Separation, Propagation, Heuristics

Selected tricks from an advanced MINLP solver

Ambros Gleixner, Zuse Institute Berlin

joint work with T. Berthold, B. Müller, F. Serrano, R. Schwarz, and S. Weltge

MINO / COST Spring School on Mixed Integer Nonlinear Programming and Applications · Paris · 8 April 2016

SPONSORED BY THE
Federal Ministry of Education and Research

SEPARATION: TIGHT OUTER APPROXIMATION

Several methods to solve convex mixed integer non-linear problems

- Outer approximation [Duran and Grossmann, 1986, Yuan et al., 1988, Fletcher and Leyffer, 1994]
- Extended cutting plane [Westerlund and Petterson, 1995]
- LP/NLP-based Branch-and-Bound [Quesada and Grossmann, 1992, Bonami et al., 2008, Abhishek et al, 2010, Achterberg, 2007, Berthold et al., 2009]
- Extended supporting hyperplane [Kronqvist et al., 2015]

Crucial technique: outer approximate feasible region by linearization

Example

Consider the region defined by $f(x) = x^2 \leq 1$ and the point $x^* = 2$ that we want to separate

Example

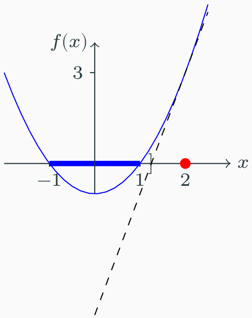
Consider the region defined by $f(x) = x^2 \leq 1$ and the point $x^* = 2$ that we want to separate

Separating hyperplane:

$$f(x^*) + \nabla f(x^*)(x - x^*) \leq 1 \Leftrightarrow 4 + 2 \cdot 2(x - 2) \leq 1 \Leftrightarrow x \leq \frac{5}{4}$$

Example

Consider the region defined by $f(x) = x^2 \leq 1$ and the point $x^* = 2$ that we want to separate

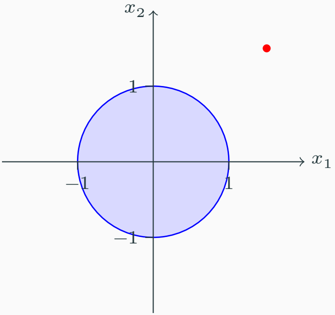


Separating hyperplane:

$$f(x^*) + \nabla f(x^*)(x - x^*) \leq 1 \Leftrightarrow 4 + 2 \cdot 2(x - 2) \leq 1 \Leftrightarrow x \leq \frac{5}{4}$$

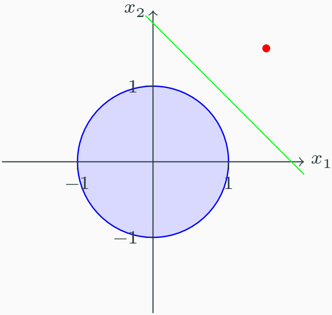
Example

$f(x) = x_1^2 + x_2^2 \leq 1$ and the point $(x_1^*, x_2^*) = (\frac{3}{2}, \frac{3}{2})$



Example

$f(x) = x_1^2 + x_2^2 \leq 1$ and the point $(x_1^*, x_2^*) = (\frac{3}{2}, \frac{3}{2})$

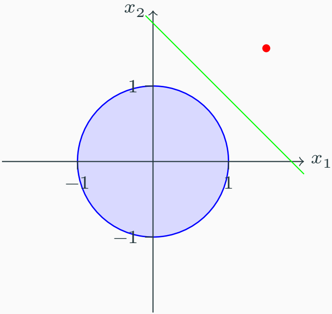


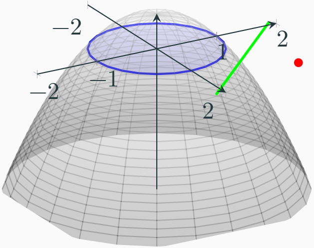
Separating hyperplane:

$$f(x^*) + \nabla f(x^*)(x - x^*) \leq 1 \Leftrightarrow x_1 + x_2 \leq \frac{11}{6}$$

Example

$f(x) = x_1^2 + x_2^2 \leq 1$ and the point $(x_1^*, x_2^*) = (\frac{3}{2}, \frac{3}{2})$



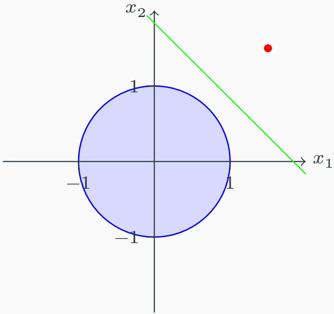


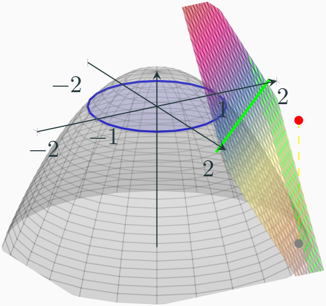
Separating hyperplane:

$$f(x^*) + \nabla f(x^*)(x - x^*) \leq 1 \Leftrightarrow x_1 + x_2 \leq \frac{11}{6}$$

Example

$f(x) = x_1^2 + x_2^2 \leq 1$ and the point $(x_1^*, x_2^*) = (\frac{3}{2}, \frac{3}{2})$





Separating hyperplane:

$$f(x^*) + \nabla f(x^*)(x - x^*) \leq 1 \Leftrightarrow x_1 + x_2 \leq \frac{11}{6}$$

We will take a constraint-wise perspective and focus on convex quadratic functions.

3 approaches:

1. pushing the hyperplane
2. changing the reference point
3. changing the formulation

We will take a constraint-wise perspective and focus on convex quadratic functions.

3 approaches:

1. pushing the hyperplane
2. changing the reference point
3. changing the formulation

We will take a constraint-wise perspective and focus on convex quadratic functions.

3 approaches:

1. pushing the hyperplane
2. changing the reference point
3. changing the formulation

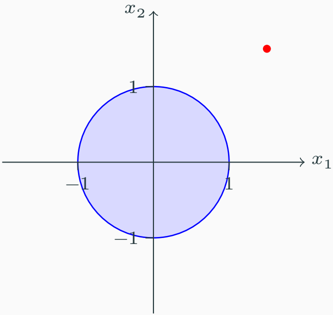
We will take a constraint-wise perspective and focus on convex quadratic functions.

3 approaches:

1. pushing the hyperplane
2. changing the reference point
3. changing the formulation

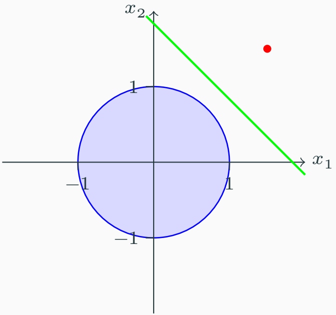
PUSHING THE HYPERPLANE

- Region $S = \{x \colon f(x) \leq c\}$ and $\bar{x} \notin S$, f convex



- Region $S = \{x \colon f(x) \leq c\}$ and $\bar{x} \notin S$, f convex
- Separating hyperplane:

$$f(\bar{x}) + \nabla f(\bar{x})(x - \bar{x}) \leq c$$

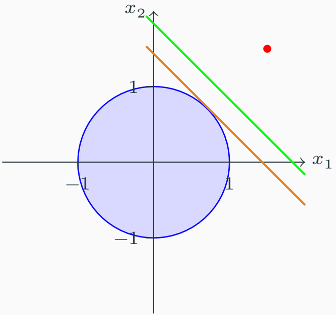


- Region $S = \{x \colon f(x) \leq c\}$ and $\bar{x} \notin S$, f convex
- Separating hyperplane:

$$f(\bar{x}) + \nabla f(\bar{x})(x - \bar{x}) \leq c$$

- Pushing the hyperplane:

$$\max_{x \in \mathbb{R}^n} \nabla f(\bar{x}) x$$
$$\text{s.t } f(x) \leq c$$

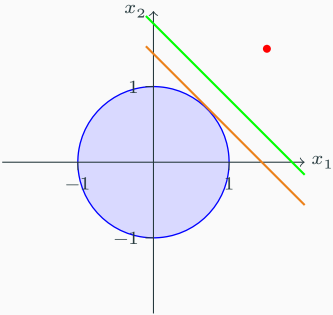


- Region $S = \{x\colon f(x) \leq c\}$ and $\bar{x} \notin S$, f convex
- Separating hyperplane:

$$f(\bar{x}) + \nabla f(\bar{x})(x - \bar{x}) \leq c$$

- Pushing the hyperplane:

$$\max_{x \in \mathbb{R}^n} \nabla f(\bar{x}) x$$
$$\text{s.t } f(x) \leq c$$



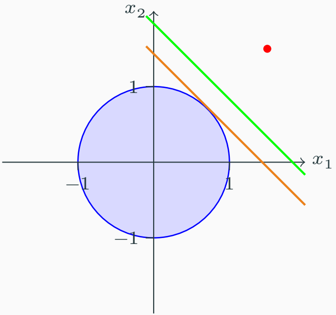
- Relatively expensive in general, simple for quadratics $f(x) = x^T A x + b^T x$

- Region $S = \{x \colon f(x) \leq c\}$ and $\bar{x} \notin S$, f convex
- Separating hyperplane:

$$f(\bar{x}) + \nabla f(\bar{x})(x - \bar{x}) \leq c$$

- Pushing the hyperplane:

$$\max_{x \in \mathbb{R}^n} \nabla f(\bar{x}) x$$
$$\text{s.t } f(x) \leq c$$



- Relatively expensive in general, simple for quadratics $f(x) = x^T A x + b^T x$
- Problem: makes sense only for strictly convex quadratics, or convex quadratics such that $b \in \text{Range}(A)$.

Given f representing S and $\bar{x} \notin S$: when is the separating hyperplane of f at $\bar{x}$ supporting?

Given f representing S and $\bar{x} \notin S$: when is the separating hyperplane of f at $\bar{x}$ supporting?

Proposition

$f\colon \mathbb{R}^n \to \mathbb{R}$ convex function, $S = \{x \in \mathbb{R}^n \mid f(x) \leq c\}$. Let $\bar{x} \notin S$ and suppose f is differentiable at $\bar{x}$. Then, the valid inequality (for S)

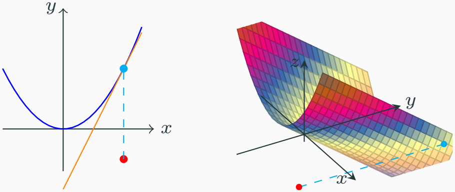
$$f(\bar{x}) + \nabla f(\bar{x})(x - \bar{x}) \leq c$$

is a supporting hyperplane of S, if and only if there exists $x_0 \in \partial S$ such that $f(x_0 + \lambda(\bar{x} - x_0))$ is affinely linear in λ.

In other words, if and only if, there is a segment joining $\bar{x}$ to the boundary of S and f is affinely linear in this segment.

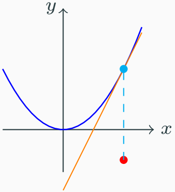
- Functions with purely linear terms always satisfy this.

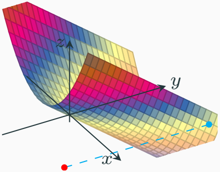
  Example: $x^2 - y \leq 0$



- Functions with purely linear terms always satisfy this.

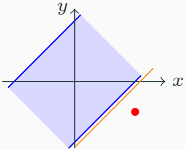
  Example: $x^2 - y \leq 0$





- Convex quadratics $f(x) = x^T Ax + b^T x$, $b \in \text{Range}(A)$ never satisfy this.

  Example: $x^2 - 2xy + y^2 \leq 1$



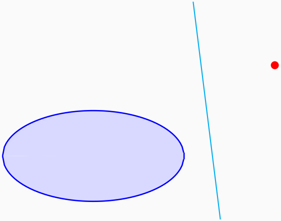
CHANGING THE REFERENCE POINT

Goal: move point $\bar{x}$ to be separated onto the boundary ∂S

Goal: move point $\bar{x}$ to be separated onto the boundary ∂S

Subgradient projection [Censor and Lent, 1980]:

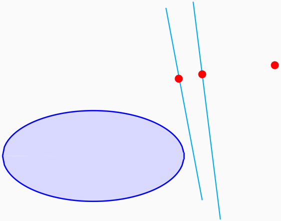
- Compute separating hyperplane for $x_0 := \bar{x}$.



Goal: move point $\bar{x}$ to be separated onto the boundary ∂S

Subgradient projection [Censor and Lent, 1980]:

- Compute separating hyperplane for $x_0 := \bar{x}$.
- $x_1 :=$ projection of x_0 onto separating hyperplane and repeat

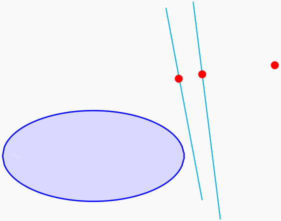


Goal: move point $\bar{x}$ to be separated onto the boundary ∂S

Subgradient projection [Censor and Lent, 1980]:

- Compute separating hyperplane for $x_0 := \bar{x}$.
- $x_1 :=$ projection of x_0 onto separating hyperplane and repeat
- Defines the following iteration:

$$x_{k+1} = x_k - \frac{f(x_k)\nabla f(x_k)}{\|\nabla f(x_k)\|^2}$$

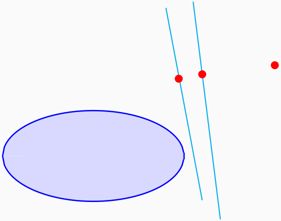


Goal: move point $\bar{x}$ to be separated onto the boundary ∂S

Subgradient projection [Censor and Lent, 1980]:

- Compute separating hyperplane for $x_0 := \bar{x}$.
- $x_1 :=$ projection of x_0 onto separating hyperplane and repeat
- Defines the following iteration:

$$x_{k+1} = x_k - \frac{f(x_k)\nabla f(x_k)}{\|\nabla f(x_k)\|^2}$$



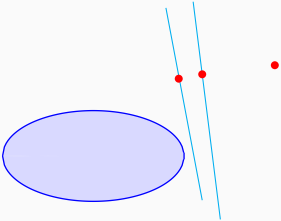
- Stop after few iterations or until the distance from x_0 to the current hyperplane decreases.

Goal: move point $\bar{x}$ to be separated onto the boundary ∂S

Subgradient projection [Censor and Lent, 1980]:

- Compute separating hyperplane for $x_0 := \bar{x}$.
- $x_1 :=$ projection of x_0 onto separating hyperplane and repeat
- Defines the following iteration:

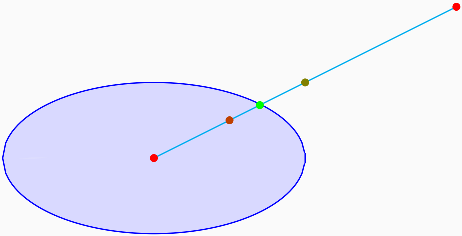
$$x_{k+1} = x_k - \lambda_k \frac{f(x_k)\nabla f(x_k)}{\|\nabla f(x_k)\|^2}$$



- Stop after few iterations or until the distance from x_0 to the current hyperplane decreases.
- Variations [Haugazeau, 1968, Bauschke and Combettes, 2001] consider projecting x_0 over the hyperplane that separates x_k

Goal: move point $\bar{x}$ to be separated onto the boundary ∂S

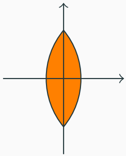
Line search towards an interior point



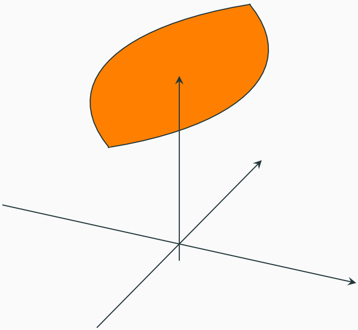
- necessity to compute interior point
- compare extended supporting hyperplane [Kronqvist et al., 2015]: line search with global interior point for all convex constraints

CHANGING THE FORMULATION

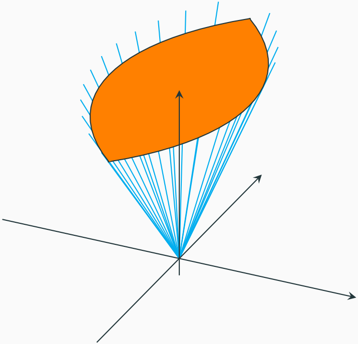
- Such a representation should behave like a cone



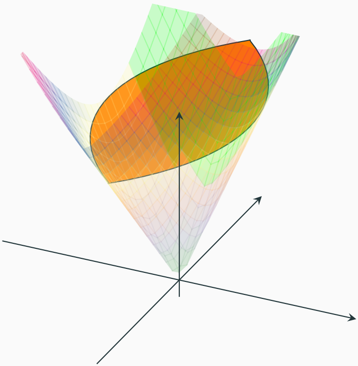
- Such a representation should behave like a cone
- $0 \in \mathring{S}$ and consider $S \times \{1\} \subset \mathbb{R}^{n+1}$



- Such a representation should behave like a cone
- $0 \in \mathring{S}$ and consider $S \times \{1\} \subset \mathbb{R}^{n+1}$
- Lines from origin to ∂S

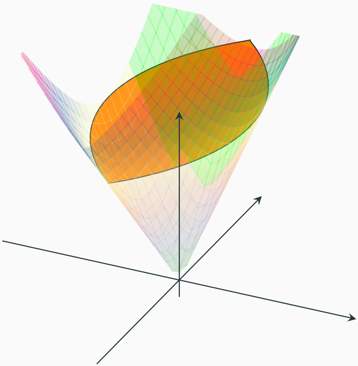


- Such a representation should behave like a cone
- $0 \in \mathring{S}$ and consider $S \times \{1\} \subset \mathbb{R}^{n+1}$
- Lines from origin to ∂S



- Such a representation should behave like a cone
- $0 \in \mathring{S}$ and consider $S \times \{1\} \subset \mathbb{R}^{n+1}$
- Lines from origin to ∂S
- Gauge function of S $(0 \in \mathring{S})$

$$\varphi_S(x) = \{ \inf t : t > 0, \; x \in tS \}$$

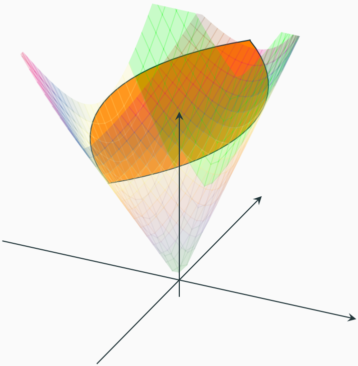


- Such a representation should behave like a cone
- $0 \in \mathring{S}$ and consider $S \times \{1\} \subset \mathbb{R}^{n+1}$
- Lines from origin to ∂S
- Gauge function of S ($0 \in \mathring{S}$)

$$\varphi_S(x) = \{\, \inf t \;:\; t > 0,\; x \in tS \,\}$$

- The gauge φ_S is convex, positively homogeneous ($\varphi_S(\lambda x) = \lambda \varphi_S(x),\ \lambda > 0$) and

$$S = \{x \colon \varphi_S(x) \leq 1\}$$



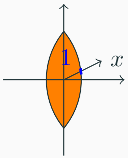
- Such a representation should behave like a cone
- $0 \in \mathring{S}$ and consider $S \times \{1\} \subset \mathbb{R}^{n+1}$
- Lines from origin to ∂S
- Gauge function of S $(0 \in \mathring{S})$

$$\varphi_S(x) = \{ \inf t : \; t > 0, \; x \in tS \}$$

- The gauge φ_S is convex, positively homogeneous $(\varphi_S(\lambda x) = \lambda \varphi_S(x), \; \lambda > 0)$ and

$$S = \{x \colon \varphi_S(x) \leq 1\}$$

- $\varphi_S(x)$ measures the relative distance from x to ∂S in the direction of x



- Let $S = \{x \colon f(x) \leq c\}$, $s_0 \in \mathring{S}$. The function that represents S is

$$\varphi_{S,s_0}(x) = \varphi_{S-s_0}(x - s_0)$$

- Let $S = \{x \colon f(x) \leq c\}$, $s_0 \in \mathring{S}$. The function that represents S is

$$\varphi_{S,s_0}(x) = \varphi_{S-s_0}(x - s_0)$$

- If $\bar{x} \notin S$ then there is $\sigma \in (0,1)$ such that

$$f(s_0 + \sigma(\bar{x} - s_0)) = c$$

and

$$\varphi_{S-s_0}(\bar{x}) = \frac{1}{\sigma}$$

HOW TO COMPUTE THE GAUGE?

- Let $S = \{x \colon f(x) \leq c\}$, $s_0 \in \mathring{S}$. The function that represents S is

$$\varphi_{S,s_0}(x) = \varphi_{S-s_0}(x - s_0)$$

- If $\bar{x} \notin S$ then there is $\sigma \in (0,1)$ such that

$$f(s_0 + \sigma(\bar{x} - s_0)) = c$$

and

$$\varphi_{S-s_0}(\bar{x}) = \frac{1}{\sigma}$$

- Such σ can be found using Newton's method, binary search, etc

- Let $f(x) = x^T Ax + b^T x$, $S = \{x\colon f(x) \leq c\}$ and $c > 0$, then

$$\varphi_S(x) = \frac{b^T x + \sqrt{(b^T x)^2 + 4cx^T Ax}}{2c}$$

- Let $f(x) = x^T Ax + b^T x$, $S = \{x\colon f(x) \le c\}$ and $c > 0$, then

$$\varphi_S(x) = \frac{b^T x + \sqrt{(b^T x)^2 + 4cx^T Ax}}{2c}$$

- If $s_0 \in \mathring{S}$, after some algebra

$$\varphi_{S,s_0}(x) = \frac{b_\varphi^T x - f(s_0) - c_\varphi + \sqrt{(b_\varphi^T x - f(s_0) - c_\varphi)^2 + 4(c - f(s_0))(f(x) - b_\varphi^T x + c_\varphi)}}{2(c - f(s_0))}$$

where

$$b_\varphi = b + 2As_0$$
$$c_\varphi = s_0^T As_0$$

- Just need to store a vector and a couple of reals

Every "optimal" representation is a rule for changing points:

- Follows from
$$\frac{x - s_0}{\varphi_{S,s_0}(x)} + s_0 \in \partial S$$
when $x \notin S$
- This is nice, since for practical reasons we might not want to change the constraint $f(x) \leq c$ to $\varphi(x) \leq 1$ (numerical tolerances, etc).

Is there any potential for some of these ideas?

Setup

- Testset: 238 instances from MINLPLib2 with at least 1 convex quadratic constraint after presolve
- Compare SCIP 3.2 with gauge (only for quadratics) vs SCIP without gauge
- 2 hours time limit
- Strategy: use any interior point (via Ipopt) + treat linear part as constant

Is there any potential for some of these ideas?

Setup

- Testset: 238 instances from MINLPLib2 with at least 1 convex quadratic constraint after presolve
- Compare SCIP 3.2 with gauge (only for quadratics) vs SCIP without gauge
- 2 hours time limit
- Strategy: use any interior point (via Ipopt) + treat linear part as constant

Results

- $+1$ instance solved
- 108 instances solved by both of them
- 10% less nodes
- 13% speedup

PROPAGATION: OBBT

- **domain reduction procedures in many areas**
  - artificial intelligence, constraint programming, satisfiability testing, ...
  - linear and integer programming
  - global optimization and mixed-integer nonlinear programming
- **general advantage**: smaller domains $\rightsquigarrow$ smaller search space
- **specifically for nonconvex MINLP**
  - branching on continuous variables/infinite domains
  - tight domains $\rightsquigarrow$ tight relaxation

FBBT AND OBBT

FBBT: Interval arithmetic on an expression graph

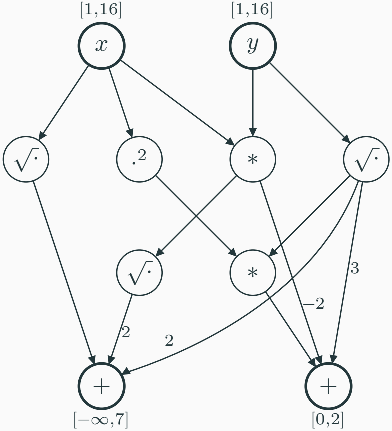
[Messine 1997, Schichl and Neumaier 2005]

FBBT: Interval arithmetic on an expression graph

[Messine 1997, Schichl and Neumaier 2005]

Example

$$\sqrt{x} + 2\sqrt{xy} + 2\sqrt{y} \in [-\infty, 7]$$
$$x^2\sqrt{y} - 2xy + 3\sqrt{y} \in [0, 2]$$
$$x, y \in [1, 16]$$



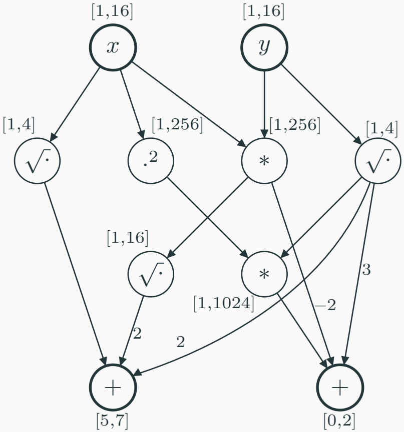
FBBT: Interval arithmetic on an expression graph

[Messine 1997, Schichl and Neumaier 2005]

Example

$$\sqrt{x} + 2\sqrt{xy} + 2\sqrt{y} \in [-\infty, 7]$$
$$x^2\sqrt{y} - 2xy + 3\sqrt{y} \in [0, 2]$$
$$x, y \in [1, 16]$$

- forward propagation



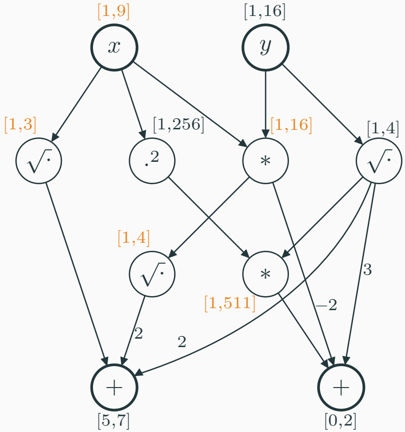
FBBT: Interval arithmetic on an expression graph

[Messine 1997, Schichl and Neumaier 2005]

Example

$$\sqrt{x} + 2\sqrt{xy} + 2\sqrt{y} \in [-\infty, 7]$$
$$x^2\sqrt{y} - 2xy + 3\sqrt{y} \in [0, 2]$$
$$x, y \in [1, 16]$$

- forward propagation
- backward propagation

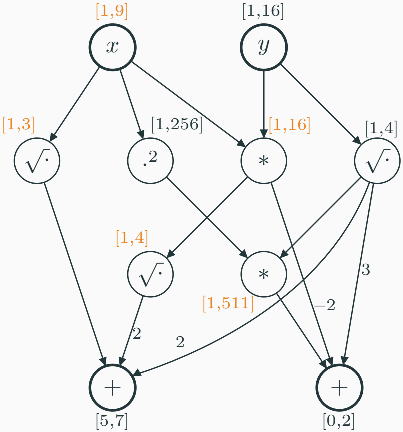


FBBT: Interval arithmetic on an expression graph

[Messine 1997, Schichl and Neumaier 2005]

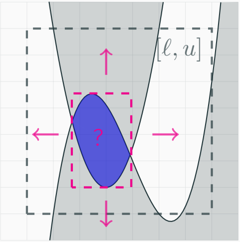
Example

$$\sqrt{x} + 2\sqrt{xy} + 2\sqrt{y} \in [-\infty, 7]$$
$$x^2\sqrt{y} - 2xy + 3\sqrt{y} \in [0, 2]$$
$$x, y \in [1, 16]$$



- forward propagation
- backward propagation
- even for linear constraints, iterative FBBT may stall

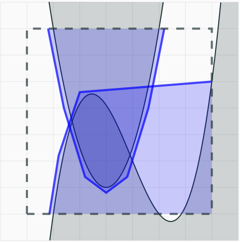
  [see Belotti, Cafieri, Lee, Liberti 2010]



$$\text{min/max}\,\{\, x_k \mid g_1(x), \dots, g_m(x) \le 0,\; x \in [\ell, u],\; x_j \in \mathbb{Z} \text{ for } j \in \mathcal{I} \}$$

OBBT is a standard technique in nonconvex MINLP algorithms

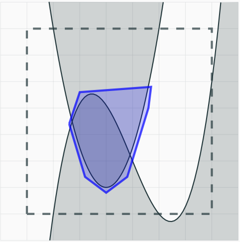
[Quesada and Grossmann 1993, Maranas and Floudas 1997, Smith and Pantelides 1999, Zamora and Grossmann 1999, Adjiman et al. 1998, 2000, Nowak and Vigerske 2006, Belotti et al. 2009, Caprara and Locatelli 2010, Misener and Floudas 2012, Gleixner and Weltge 2013, ...]



$$\min/\max \{ x_k \mid Ax \geq b, \\ x \in [\ell, u] \}$$

OBBT is a standard technique in nonconvex MINLP algorithms

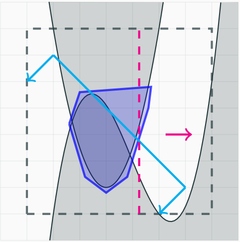
[Quesada and Grossmann 1993, Maranas and Floudas 1997, Smith and Pantelides 1999, Zamora and Grossmann 1999, Adjiman et al. 1998, 2000, Nowak and Vigerske 2006, Belotti et al. 2009, Caprara and Locatelli 2010, Misener and Floudas 2012, Gleixner and Weltge 2013, ...]



$$\text{min/max}\,\{\, x_k \mid\; Ax \geq b,\; c^\mathsf{T} x \leq z^*,\; x \in [\ell, u] \,\}$$

OBBT is a standard technique in nonconvex MINLP algorithms

[Quesada and Grossmann 1993, Maranas and Floudas 1997, Smith and Pantelides 1999, Zamora and Grossmann 1999, Adjiman et al. 1998, 2000, Nowak and Vigerske 2006, Belotti et al. 2009, Caprara and Locatelli 2010, Misener and Floudas 2012, Gleixner and Weltge 2013, ...]

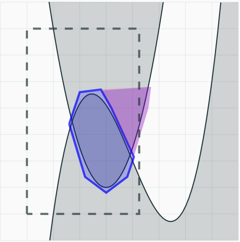


$$\min/\max \{ x_k \mid Ax \geq b, \quad c^\mathsf{T} x \leq z^*, \quad x \in [\ell, u] \}$$

OBBT is a standard technique in nonconvex MINLP algorithms

- objective cutoff constraint $\rightsquigarrow$ optimality-based procedure

[Quesada and Grossmann 1993, Maranas and Floudas 1997, Smith and Pantelides 1999, Zamora and Grossmann 1999, Adjiman et al. 1998, 2000, Nowak and Vigerske 2006, Belotti et al. 2009, Caprara and Locatelli 2010, Misener and Floudas 2012, Gleixner and Weltge 2013, ...]

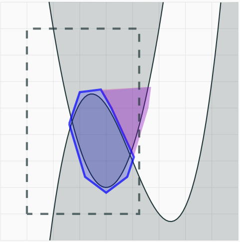


$$\min/\max \{ x_k \mid Ax \geq b, \quad c^\top x \leq z^*, \quad x \in [\ell, u] \}$$

OBBT is a standard technique in nonconvex MINLP algorithms

- objective cutoff constraint $\rightsquigarrow$ optimality-based procedure

[Quesada and Grossmann 1993, Maranas and Floudas 1997, Smith and Pantelides 1999, Zamora and Grossmann 1999, Adjiman et al. 1998, 2000, Nowak and Vigerske 2006, Belotti et al. 2009, Caprara and Locatelli 2010, Misener and Floudas 2012, Gleixner and Weltge 2013, ...]

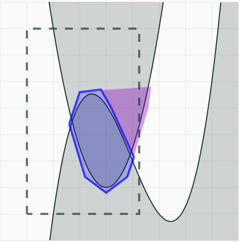


$$\text{min/max}\left\{ x_k \mid Ax \geq b, \; c^\mathsf{T} x \leq z^*, \; x \in [\ell, u] \right\}$$

OBBT is a standard technique in nonconvex MINLP algorithms

- objective cutoff constraint ⇝ optimality-based procedure
- exploits dependencies between (linear) constraints

[Quesada and Grossmann 1993, Maranas and Floudas 1997, Smith and Pantelides 1999, Zamora and Grossmann 1999, Adjiman et al. 1998, 2000, Nowak and Vigerske 2006, Belotti et al. 2009, Caprara and Locatelli 2010, Misener and Floudas 2012, Gleixner and Weltge 2013, ...]



$$\min/\max \{ x_k \mid Ax \geq b, \; c^\mathsf{T} x \leq z^*, \; x \in [\ell, u] \}$$

OBBT is a standard technique in nonconvex MINLP algorithms

- objective cutoff constraint $\rightsquigarrow$ optimality-based procedure
- exploits dependencies between (linear) constraints
- polynomial, but comparatively expensive

[Quesada and Grossmann 1993, Maranas and Floudas 1997, Smith and Pantelides 1999, Zamora and Grossmann 1999, Adjiman et al. 1998, 2000, Nowak and Vigerske 2006, Belotti et al. 2009, Caprara and Locatelli 2010, Misener and Floudas 2012, Gleixner and Weltge 2013, ...]

BOUND FILTERING FOR OBBT

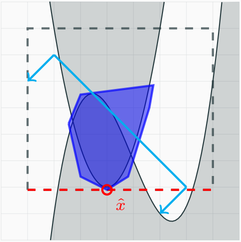
Simple observation

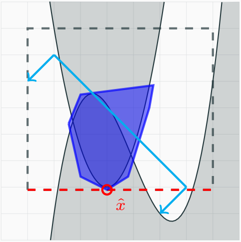
If a LP-feasible solution $\hat{x}$ is tight at a bound,

$$\hat{x} \in \{Ax \leq b, c^{\mathsf{T}}x \leq z^*, x \in [\ell, u]\}$$

and $\hat{x}_k \in \{\ell_k, u_k\}$,

this bound cannot be tightened by OBBT.





Simple observation

If a LP-feasible solution $\hat{x}$ is tight at a bound,

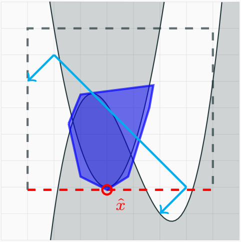
$$\hat{x} \in \{Ax \leq b, c^\mathsf{T} x \leq z^*, x \in [\ell, u]\}$$

$$\text{and } \hat{x}_k \in \{\ell_k, u_k\},$$

this bound cannot be tightened by OBBT.

Save LP solves without bound tightening success

- simple filtering: inspect available solutions
  - optimum of the LP relaxation (satisfies objective cutoff)
  - OBBT-LPs encountered along the way



Simple observation

If a LP-feasible solution $\hat{x}$ is tight at a bound,

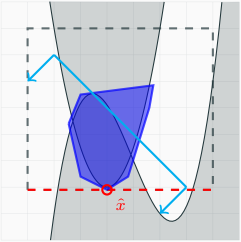
$$\hat{x} \in \{Ax \leq b, c^{\mathsf{T}}x \leq z^*, x \in [\ell, u]\}$$

and $\hat{x}_k \in \{\ell_k, u_k\}$,

this bound cannot be tightened by OBBT.

Save LP solves without bound tightening success

- simple filtering: inspect available solutions
  - optimum of the LP relaxation (satisfies objective cutoff)
  - OBBT-LPs encountered along the way
- aggressive filtering: find solutions tight at many bounds
  - minimize/maximize $\sum x_i$ ...



Simple observation

If a LP-feasible solution $\hat{x}$ is tight at a bound,

$$\hat{x} \in \{Ax \leq b, c^\mathsf{T} x \leq z^*, x \in [\ell, u]\}$$

$$\text{and } \hat{x}_k \in \{\ell_k, u_k\},$$

this bound cannot be tightened by OBBT.

Save LP solves without bound tightening success

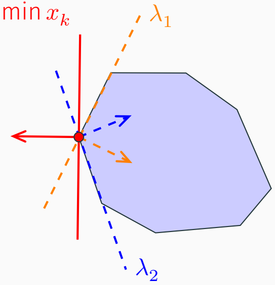
- simple filtering: inspect available solutions
  - optimum of the LP relaxation (satisfies objective cutoff) -17%
  - OBBT-LPs encountered along the way -37%
- aggressive filtering: find solutions tight at many bounds
  - minimize/maximize $\sum x_i$ ... -24%

- significantly reduces number of LP solves -78%

LAGRANGIAN VARIABLE BOUNDS

A compact explanation for OBBT's bound tightening

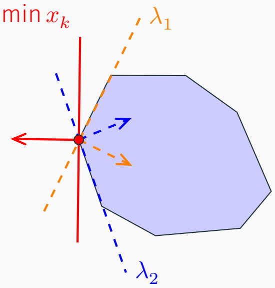
$Ax \geq b,\ \lambda \geq 0$ optimal



A compact explanation for OBBT's bound tightening

$Ax \geq b,\ \lambda \geq 0$ optimal

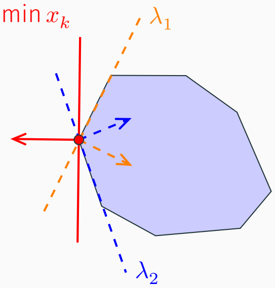
$\Rightarrow\ \lambda^{\mathsf{T}} A x \geq \lambda^{\mathsf{T}} b$



A compact explanation for OBBT's bound tightening

$Ax \geq b,\ \lambda \geq 0$ optimal

$$\Rightarrow \lambda^{\mathsf{T}} A x \geq \lambda^{\mathsf{T}} b \Leftrightarrow \sum_i \tilde{a}_i x_i \geq \lambda^{\mathsf{T}} b$$

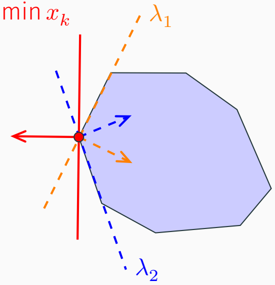


A compact explanation for OBBT's bound tightening

$Ax \geq b,\ \lambda \geq 0$ optimal

$$\Rightarrow \lambda^\mathsf{T} A x \geq \lambda^\mathsf{T} b \Leftrightarrow \sum_i \tilde{a}_i x_i \geq \lambda^\mathsf{T} b$$

$$\Rightarrow x_k \geq (1 - \tilde{a}_k) x_k - \sum_{i \neq k} \tilde{a}_i x_i + \lambda^\mathsf{T} b$$



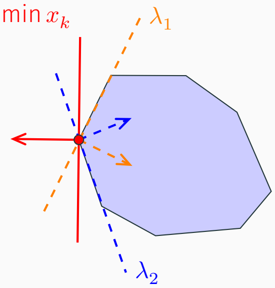
A compact explanation for OBBT's bound tightening

$Ax \geq b,\ \lambda \geq 0$ optimal

$$\Rightarrow \lambda^\mathsf{T} Ax \geq \lambda^\mathsf{T} b \Leftrightarrow \sum_i \tilde{a}_i x_i \geq \lambda^\mathsf{T} b$$

$$\Rightarrow x_k \geq (1 - \tilde{a}_k) x_k - \sum_{i \neq k} \tilde{a}_i x_i + \lambda^\mathsf{T} b$$

$$\Leftrightarrow x_k \geq \sum_i r_i x_i + \lambda^\mathsf{T} b$$



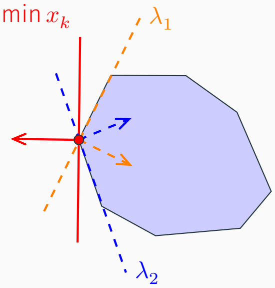
A compact explanation for OBBT's bound tightening

$Ax \geq b,\ \lambda \geq 0$ optimal

$$\Rightarrow \lambda^\mathsf{T} Ax \geq \lambda^\mathsf{T} b \Leftrightarrow \sum_i \tilde{a}_i x_i \geq \lambda^\mathsf{T} b$$

$$\Rightarrow x_k \geq (1 - \tilde{a}_k) x_k - \sum_{i \neq k} \tilde{a}_i x_i + \lambda^\mathsf{T} b$$

$$\Leftrightarrow x_k \geq \sum_i r_i x_i + \lambda^\mathsf{T} b$$



- if ℓ_k is tightened, then $\tilde{a}_k = 1 \Leftrightarrow r_k = 0$ (complementary slackness)

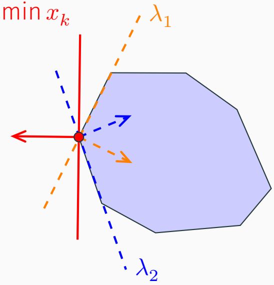
A compact explanation for OBBT's bound tightening

$$Ax \geq b,\ \lambda \geq 0 \text{ optimal}$$

$$\Rightarrow \lambda^\mathsf{T} A x \geq \lambda^\mathsf{T} b \Leftrightarrow \sum_i \tilde{a}_i x_i \geq \lambda^\mathsf{T} b$$

$$\Rightarrow x_k \geq (1 - \tilde{a}_k) x_k - \sum_{i \neq k} \tilde{a}_i x_i + \lambda^\mathsf{T} b$$

$$\Leftrightarrow x_k \geq \sum_i r_i x_i + \lambda^\mathsf{T} b$$



- if ℓ_k is tightened, then $\tilde{a}_k = 1 \Leftrightarrow r_k = 0$ (complementary slackness)
- with multiplier $\mu \leq 0$ for objective cutoff constraint:

$$x_k \geq \sum_i r_i x_i + \mu z^* + \lambda^\mathsf{T} b \qquad \text{(LVB)}$$

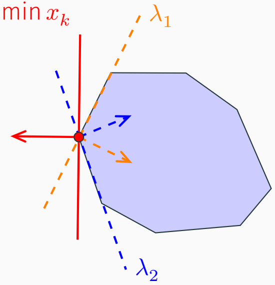
A compact explanation for OBBT's bound tightening

$Ax \geq b,\ \lambda \geq 0$ optimal

$$\Rightarrow \lambda^\mathsf{T} Ax \geq \lambda^\mathsf{T} b \Leftrightarrow \sum_i \tilde{a}_i x_i \geq \lambda^\mathsf{T} b$$

$$\Rightarrow x_k \geq (1 - \tilde{a}_k) x_k - \sum_{i \neq k} \tilde{a}_i x_i + \lambda^\mathsf{T} b$$

$$\Leftrightarrow x_k \geq \sum_i r_i x_i + \lambda^\mathsf{T} b$$



- if ℓ_k is tightened, then $\tilde{a}_k = 1 \Leftrightarrow r_k = 0$ (complementary slackness)
- with multiplier $\mu \leq 0$ for objective cutoff constraint:

$$x_k \geq \sum_{i: r_i > 0} r_i \ell_i + \sum_{i: r_i < 0} r_i u_i + \mu z^* + \lambda^\mathsf{T} b \qquad \text{(LVB)}$$

The right-hand side of $x_k \geq r^\mathsf{T} x + \mu z^* + \lambda^\mathsf{T} b$ becomes tighter

- if some ℓ_i increases for $r_i > 0$
- if some u_i decreases for $r_i < 0$
- if a better primal solution is found and $\mu < 0$

The right-hand side of $x_k \geq r^\mathsf{T} x + \mu z^* + \lambda^\mathsf{T} b$ becomes tighter

- if some ℓ_i increases for $r_i > 0$
- if some u_i decreases for $r_i < 0$
- if a better primal solution is found and $\mu < 0$

Learn LVBs during root OBBT and propagate again

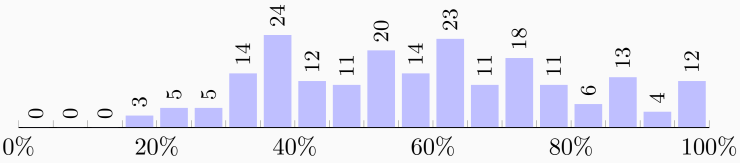
- locally at nodes of the branch-and-bound tree
- globally if a better primal solution is found
- compare “duality-based reduction” [Tawarmalani and Sahinidis 2004]

This promises a computationally cheap approximation of OBBT in the tree.

COMPUTATIONAL EXPERIMENTS

How many nontrivial LVBs can be generated?

- 211 instances from MINLPLib
- full OBBT for variables in nonlinear constraints, once at root node
- count LVBs with some $r_i \neq 0$, $i \neq k$, or $\mu \neq 0$



Histogram: rate of generated LVBs/OBBT LP distributed over test set

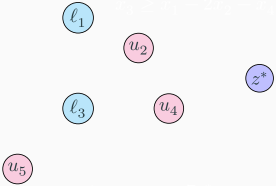
- always $\geq 15\%$, 132 times $\geq 50\%$

```
SCIP> read LiCrudeOil_ex03
SCIP> optimize
SCIP> display statistics

Propagators          : #Propagate   #ResProp    Cutoffs     DomReds
  genvbounds         :     249816        134         40       47220
  obbt               :          1          0          0          25
  probing            :          0          0          0           0
  pseudoobj          :     846391          0          0           0
  redcost            :     506542          0          0          37
  rootredcost        :          2          0          0           2
  vbounds            :     146488          0          0           0
Propagator Timings   :  TotalTime  SetupTime  Propagate     ResProp
  genvbounds         :       0.75       0.00       0.75        0.00
  obbt               :       3.77       0.00       3.77        0.00
  probing            :       0.04       0.00       0.00        0.00
  pseudoobj          :       4.08       0.00       4.08        0.00
  redcost            :       4.05       0.00       4.05        0.00
  rootredcost        :       0.39       0.00       0.39        0.00
  vbounds            :       0.79       0.00       0.79        0.00
  ...
```

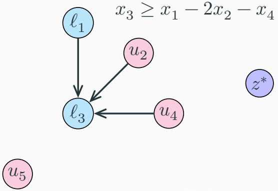
```
SCIP> read kallrath_circles.c6ax
SCIP> optimize
SCIP> display statistics

Propagators          : #Propagate    #ResProp     Cutoffs     DomReds
  genvbounds         :    1198345          0           3       81048
  obbt               :          1          0           0           0
  probing            :          0          0           0           0
  pseudoobj          :    9362769          0           0         212
  redcost            :          0          0           0           0
  rootredcost        :          8          0           0           3
  vbounds            :          0          0           0           0
Propagator Timings   :  TotalTime  SetupTime   Propagate     ResProp
  genvbounds         :       4.79       0.00        4.79        0.00
  obbt               :       0.00       0.00        0.00        0.00
  probing            :       0.00       0.00        0.00        0.00
  pseudoobj          :       5.53       0.00        5.53        0.00
  redcost            :       3.09       0.00        3.09        0.00
  rootredcost        :       2.83       0.00        2.83        0.00
  vbounds            :       3.05       0.00        3.05        0.00
  ...
```

- only propagate from right-hand side to left-hand side variable

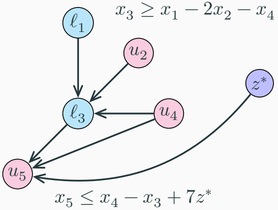
- only propagate from right-hand side to left-hand side variable
- build a directed dependency graph of lower bounds $\ell_1, \dots, \ell_n$, upper bounds $u_1, \dots, u_n$, and incumbent z^*



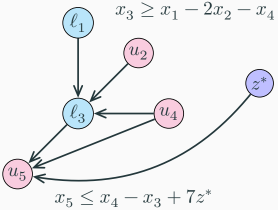
- only propagate from right-hand side to left-hand side variable
- build a directed dependency graph of lower bounds $\ell_1, \dots, \ell_n$, upper bounds $u_1, \dots, u_n$, and incumbent z^*



- only propagate from right-hand side to left-hand side variable
- build a directed dependency graph of lower bounds $\ell_1, \dots, \ell_n$, upper bounds $u_1, \dots, u_n$, and incumbent z^*

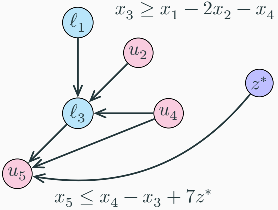


- only propagate from right-hand side to left-hand side variable
- build a directed dependency graph of lower bounds $\ell_1, \ldots, \ell_n$, upper bounds $u_1, \ldots, u_n$, and incumbent z^*
- *propagate in (almost) topological order*
  - fixed point in one sweep if acyclic
- *propagate only connected components* with bound changes



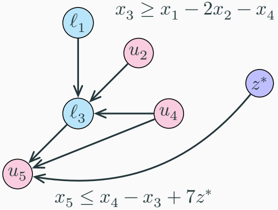
PROPAGATING LAGRANGIAN VARIABLE BOUNDS

- only propagate from right-hand side to left-hand side variable
- build a directed dependency graph of lower bounds $\ell_1, \dots, \ell_n$, upper bounds $u_1, \dots, u_n$, and incumbent z^*
- *propagate in (almost) topological order*
  - fixed point in one sweep if acyclic
- *propagate only connected components* with bound changes



	props	domreds	cutoffs	time [s]
plain	3898	856	71	1.20
sorted	1251	858	69	0.49
relative	-68%	+0%	-3%	-59%

- only propagate from right-hand side to left-hand side variable
- build a directed dependency graph of lower bounds $\ell_1, \dots, \ell_n$, upper bounds $u_1, \dots, u_n$, and incumbent z^*
- *propagate in (almost) topological order*
  - fixed point in one sweep if acyclic
- *propagate only connected components* with bound changes



	props	domreds	cutoffs	time [s]
plain	3898	856	71	1.20
sorted	1251	858	69	0.49
relative	-68%	+0%	-3%	-59%

- always $\leq 2\%$ *of total running time* (except for two easy instances)

What is the performance impact of OBBT and LVBs?

- SCIP 3.1 without OBBT, with OBBT only, and OBBT+LVB
- 605 mixed-integer and 347 continuous MINLPs from MINLPLib2

What is the performance impact of OBBT and LVBs?

- SCIP 3.1 without OBBT, with OBBT only, and OBBT+LVB
- 605 mixed-integer and 347 continuous MINLPs from MINLPLib2

Solved instances

- with OBBT: $+5$ mixed-integer / $+4$ NLP
- with OBBT+LVB: $+9$ mixed-integer / $+4$ NLP

What is the performance impact of OBBT and LVBs?

- SCIP 3.1 without OBBT, with OBBT only, and OBBT+LVB
- 605 mixed-integer and 347 continuous MINLPs from MINLPLib2

Solved instances

- with OBBT: $+5$ mixed-integer / $+4$ NLP
- with OBBT+LVB: $+9$ mixed-integer / $+4$ NLP

Performance OBBT+LVB vs. default

- mixed-integer: 2.3% faster
- continuous: 6.5% faster
- harder instances (at least 100 s): about 18% faster on both test sets
- often slowdown on easy, but game changer for hard/unsolved instances

PRIMAL HEURISTICS: UNDERCOVER

Large Neighborhood Search Heuristics

- common MIP heuristics: fix variables $\rightsquigarrow$ easy subproblem $\rightsquigarrow$ solve
- MIP: “easy” = few integralities
- MINLP: “easy” = few nonlinearities

Large Neighborhood Search Heuristics

- common MIP heuristics: fix variables $\rightsquigarrow$ easy subproblem $\rightsquigarrow$ solve
- MIP: "easy" = few integralities
- MINLP: "easy" = few nonlinearities

Observation

- any MINLP can be reduced to a MIP by fixing (sufficiently many) variables

Large Neighborhood Search Heuristics

- common MIP heuristics: fix variables $\rightsquigarrow$ easy subproblem $\rightsquigarrow$ solve
- MIP: "easy" = few integralities
- MINLP: "easy" = few nonlinearities

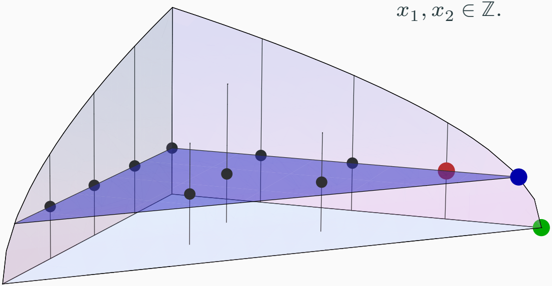
Observation

- any MINLP can be reduced to a MIP by fixing (sufficiently many) variables

Idea

- fix minimum number of variables to obtain a sub-MIP
- solution of LP/NLP relaxation as fixing values

$$
\begin{aligned}
\max \quad & x_2 + x_3 \\
\text{s.t.} \quad & x_1 + x_2 + x_3^2 \leq 4, \\
& x_1, x_2, x_3 \geq 0, \\
& x_1, x_2 \in \mathbb{Z}.
\end{aligned}
$$



Fixing x_3 to any value within its bounds yields a linear subproblem.

Definition Let us be given

- a domain box $[L, U] = \bigtimes_i [L_i, U_i]$,
- constraint functions $g_j : [L, U] \to \mathbb{R}$, $x \mapsto g_j(x)$ on $[L, U]$, and
- a set $\mathcal{C} \subseteq \mathcal{N} := \{1, \dots, n\}$ of variable indices.

Definition Let us be given

- a domain box $[L, U] = \bigtimes_i [L_i, U_i]$,
- constraint functions $g_j : [L, U] \to \mathbb{R}$, $x \mapsto g_j(x)$ on $[L, U]$, and
- a set $\mathcal{C} \subseteq \mathcal{N} := \{1, \dots, n\}$ of variable indices.

We call $\mathcal{C}$ a **cover of g_j** if and only if for all $\bar{x} \in [L, U]$ the set

$$\{(x, g_j(x)) \mid x \in [L, U], x_k = \bar{x}_k \text{ for all } k \in \mathcal{C}\}$$

is an affine set intersected with $[L, U] \times \mathbb{R}$.

Definition Let us be given

- a domain box $[L, U] = \bigtimes_i [L_i, U_i]$,
- constraint functions $g_j : [L, U] \to \mathbb{R}$, $x \mapsto g_j(x)$ on $[L, U]$, and
- a set $\mathcal{C} \subseteq \mathcal{N} := \{1, \dots, n\}$ of variable indices.

We call $\mathcal{C}$ a **cover of g_j** if and only if for all $\bar{x} \in [L, U]$ the set

$$\{(x, g_j(x)) \mid x \in [L, U], x_k = \bar{x}_k \text{ for all } k \in \mathcal{C}\}$$

is an affine set intersected with $[L, U] \times \mathbb{R}$.

We call $\mathcal{C}$ a **cover of the MINLP** if and only if $\mathcal{C}$ is a cover for $g_1, \dots, g_m$.

Definition Let P be an MINLP with $g_1, \dots, g_m$ twice continuously differentiable on the interior of $[L, U]$.

We call $G_P = (V_P, E_P)$ the co-occurrence graph of P with

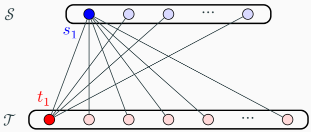
- node set $V_P = \{1, \dots, n\}$ and
- edge set $E_P = \{ij \mid i, j \in V, \exists k \in \{1, \dots, m\} : \frac{\partial^2}{\partial x_i \partial x_j} g_k(x) \not\equiv 0\}$

Definition Let P be an MINLP with $g_1, \ldots, g_m$ twice continuously differentiable on the interior of $[L, U]$.

We call $G_P = (V_P, E_P)$ the co-occurrence graph of P with

- node set $V_P = \{1, \ldots, n\}$ and
- edge set $E_P = \{ij \mid i, j \in V, \exists k \in \{1, \ldots, m\} : \frac{\partial^2}{\partial x_i \partial x_j} g_k(x) \not\equiv 0\}$

Example



min ... s.t. $s_1 t_i \le a_i$ for all $i = 1, \ldots$

$s_j t_1 \le b_j$ for all $j = 1, \ldots$

Definition Let P be an MINLP with $g_1, \dots, g_m$ twice continuously differentiable on the interior of $[L, U]$.

We call $G_P = (V_P, E_P)$ the co-occurrence graph of P with

- node set $V_P = \{1, \dots, n\}$ and
- edge set $E_P = \{ij \mid i, j \in V, \exists k \in \{1, \dots, m\} : \frac{\partial^2}{\partial x_i \partial x_j} g_k(x) \not\equiv 0\}$

Theorem [Berthold and G. 2010, 2014]
$\mathcal{C} \subseteq \{1, \dots, n\}$ is a cover of P if and only if it is a *vertex cover* of the co-occurrence graph G_P.

Definition Let P be an MINLP with $g_1, \dots, g_m$ twice continuously differentiable on the interior of $[L, U]$.

We call $G_P = (V_P, E_P)$ the co-occurrence graph of P with

- node set $V_P = \{1, \dots, n\}$ and
- edge set $E_P = \{ij \mid i, j \in V, \exists k \in \{1, \dots, m\} : \frac{\partial^2}{\partial x_i \partial x_j} g_k(x) \not\equiv 0\}$

Theorem [Berthold and G. 2010, 2014]
$\mathcal{C} \subseteq \{1, \dots, n\}$ is a cover of P if and only if it is a *vertex cover* of the co-occurrence graph G_P.

Corollary Computing a minimum cover of an MINLP is $\mathcal{NP}$-hard.

Auxiliary binary variables

$$\alpha_k = 1 :\Leftrightarrow x_k \text{ is fixed in } P$$

$\mathcal{C}(\alpha) := \{k \mid \alpha_k = 1\}$ is a cover of P if and only if

$$\alpha_k = 1 \quad \text{for all loops } kk \in E_P, \tag{1}$$

$$\alpha_k + \alpha_j \geq 1 \quad \text{for all edges } kj \in E_p, k > j. \tag{2}$$

$\rightsquigarrow$ Covering problem

$$\min\left\{\sum_{k=1}^{n} \alpha_k : (1), (2), \alpha \in \{0,1\}^n\right\}. \tag{3}$$

Auxiliary binary variables

$$\alpha_k = 1 :\Leftrightarrow x_k \text{ is fixed in } P$$

$\mathcal{C}(\alpha) := \{k \mid \alpha_k = 1\}$ is a cover of P if and only if

$$\alpha_k = 1 \quad \text{for all loops } kk \in E_P, \tag{1}$$

$$\alpha_k + \alpha_j \geq 1 \quad \text{for all edges } kj \in E_p, k > j. \tag{2}$$

⇝ Covering problem

$$\min\left\{\sum_{k=1}^{n} \alpha_k : (1), (2), \alpha \in \{0,1\}^n\right\}. \tag{3}$$

Auxiliary binary variables

$$\alpha_k = 1 :\Leftrightarrow x_k \text{ is fixed in } P$$

$\mathcal{C}(\alpha) := \{k \mid \alpha_k = 1\}$ is a cover of P if and only if

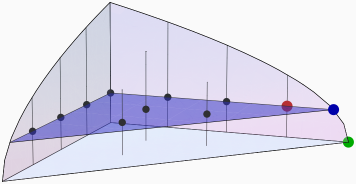
$$\alpha_k = 1 \qquad \text{for all loops } kk \in E_P, \tag{1}$$

$$\alpha_k + \alpha_j \geq 1 \qquad \text{for all edges } kj \in E_p, k > j. \tag{2}$$

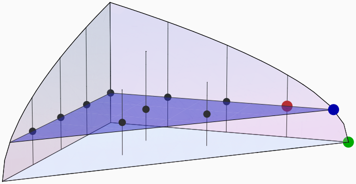
$\rightsquigarrow$ Covering problem

$$\min\left\{\sum_{k=1}^{n} \alpha_k : (1), (2), \alpha \in \{0,1\}^n\right\}. \tag{3}$$

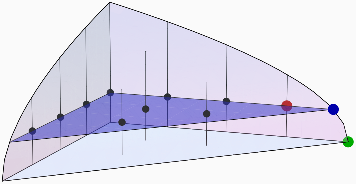
```
Input: MINLP P
begin
    compute a solution x̄
    of an approximation of P;
    round x̄_k for all k ∈ 𝓘;
    determine a
    cover 𝒞 of P;
    solve the sub-MIP of P
    given by fixing x_k = x̄_k for
    all k ∈ 𝒞;
```



Input: MINLP P
begin
  compute a solution $\bar{x}$ of an approximation of P;
  round $\bar{x}_k$ for all $k \in \mathcal{I}$;
  determine a cover $\mathcal{C}$ of P;
  solve the sub-MIP of P given by fixing $x_k = \bar{x}_k$ for all $k \in \mathcal{C}$;



```
Input: MINLP P
begin
    compute a solution x̄ of an approximation of P;
    round x̄_k for all k ∈ 𝓘;
    determine a cover 𝒞 of P;
    solve the sub-MIP of P given by fixing x_k = x̄_k for all k ∈ 𝒞;
```



Input: MINLP P

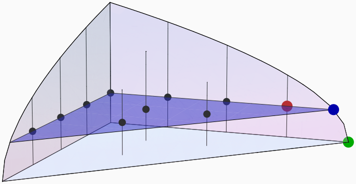
begin

compute a solution $\bar{x}$ of an approximation of P;

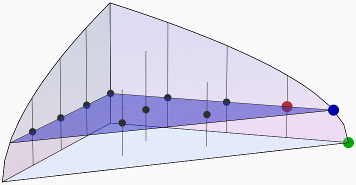
round $\bar{x}_k$ for all $k \in \mathcal{I}$;

determine a cover $\mathcal{C}$ of P;

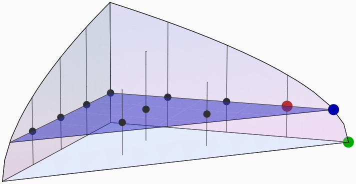
solve the sub-MIP of P given by fixing $x_k = \bar{x}_k$ for all $k \in \mathcal{C}$;



```
Input: MINLP P
begin
    compute a solution x̄ of an approximation of P;
    round x̄_k for all k ∈ I;
    determine a cover C of P;
    solve the sub-MIP of P given by fixing x_k = x̄_k for all k ∈ C;
```



Input: MINLP P
begin
compute a solution $\bar{x}$ of an approximation of P;
round $\bar{x}_k$ for all $k \in \mathcal{I}$;
determine a cover $\mathcal{C}$ of P;
solve the sub-MIP of P given by fixing $x_k = \bar{x}_k$ for all $k \in \mathcal{C}$;



Remark:

- MIP heuristics: trade-off fixing **many vs. few** variables

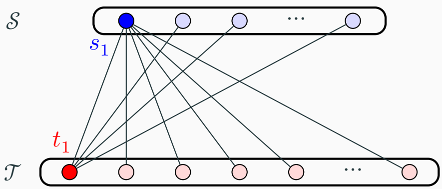
  here: eliminate nonlinearities by fixing **as few as possible** variables

  $\rightarrow$ minimum cover!

The co-occurence graph of the bilinear program

$$\min \quad \dots \quad \text{s.t.} \quad s_1 t_i \le a_i \text{ for all } i = 1, \dots,$$
$$s_j t_1 \le b_j \text{ for all } j = 1, \dots,$$

is



The cover $\mathcal{S}$ of complicating variables may be arbitrarily large compared to the minimum cover $\{s_1, t_1\}$.

Fix-and-propagate [compare FischettiSalvagnin09]

- fix variables sequentially and tighten bounds after each fixing
- project fixing values to tightened domains

Backtrack

- try alternative fixing values if infeasible

Analyze infeasibility

- learn conflict/nogood constraints

NLP postprocessing

- fix integer variables in sub-MIP solution
- solve resulting sub-NLP to local optimality

Fix-and-propagate [compare FischettiSalvagnin09]

- fix variables sequentially and tighten bounds after each fixing
- project fixing values to tightened domains

Backtrack

- try alternative fixing values if infeasible

Analyze infeasibility

- learn conflict/nogood constraints

NLP postprocessing

- fix integer variables in sub-MIP solution
- solve resulting sub-NLP to local optimality

Fix-and-propagate [compare FischettiSalvagnin09]

- fix variables sequentially and tighten bounds after each fixing
- project fixing values to tightened domains

Backtrack

- try alternative fixing values if infeasible

Analyze infeasibility

- learn conflict/nogood constraints

NLP postprocessing

- fix integer variables in sub-MIP solution
- solve resulting sub-NLP to local optimality

Fix-and-propagate [compare FischettiSalvagnin09]

- fix variables sequentially and tighten bounds after each fixing
- project fixing values to tightened domains

Backtrack

- try alternative fixing values if infeasible

Analyze infeasibility

- learn conflict/nogood constraints

NLP postprocessing

- fix integer variables in sub-MIP solution
- solve resulting sub-NLP to local optimality

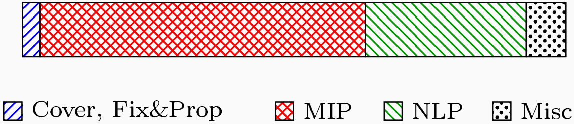
Test set

- 149 MIQCPs from GloMIQO test set

Comparison to other heuristics

- Undercover: solution for 76 instances (typically less than 0.1 sec)
- root heuristics: Baron 65, Couenne 55, SCIP 98
- lower success rate on general MINLPs

Running time distribution



SUMMARY

SUMMARY

Separation

- several methods to separate by tight (supporting) hyperplanes
- reformulation vs. changing the separated point

Separation

- several methods to separate by tight (supporting) hyperplanes
- reformulation vs. changing the separated point

Propagation

- optimization-based bound tightening
- propagate valid inequalities from dual information

Separation

- several methods to separate by tight (supporting) hyperplanes
- reformulation vs. changing the separated point

Propagation

- optimization-based bound tightening
- propagate valid inequalities from dual information

Heuristics

- Undercover: largest sub-MIP of an MINLP
- automatic structure detection via co-occurence graph

A toolbox for generating and solving constraint integer programs.

Free for academic use, available in source code at http://scip.zib.de/.

ZIMPL

- model and generate LPs, MIPs, and MINLPs

SCIP

- MIP, MINLP and CIP solver, branch-cut-and-price framework

SoPlex

- revised primal and dual simplex algorithm

GCG

- generic branch-cut-and-price solver

UG

- framework for parallelization of MIP and MINLP solvers

- 27 active developers
  - 4 running Bachelor and Master projects
  - 15 running PhD projects
  - 8 postdocs and professors
- 4 development centers in Germany
  - Aachen: GCG
  - Berlin: SCIP, SoPlex, UG, ZIMPL
  - Darmstadt: SCIP and SCIP-SDP
  - Erlangen-Nürnberg: SCIP

- many international contributors and users
  - more than 8 000 downloads per year from over 100 countries
- careers
  - 10 awards for Masters and PhD theses: MOS, EURO, GOR, DMV
  - 7 former developers are now building commercial optimization software at CPLEX, FICO Xpress, Gurobi, MOSEK, and GAMS