Heuristics for nonconvex MINLP

Pietro Belotti, Timo Berthold
FICO, Xpress Optimization Team, Birmingham, UK
pietrobelotti@fico.com

18th Combinatorial Optimization Workshop, Aussois, 9 Jan 2014

FICO

$$
\begin{aligned}
\min \quad & f(\mathbf{x}) \\
s.t. \quad & \mathbf{g}(\mathbf{x}) \leq \mathbf{0} \\
& \mathbf{x} \in (\mathbb{Z}^p \times \mathbb{R}^{n-p}) \cap [\boldsymbol{\ell}, \mathbf{u}]
\end{aligned}
$$

Mixed-Integer Non-Linear Programming

FICO

$$
\begin{aligned}
\min \quad & f(\mathbf{x}) \\
s.t. \quad & \mathbf{g}(\mathbf{x}) \leq \mathbf{0} \\
& \mathbf{x} \in (\mathbb{Z}^p \times \mathbb{R}^{n-p}) \cap [\boldsymbol{\ell}, \mathbf{u}]
\end{aligned}
$$

Assume $f : \mathbb{R}^n \to \mathbb{R}$ and $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^m$ are factorable

Feasibility Pump heuristics[1]

Feasibility Pump heuristics are suited for problems where

- integrality constraints: $\mathbf{x} \in \mathcal{I} = \mathbb{Z}^p \times \mathbb{R}^{n-p}$ and
- continuous constraints: $\mathbf{x} \in \mathcal{C} = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{g}(\mathbf{x}) \leq \mathbf{0}\}$

coexist, such as MILP and MINLP.

[1] M. Fischetti, F. Glover, A. Lodi, "The feasibility pump", *Math. Prog.* 104(1):91-104, 2005.

Feasibility Pump heuristics[1]

Feasibility Pump heuristics are suited for problems where

- integrality constraints: $\mathbf{x} \in \mathcal{I} = \mathbb{Z}^p \times \mathbb{R}^{n-p}$ and
- continuous constraints: $\mathbf{x} \in \mathcal{C} = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{g}(\mathbf{x}) \leq \mathbf{0}\}$

coexist, such as MILP and MINLP.

Idea: generate two sequences of (possibly infeasible) solutions:

- $(\hat{\mathbf{x}}^k)_{k \in N}$: satisfies integrality, i.e. $\hat{\mathbf{x}}^k \in \mathcal{I}$
- $(\tilde{\mathbf{x}}^k)_{k \in N}$: satisfies continuous constraints, i.e., $\tilde{\mathbf{x}}^k \in \mathcal{C}$

The next point of sequence $(\tilde{\mathbf{x}}^k)$ is the closest to the previous point of sequence $(\hat{\mathbf{x}}^k)$ and viceversa.

[1] M. Fischetti, F. Glover, A. Lodi, "The feasibility pump", *Math. Prog.* 104(1):91-104, 2005.

Feasibility Pump heuristics

- Initialization: $\tilde{\mathbf{x}}^0 \in \mathcal{C}$.
- $k \leftarrow 1$
- **repeat**
- $\quad \hat{\mathbf{x}}^k \in \mathrm{argmin}_{\mathbf{x} \in \mathcal{I}} ||\mathbf{x} - \tilde{\mathbf{x}}^{k-1}||$
- $\quad \tilde{\mathbf{x}}^k \in \mathrm{argmin}_{\mathbf{x} \in \mathcal{C}} ||\mathbf{x} - \hat{\mathbf{x}}^{k}||$
- $\quad k \leftarrow k + 1$
- **until** $\hat{\mathbf{x}}^k$ or $\tilde{\mathbf{x}}^k$ is feasible

MILP: The Feasibility Pump

FICO

Consider

$$(\textbf{MILP}) \quad \min\{\mathbf{c}^\top \mathbf{x} : A\mathbf{x} \le \mathbf{b}, \mathbf{x} \in \mathbb{Z}^p \times \mathbb{R}^{n-p}\}.$$

Set $\tilde{\mathbf{x}}^0 \in \mathrm{argmin}\{\mathbf{c}^\top \mathbf{x} : A\mathbf{x} \le \mathbf{b}\}$ and $k = 1$. Then

$$\begin{aligned} \hat{\mathbf{x}}^k &= \lfloor \tilde{\mathbf{x}}^k \rceil \\ \tilde{\mathbf{x}}^k &\in \mathrm{argmin}\{||\mathbf{x} - \hat{\mathbf{x}}^{k-1}||_1 : A\mathbf{x} \le \mathbf{b}\} \end{aligned}$$

MILP variants

The original FP finds a feasible, but often bad, initial solution.

- The objective function $c^\top x$ is ignored

$\Rightarrow$ Objective FP[2]:

$$\begin{aligned} \hat{\mathbf{x}}^k &= \lfloor \tilde{\mathbf{x}}^k \rceil \\ \tilde{\mathbf{x}}^k &\in \operatorname{argmin}\{\alpha \|\mathbf{x} - \hat{\mathbf{x}}^{k-1}\|_1 + \beta \mathbf{c}^\top \mathbf{x} : A\mathbf{x} \leq \mathbf{b}\} \end{aligned}$$

Feasibility Pump 2.0[3]

- better rounding heuristics than $\lfloor \mathbf{x} \rceil$
- perturbation in the event of cycling

[2] T. Achterberg, T. Berthold, "Improving the feasibility pump". *Discrete Optimization* 4(1):77-86, 2007.

[3] M. Fischetti, D. Salvagnin, "Feasibility pump 2.0", *Math. Prog. Comp.* 1(2-3):201-222, 2009.

Convex MINLP: The enhanced Feasibility Pump[4]

The first attempt to port the FP to the MINLP realm.

- The two sequences obtained by solving two simpler problems:
  - an NLP problem
  - a MILP *approximation* of the MINLP
- MILP provided by Outer Approximation:

$$
\begin{array}{ll}
\min & z \\
s.t. & f(\tilde{\mathbf{x}}) + \nabla f(\tilde{\mathbf{x}})(\mathbf{x} - \tilde{\mathbf{x}}) \leq z \\
 & g_i(\tilde{\mathbf{x}}) + \nabla g_i(\tilde{\mathbf{x}})(\mathbf{x} - \tilde{\mathbf{x}}) \leq 0 \quad \forall i = 1, 2, \ldots, m,
\end{array}
$$

[4] P. Bonami, G. Cornuéjols, A. Lodi, F. Margot, "A Feasibility Pump for Mixed Integer Nonlinear Programs." *Math. Prog.* 119(2):331-352, 2009.

Nonconvex MINLP

- integrality constraints: $\mathbf{x} \in \mathcal{I} = \mathbb{Z}^p \times \mathbb{R}^{n-p}$ and
- nonconvex continuous constraints:
  $\mathbf{x} \in \mathcal{C} = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{g}(\mathbf{x}) \leq \mathbf{0}\}$

Previous attempts try to solve (even heuristically) the continuous nonconvex problem through a global solver or multi-start[5]

[5] C. D'Ambrosio, A. Frangioni, L. Liberti, A. Lodi, "Experiments with a feasibility pump approach for nonconvex MINLPs", *Proceedings of SEA 2010.*

Nonconvex Feasibility Pump

Take advantage of features of both the solver and the problem.

- convexification cuts: use a **valid** LP relaxation rather than Outer Approximation
- nonlinearity: use second-order information

Nonconvex problems: lower bounds

$$
\begin{array}{lll}
(\mathbf{P}_0) & \min & f(\mathbf{x}) \\
 & s.t. & \mathbf{g}(\mathbf{x}) \leq \mathbf{0} \\
 & & \mathbf{x} \in (\mathbb{Z}^p \times \mathbb{R}^{n-p}) \cap [\boldsymbol{\ell}, \mathbf{u}]
\end{array}
$$

Nonconvex problems: lower bounds

$$\begin{array}{lll} (\mathbf{P}_0) & \min & f(\mathbf{x}) \\ & s.t. & \mathbf{g}(\mathbf{x}) \leq \mathbf{0} \\ & & \mathbf{x} \in (\mathbb{Z}^p \times \mathbb{R}^{n-p}) \cap [\boldsymbol{\ell}, \mathbf{u}] \end{array}$$

is **reformulated** as an equivalent problem

$$\begin{array}{llll} (\mathbf{P}) & \min & x_{n+q} & \\ & s.t. & x_k = \vartheta_k(\mathbf{x}) & n+1 \leq k \leq n+q \\ & & \mathbf{x} \in (\mathbb{Z}^s \times \mathbb{R}^{n+q-s}) \cap [\boldsymbol{\ell}', \mathbf{u}'] & \end{array}$$

Nonconvex problems: lower bounds

$$
(\mathbf{P}_0) \quad \min \quad f(\mathbf{x}) \\
s.t. \quad \mathbf{g}(\mathbf{x}) \leq \mathbf{0} \\
\mathbf{x} \in (\mathbb{Z}^p \times \mathbb{R}^{n-p}) \cap [\boldsymbol{\ell}, \mathbf{u}]
$$

is **reformulated** as an equivalent problem

$$
(\mathbf{P}) \quad \min \quad x_{n+q} \\
s.t. \quad x_k = \vartheta_k(\mathbf{x}) \qquad n+1 \leq k \leq n+q \\
\mathbf{x} \in (\mathbb{Z}^s \times \mathbb{R}^{n+q-s}) \cap [\boldsymbol{\ell}', \mathbf{u}']
$$

and each $x_k = \vartheta_k(\mathbf{x})$ is **linearized**:

$$
(\mathbf{MILP}) \quad \min \quad x_{n+q} \\
s.t. \quad a^k x_k + A^k \mathbf{x} \geq \mathbf{b}^k \qquad n+1 \leq k \leq n+q \\
\mathbf{x} \in (\mathbb{Z}^s \times \mathbb{R}^{n+q-s}) \cap [\boldsymbol{\ell}', \mathbf{u}']
$$

NLP points and MILP points

$$
\begin{array}{llll}
(\mathbf{P}) & \min & x_{n+q} & \\
& s.t. & x_k = \vartheta_k(\mathbf{x}) & n+1 \leq k \leq n+q \\
& & \mathbf{x} \in (\mathbb{Z}^s \times \mathbb{R}^{n+q-s}) \cap [\boldsymbol{\ell}', \mathbf{u}'] &
\end{array}
$$

NLP points and MILP points

$$
\begin{array}{llll}
(\mathbf{P}) & \min & x_{n+q} & \\
& s.t. & x_k = \vartheta_k(\mathbf{x}) & n+1 \le k \le n+q \\
& & \mathbf{x} \in (\mathbb{Z}^s \times \mathbb{R}^{n+q-s}) \cap [\boldsymbol{\ell}', \mathbf{u}'] &
\end{array}
$$

$$
\begin{array}{llll}
(\mathbf{MILP}) & \min & x_{n+q} & \\
& s.t. & a^k x_k + A^k \mathbf{x} \ge \mathbf{b}^k & n+1 \le k \le n+q \\
& & \mathbf{x} \in (\mathbb{Z}^s \times \mathbb{R}^{n+q-s}) \cap [\boldsymbol{\ell}', \mathbf{u}'] &
\end{array}
$$

NLP points and MILP points

FICO

$$(\textbf{NLP}) \quad \min \quad x_{n+q}$$
$$s.t. \quad x_k = \vartheta_k(\mathbf{x}) \qquad n+1 \le k \le n+q$$
$$\mathbf{x} \in (\mathbb{R}^s \times \mathbb{R}^{n+q-s}) \cap [\boldsymbol{\ell}', \mathbf{u}']$$

$$(\textbf{MILP}) \quad \min \quad x_{n+q}$$
$$s.t. \quad a^k x_k + A^k \mathbf{x} \ge \mathbf{b}^k \qquad n+1 \le k \le n+q$$
$$\mathbf{x} \in (\mathbb{Z}^s \times \mathbb{R}^{n+q-s}) \cap [\boldsymbol{\ell}', \mathbf{u}']$$

$$\mathcal{C} = \{\mathbf{x} \in \mathbb{R}^{n+q} \cap [\boldsymbol{\ell}', \mathbf{u}'] : x_k = \vartheta_k(\mathbf{x}),\ \forall k\}$$

$$\mathcal{I} = \{\mathbf{x} \in \mathbb{Z}^s \times \mathbb{R}^{n+q-s} \cap [\boldsymbol{\ell}', \mathbf{u}'] : a^k x_k + A^k \mathbf{x} \ge b^k,\ \forall k\}$$

Nonconvex Feasibility Pump

- Initialization: $\tilde{\mathbf{x}}^0 \in \operatorname{argmin}\{f(\mathbf{x}) : \mathbf{g}(\mathbf{x}) \leq \mathbf{0}\} \in \mathcal{C}$.
- $k \leftarrow 1$
- **repeat**
- $\quad \hat{\mathbf{x}}^k \in \operatorname{argmin}_{\mathbf{x}\in\mathcal{I}} \Delta'(\mathbf{x}, \tilde{\mathbf{x}}^{k-1})$
- $\quad \tilde{\mathbf{x}}^k \in \operatorname{argmin}_{\mathbf{x}\in\mathcal{C}} \Delta''(\mathbf{x}, \hat{\mathbf{x}}^k)$
- $\quad k \leftarrow k+1$
- **until** $\hat{\mathbf{x}}^k$ or $\tilde{\mathbf{x}}^k$ is feasible

Δ' and Δ'' combine ℓ_p distance, objective function, and second order information from the problem

Distance measures: second order information

Consider the (locally!) optimal solution $\tilde{\mathbf{x}}^0$ of the continuous relaxation of the MINLP.

- If no active constraints, the Hessian of the objective is PSD:

$$\nabla^2 f(\tilde{\mathbf{x}}^0) \succeq 0$$

- In general, the Hessian of the Lagrangian provides useful info
- We consider it only in the null space of the gradient:

$$(\nabla f)^\top \mathbf{x} = x_{n+q} = f(\tilde{\mathbf{x}}^0)$$

Distance measures: second order information

Construct PSD matrix P from the Hessian of the Lagrangian by

- Obtaining a projection on the cone of PSD matrices

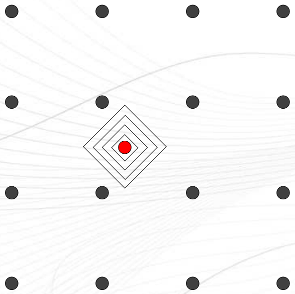
... (by eliminating negative eigenvalues)

Then use:

NLP: $\Delta''(\mathbf{x}, \bar{\mathbf{x}}) := \alpha''||\mathbf{x} - \bar{\mathbf{x}}||_2 + \beta''(\mathbf{x} - \bar{\mathbf{x}})^\top P(\mathbf{x} - \bar{\mathbf{x}}) + \gamma x_{n+q}$

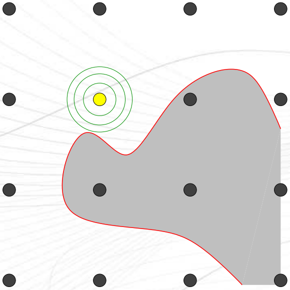
MILP: $\Delta'(\mathbf{x}, \bar{\mathbf{x}}) := \alpha'||\mathbf{x} - \bar{\mathbf{x}}||_1 + \beta'||P^{\frac{1}{2}}(\mathbf{x} - \bar{\mathbf{x}})||_1 + \gamma x_{n+q}$

Distance measures: unconstrained case

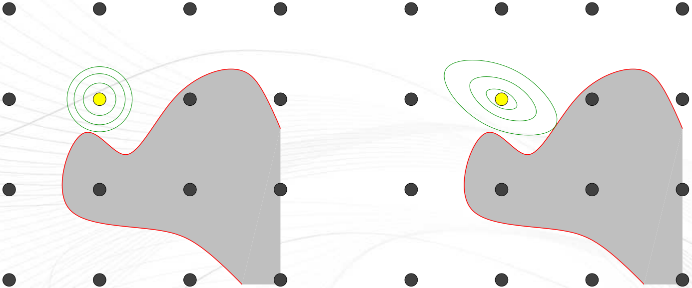


Distance measures: unconstrained case

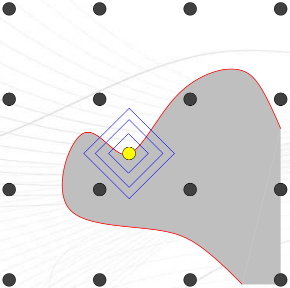
Distance measures: solving NLP



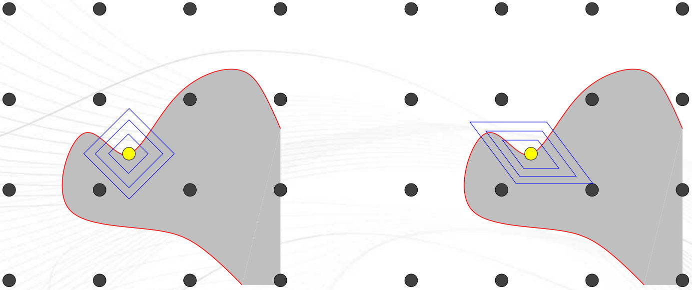
Distance measures: solving NLP



Distance measures: solving MILP



Distance measures: solving MILP



(Not) solving the MILP exactly

$(\hat{\mathbf{x}}^k)$ obtained by solving an MILP.

- Optimality takes time and is unnecessary

$\Rightarrow$ Impose node limit (10k) or

$\Rightarrow$ Use heuristics

- Optional: *adaptive* MILP method selection
  - If succeeded five times in a row, switch to cheaper method
  - If current method returns no solution, switch to more expensive one and rerun

Computational tests

Implemented in Couenne, an Open-Source MINLP solver[6].

- Available in `stable/0.4`
- Run at most once at every node until depth 5

... and with decreasing frequency afterward

- Time limit: 2 hours
- MILP (+heuristics): **SCIP**[7]
- NLP solver: **Ipopt**[8]

[6] http://www.coin-or.org/Couenne
[7] http://scip.zib.de
[8] http://www.coin-or.org/Ipopt

Computational tests: full Couenne run

Tested on 266 MINLP instances from

- MINLPLIB[9],
- MACMINLP[10].

- 180 were solved by **all variants** in less than a minute

$\Rightarrow$ ignored, 86 instances remaining

[9] http://www.gamsworld.org/minlp/minlplib.htm
[10] http://wiki.mcs.anl.gov/leyffer/index.php/MacMINLP

Compared four variants

- No heuristic (MINLP solutions found after LP solver)
- Greedy rounding heuristic (default in Couenne)
- FP1: Feasibility pump where Δ', Δ'': resp. ℓ_1 and ℓ_2 distance
- FP2: Feasibility pump with
  - NLP: .25 distance, .75 Hessian
  - MILP: .75 distance, .25 objective

Summary

	None	GreedyR	FP1	FP2
Instances solved[11]	24	34	41	34
(Strictly) better LB[12]	5	10	8	3
Best time[13]	7	11	3	3

[11] before time limit

[12] for the 35 instances not solved by any

[13] for the 24 instances solved by all

Standalone FP on 200 instances from MINLPLIB

MILP solver	best obj	worst obj	#sol	t(geo)	t(sh.geo)
adaptive	0	0	114	4.1	16.5
MILP	0	0	114	5.2	19.2
MILP, emph: feas	6	12	107	7.3	24.6
MILP, 5k nodes	3	3	113	5.2	19.1
RENS	8	24	105	3.0	13.5
ObjFP	5	22	107	3.1	13.5

Partial overlap with 250 instances from the storm[14] paper.

- ☹ Storm paper: solution found in ≈ 200 instances out of 250
- ☺ Some of our solutions (< 10) were better

[14] C. D'Ambrosio, A. Frangioni, L. Liberti, A. Lodi, "A storm of feasibility pumps for nonconvex MINLP", *Math Prog B* 136(2):375-402, 2012.

Tests on Distance/Hessian/Objective weights

Weights differ for NLP and MILP (full MILP solve)
For distance, Hessian, and objective term in Δ' and Δ'':

- $\star$: 1 (exclusive)
- $-$: decreasing as $.95^{\text{iter}}$,
- $+$: increasing as $1 - .95^{\text{iter}}$

Tests on Distance/Hessian/Objective weights

NLP			MILP					
dist	Hess	obj	dist	Hess	obj	best obj	worst obj	#feas
⋆			⋆			0	0	111
+	+	−	⋆			0	3	111
+	−	−	⋆			0	3	111
+		−	⋆			0	2	111
⋆			+	+	−	5	15	112
⋆			+	−	−	7	11	114
⋆			+		−	3	7	108
+	+	−	+	+	−	6	13	112
+	−	−	+	+	−	6	14	111
+	−	−	+	−	−	7	12	112
+		−	+		−	3	7	108

Thank you

Some results

CPU times (in seconds) or, if above time limit (2h), the lower bound in brackets

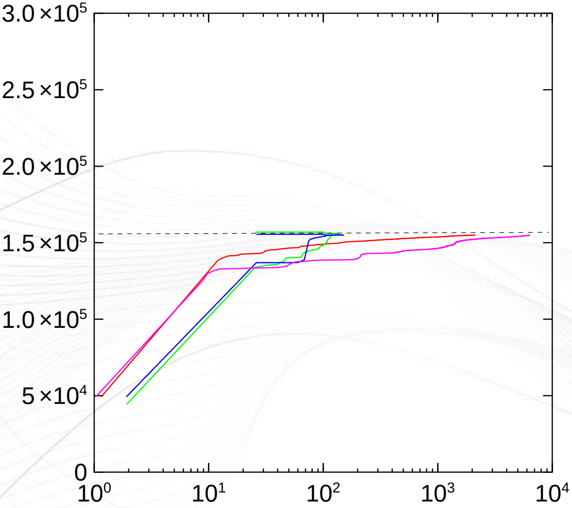
Name	#var	#int	#con	None	GR	FP1	FP2
pump	24	9	34	(78303)	(74524)	4141	5127
synheat	53	12	65	(81794)	(87461)	(154997)	1665
synh-mod	53	12	61	(81990)	(85248)	4218	1510
armin	102	80	156	(24.6)	5192	591	771
parallel	152	20	96	(-1755)	(-21)	3191	631
multist	185	18	265	(-39525)	(-24751)	(-7681)	(-7801)
product	948	92	1031	(-2238)	(-2238)	1569	2043
oil	1361	19	1417	344	1380	(-0.9)	1084

Some results (cont.)

CPU times (in seconds) or, if above 2h, lower bound in brackets

Name	#var	#int	#con	None	GR	FP1	FP2
st_e35	29	7	33	(54416)	(55302)	11	19
tln5	35	35	30	(7.4)	(6.75)	195	765
trimlon5	35	35	30	(7.4)	(6.77)	219	906
tln6	48	48	36	(9.0)	(8.89)	(12.2)	(11.3)
tln7	63	63	42	(5.4)	(5.37)	(10.5)	(9.8)
tln12	168	168	72	(20.6)	(22.25)	(23)	(24.1)
trimlon6	168	168	72	(20.5)	(22.33)	(24.0)	(23.9)
tls6	213	177	120	(1.7)	(1.76)	(6.6)	(8.5)
tls12	792	648	372	(9.9)	(9.30)	(12.1)	(12.5)
space-25	893	750	235	(105.1)	(105.1)	(71.8)	59

minlplib/ex1233



None RH FP2 FP1

Spatial B&B with convexification

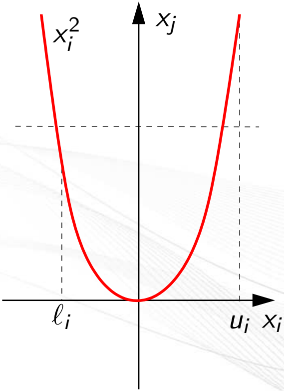
Apply a branch and bound where, at each node:

- ▶(BT) Tighten variables' bounds
- (Lb) **repeat** k times:
  - add cuts of linear env.
  - **if** no cuts found, **break**
  - get *lower bd.*, soln. $(\hat{x}, \hat{y})$
- (Ub) Look for a *feasible solution* with NLP solver `Ipopt`
- (Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i
- (NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x

Spatial B&B with convexification

Apply a branch and bound where, at each node:

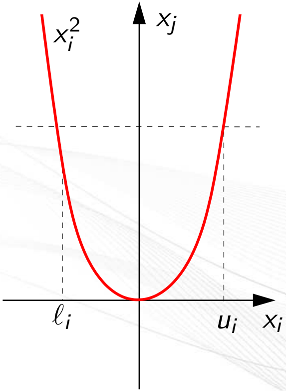
- ▶(BT) Tighten variables' bounds
- (Lb) **repeat** k times:
  - add cuts of linear env.
  - **if** no cuts found, **break**
  - get *lower bd.*, soln. $(\hat{x}, \hat{y})$
- (Ub) Look for a *feasible solution* with NLP solver `Ipopt`
- (Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i
- (NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

- ▶(BT) Tighten variables' bounds
- (Lb) **repeat** k times:
  - add cuts of linear env.
  - **if** no cuts found, **break**
  - get *lower bd.*, soln. $(\hat{x}, \hat{y})$
- (Ub) Look for a *feasible solution* with NLP solver `Ipopt`
- (Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i
- (NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

(BT) Tighten variables' bounds

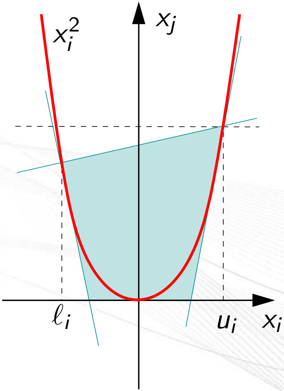
(Lb) **repeat** k times:

- add cuts of linear env.
- **if** no cuts found, **break**
- get *lower bd.*, soln. $(\hat{x}, \hat{y})$

(Ub) Look for a *feasible solution* with NLP solver `Ipopt`

(Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i

(NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

(BT) Tighten variables' bounds

(Lb) **repeat** k times:

  add cuts of linear env.

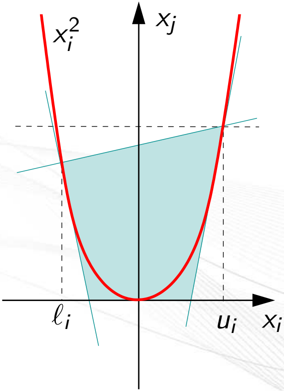
► **if** no cuts found, **break**

  get *lower bd.*, soln. $(\hat{x}, \hat{y})$

(Ub) Look for a *feasible solution* with NLP solver `Ipopt`

(Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i

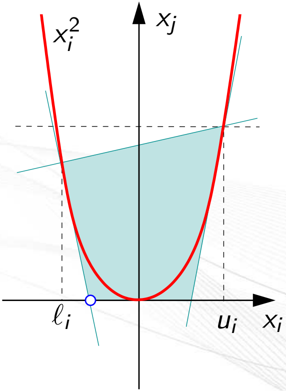
(NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

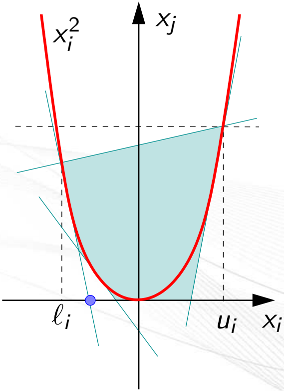
- (BT) Tighten variables' bounds
- (Lb) **repeat** k times:
  - add cuts of linear env.
  - **if** no cuts found, **break**
  - ► get *lower bd.*, soln. $(\hat{x}, \hat{y})$
- (Ub) Look for a *feasible solution* with NLP solver `Ipopt`
- (Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i
- (NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

- (BT) Tighten variables' bounds
- (Lb) **repeat** k times:
  - ▶ add cuts of linear env.
  - **if** no cuts found, **break**
  - get *lower bd.*, soln. $(\hat{x}, \hat{y})$
- (Ub) Look for a *feasible solution* with NLP solver `Ipopt`
- (Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i
- (NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

(BT) Tighten variables' bounds

(Lb) **repeat** k times:

add cuts of linear env.

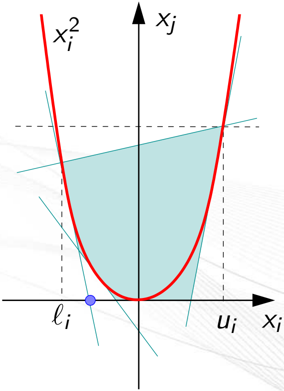
▶ **if** no cuts found, **break**

get *lower bd.*, soln. $(\hat{x}, \hat{y})$

(Ub) Look for a *feasible solution* with NLP solver `Ipopt`

(Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i

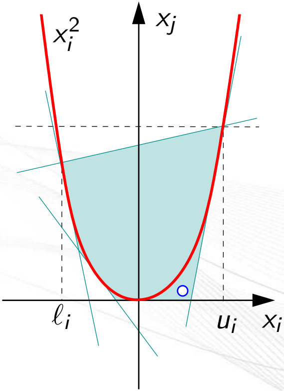
(NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

- (BT) Tighten variables' bounds
- (Lb) **repeat** k times:
  - add cuts of linear env.
  - **if** no cuts found, **break**
  - ▶ get *lower bd.*, soln. $(\hat{x}, \hat{y})$
- (Ub) Look for a *feasible solution* with NLP solver `Ipopt`
- (Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i
- (NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

(BT) Tighten variables' bounds

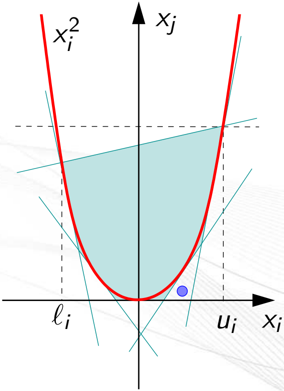
(Lb) **repeat** k times:

- add cuts of linear env.
- **if** no cuts found, **break**
- get *lower bd.*, soln. $(\hat{x}, \hat{y})$

(Ub) Look for a *feasible solution* with NLP solver `Ipopt`

(Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i

(NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

(BT) Tighten variables' bounds

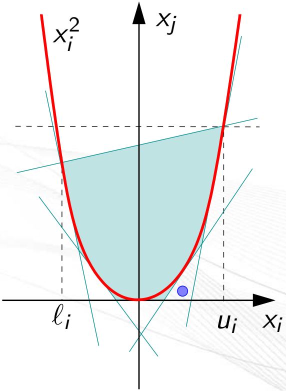
(Lb) **repeat** k times:

- add cuts of linear env.
- ▶ **if** no cuts found, **break**
- get *lower bd.*, soln. $(\hat{x}, \hat{y})$

(Ub) Look for a *feasible solution* with NLP solver `Ipopt`

(Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i

(NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

(BT) Tighten variables' bounds

(Lb) **repeat** k times:

add cuts of linear env.

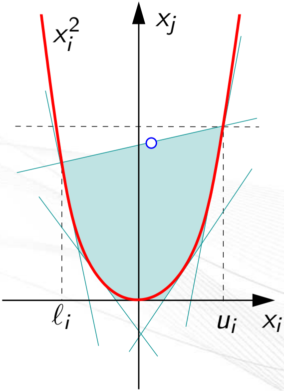
if no cuts found, **break**

▶ get *lower bd.*, soln. $(\hat{x}, \hat{y})$

(Ub) Look for a *feasible solution* with NLP solver `Ipopt`

(Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i

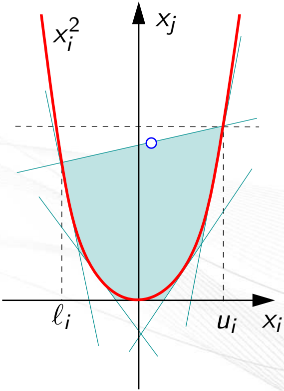
(NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

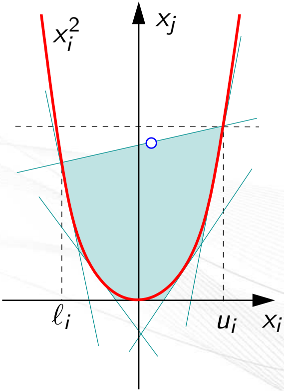
- (BT) Tighten variables' bounds
- (Lb) **repeat** k times:
  - add cuts of linear env.
  - **if** no cuts found, **break**
  - get *lower bd.*, soln. $(\hat{x}, \hat{y})$
- (Ub) Look for a *feasible solution* with NLP solver `Ipopt`
- (Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i
- (NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

- (BT) Tighten variables' bounds
- (Lb) **repeat** k times:
  - add cuts of linear env.
  - ▶ **if** no cuts found, **break**
  - get *lower bd.*, soln. $(\hat{x}, \hat{y})$
- (Ub) Look for a *feasible solution* with NLP solver `Ipopt`
- (Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i
- (NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

(BT) Tighten variables' bounds

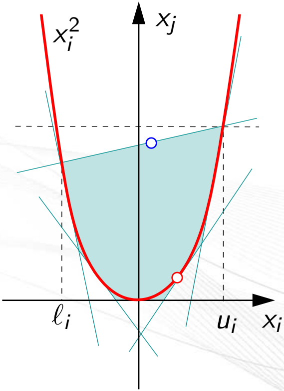
(Lb) **repeat** k times:

- add cuts of linear env.
- **if** no cuts found, **break**
- get *lower bd.*, soln. $(\hat{x}, \hat{y})$

▶(Ub) Look for a *feasible solution* with NLP solver `Ipopt`

(Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i

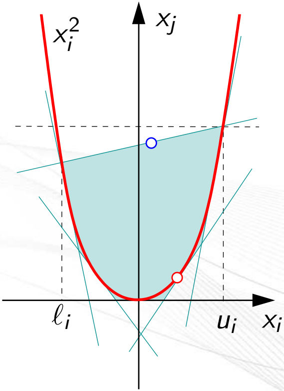
(NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

- (BT) Tighten variables' bounds
- (Lb) **repeat** k times:
  - add cuts of linear env.
  - **if** no cuts found, **break**
  - get *lower bd.*, soln. $(\hat{x}, \hat{y})$
- (Ub) Look for a *feasible solution* with NLP solver `Ipopt`
- ▶(Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i
- (NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Spatial B&B with convexification

Apply a branch and bound where, at each node:

(BT) Tighten variables' bounds

(Lb) **repeat** k times:

add cuts of linear env.

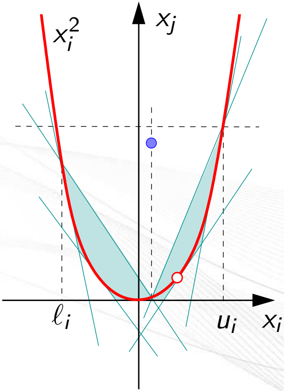
if no cuts found, **break**

get *lower bd.*, soln. $(\hat{x}, \hat{y})$

(Ub) Look for a *feasible solution* with NLP solver Ipopt

(Int Br.) If a variable $\hat{y}_i$ is fractional, branch on y_i

▶(NL Br.) If $w_i = h_i(x)$ *infeasible*, i.e. $\hat{w}_i \neq h_i(\hat{x})$, branch on x



Create linear envelope for each operator in the objective/constraints. Ex.: $y^2 \log(x^3 + 5) \geq 7$

Reformulation-Linearization

Create linear envelope for each operator in the objective/constraints. Ex.: $y^2 \log(x^3 + 5) \geq 7$

$$w_1 = x^3$$

$$w_2 = w_1 + 5$$

$$w_3 = \log w_2$$

$$w_4 = y^2$$

$$w_5 = w_4 w_3$$

$$w_5 \geq 7$$

Reformulation-Linearization

Create linear envelope for each operator in the objective/constraints. Ex.: $y^2 \log(x^3 + 5) \geq 7$

$$w_1 = x^3$$

$$w_2 = w_1 + 5$$

$$w_3 = \log w_2$$

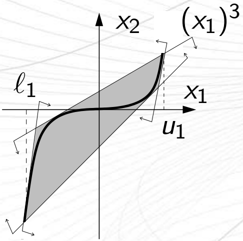
$$w_4 = y^2$$

$$w_5 = w_4 w_3$$

$$w_5 \geq 7$$

Reformulation-Linearization

Create linear envelope for each operator in the objective/constraints. Ex.: $y^2 \log(x^3 + 5) \geq 7$



$$w_1 = x^3$$

$$w_2 = w_1 + 5$$

$$w_3 = \log w_2$$

$$w_4 = y^2$$

$$w_5 = w_4 w_3$$

$$w_5 \geq 7$$

$$a_1 w_1 + a_2 x \leq b_1$$
$$a_3 w_1 + a_4 x \leq b_2$$

Reformulation-Linearization

Create linear envelope for each operator in the objective/constraints. Ex.: $y^2 \log(x^3 + 5) \geq 7$

$$w_1 = x^3 \qquad\qquad a_1 w_1 + a_2 x \leq b_1$$

$$a_3 w_1 + a_4 x \leq b_2$$

$$w_2 = w_1 + 5 \qquad\qquad w_2 = w_1 + 5$$

$$w_3 = \log w_2$$

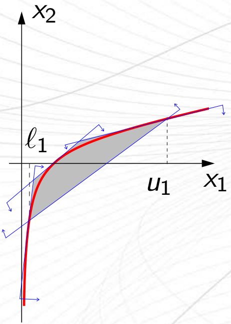
$$w_4 = y^2$$

$$w_5 = w_4 w_3$$

$$w_5 \geq 7$$

Reformulation-Linearization

Create linear envelope for each operator in the objective/constraints. Ex.: $y^2 \log(x^3 + 5) \geq 7$

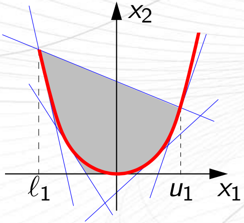


$$
\begin{aligned}
w_1 &= x^3 \\
w_2 &= w_1 + 5 \\
w_3 &= \log w_2 \\
w_4 &= y^2 \\
w_5 &= w_4 w_3 \\
w_5 &\geq 7
\end{aligned}
$$

$$
\begin{aligned}
a_1 w_1 + a_2 x &\leq b_1 \\
a_3 w_1 + a_4 x &\leq b_2 \\
w_2 &= w_1 + 5 \\
a_5 w_3 + a_6 w_2 &\leq b_3 \\
a_7 w_3 + a_8 w_2 &\leq b_4
\end{aligned}
$$

Reformulation-Linearization

Create linear envelope for each operator in the objective/constraints. Ex.: $y^2 \log(x^3 + 5) \geq 7$



$$
\begin{aligned}
w_1 &= x^3 \\
w_2 &= w_1 + 5 \\
w_3 &= \log w_2 \\
w_4 &= y^2 \\
w_5 &= w_4 w_3 \\
w_5 &\geq 7
\end{aligned}
$$

$$
\begin{aligned}
a_1 w_1 + a_2 x &\leq b_1 \\
a_3 w_1 + a_4 x &\leq b_2 \\
w_2 &= w_1 + 5 \\
a_5 w_3 + a_6 w_2 &\leq b_3 \\
a_7 w_3 + a_8 w_2 &\leq b_4 \\
a_9 w_4 + a_{10} y &\leq b_5 \\
a_{11} w_4 + a_{12} y &\leq b_6
\end{aligned}
$$

Reformulation-Linearization

Create linear envelope for each operator in the objective/constraints. Ex.: $y^2 \log(x^3 + 5) \geq 7$

$$
\begin{aligned}
w_1 &= x^3 & \quad & a_1 w_1 + a_2 x \leq b_1 \\
 & & & a_3 w_1 + a_4 x \leq b_2 \\
w_2 &= w_1 + 5 & & w_2 = w_1 + 5 \\
w_3 &= \log w_2 & & a_5 w_3 + a_6 w_2 \leq b_3 \\
 & & & a_7 w_3 + a_8 w_2 \leq b_4 \\
w_4 &= y^2 & & a_9 w_4 + a_{10} y \leq b_5 \\
 & & & a_{11} w_4 + a_{12} y \leq b_6 \\
w_5 &= w_4 w_3 & & a_{13} w_5 + a_{14} w_4 + a_{15} w_3 \leq b_7 \\
 & & & a_{16} w_5 + a_{17} w_4 + a_{18} w_3 \leq b_8 \\
w_5 &\geq 7 & &
\end{aligned}
$$

Reformulation-Linearization

Create linear envelope for each operator in the objective/constraints. Ex.: $y^2 \log(x^3 + 5) \geq 7$

$w_1 = x^3$	$a_1 w_1 + a_2 x \leq b_1$ $a_3 w_1 + a_4 x \leq b_2$
$w_2 = w_1 + 5$	$w_2 = w_1 + 5$
$w_3 = \log w_2$	$a_5 w_3 + a_6 w_2 \leq b_3$ $a_7 w_3 + a_8 w_2 \leq b_4$
$w_4 = y^2$	$a_9 w_4 + a_{10} y \leq b_5$ $a_{11} w_4 + a_{12} y \leq b_6$
$w_5 = w_4 w_3$	$a_{13} w_5 + a_{14} w_4 + a_{15} w_3 \leq b_7$ $a_{16} w_5 + a_{17} w_4 + a_{18} w_3 \leq b_8$
$w_5 \geq 7$	$w_5 \geq 7$

Reformulation-Linearization

Create linear envelope for each operator in the objective/constraints. Ex.: $y^2 \log(x^3 + 5) \geq 7$

$$
\begin{array}{l}
w_1 = x^3 \\
w_2 = w_1 + 5 \\
w_3 = \log w_2 \\
w_4 = y^2 \\
w_5 = w_4 w_3 \\
w_5 \geq 7
\end{array}
\quad \subseteq \quad
\begin{array}{l}
a_1 w_1 + a_2 x \leq b_1 \\
a_3 w_1 + a_4 x \leq b_2 \\
w_2 = w_1 + 5 \\
a_5 w_3 + a_6 w_2 \leq b_3 \\
a_7 w_3 + a_8 w_2 \leq b_4 \\
a_9 w_4 + a_{10} y \leq b_5 \\
a_{11} w_4 + a_{12} y \leq b_6 \\
a_{13} w_5 + a_{14} w_4 + a_{15} w_3 \leq b_7 \\
a_{16} w_5 + a_{17} w_4 + a_{18} w_3 \leq b_8 \\
w_5 \geq 7
\end{array}
$$