# Global Optimization with SCIP 8

**Ksenia Bestuzheva, Antonia Chmiela, Benjamin Müller, Felipe Serrano, Stefan Vigerske, Fabian Wegscheider**

HUGO 2022 – XV. Workshop on Global Optimization
September 8th

# TOC

# Introduction

# What is SCIP?

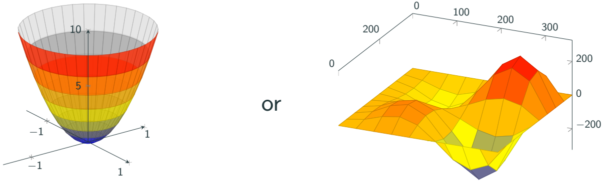
**SCIP (Solving Constraint Integer Programs) . . .**

- provides a full-scale MIP and MINLP solver,
- incorporates
  - MIP features (cutting planes, LP relaxation), and
  - MINLP features (spatial branch-and-bound, NLP relaxation)
  - CP features (domain propagation),
  - SAT-solving features (conflict analysis, restarts),
- is a branch-cut-and-price framework,
- has a modular structure via plugins,
- is free for academic purposes,
- and available in source-code under https://www.scipopt.org

# Mixed-Integer Nonlinear Programming

$$
\begin{aligned}
\min\ & c^\mathsf{T} x \\
\text{s.t. } & g_k(x) \le 0 && \forall k \in [m] \\
& x_i \in \mathbb{Z} && \forall i \in \mathcal{I} \subseteq [n] \\
& x_i \in [\ell_i, u_i] && \forall i \in [n]
\end{aligned}
$$

The functions $g_k : [\ell, u] \to \mathbb{R}$ can be




convex nonconvex

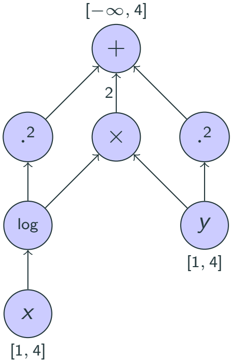
and are given in algebraic form.

# Expression Trees

The algebraic structure of nonlinear constraints is stored in a directed acyclic graph:

- nodes: variables, operations
- arcs: flow of computation

$$\log(x)^2 + 2\log(x)y + y^2 \in [-\infty, 4]$$
$$x, y \in [1, 4]$$

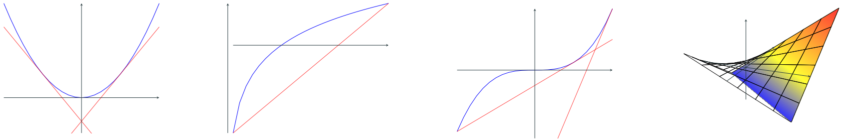


# Branch and Bound

SCIP solves MINLPs by spatial Branch & Bound.

LP relaxation via convexification and linearization:

convex functions    concave functions    $x^k \quad (k \in 2\mathbb{Z}+1)$    $x \cdot y$

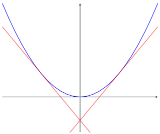


# Branch and Bound

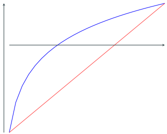
SCIP solves MINLPs by spatial Branch & Bound.

LP relaxation via convexification and linearization:

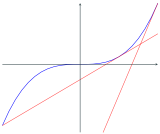
convex functions



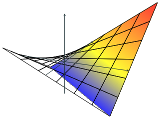
concave functions



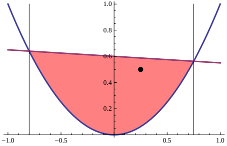
$x^k \quad (k \in 2\mathbb{Z}+1)$

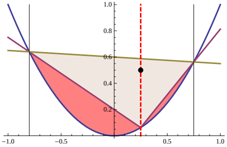


$x \cdot y$



Branching on variables in violated nonconvex constraints:



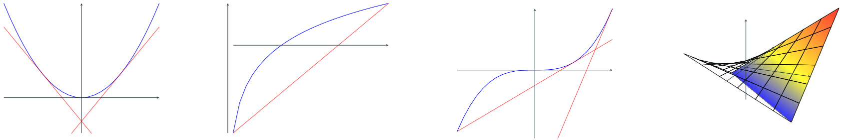



# Branch and Bound

SCIP solves MINLPs by spatial Branch & Bound.

LP relaxation via convexification and linearization:

convex functions — concave functions — $x^k \quad (k \in 2\mathbb{Z}+1)$ — $x \cdot y$



Branching on variables in violated nonconvex constraints:


. . . and bound tightening (FBBT, OBBT), primal heuristics (e.g., sub-NLP/MIP/MINLP), other special techniques

# SCIP 7 (and before)

# Reformulation in Presolve

Goal: Reformulate constraints such that only elementary cases (convex, concave, odd power, quadratic) remain by introducing new variables and new constraints.

Consider

$$\begin{aligned} &\min z \\ &\text{s.t. } \exp(\ln(1000) + 1 + x\,y) \le z \\ &\quad\quad x^2 + y^2 \le 2 \end{aligned}$$

# Reformulation in Presolve

Goal: Reformulate constraints such that only elementary cases (convex, concave, odd power, quadratic) remain by introducing new variables and new constraints.

Consider

$$\begin{aligned} &\min z \\ &\text{s.t. } \exp(\ln(1000) + 1 + x\,y) \leq z \\ &\qquad x^2 + y^2 \leq 2 \end{aligned}$$

Reformulation takes apart $\exp(\ln(1000) + 1 + x\,y)$, thus SCIP actually solves the extended formulation

$$\begin{aligned} &\min z \\ &\text{s.t. } \exp(w) \leq z \\ &\qquad \ln(1000) + 1 + x\,y = w \\ &\qquad x^2 + y^2 \leq 2 \end{aligned}$$

# Issue with explicit reformulation

SCIP solves reformulated problem fine:

```
SCIP Status        : problem is solved [optimal solution found]
Solving Time (sec) : 0.08
Solving Nodes      : 5
Primal Bound       : +9.99999656552062e+02 (3 solutions)
Dual Bound         : +9.99999656552062e+02
Gap                : 0.00 %
```

# Issue with explicit reformulation

SCIP solves reformulated problem fine:

```
SCIP Status        : problem is solved [optimal solution found]
Solving Time (sec) : 0.08
Solving Nodes      : 5
Primal Bound       : +9.99999656552062e+02 (3 solutions)
Dual Bound         : +9.99999656552062e+02
Gap                : 0.00 %
```

Solution $(x, y, z, w) = (-1.000574549, 0.999425451, 999.999656552, 6.907754936)$ looks ok:

| | Violation |
|---|---|
| $\min z$ | |
| s.t. $\exp(w) \leq z$ | $0.4659 \cdot 10^{-6} \leq$ `feastol` ✓ |
| $\ln(1000) + 1 + x\,y = w$ | $0.6731 \cdot 10^{-6} \leq$ `feastol` ✓ |
| $x^2 + y^2 \leq 2$ | $0.6602 \cdot 10^{-6} \leq$ `feastol` ✓ |

# Issue with explicit reformulation

SCIP solves reformulated problem ~~fine~~:

```
SCIP Status        : problem is solved [optimal solution found]
Solving Time (sec) : 0.08
Solving Nodes      : 5
Primal Bound       : +9.99999656552062e+02 (3 solutions)
Dual Bound         : +9.99999656552062e+02
Gap                : 0.00 %
```

Solution $(x, y, z, w) = (-1.000574549, 0.999425451, 999.999656552, 6.907754936)$ looks ok:

| | Violation |
|---|---|
| $\min z$ | |
| s.t. $\exp(w) \leq z$ | $0.4659 \cdot 10^{-6} \leq$ `feastol` ✓ |
| $\ln(1000) + 1 + x\,y = w$ | $0.6731 \cdot 10^{-6} \leq$ `feastol` ✓ |
| $x^2 + y^2 \leq 2$ | $0.6602 \cdot 10^{-6} \leq$ `feastol` ✓ |

However, original $\exp(\ln(1000) + 1 + x\,y) \leq z$ has too large violation:

```
[nonlinear] <e1>: exp((7.9077552789821368151 +1 (<x> * <y>)))-1<z>[C]  <= 0;
violation: right hand side is violated by 0.000673453314561812
best solution is not feasible in original problem
```

# Problem with classic approach

$\Rightarrow$ Explicit reformulation of constraints ...

- ... loses the connection to the original problem.
- ... loses distinction between original and auxiliary variables. Thus, we may branch on auxiliary variables.
- ... prevents simultaneous exploitation of overlapping structures.

# SCIP 8: updated framework

# Main Ideas

**Avoid explicit split-up of constraints**.

- introduce extended formulation as annotation to the original formulation
- use extended formulation for relaxation
- use original formulation for feasibility checking
- to resolve infeasibility in original constraints, tighten relaxation of extended formulation

# Main Ideas

**Avoid explicit split-up of constraints**.

- introduce extended formulation as annotation to the original formulation
- use extended formulation for relaxation
- use original formulation for feasibility checking
- to resolve infeasibility in original constraints, tighten relaxation of extended formulation

**Everything nonlinear is an expression**.

- represent all nonlinear constraints in one expression graph (DAG)
- all algorithms (check, separation, propagation, etc.) work on the same expression graph, no more specialized nonlinear constraints

# Main Ideas

**Avoid explicit split-up of constraints**.

- introduce extended formulation as annotation to the original formulation
- use extended formulation for relaxation
- use original formulation for feasibility checking
- to resolve infeasibility in original constraints, tighten relaxation of extended formulation

**Everything nonlinear is an expression**.

- represent all nonlinear constraints in one expression graph (DAG)
- all algorithms (check, separation, propagation, etc.) work on the same expression graph, no more specialized nonlinear constraints
- separate expression operators ($+$, $\times$) and high-level structures (quadratic, semi-continuous, second order cone, etc.)

# Expression Handlers

Each operator type (+, ×, pow, etc.) is implemented by an **expression handler**, which can provide a number of callbacks:

- evaluate and differentiate expression w.r.t. operands
- interval evaluation and tighten bounds on operands
- provide linear under- and over-estimators
- inform about curvature, monotonicity, integrality
- simplify, compare, print, parse, hash, copy, etc.

Expression handlers are like other **SCIP plugins**.
New ones can be added by users (YOU!).

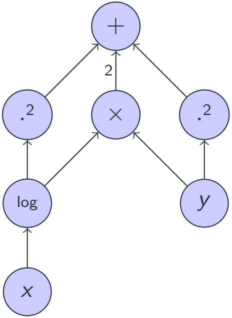
Available handler: abs, cos, entropy, exp, log, pow, product, signpow, sin, sum, value, var

# Example: Extended Formulation (exprhdlr only)

**Constraint**:

$$\log(x)^2 + 2\log(x)y + y^2 \leq 4$$

This formulation is used to check feasibility and presolve.



# Example: Extended Formulation (exprhdlr only)

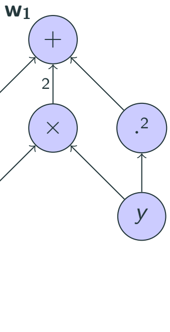
**Constraint**:

$$\log(x)^2 + 2\log(x)y + y^2 \leq 4$$

This formulation is used to check feasibility and presolve.

**Extended Formulation**:

$$w_1 \leq 4$$
$$\log(x)^2 + 2\log(x)y + y^2 = w_1$$



# Example: Extended Formulation (exprhdlr only)

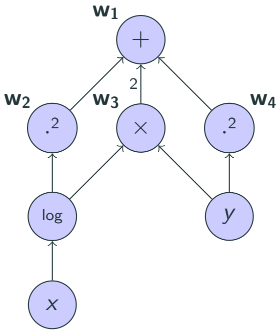
**Constraint**:

$$\log(x)^2 + 2\log(x)y + y^2 \leq 4$$

This formulation is used to check feasibility and presolve.

**Extended Formulation**:

$$\begin{aligned} w_1 &\leq 4 \\ w_2 + 2w_3 + w_4 &= w_1 \\ \log(x)^2 &= w_2 \\ \log(x)y &= w_3 \\ y^2 &= w_4 \end{aligned}$$



# Example: Extended Formulation (exprhdlr only)

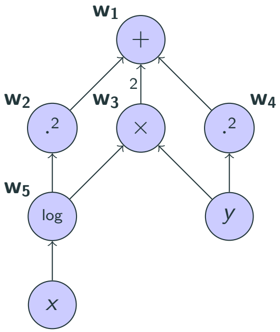
**Constraint**:

$$\log(x)^2 + 2\log(x)y + y^2 \leq 4$$

This formulation is used to check feasibility and presolve.

**Extended Formulation**:

$$\begin{aligned} w_1 &\leq 4 \\ w_2 + 2w_3 + w_4 &= w_1 \\ w_5^2 &= w_2 \\ w_5 y &= w_3 \\ y^2 &= w_4 \\ \log(x) &= w_5 \end{aligned}$$




Used to construct LP relaxation.

# Nonlinearity Handlers

But $\log(x)^2 + 2\log(x)y + y^2 \leq 4$ is convex and quadratic in $(\log(x), y)$.

# Nonlinearity Handlers

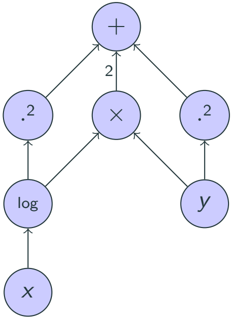
But $\log(x)^2 + 2\log(x)y + y^2 \leq 4$ is convex and quadratic in $(\log(x), y)$.

To explore structure, we now have **Nonlinearity Handler**:

- Adds additional **separation** and/or **propagation** algorithms for structures that can be identified in the expression graph.
- Attached to nodes in expression graph, but does not define expressions nor constraints.
- Examples: quadratics, convex and concave, second order cone, ...
- Several nlhdlrs can be attached to a node in the expression graph.

# Example: Extended Formulation (with nlhdlrs)

**Constraint**: $\log(x)^2 + 2\log(x)y + y^2 \leq 4$

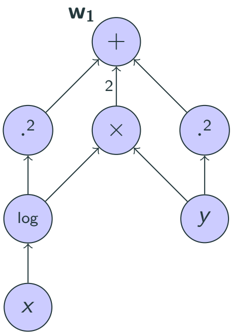


# Example: Extended Formulation (with nlhdlrs)

**Constraint**: $\log(x)^2 + 2\log(x)y + y^2 \leq 4$

1. Annotate root with auxiliary variable $w_1$.

$$w_1 \leq 4$$

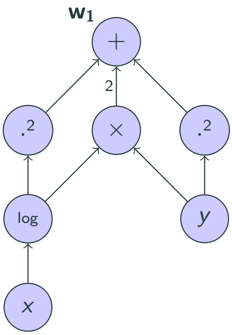


# Example: Extended Formulation (with nlhdlrs)

**Constraint**: $\log(x)^2 + 2\log(x)y + y^2 \leq 4$

$w_1 \leq 4$

1. Annotate root with auxiliary variable $w_1$.
2. Run `detect` of all nlhdlrs on $+$ node.



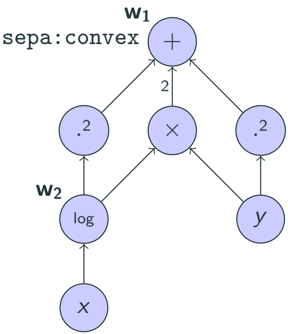
# Example: Extended Formulation (with nlhdlrs)

**Constraint**: $\log(x)^2 + 2\log(x)y + y^2 \leq 4$

1. Annotate root with auxiliary variable $w_1$.
2. Run `detect` of all nlhdlrs on $+$ node.
   - `nlhdlr_convex`
     - detects a convex quadratic structure,
     - signals that it can compute underestimators,
     - but requests an auxiliary variable $w_2$ for log node.

$$w_1 \leq 4$$

$$w_2^2 + 2w_2y + y^2 \leq w_1$$



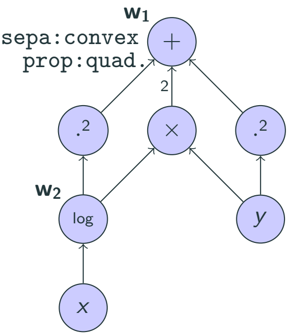
# Example: Extended Formulation (with nlhdlrs)

**Constraint**: $\log(x)^2 + 2\log(x)y + y^2 \leq 4$

1. Annotate root with auxiliary variable $w_1$.
2. Run `detect` of all nlhdlrs on $+$ node.
   - `nlhdlr_convex`
     - detects a convex quadratic structure,
     - signals that it can compute underestimators,
     - but requests an auxiliary variable $w_2$ for log node.
   - `nlhdlr_quadratic`
     - also detects a quadratic structure,
     - signals that it can do domain propagation, and
     - notifies that it will use bounds of nodes log and $y$.

$$w_1 \leq 4$$

$$w_2^2 + 2w_2 y + y^2 \leq w_1$$



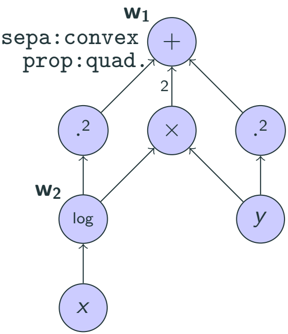
# Example: Extended Formulation (with nlhdlrs)

**Constraint**: $\log(x)^2 + 2\log(x)y + y^2 \leq 4$

1. Annotate root with auxiliary variable $w_1$.
2. Run `detect` of all nlhdlrs on $+$ node.
   - `nlhdlr_convex`
     - detects a convex quadratic structure,
     - signals that it can compute underestimators,
     - but requests an auxiliary variable $w_2$ for log node.
   - `nlhdlr_quadratic`
     - also detects a quadratic structure,
     - signals that it can do domain propagation, and
     - notifies that it will use bounds of nodes log and $y$.
3. Run `detect` of all nlhdlrs on log node.

$$w_1 \leq 4$$

$$w_2^2 + 2w_2 y + y^2 \leq w_1$$

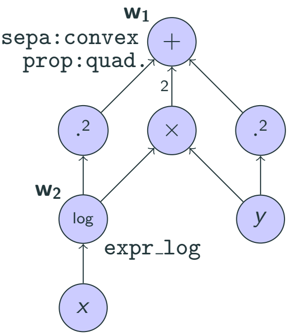


# Example: Extended Formulation (with nlhdlrs)

**Constraint**: $\log(x)^2 + 2\log(x)y + y^2 \le 4$

1. Annotate root with auxiliary variable $w_1$.
2. Run `detect` of all nlhdlrs on $+$ node.
   - `nlhdlr_convex`
     - detects a convex quadratic structure,
     - signals that it can compute underestimators,
     - but requests an auxiliary variable $w_2$ for log node.
   - `nlhdlr_quadratic`
     - also detects a quadratic structure,
     - signals that it can do domain propagation, and
     - notifies that it will use bounds of nodes log and $y$.
3. Run `detect` of all nlhdlrs on log node.
   - No specialized nlhdlr signals success.
     The expression handler will be used for both under/overestimation and propagation.

$$w_1 \le 4$$
$$w_2^2 + 2w_2y + y^2 \le w_1$$
$$\log(x) = w_2$$



## MINLP features in SCIP (teasers only)

SCIP features particular for nonlinear structures

$+$ = new or improved in SCIP 8

# Presolve

- simplify expressions

+ identify common subexpressions

# Presolve

- simplify expressions

+ identify common subexpressions

+ linearize products of binary variables

# Presolve

- simplify expressions

+ identify common subexpressions

+ linearize products of binary variables

- try to restrict variables appearing in one constraint only to its bounds, e.g., $xy + yz \leq u \rightarrow y \in \{\ell_y, u_y\}$ [Hansen et.al., 1993]

# Presolve

- simplify expressions

+ identify common subexpressions

+ linearize products of binary variables

- try to restrict variables appearing in one constraint only to its bounds,
  e.g., $xy + yz \leq u \rightarrow y \in \{\ell_y, u_y\}$ [Hansen et.al., 1993]

- if QP, add KKT as redundant constraints

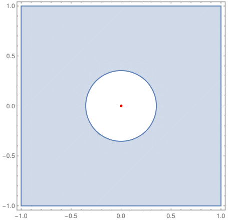
# Presolve

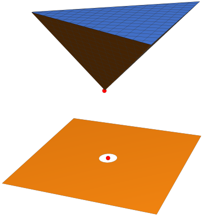
- simplify expressions
+ identify common subexpressions
+ linearize products of binary variables
- try to restrict variables appearing in one constraint only to its bounds, e.g., $xy + yz \leq u \rightarrow y \in \{\ell_y, u_y\}$ [Hansen et.al., 1993]
- if QP, add KKT as redundant constraints
+ symmetry detection using expression graph [Liberti 2010, Wegscheider 2019]

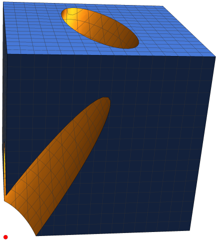
Nonlinear handler for quadratic subexpressions:

- provides domain propagation (variable bound tightening)

\+ intersection cuts for nonconvex quadratics [Chmiela, Muñoz, Serrano 2021]





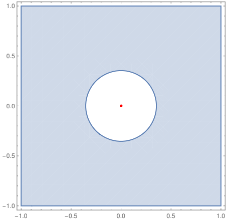
\+ also as separator for implied quadratics $\det(2 \times 2 \text{ minors of } X) = 0$, $X = xx^{\mathsf{T}}$

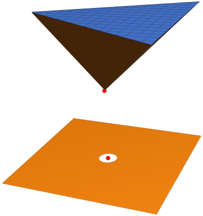
# Quadratics

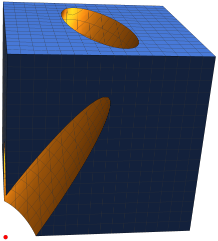
Nonlinear handler for quadratic subexpressions:

- provides domain propagation (variable bound tightening)

\+ intersection cuts for nonconvex quadratics [Chmiela, Muñoz, Serrano 2021]





\+ also as separator for implied quadratics $\det(2 \times 2 \text{ minors of } X) = 0$, $X = xx^{\mathsf{T}}$

Separator for implied PSD constraint ($X \succeq xx^{\mathsf{T}}$):

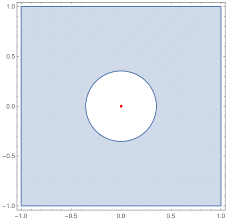
\+ SDP-cuts for $2 \times 2$ principal minors of $X - xx^{\mathsf{T}} \succeq 0$

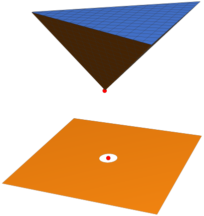
# Quadratics

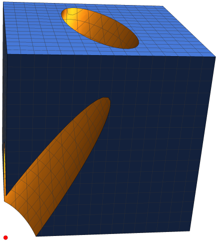
Nonlinear handler for quadratic subexpressions:

- provides domain propagation (variable bound tightening)

$+$ intersection cuts for nonconvex quadratics [Chmiela, Muñoz, Serrano 2021]





$+$ also as separator for implied quadratics $\det(2 \times 2 \text{ minors of } X) = 0$, $X = xx^{\mathsf{T}}$

Separator for implied PSD constraint ($X \succeq xx^{\mathsf{T}}$):

$+$ SDP-cuts for $2 \times 2$ principal minors of $X - xx^{\mathsf{T}} \succeq 0$

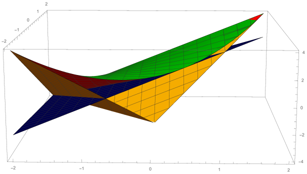
Separator for edge-concave quadratics:

- aggregate quadratic constraints to be edge-concave
- separate facets from vertex-polyhedral convex hull [Misener, Floudas 2012]

# Bilinear

Nonlinear handler for bilinear expressions:

- convexify and domain propagation for $xy$ w.r.t. additional inequalities on $x$, $y$, e.g., $x \leq y$



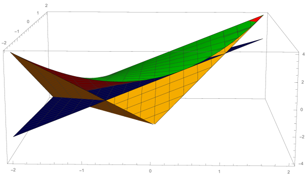
- inequalities found by 2D-projection of LP relaxation ($\approx$ OBBT)

[Linderoth 2004, Hijazi 2015, Locatelli 2016, Müller, Serrano, Gleixner 2020]

# Bilinear

Nonlinear handler for bilinear expressions:

- convexify and domain propagation for $xy$ w.r.t. additional inequalities on $x$, $y$, e.g., $x \leq y$



- inequalities found by 2D-projection of LP relaxation ($\approx$ OBBT)

[Linderoth 2004, Hijazi 2015, Locatelli 2016, Müller, Serrano, Gleixner 2020]

Reformulation Linearization Technique for bilinear products:

+ cuts from multiplication of LP rows and bounds
+ also for implicit products in mixed-binary linear problems

[Adams, Sherali 1986, Achterberg, Bestuzheva, Gleixner 2022+]

# Second-Order Cones

Nonlinear handler for Second-Order Cones:

+ detect SOC constraints from quadratics and some Euclidean norms

# Second-Order Cones

Nonlinear handler for Second-Order Cones:

+ detect SOC constraints from quadratics and some Euclidean norms

- separate using disaggregated formulation

[Vielma, et.al. 2016]

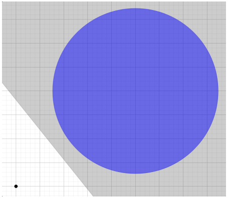
⇓

# Convexity and Concavity

Nonlinear handler for convex and concave expressions:

+ find convex/concave subexpressions using composition rules
  - gradient cuts on convex functions
  - facets of convex hull on concave function
+ prefer extended formulations for convex case

[Tawarmalani Sahinidis, 2005]



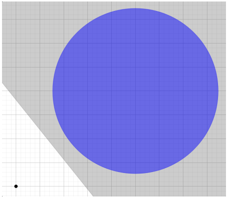
[Veinott 1967, Kronqvist, Lundell, Westerlund 2016, Serrano, Schwarz, Gleixner 2020]

# Convexity and Concavity

Nonlinear handler for convex and concave expressions:

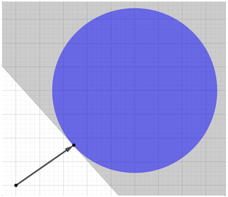
+ find convex/concave subexpressions using composition rules
  - gradient cuts on convex functions
  - facets of convex hull on concave function
+ prefer extended formulations for convex case

[Tawarmalani Sahinidis, 2005]



Separator for supporting hyperplanes:

- separators to linearize at boundary of convex NLP relaxation
- projection of given point, or linesearch to interior point



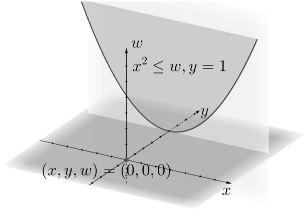
[Veinott 1967, Kronqvist, Lundell, Westerlund 2016, Serrano, Schwarz, Gleixner 2020]

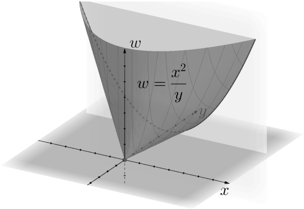
# Perspective Strengthening

Nonlinear handler for perspective strengthening:

+ detect expressions in semi-continuous variables ($\ell_x y \leq x \leq u_x y$, $y \in \{0, 1\}$)
+ strengthen under/overestimators for such expressions, e.g.,

$$f(\hat{x}) + \nabla f(\hat{x})(x - \hat{x}) \leq w \quad \Rightarrow \quad f(\hat{x})y + \nabla f(\hat{x})(x - \hat{x}y) \leq w$$




[Frangioni, Gentile 2006, Bestuzheva, Gleixner, Vigerske 2021]

# Quotients

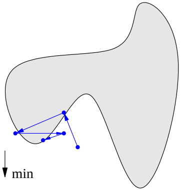
Nonlinear handler for quotients:

- to avoid ambiguity, there is no expression type for quotients

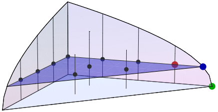
+ detect $\frac{ax+b}{cy+d}$ in nonlinear handler
+ provide bound tightening
+ provide linear under/overestimates [Zamora, Grossmann 1998]

# Primal Heuristics

- fix all integer vars, solve NLP to local optimality




- multistart with constraint consensus for NLPs [Smith, Chinneck, Aitken 2013]
- NLP-diving
- solve sequence of regularized NLP reformulations of MINLP ("MPEC")
- fix nonlinear vars, solve remaining sub-MIP [Berthold, Gleixner 2014]

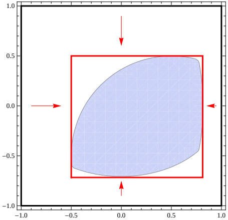


- other large neighborhood search heuristics, solving sub-MINLPs [Berthold, Heinz, Pfetsch, Vigerske 2011]

NLP solving via Ipopt and CppAD

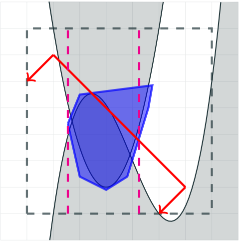
# Domain Propagation (Variable Bound Tightening)

- Feasibility-Based Bound Tightening (FBBT): as in constraint programming




- Optimization-Based Bound Tightening (OBBT): optimize var. over LP

[Gleixner, Berthold, Müller, Weltge 2017]



- OBBT over NLP relaxation

# Interfaces

Readers for MINLP included:

- MPS, LP, PIP, AMPL NL, OSiL, ZIMPL

Interfaces:

- C, Java
- Python
- Julia, Matlab
- AMPL, GAMS
- ...

# Benchmark

# Benchmark Setting

Test set:

- 183 “solvable” instances from MINLPLib
- varying degree of integrality
- varying degree of nonlinearity
- avoid too many instances with similar name
- all solvers can handle (no sine, cosine, signpower)

# Benchmark Setting

Test set:

- 183 “solvable” instances from MINLPLib
- varying degree of integrality
- varying degree of nonlinearity
- avoid too many instances with similar name
- all solvers can handle (no sine, cosine, signpower)

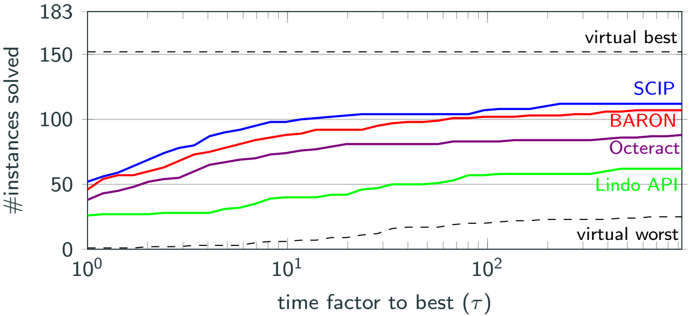
Settings:

- relative gap tolerance $10^{-4}$
- feasibility tolerance $10^{-6}$
- bound unbounded variables by $10^{12}$
- 2 hours time limit
- 1 thread
- 50 GB RAM

GAMS: all included global (still updated) MINLP solvers

# Comparison

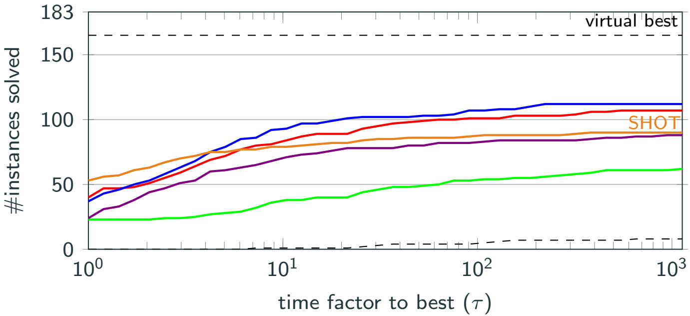
| | solved | timeout | mean time* |
|---|---|---|---|
| BARON 22.9.1 | 107 | 72 | 616s |
| Lindo API 13.0.340 | 62 | 80 | 1116s |
| Octeract 4.4.1 | 88 | 85 | 849s |
| SCIP 8.0 (f0a1490e3b) | 112 | 66 | 497s |
| virtual best | 152 | – | 109s |




*w.r.t. 135 instances on which no solver returned wrong results

# Comparison incl. SHOT with Gurobi

| | solved | timeout | mean time† |
|---|---|---|---|
| BARON 22.9.1 | 107 | 72 | 627s |
| Lindo API 13.0.340 | 62 | 80 | 1155s |
| Octeract 4.4.1 | 88 | 85 | 824s |
| SCIP 8.0 (f0a1490e3b) | 112 | 66 | 470s |
| SHOT 1.1 (033622c6) with Gurobi 9.5.2 | 90 | 18 | (59s) |
| virtual best | 165 | – | (29s) |




† w.r.t. 129 instances on which no solver returned wrong results

# End!

For details, see also the release report of SCIP 8 (and previous, and reference therein):

K. Bestuzheva, M. Besançon, W. Chen, A. Chmiela, T. Donkiewicz, J. Doornmalen, L. Eifler, O. Gaul, G. Gamrath, A. Gleixner, L. Gottwald, C. Graczyk, K. Halbig, A. Hoen, C. Hojny, R. Hulst, T. Koch, M. Lübbecke, S. Maher, F. Matter, E. Mühmer, B. Müller, M. Pfetsch, D. Rehfeldt, S. Schlein, F. Schlösser, F. Serrano, Y. Shinano, B. Sofranac, M. Turner, S. Vigerske, F. Wegscheider, P. Wellner, D. Weninger, J. Witzig, *The SCIP Optimization Suite 8.0*, ZIB Report 21-41, 2021

# PS: Testset

arki0005 arki0019 arki0020 autocorr_bern25-13 autocorr_bern30-04 autocorr_bern35-04 ball_mk4_15 batchs201210m bayes2_20 bayes2_30 bayes2_50 blend146 blend480 blend852 camshape100 camshape200 cardqp_inlp cardqp_iqp carton9 casctanks cecil_13 celar6-sub0 cesam2log chimera_k64maxcut-01 chimera_k64maxcut-02 chimera_lga-01 chp_shorttermplan1a chp_shorttermplan2a chp_shorttermplan2b clay0305hfsg color_lab3_3x0 crossdock_15x7 crossdock_15x8 crudeoil_pooling_ct3 crudeoil_pooling_dt2 crudeoil_pooling_dt4 cvxnonsep_psig20 cvxnonsep_psig30 edgecross10-050 edgecross10-070 edgecross14-039 eg_all_s eigena2 elec25 ex1223b ex1233 ex1252 ex4 ex5_3_3 ex5_4_4 ex6_1_3 ex6_2_10 ex6_2_6 ex7_2_3 ex8_5_2 ex8_5_6 faclay20h flay05h flay05m flay06m fo7_2 fo7_ar2_1 fo7_ar3_1 gabriel01 gabriel02 gabriel04 gastrans135 gastrans582_cold13 gastrans582_cold17_95 graphpart_3pm-0444-0444 graphpart_clique-30 graphpart_clique-40 gsg_0001 hadamard_5 heatexch_spec2 ibs2 ising2_5-300_5555 kall_circles_c8a kall_congruentcircles_c61 kall_diffcircles_9 kissing2 kport20 kriging_peaks-red020 kriging_peaks-red030 kriging_peaks-red050 lop97ic maxcsp-geo50-20-d4-75-36 maxcsp-langford-3-11 meanvar-orl400_05_e_7 methanol100 methanol200 milinfract multiplants_mtg2 multiplants_mtg5 multiplants_stg6 nd_netgen-2000-2-5-a-a-ns_7 nd_netgen-3000-1-1-b-b-ns_7 netmod_dol1 netmod_dol2 nous1 nvs04 nvs05 nvs06 orth_d3m6 parabol5_2_3 parallel p_ball_10b_5p_2d_h p_ball_20b_5p_3d_m p_ball_30b_5p_2d_m pedigree_ex1058 pedigree_ex485 pedigree_sp_top4_350tr pinene100 pinene200 pinene50 pointpack08 pointpack10 pooling_digabel16 pooling_epa1 pooling_epa2 popdynm100 popdynm25 popdynm50 portfol_robust100_09 portfol_robust200_03 portfol_shortfall100_04 powerflow0009r powerflow0030r qspp_0_10_0_1_10_1 qspp_0_13_0_1_10_1 qspp_0_14_0_1_10_1 radar-2000-10-a-6_lat_7 radar-3000-10-a-8_lat_7 rocket100 rocket200 rocket50 rsyn0820m03m rsyn0820m04m rsyn0840m04m sepasequ_convent sfacloc2_2_80 sfacloc2_3_90 sfacloc2_4_90 sjup2 sonet19v5 sonet20v6 sonet24v2 sonetgr17 sporttournament22 sporttournament26 sporttournament28 squfl020-150 squfl030-150 squfl030-150persp sssd15-06persp sssd15-08 sssd15-08persp st_e32 st_e36 steenbrf supplychainp1_022020 supplychainr1_022020 syn40m04hfsg synheat telecomsp_pacbell tln7 tls4 toroidal3g7_6666 tspn10 tspn12 unitcommit_200_100_1_mod_8 unitcommit_200_100_2_mod_8 unitcommit_50_20_2_mod_8 wastepaper3 wastepaper4 wastewater05m2 wastewater13m2 wastewater14m1 waterno2_03 waterno2_04 waterund11 waterund14 waterund22