mathbb{Z}.
\end{array}
$$

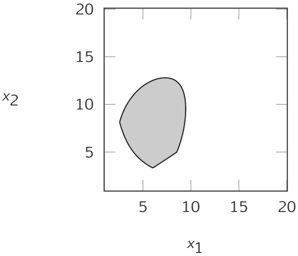




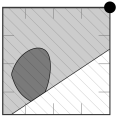
An example

$$\begin{aligned}
\text{minimize} \quad & c^T x = -x_1 - x_2 \\
\text{subject to} \quad & g_1(x_1, x_2) = 0.15(x_1 - 8)^2 + 0.1(x_2 - 6)^2 + 0.025e^{x_1} x_2^{-2} - 5 \leq 0 \\
& g_2(x_1, x_2) = 1/x_1 + 1/x2 - x_1^{0.5} x_2^{0.5} + 4 \leq 0 \\
& 2x_1 - 3x_2 - 2 \leq 0 \\
& 1 \leq x_1 \leq 20, \quad 1 \leq x_2 \leq 20, \quad x_1 \in \mathbb{R}, \quad x_2 \in \mathbb{Z}.
\end{aligned}$$



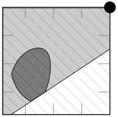


- In each iteration k of the algorithm a MILP problem is solved to obtain the solution point (x_1^k, x_2^k).
    - First iteration gives $x_1^1 = 20, x_2^1 = 20$.



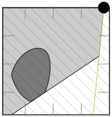
- In each iteration k of the algorithm a MILP problem is solved to obtain the solution point (x_1^k, x_2^k).
  - First iteration gives $x_1^1 = 20, x_2^1 = 20$.
  - $g_1(x_1^1, x_2^1) = 30359.0, \quad g_2(x_1^1, x_2^1) = -15.9$.

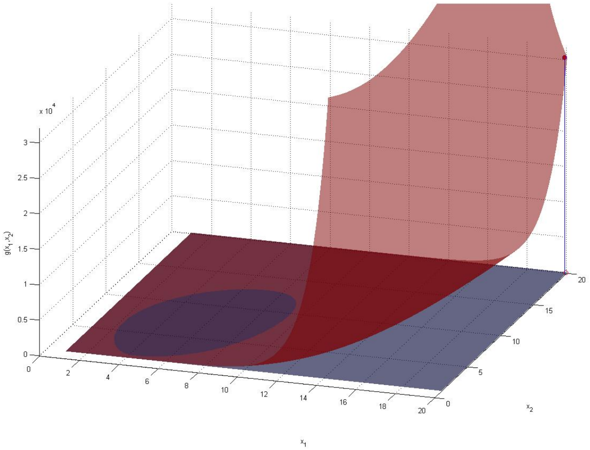
- In each iteration k of the algorithm a MILP problem is solved to obtain the solution point (x_1^k, x_2^k).
  - First iteration gives $x_1^1 = 20, x_2^1 = 20$.
  - $g_1(x_1^1, x_2^1) = 30359.0, \quad g_2(x_1^1, x_2^1) = -15.9$.



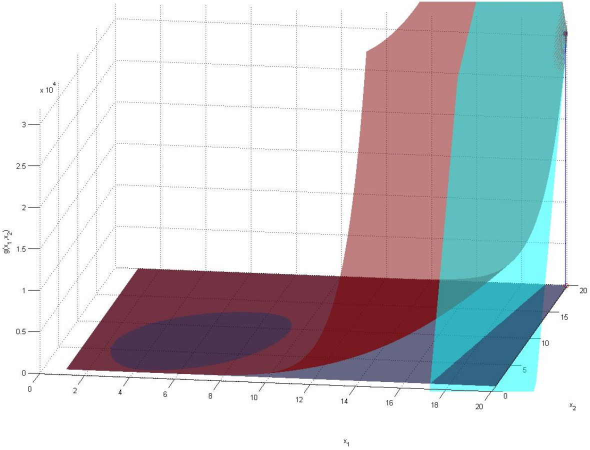
- A new cutting plane is generated for the violated nonlinear constraint g_1:

$$g_1(x_1^1, x_2^1) + \nabla g_1(x_1^1, x_2^1)^T (x - x_1^1, x - x_2^1) \leq 0$$



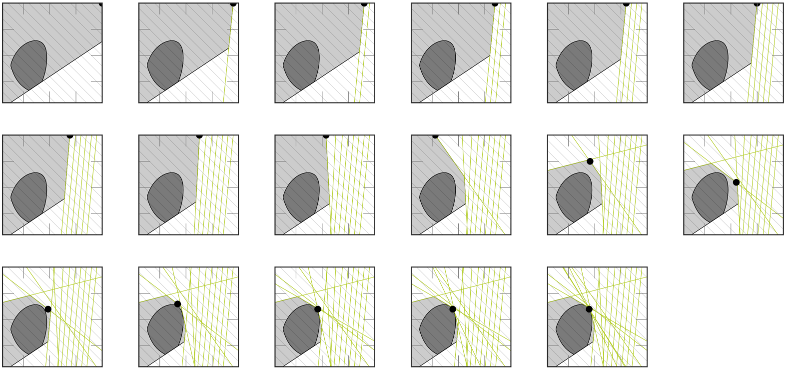


- The nonlinear function $g_1(x_1, x_2)$

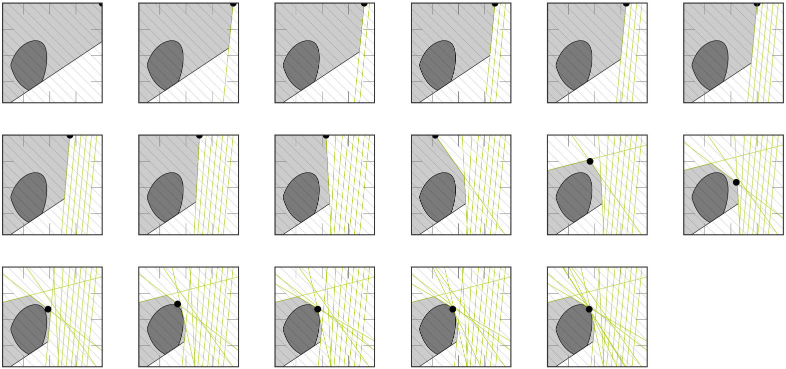


- The nonlinear function $g_1(x_1, x_2)$
  - and the cutting plane generated at $x_1 = 20, x_2 = 20$

- To solve the problem with the ECP algorithm ($\epsilon = 0.001$) it takes 17 iterations (17 MILP problems solved to optimality).

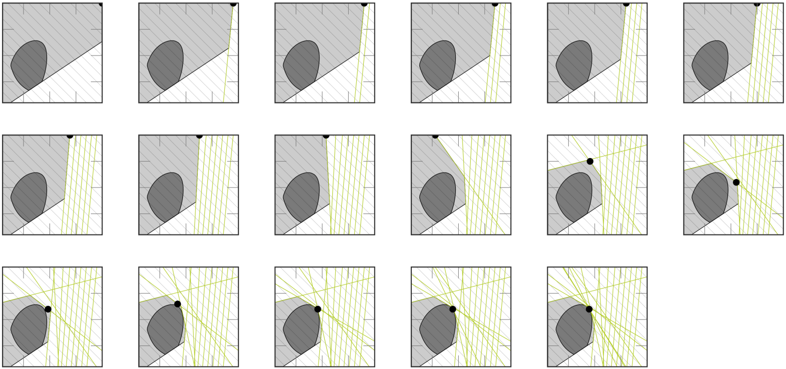


- To solve the problem with the ECP algorithm ($\epsilon = 0.001$) it takes 17 iterations (17 MILP problems solved to optimality).



- How can we improve performance?

- To solve the problem with the ECP algorithm ($\epsilon = 0.001$) it takes 17 iterations (17 MILP problems solved to optimality).



- How can we improve performance?
  Generate cutting planes on the boundary of the feasible set!

The ESH algorithm

- A new interior point based algorithm for solving convex MINLP problems to global optimality.

- A new interior point based algorithm for solving convex MINLP problems to global optimality.
- Roots:
  - The extended cutting plane algorithm 1995

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

- A new interior point based algorithm for solving convex MINLP problems to global optimality.

- Roots:
  - The extended cutting plane algorithm 1995
  - Kelley's cutting plane algorithm 1960

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

- A new interior point based algorithm for solving convex MINLP problems to global optimality.
- Roots:
  - The extended cutting plane algorithm 1995
  - Kelley's cutting plane algorithm 1960
  - The supporting hyperplane method 1967 [1]

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

- A new interior point based algorithm for solving convex MINLP problems to global optimality.

- Roots:
  - The extended cutting plane algorithm 1995
  - Kelley's cutting plane algorithm 1960
  - The supporting hyperplane method 1967 [1]

- Cutting planes are replaced with supporting hyperplanes using a line search procedure to find the generation point. An interior point is required for the line search.

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

- A new interior point based algorithm for solving convex MINLP problems to global optimality.

- Roots:
  - The extended cutting plane algorithm 1995
  - Kelley's cutting plane algorithm 1960
  - The supporting hyperplane method 1967 [1]

- Cutting planes are replaced with supporting hyperplanes using a line search procedure to find the generation point. An interior point is required for the line search.

- Two LP preprocessing steps are utilized to quickly get a tight linear relaxation of the part of the feasible region defined by the convex/quasiconvex constraints.

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

The MINLP problem

- The algorithm finds the optimal solution x^* to the following convex MINLP problem:

$$\boxed{x^* = \underset{x \in C \cap L \cap Y}{\arg\min}\, c^T x} \tag{P}$$

where $x = [x_1, x_2, \ldots, x_N]^T$ belongs to the compact set

$$X = \left\{ x \,\middle|\, \underline{x}_i \leq x_i \leq \overline{x}_i,\ i = 1, \ldots, N \right\} \subset \mathbb{R}^n,$$

the feasible region is defined by $C \cap L \cap Y$,

$$\begin{aligned} C &= \{x \,|\, g_m(x) \leq 0,\ m = 1, \ldots, M,\ x \in X\}, \\ L &= \{x \,|\, Ax \leq a,\ Bx = b,\ x \in X\}, \\ Y &= \{x \,|\, x_i \in \mathbb{Z},\ i \in I_{\mathbb{Z}},\ x \in X\}, \end{aligned}$$

and C is a convex set.

Steps in the ESH algorithm

NLP: Obtain a feasible, relaxed interior point (a point in the set C) by solving a NLP problem.

Steps in the ESH algorithm

NLP: Obtain a feasible, relaxed interior point (a point in the set C) by solving a NLP problem.

LP1: Solve simple LP problems (initially in X) to add additional supporting hyperplanes.

Steps in the ESH algorithm

NLP: Obtain a feasible, relaxed interior point (a point in the set C) by solving a NLP problem.

LP1: Solve simple LP problems (initially in X) to add additional supporting hyperplanes.

LP2: Solve simple LP problems (including the constraints in L) to add additional supporting hyperplanes.

Steps in the ESH algorithm

NLP: Obtain a feasible, relaxed interior point (a point in the set C) by solving a NLP problem.

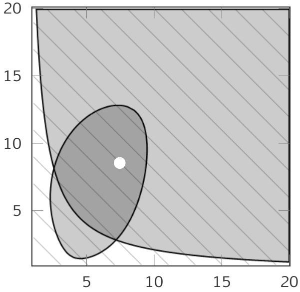
LP1: Solve simple LP problems (initially in X) to add additional supporting hyperplanes.

LP2: Solve simple LP problems (including the constraints in L) to add additional supporting hyperplanes.

MILP: Solve MILP problems to find the optimal solution to (P).

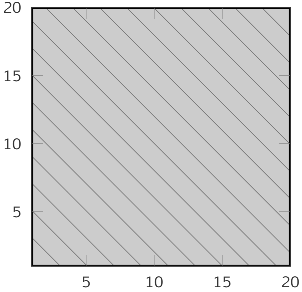
NLP step

- If an interior point is not given, obtain a feasible, relaxed interior point (satisfying all the nonlinear constraints in C) by solving a NLP problem.



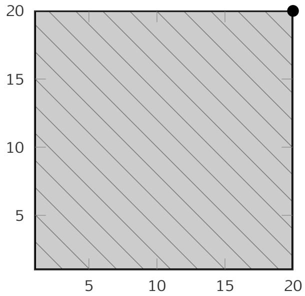
LP1 step (optional)

- Solve simple LP problems (initially in X) and conduct a line search procedure to obtain supporting hyperplanes giving a first linear relaxation of the convex set C.



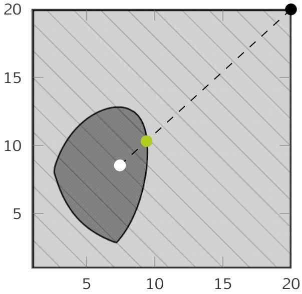
LP1 step (optional)

- Solve simple LP problems (initially in X) and conduct a line search procedure to obtain supporting hyperplanes giving a first linear relaxation of the convex set C.



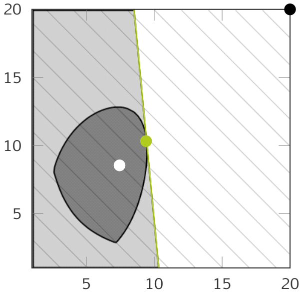
LP1 step (optional)

- Solve simple LP problems (initially in X) and conduct a line search procedure to obtain supporting hyperplanes giving a first linear relaxation of the convex set C.



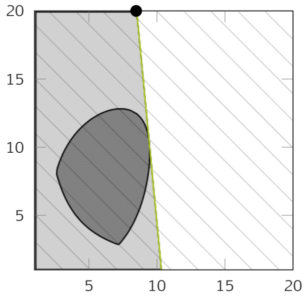
LP1 step (optional)

- Solve simple LP problems (initially in X) and conduct a line search procedure to obtain supporting hyperplanes giving a first linear relaxation of the convex set C.



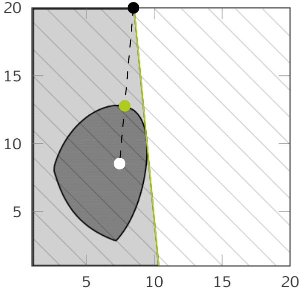
LP1 step (optional)

- Solve simple LP problems (initially in X) and conduct a line search procedure to obtain supporting hyperplanes giving a first linear relaxation of the convex set C.



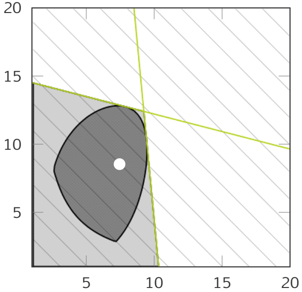
LP1 step (optional)

- Solve simple LP problems (initially in X) and conduct a line search procedure to obtain supporting hyperplanes giving a first linear relaxation of the convex set C.



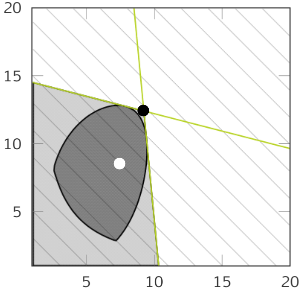
LP1 step (optional)

- Solve simple LP problems (initially in X) and conduct a line search procedure to obtain supporting hyperplanes giving a first linear relaxation of the convex set C.



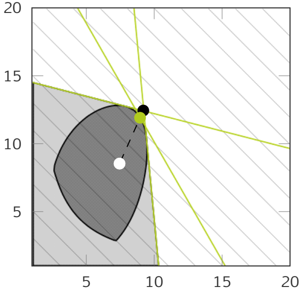
LP1 step (optional)

- Solve simple LP problems (initially in X) and conduct a line search procedure to obtain supporting hyperplanes giving a first linear relaxation of the convex set C.



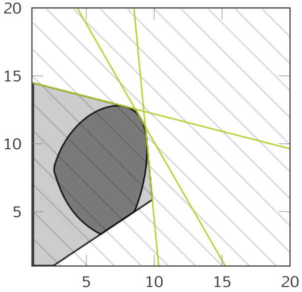
LP1 step (optional)

- Solve simple LP problems (initially in X) and conduct a line search procedure to obtain supporting hyperplanes giving a first linear relaxation of the convex set C.



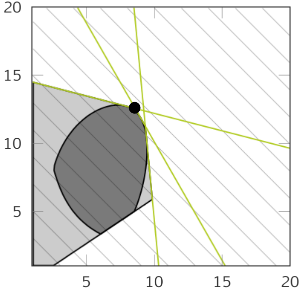
LP2 step (optional)

- Continue with a corresponding procedure as in LP1 but now also including the linear constraints in L in the original problem.



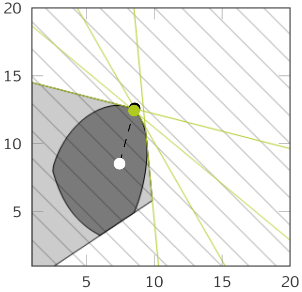
LP2 step (optional)

- Continue with a corresponding procedure as in LP1 but now also including the linear constraints in L in the original problem.



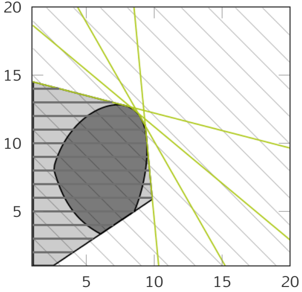
LP2 step (optional)

- Continue with a corresponding procedure as in LP1 but now also including the linear constraints in L in the original problem.



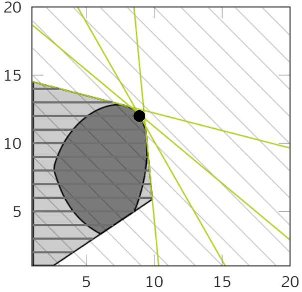
MILP step

- Finally include the integer requirements and solve MILP problems using a corresponding procedure to find the optimal solution to (P).



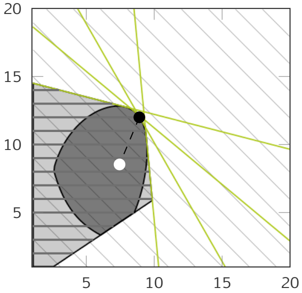
MILP step

- Finally include the integer requirements and solve MILP problems using a corresponding procedure to find the optimal solution to (P).



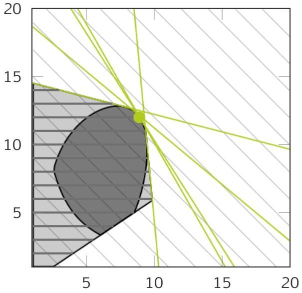
MILP step

- Finally include the integer requirements and solve MILP problems using a corresponding procedure to find the optimal solution to (P).



MILP step

- Finally include the integer requirements and solve MILP problems using a corresponding procedure to find the optimal solution to (P).



NLP step

- A point in C is required as an endpoint for the line searches to be conducted in the LP1-, LP2- and MILP-steps.

NLP step

- A point in C is required as an endpoint for the line searches to be conducted in the LP1-, LP2- and MILP-steps.
- Assuming that (P) has a solution, the internal point can be obtained from the following NLP problem:

$$\boxed{\begin{aligned} \tilde{x}_{\text{NLP}} &= \operatorname*{arg\,min}_{x \in X} F(x), \\ \text{where } F(x) &:= \max_{m=1,\dots,M} \{g_m(x)\}. \end{aligned}} \tag{P-NLP}$$

NLP step

- A point in C is required as an endpoint for the line searches to be conducted in the LP1-, LP2- and MILP-steps.
- Assuming that (P) has a solution, the internal point can be obtained from the following NLP problem:

$$\boxed{\begin{aligned} \tilde{x}_{\mathrm{NLP}} &= \underset{x \in X}{\arg\min}\, F(x), \\ \text{where } F(x) &:= \max_{m=1,\ldots,M} \{g_m(x)\}. \end{aligned}} \tag{P-NLP}$$

- F is convex/quasiconvex since it is the maximum of convex/quasiconvex functions.

NLP step

- A point in C is required as an endpoint for the line searches to be conducted in the LP1-, LP2- and MILP-steps.
- Assuming that (P) has a solution, the internal point can be obtained from the following NLP problem:

$$\boxed{\begin{aligned} \tilde{x}_{\text{NLP}} &= \arg\min_{x \in X} F(x), \\ \text{where } F(x) &:= \max_{m=1,\dots,M} \{g_m(x)\}. \end{aligned}} \tag{P-NLP}$$

- F is convex/quasiconvex since it is the maximum of convex/quasiconvex functions.
- (P-NLP) may be nonsmooth (if $M > 1$) even if g_m is smooth.

NLP step

- A point in C is required as an endpoint for the line searches to be conducted in the LP1-, LP2- and MILP-steps.
- Assuming that (P) has a solution, the internal point can be obtained from the following NLP problem:

$$\boxed{\begin{gathered}\tilde{x}_{\mathrm{NLP}} = \arg\min_{x \in X} F(x), \\ \text{where } F(x) := \max_{m=1,\ldots,M} \{g_m(x)\}.\end{gathered}} \tag{P-NLP}$$

- F is convex/quasiconvex since it is the maximum of convex/quasiconvex functions.
- (P-NLP) may be nonsmooth (if $M > 1$) even if g_m is smooth.
- The point $\tilde{x}_{\mathrm{NLP}}$ need not be optimal but then fulfill $F(\tilde{x}_{\mathrm{NLP}}) < 0$.

NLP step

- A point in C is required as an endpoint for the line searches to be conducted in the LP1-, LP2- and MILP-steps.
- Assuming that (P) has a solution, the internal point can be obtained from the following NLP problem:

$$\boxed{\begin{aligned} \tilde{x}_{\text{NLP}} &= \operatorname*{arg\,min}_{x \in X} F(x), \\ \text{where } F(x) &:= \max_{m=1,\dots,M} \{g_m(x)\}. \end{aligned}} \tag{P-NLP}$$

- F is convex/quasiconvex since it is the maximum of convex/quasiconvex functions.
- (P-NLP) may be nonsmooth (if $M > 1$) even if g_m is smooth.
- The point $\tilde{x}_{\text{NLP}}$ need not be optimal but then fulfill $F(\tilde{x}_{\text{NLP}}) < 0$.
- The NLP step can also be formulated as a smooth NLP problem if all functions g_m are smooth and convex.

LP1 step

- Starting from $k = 1$, $\Omega_0 = X$, the problem

$$\boxed{\tilde{x}^k_{\mathrm{LP}} = \underset{\Omega_{k-1}}{\arg\min}\ c^T x} \tag{P-LP1}$$

is solved, and the point x^k is obtained by a line search for $F(x^k) = 0$ between the internal point $\tilde{x}_{\mathrm{NLP}}$ and the solution point to (P-LP1) $\tilde{x}^k_{\mathrm{LP}}$:

$$x^k = \lambda \tilde{x}_{\mathrm{NLP}} + (1-\lambda)\tilde{x}^k_{\mathrm{LP}}, \quad \lambda \in [0,1].$$

LP1 step

- Starting from $k = 1$, $\Omega_0 = X$, the problem

$$\boxed{\tilde{x}^k_{\mathrm{LP}} = \operatorname*{arg\,min}_{\Omega_{k-1}} \; c^T x} \tag{P-LP1}$$

  is solved, and the point x^k is obtained by a line search for $F(x^k) = 0$ between the internal point $\tilde{x}_{\mathrm{NLP}}$ and the solution point to (P-LP1) $\tilde{x}^k_{\mathrm{LP}}$:

$$x^k = \lambda \tilde{x}_{\mathrm{NLP}} + (1-\lambda)\tilde{x}^k_{\mathrm{LP}}, \quad \lambda \in [0,1].$$

  Supporting hyperplanes (SHs)

$$l_k := F(x^k) + \xi_F(x^k)^T (x - x^k) \leq 0$$

  are generated and added to Ω_k.
  $\xi_F(x^k)^T$ is a gradient or subgradient of F at x^k.

LP1 step

- Starting from $k = 1$, $\Omega_0 = X$, the problem

$$\boxed{\tilde{x}_{\text{LP}}^{k} = \underset{\Omega_{k-1}}{\arg\min}\ c^T x} \qquad \text{(P-LP1)}$$

is solved, and the point x^k is obtained by a line search for $F(x^k) = 0$ between the internal point $\tilde{x}_{\text{NLP}}$ and the solution point to (P-LP1) $\tilde{x}_{\text{LP}}^{k}$:

$$x^k = \lambda \tilde{x}_{\text{NLP}} + (1 - \lambda)\tilde{x}_{\text{LP}}^{k}, \quad \lambda \in [0, 1].$$

Supporting hyperplanes (SHs)

$$l_k := F(x^k) + \xi_F(x^k)^T(x - x^k) \leq 0$$

are generated and added to Ω_k.
$\xi_F(x^k)^T$ is a gradient or subgradient of F at x^k.
- Repeted untill $F(\tilde{x}_{\text{LP}}^{k}) < \epsilon_{\text{LP1}}$ or a maximum number of SHs

LP2 step

- This step is otherwise identical to LP1, with the exception that the linear constraints in L are now also included, *i.e.,*

$$\boxed{\tilde{x}_{\mathrm{LP}}^{k} = \underset{\Omega_{k-1} \cap L}{\arg\min}\; c^T x} \qquad \text{(P-LP2)}$$

LP2 step

- This step is otherwise identical to LP1, with the exception that the linear constraints in L are now also included, *i.e.,*

$$\boxed{\tilde{x}_{\mathrm{LP}}^{k} = \underset{\Omega_{k-1} \cap L}{\arg\min}\ c^T x} \tag{P-LP2}$$

- (P-LP2) is repeatedly solved until $F(\tilde{x}_{\mathrm{LP}}^{k}) < \epsilon_{\mathrm{LP2}}$ or a maximum number of SHs have additionally been generated.

MILP step

- Finally, in order to also fulfill the integer requirements of problem (P), a MILP step is performed.

MILP step

- Finally, in order to also fulfill the integer requirements of problem (P), a MILP step is performed.
- This step is otherwise identical to LP2, with the exception that the integer requirements in Y are now additionally considered, *i.e.,*

$$\boxed{\tilde{x}^k_{\text{MILP}} = \underset{\Omega_{k-1} \cap L \cap Y}{\arg\min} \; c^T x.} \tag{P-MILP}$$

MILP step

- Finally, in order to also fulfill the integer requirements of problem (P), a MILP step is performed.
- This step is otherwise identical to LP2, with the exception that the integer requirements in Y are now additionally considered, *i.e.,*

$$\boxed{\tilde{x}^k_{\text{MILP}} = \underset{\Omega_{k-1} \cap L \cap Y}{\arg\min} \; c^T x.} \tag{P-MILP}$$

- (P-MILP) is repeatedly solved until $F(\tilde{x}^k_{\text{MILP}}) < \epsilon_{\text{MILP}}$.

MILP step

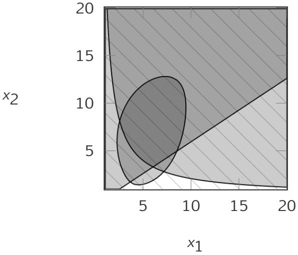
- Finally, in order to also fulfill the integer requirements of problem (P), a MILP step is performed.
- This step is otherwise identical to LP2, with the exception that the integer requirements in Y are now additionally considered, *i.e.,*

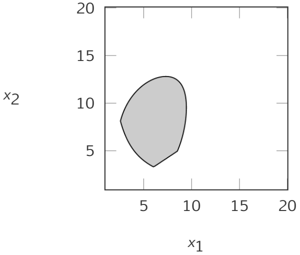
$$\boxed{\tilde{x}^k_{\text{MILP}} = \underset{\Omega_{k-1} \cap L \cap Y}{\arg\min}\ c^T x.} \tag{P-MILP}$$

- (P-MILP) is repeatedly solved until $F(\tilde{x}^k_{\text{MILP}}) < \epsilon_{\text{MILP}}$.
- Intermediate (P-MILP) problems do not need to be solved to optimality, but in order to guarantee an optimal solution of (P), the final MILP solution must be optimal.

Now, consider the same example as earlier

$$
\begin{aligned}
\text{minimize} \quad & c^T x = -x_1 - x_2 \\
\text{subject to} \quad & 0.15(x_1-8)^2 + 0.1(x_2-6)^2 + 0.025 e^{x_1} x_2^{-2} - 5 \leq 0 \\
& 1/x_1 + 1/x2 - x_1^{0.5} x_2^{0.5} + 4 \leq 0 \\
& 2x_1 - 3x_2 - 2 \leq 0 \\
& 1 \leq x_1 \leq 20, \quad 1 \leq x_2 \leq 20, \quad x_1 \in \mathbb{R}, \quad x_2 \in \mathbb{Z}.
\end{aligned}
$$



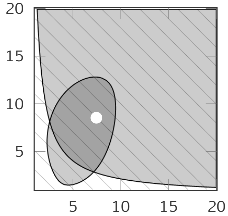


NLP step – find an interior point

$$\tilde{x}_{\mathrm{NLP}} = \underset{(x_1,x_2)\in X}{\arg\min}\; F(x_1,x_2),$$

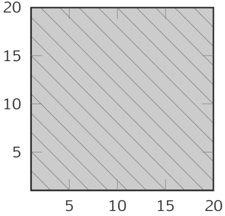
$$\text{where } F(x_1,x_2) := \max\{g_1(x_1,x_2),\, g_2(x_1,x_2)\}.$$

- The problem can be found using a suitable NLP solver.
- Not required to be the optimal point
- The optimal point here is $(7.45, 8.54)$



LP1-step – Iteration 1

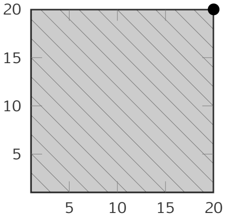
- Assume initially that $\Omega_0 = X$.



LP1-step – Iteration 1

- Assume initially that $\Omega_0 = X$.
- $k = 1$, solve LP in Ω,

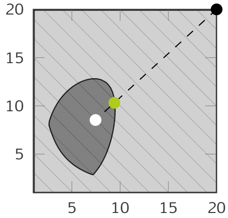
$$\tilde{x}^k_{\text{LP}} = \operatorname*{arg\,min}_{\Omega_{k-1}} c^T x.$$



LP1-step – Iteration 1

- Assume initially that $\Omega_0 = X$.
- $k = 1$, solve LP in Ω,

$$\tilde{x}_{\mathrm{LP}}^{k} = \operatorname*{arg\,min}_{\Omega_{k-1}} \; c^T x.$$



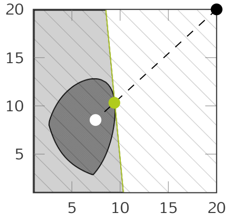
- Do line search

$$x^k = \lambda \tilde{x}_{\mathrm{NLP}} + (1-\lambda)\tilde{x}_{\mathrm{LP}}^{k}.$$

LP1-step – Iteration 1

- Assume initially that $\Omega_0 = X$.
- $k = 1$, solve LP in Ω,

$$\tilde{x}_{\mathrm{LP}}^k = \operatorname*{arg\,min}_{\Omega_{k-1}} \; c^T x.$$



- Do line search

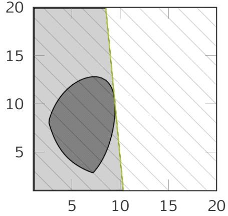
$$x^k = \lambda \tilde{x}_{\mathrm{NLP}} + (1-\lambda)\tilde{x}_{\mathrm{LP}}^k.$$

- Generate supporting hyperplane in x^k and add to Ω.

LP1-step – Iteration 2

- $\Omega_1 = \{x | l_1(x) \leq 0,\ x \in X\}$.

  $l_1(x) = 3.26x_1 + 0.313x_2 - 33.9$



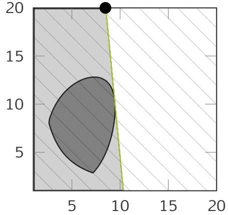
LP1-step – Iteration 2

- $\Omega_1 = \{x | l_1(x) \leq 0,\ x \in X\}$.

  $$l_1(x) = 3.26x_1 + 0.313x_2 - 33.9$$

- $k = 2$, solve LP in Ω,

  $$\tilde{x}^k_{\mathrm{LP}} = \arg\min_{\Omega_{k-1}}\ c^T x.$$



LP1-step – Iteration 2

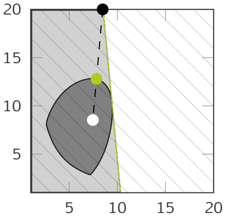
- $\Omega_1 = \{x | l_1(x) \leq 0,\ x \in X\}$.

  $$l_1(x) = 3.26x_1 + 0.313x_2 - 33.9$$

- $k = 2$, solve LP in Ω,

  $$\tilde{x}^k_{\text{LP}} = \arg\min_{\Omega_{k-1}} \ c^T x.$$

- Do line search $x^k = \lambda \tilde{x}_{\text{NLP}} + (1 - \lambda)\tilde{x}^k_{\text{LP}}$.



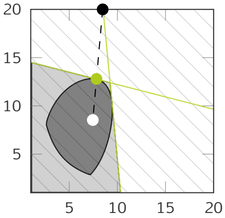
LP1-step – Iteration 2

- $\Omega_1 = \{x | l_1(x) \leq 0,\ x \in X\}$.

  $$l_1(x) = 3.26x_1 + 0.313x_2 - 33.9$$

- $k = 2$, solve LP in Ω,

  $$\tilde{x}_{\text{LP}}^k = \arg\min_{\Omega_{k-1}}\ c^T x.$$

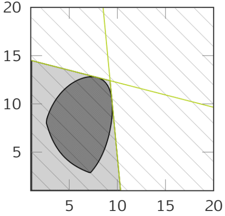


- Do line search $x^k = \lambda \tilde{x}_{\text{NLP}} + (1-\lambda)\tilde{x}_{\text{LP}}^k$.
- Generate supporting hyperplane in x^k and add to Ω.

LP1-step – Iteration 3

- $\Omega_2 = \{x | l_j(x) \leq 0,\ j \in \{1,2\},\ x \in X\}$

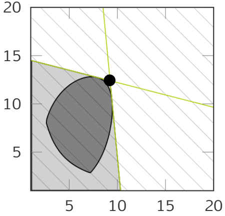
$$l_1(x) = 3.26x_1 + 0.313x_2 - 33.9$$
$$l_2(x) = 0.332x_1 + 1.30x_2 - 19.2$$



LP1-step – Iteration 3

- $\Omega_2 = \{x | l_j(x) \leq 0,\ j \in \{1,2\},\ x \in X\}$

$$l_1(x) = 3.26x_1 + 0.313x_2 - 33.9$$
$$l_2(x) = 0.332x_1 + 1.30x_2 - 19.2$$



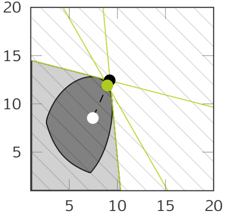
- $k = 3$, solve LP in Ω,

$$\tilde{x}_{\mathrm{LP}}^k = \arg\min_{\Omega_{k-1}}\ c^T x.$$

LP1-step – Iteration 3

- $\Omega_2 = \{x | l_j(x) \leq 0,\ j \in \{1,2\},\ x \in X\}$

$$l_1(x) = 3.26x_1 + 0.313x_2 - 33.9$$
$$l_2(x) = 0.332x_1 + 1.30x_2 - 19.2$$



- $k = 3$, solve LP in Ω,

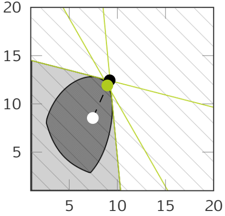
$$\tilde{x}_{\text{LP}}^k = \arg\min_{\Omega_{k-1}} \ c^T x.$$

- Do line search, generate supporting hyperplane and add to Ω.

LP1-step – Iteration 3

- $\Omega_2 = \{x | l_j(x) \leq 0,\ j \in \{1, 2\},\ x \in X\}$

$$l_1(x) = 3.26x_1 + 0.313x_2 - 33.9$$
$$l_2(x) = 0.332x_1 + 1.30x_2 - 19.2$$



- $k = 3$, solve LP in Ω,

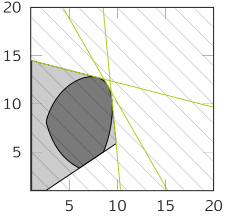
$$\tilde{x}_{\mathrm{LP}}^k = \arg\min_{\Omega_{k-1}} \ c^T x.$$

- Do line search, generate supporting hyperplane and add to Ω.
- Terminate LP1-step since $F(\tilde{x}_{\mathrm{LP}}^k) < \epsilon_{\mathrm{LP1}}$.

LP2-step – Iteration 4

- $\Omega_3 = \{x | l_j(x) \leq 0,\ j \in \{1,2,3\},\ x \in X\}$

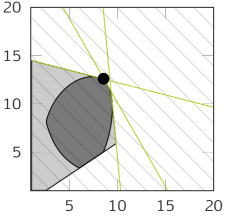
$$l_1(x) = 3.26x_1 + 0.313x_2 - 33.9$$
$$l_2(x) = 0.332x_1 + 1.30x_2 - 19.2$$
$$l_3(x) = 1.66x_1 + 0.951x_2 - 26.2$$



LP2-step – Iteration 4

- $\Omega_3 = \{x | l_j(x) \leq 0,\ j \in \{1, 2, 3\},\ x \in X\}$

$$l_1(x) = 3.26x_1 + 0.313x_2 - 33.9$$
$$l_2(x) = 0.332x_1 + 1.30x_2 - 19.2$$
$$l_3(x) = 1.66x_1 + 0.951x_2 - 26.2$$



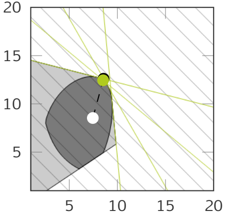
- $k = 4$, solve LP now in $\Omega \cap L$,

$$\tilde{x}_{\text{LP}}^k = \arg\min_{\Omega_{k-1} \cap L}\ c^T x.$$

LP2-step – Iteration 4

- $\Omega_3 = \{x | l_j(x) \leq 0,\ j \in \{1,2,3\},\ x \in X\}$

$$l_1(x) = 3.26x_1 + 0.313x_2 - 33.9$$
$$l_2(x) = 0.332x_1 + 1.30x_2 - 19.2$$
$$l_3(x) = 1.66x_1 + 0.951x_2 - 26.2$$



- $k = 4$, solve LP now in $\Omega \cap L$,

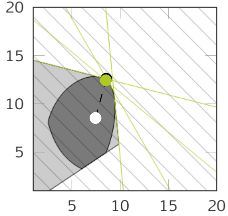
$$\tilde{x}^k_{\text{LP}} = \arg\min_{\Omega_{k-1} \cap L}\ c^T x.$$

- Do line search, generate supporting hyperplane and add to Ω.

LP2-step – Iteration 4

- $\Omega_3 = \{x | l_j(x) \leq 0,\ j \in \{1,2,3\},\ x \in X\}$

$$l_1(x) = 3.26x_1 + 0.313x_2 - 33.9$$
$$l_2(x) = 0.332x_1 + 1.30x_2 - 19.2$$
$$l_3(x) = 1.66x_1 + 0.951x_2 - 26.2$$

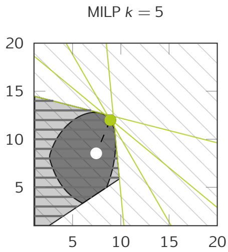


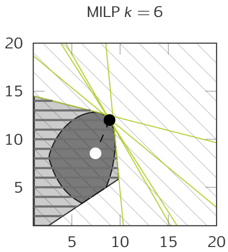
- $k = 4$, solve LP now in $\Omega \cap L$,

$$\tilde{x}_{\mathrm{LP}}^k = \arg\min_{\Omega_{k-1} \cap L}\ c^T x.$$

- Do line search, generate supporting hyperplane and add to Ω.
- Terminate LP2-step since $F(\tilde{x}_{\mathrm{LP}}^k) < \epsilon_{\mathrm{LP2}}$.

MILP step – Iterations 5 and 6





- In this step the integer requirements in Y are also considered, *i.e.,* initially $k = 5, \Omega = \Omega_{k-1} \cap L \cap Y$.
- The MILP steps are required to guarantee an integer-feasible solution.

Solution and comparisons to other solvers

- Solving the MINLP problem with the supporting hyperplane algorithm gives the following solution

Type	Iteration	Obj. funct.	x_1	x_2	$F(x_1, x_2)$
LP1	1	−40.0000	20.0000	20.0000	30 359
LP1	2	−28.4720	8.47199	20.0000	14.9321
LP1	3	−21.6378	9.19722	12.4406	0.957382
LP2	4	−21.1639	8.56022	12.6037	0.229455
MILP	5	−20.9065	8.90647	12	0.00442134
MILP	6	−20.9036	8.90362	12	$4.22619 \cdot 10^{-6}$

Solution and comparisons to other solvers

- Solving the MINLP problem with the supporting hyperplane algorithm gives the following solution

Type	Iteration	Obj. funct.	x_1	x_2	$F(x_1, x_2)$
LP1	1	−40.0000	20.0000	20.0000	30 359
LP1	2	−28.4720	8.47199	20.0000	14.9321
LP1	3	−21.6378	9.19722	12.4406	0.957382
LP2	4	−21.1639	8.56022	12.6037	0.229455
MILP	5	−20.9065	8.90647	12	0.00442134
MILP	6	−20.9036	8.90362	12	$4.22619 \cdot 10^{-6}$

- Solution times compared to some other MINLP solvers:

Solver	Subproblems solved	Time (s)	Implementation
ESH	1 NLP + 4 LP + 2 MILP	0.04	Early prototype
ECP	21 MILP (10 OPT) + 1 NLP	1.4	GAMS 24.2 + CPLEX
DICOPT	10 NLP + 10 MILP	1.5	GAMS 24.2 + CONOPT + CPLEX

Solver comparison

- The test problems are taken from MINLPlib2, which is a collection Mixed Integer Nonlinear Programming models.

Solver	fo7-ar4-1	fo9-ar3-1	jit1	m7-ar5-1
ESH	**32.51**	**169.15**	0.12	**1.06**
ECP	79.45	612.68	0.25	4.84
ANTIGONE	33.58	*	1.36	1.61
BARON	*	*	0.1	166.62
DICOPT	#	#	#	#
SBB	#	#	0.03	#
SCIP	34.12	393.22	**0.01**	13.15
Variables	112	180	25	112
Binaries	0	0	0	0
Integers	42	72	4	42
Type	MINLP	MINLP	MINLP	MINLP

Solver comparison

Solver	batchs101006m	enpro56pb	o7	rsyn0805m04h	rsyn0830m04h	sssd25-08
ESH	17.89	1.34	**461.75**	6.64	111.63	**58.71**
ECP	17.8	2.37	*	4.46	25.02	*
ANTIGONE	15.68	0.75	*	*	*	668.71
BARON	161.81	7.75	*	66.87	*	*
DICOPT	**1.75**	**0.34**	#	**2.31**	**4.57**	*
SBB	#	#	#	10.8	171.91	#
SCIP	11.68	1.48	*	13.63	*	*
Variables	278	127	114	1,400	1,956	256
Binaries	129	73	42	296	416	224
Integers	0	0	0	0	0	0
Type	MBNLP	MBNLP	MBNLP	MBNLP	MBNLP	MBNLP

Solver comparison

Solver	alan	fac3	netmod-dol2	netmod-kar1	slay05h
ESH	**0**.01	0.76	94.19	21.87	38.49
ECP	0.28	0.22	467.35	82.54	84.46
ANTIGONE	0.33	92.75	113.9	157.01	0.61
BARON	0.14	1.31	*	#	1,067.14
DICOPT	0.14	0.53	#	#	**0.19**
SBB	**0**.01	**0.20**	#	23.26	6.13
SCIP	**0.01**	0.23	**43.57**	**4.04**	1.24
Variables	8	66	1,998	456	230
Binaries	4	12	462	136	40
Integers	0	0	0	0	0
Type	MBQP	MBQP	MBQP	MBQP	MBQP

Solver comparison

Solver	du-opt	ex4
ESH	33.3	1.01
ECP	22.98	0.75
ANTIGONE	*	**0.22**
BARON	13.74	2.62
DICOPT	#	0.44
SBB	**0.33**	1.06
SCIP	0.7	0.45
Variables	20	36
Binaries	0	25
Integers	13	0
Type	MIQP	MBQCQP

Future work

Implementations of the algorithm

- Mathematica / Wolfram Language. Early prototype “available”.
- COIN-OR: Utilize the Optimization Services and Open Solver Interface APIs.
- GAMS

Future work

Implementations of the algorithm

- Mathematica / Wolfram Language. Early prototype "available".
- COIN-OR: Utilize the Optimization Services and Open Solver Interface APIs.
- GAMS

Development of the algorithm

- Pseudoconvex constraints and objective functions.
- Selection (update) strategies of the interior point.
- Strategies for the LP1/LP2 steps.

Thank you!