A Tutorial on Mixed Integer Non Linear Programming

Claudia D'Ambrosio

LIX, CNRS & École Polytechnique

Belgian Mathematical Optimization workshop
La Roche (Belgium), 26 April 2019

Outline

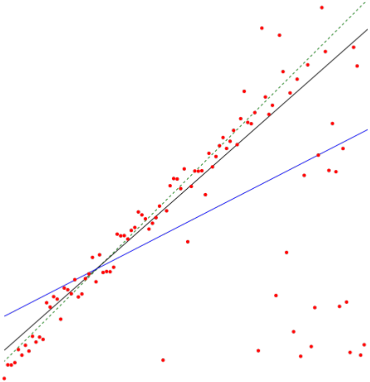
Outline

Subset selection in Linear Regression

Subset selection in Linear Regression

m data points (x_i, y_i) with $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$.

Find $\beta \in \mathbb{R}^d$ such that $\sum_{i=1}^{m}(y_i - x_i^\mathsf{T}\beta)^2$ is minimized while limiting the cardinality of β to K.



Subset selection in Linear Regression

m data points (x_i, y_i) with $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$.

Find $\beta \in \mathbb{R}^d$ such that $\sum_{i=1}^m (y_i - x_i^\mathsf{T}\beta)^2$ is minimized while limiting the cardinality of β to K.

$$\min_\beta \sum_{i=1}^m (y_i - \sum_{j=1}^d x_{ij}\beta_j)^2$$

$$|\text{supp}(\beta)| \leq K$$

D. Bertsimas, R. Shioda. Algorithm for cardinality-constrained quadratic optimization, **Computational Optimization and Applications**, 43 (1), pp. 1–22, 2009.

Subset selection in Linear Regression

$$\min_{\beta,z} \sum_{i=1}^{m} (y_i - \sum_{j=1}^{d} x_{ij}\beta_j)^2$$

$$\sum_{j=1}^{d} z_i \leq K$$

$$\underline{\beta}_j z_j \leq \beta_j \leq \overline{\beta}_j z_j \quad \forall j \leq d$$

$$z_j \in \{0, 1\} \quad \forall j \leq d$$

Robust Portfolio Selection

Robust Portfolio Selection

- n possibly risky assets
- mean return vector $\overline{\mu} \in \mathbb{R}^n$
- $x \in \mathbb{R}^n_+$: fraction of the porftolio value invested in each of the n assets

$$\begin{aligned} \min x^\mathsf{T}\overline{\Sigma}x & \\ \overline{\mu}^\mathsf{T}x &\geq R \\ \mathbf{e}^\mathsf{T}x &= 1 \\ x &\geq 0 \end{aligned}$$

where $\overline{\Sigma} \in \mathbb{R}^{n\times n}$ is the covariance return matrix, $R > 0$ is the minimum portfolio return, $\mathbf{e} \in \mathbb{R}^n$ is the all-one vector.

H. Markowitz, Portfolio Selection, **The Journal of Finance**, 7 (1), pp. 77–91, 1952.

L. Mencarelli, C. D'Ambrosio. Complex Portfolio Selection via Convex Mixed-Integer Quadratic Programming: A Survey, **International Transactions in Operational Research** 26, pp. 389–414, 2019.

Deutsche Börse buys US analytics provider Axioma in $850m deal

Philip Stafford in London APRIL 9, 2019

Deutsche Börse will purchase Axioma, a US risk and portfolio analytics provider, for $850m as part of a deal that will see the German exchanges operator set up its index business as a new company.

Outline

Mathematical Programming

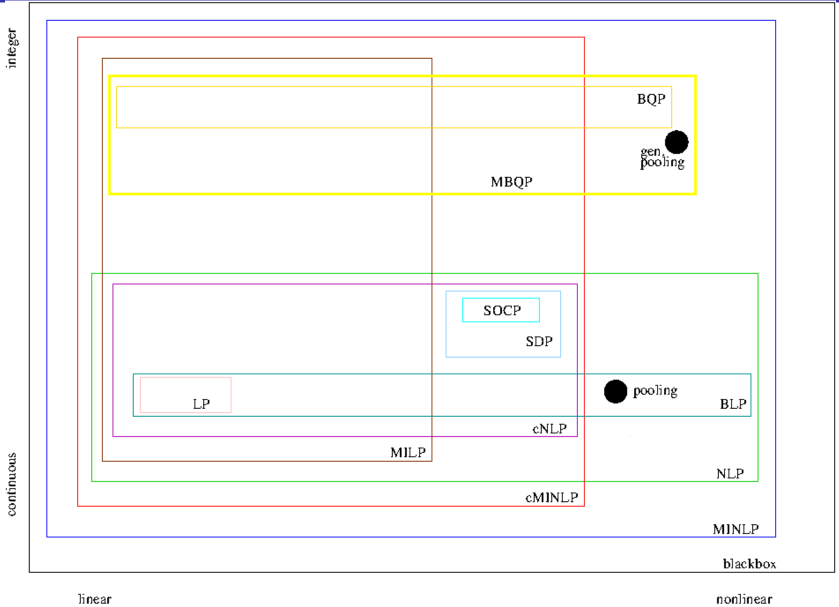
(MINLP)

$$
\begin{aligned}
\min f(x,y) & & \\
g_i(x,y) & \leq 0 \quad \forall i = 1, \ldots, m \\
x & \in X \\
y & \in Y
\end{aligned}
$$

where $f(x,y) : \mathbb{R}^n \to \mathbb{R}$, $g_i(x,y) : \mathbb{R}^n \to \mathbb{R}\ \forall i = 1, \ldots, m$, $X \subseteq \mathbb{R}^{n_1}$ $Y \subseteq \mathbb{N}^{n_2}$ and $n = n_1 + n_2$.

Hypothesis: f and g are twice continuously differentiable functions.

Main optimization problem classes



Outline

Exact reformulations

(MINLP')

$$\min h(w,z) \tag{1}$$
$$p_i(w,z) \leq 0 \quad \forall i = 1,\dots,r \tag{2}$$
$$w \in W \tag{3}$$
$$z \in Z \tag{4}$$

where $h(w,z) : \mathbb{R}^q \to \mathbb{R}$, $p_i(w,z) : \mathbb{R}^q \to \mathbb{R}\ \forall i = 1,\dots,r$, $W \subseteq \mathbb{R}^{q_1}$, $Z \subseteq \mathbb{N}^{q_2}$ and $q = q_1 + q_2$.

Exact reformulations

(MINLP')

$$\min h(w,z) \quad (1)$$
$$p_i(w,z) \leq 0 \quad \forall i = 1,\dots,r \quad (2)$$
$$w \in W \quad (3)$$
$$z \in Z \quad (4)$$

where $h(w,z) : \mathbb{R}^q \to \mathbb{R}$, $p_i(w,z) : \mathbb{R}^q \to \mathbb{R}\ \forall i = 1,\dots,r$, $W \subseteq \mathbb{R}^{q_1}$, $Z \subseteq \mathbb{N}^{q_2}$ and $q = q_1 + q_2$.

The formulation (MINLP') is an **exact reformulation** of (MINLP) if

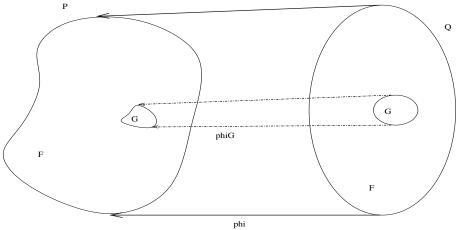
- $\forall (w', z')$ satisfying (2)-(4), $\exists (x', y')$ feasible solution of (MINLP) s.t. $\phi(w', z') = (x', y')$
- ϕ is efficiently computable
- $\forall (w', z')$ global solution of (MINLP'), then $\phi(w', z')$ is a global solution of (MINLP)
- $\forall (x', y')$ global solution of (MINLP), there is a (w', z') global solution of (MINLP')

Exact reformulations

(MINLP')

$$
\begin{aligned}
\min h(w,z) & & & (1)\\
p_i(w,z) &\leq 0 \quad \forall i = 1,\ldots,r & & (2)\\
w &\in W & & (3)\\
z &\in Z & & (4)
\end{aligned}
$$

where $h(w,z) : \mathbb{R}^q \to \mathbb{R}$, $p_i(w,z) : \mathbb{R}^q \to \mathbb{R}\ \forall i = 1,\ldots,r$, $W \subseteq \mathbb{R}^{q_1}$, $Z \subseteq \mathbb{N}^{q_2}$ and $q = q_1 + q_2$.



Exact reformulations: example 1

$$\begin{array}{rcl} \min y_1^2 + y_2^2 & & \\ 10y_1 + 5y_2 & \leq & 11 \\ y_1 & \in & \{0,1\} \\ y_2 & \in & \{0,1\} \end{array}$$

is equivalent to

$$\begin{array}{rcl} \min y_1 + y_2 & & \\ 10y_1 + 5y_2 & \leq & 11 \\ y_1 & \in & \{0,1\} \\ y_2 & \in & \{0,1\} \end{array}$$

or

$$\begin{array}{rcl} \min w_1 + w_2 & & \\ w_1(= y_1^2) & = & y_1 \\ w_2(= y_2^2) & = & y_2 \\ 10y_1 + 5y_2 & \leq & 11 \\ y_1 & \in & \{0,1\} \\ y_2 & \in & \{0,1\} \end{array}$$

Exact reformulations: example 2

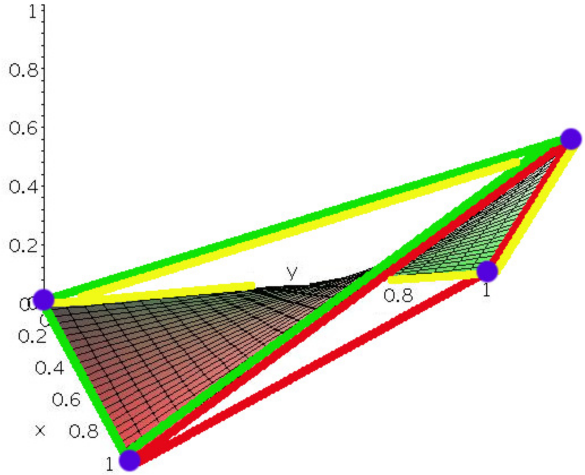
xy when y is binary

- If $\exists$ bilinear term xy where $x \in [0,1]$, $y \in \{0,1\}$
- We can construct an **exact reformulation**:
  - Replace each term xy by an added variable w
  - Adjoin Fortet's reformulation constraints:

$$
\begin{array}{rcl}
w & \geq & 0 \\
w & \geq & x + y - 1 \\
w & \leq & x \\
w & \leq & y
\end{array}
$$

  - Get a MILP reformulation
  - Solve reformulation using a MILP solver: more effective than solving MINLP

"Proof"



"Proof"

$$\begin{array}{rcl} w & \geq & 0 \\ w & \geq & x + y - 1 \\ w & \leq & x \\ w & \leq & y \end{array}$$

$y = 0$	$y = 1$
$w \geq 0$	$w \geq 0$
$w \geq x - 1$	$w \geq x$
$w \leq 0$	$w \leq 1$
$w \leq x$	$w \leq x$
$w = 0$	$w = x$

Relaxations

(rMINLP)

$$\begin{aligned} \min \underline{f(w,z)} & \\ \underline{g_i(w,z)} &\leq 0 \quad \forall i = 1,\ldots,r \\ w &\in W \\ z &\in Z \end{aligned}$$

where $X \subseteq W \subseteq \mathbb{R}^{q_1}$, $Y \subseteq Z \subseteq \mathbb{Z}^{q_2}$, $q_1 \geq n_1$, $q_2 \geq n_2$, $\underline{f(w,z)} \leq f(x,y)$ $\forall (x,y) \subseteq (w,z)$, and $\{(x,y)|g(x,y) \leq 0\} \subseteq \text{Proj}_{(x,y)}\{(w,z)|\underline{g(w,z)} \leq 0\}$.
Examples:

- continuous relaxation: when $(w,z) \in \mathbb{R}^n$, $W = X$, $Z = Y$, $\underline{f(x,y)} = f(x,y)$, $\underline{g(x,y)} = g(x,y)$
- linear relaxation: when $q = n$, $W = X$, $Z = Y$, $\underline{f(w,z)}$ and $\underline{g(w,z)}$ are linear
- convex relaxation: when $q = n$, $W = X$, $Z = Y$, $\underline{f(w,z)}$ and $\underline{g(w,z)}$ are convex

Relaxations: example

$x_1 x_2$ when x_1, x_2 continuous

- Get bilinear term $x_1 x_2$ where $x_1 \in [x_1^L, x_1^U]$, $x_2 \in [x_2^L, x_2^U]$
- We can construct a **relaxation**:
  - Replace each term $x_1 x_2$ by an added variable w
  - Adjoin following constraints:

$$
\begin{aligned}
w &\geq x_1^L x_2 + x_2^L x_1 - x_1^L x_2^L \\
w &\geq x_1^U x_2 + x_2^U x_1 - x_1^U x_2^U \\
w &\leq x_1^U x_2 + x_2^L x_1 - x_1^U x_2^L \\
w &\leq x_1^L x_2 + x_2^U x_1 - x_1^L x_2^U
\end{aligned}
$$

- These are called **McCormick's envelopes**
- Get an LP relaxation (solvable in polynomial time)

References & Software

- Fortet, *Applications de l'algèbre de Boole en recherche opérationelle*, **Revue Française de Recherche Opérationelle**, 4, pp. 251–259, 1960.
- McCormick, *Computability of global solutions to factorable nonconvex programs: Part I — Convex underestimating problems*, **Mathematical Programming**, 1976.
- Liberti, *Reformulations in Mathematical Programming: definitions and systematics*, **RAIRO-RO**, 2009.
- Liberti, Cafieri, Tarissan, *Reformulations in Mathematical Programming: a computational approach*, in Abraham et al. (eds.), **Foundations of Comput. Intel.**, 2009
- ROSE (`https://projects.coin-or.org/ROSE`)

Outline

What is a convex MINLP?

Convex Mixed Integer NonLinear Programming (MINLP).

$$
\begin{aligned}
\min f(x,y) & & \\
g(x,y) & \leq & 0 \\
x & \in & X = \{x \mid x \in \mathbb{R}^{n_1}, Dx \leq d, x^L \leq x \leq x^U\} \\
y & \in & Y = \{y \mid y \in \mathbb{Z}^{n_2}, Ay \leq a, y^L \leq y \leq y^U\}
\end{aligned}
$$

with $f(x,y) : \mathbb{R}^{n_1+n_2} \to \mathbb{R}$ and $g(x,y) : \mathbb{R}^{n_1+n_2} \to \mathbb{R}^m$ are

* continuous
* twice differentiable
* convex

functions.

- Local optima are also global optima .

“Basic” subproblems we can solve “well”

NLP relaxation

$$
\begin{array}{rcll}
\min f(x,y) & & & \\
g(x,y) & \leq & 0 & \\
x & \in & X & \\
y & \in & \{y \mid Ay \leq a\} & \\
y_j & \leq & \alpha_j^k & j \in \{1,2,\ldots,n_2\} \\
y_j & \geq & \beta_j^k & j \in \{1,2,\ldots,n_2\}
\end{array}
$$

k: current step of a Branch-and-Bound procedure;
α^k: current lower bound on y ($\alpha^k \geq y^L$);
β^k: current upper bound on y ($\beta^k \leq y^U$).

NLP restriction and Feasibility subproblem

NLP restriction for a fixed y^k:

$$\begin{aligned} \min f(x, y^k) & \\ g(x, y^k) &\leq 0 \\ x &\in X. \end{aligned}$$

Feasibility subproblem for a fixed y^k:

$$\begin{aligned} \min u & \\ g(x, y^k) &\leq u \\ x &\in X \\ u &\in \mathbb{R}_+. \end{aligned}$$

MILP relaxation

$$
\begin{array}{rcll}
& \min \gamma & & \\
f(x^k, y^k) + \nabla f(x^k, y^k)^T \begin{pmatrix} x - x^k \\ y - y^k \end{pmatrix} & \leq & \gamma & \forall k \\
g_i(x^k, y^k) + \nabla g_i(x^k, y^k)^T \begin{pmatrix} x - x^k \\ y - y^k \end{pmatrix} & \leq & 0 & \forall k \; \forall i \in I^k \\
x & \in & X & \\
y & \in & Y. &
\end{array}
$$

where $I^k \subseteq \{1, 2, \ldots, m\}$. choices:

- $I^k = \{1, 2, \ldots, m\}$
- $I^k = \{i \mid g(x^k, y^k) > 0, 1 \leq i \leq m\}$

Outline

Convex MINLP Algorithms

- Branch-and-Bound (BB).
- Outer-Approximation (OA).
- Generalized Benders Decomposition (GBD).
- Extended Cutting Plane (ECP).
- LP/NLP-based Branch-and-Bound (QG).
- Hybrid Algorithms (Hyb).

Branch-and-Bound (BB)

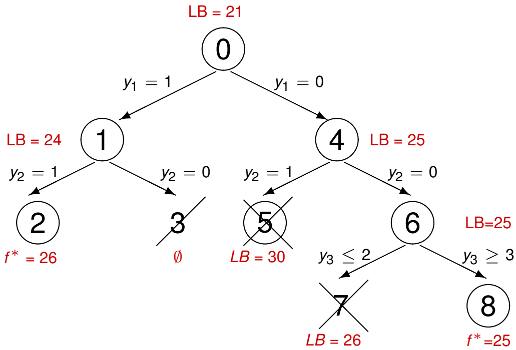
Gupta and Ravindran, 1985. Link BB for MILPs.

```
f* = +∞, Π = {P^0}, LB(P^0) = −∞ where P^0 = NLP relaxation.
while Π ≠ ∅ do
    Choose the current subproblem P ∈ Π, Π = Π \ {P}.
    Solve P obtaining (x̄, ȳ).
    if P infeasible ∨ f(x̄, ȳ) ≥ f* then
        break
    end if
    if ȳ ∈ Z^{n_2} then
        f* = f(x̄, ȳ), (x*, y*) = (x̄, ȳ).
        Update Π potentially fathoming subproblems.
    else
        Take a fractional value ȳ_j and obtain two subproblems P^1 = P with α_j^1 = ⌊ȳ_j⌋ and
        P^2 = P with β_j^2 = ⌊ȳ_j⌋ + 1.
        LB(P^1) = LB(P^2) = f(x̄, ȳ).
        Π = Π ⋃ {P^1, P^2}.
    end if
end while
return (x*, y*).
```

Fathoming is performed when:

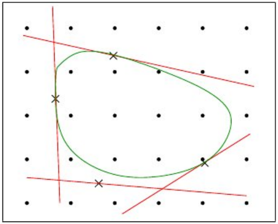
- The subproblem solution is MINLP feasible (f^*).
- The subproblem is infeasible.
- The subproblem P^k has $LB(P^k) \geq f^*$.

Branch-and-Bound (BB)



Outer-Approximation (OA)

Duran and Grossmann, 1986.



$$
\begin{array}{rcll}
\multicolumn{4}{c}{\min \gamma} \\
f(x^k, y^k) + \nabla f(x^k, y^k)^T \begin{pmatrix} x - x^k \\ y - y^k \end{pmatrix} & \leq & \gamma & \forall k \\
g_i(x^k, y^k) + \nabla g_i(x^k, y^k)^T \begin{pmatrix} x - x^k \\ y - y^k \end{pmatrix} & \leq & 0 & \forall k \; \forall i \in I^k \\
x & \in & X & \\
y & \in & Y. &
\end{array}
$$

$I^k = \{1, 2, \ldots, m\} \; \forall k = 1, \ldots, K.$

NB. The linearization constraints of MILP relaxation are not valid for non-convex MINLPs.

Outer-Approximation (OA)

$K = 1$, $f^* = +\infty$,

while $f^* \neq \text{LB}$ **do**

  Solve the current MILP relaxation (obtaining (x^K, y^K)) and update LB.

  Solve the current NLP restriction for y^K.

  if NLP restriction for y^K infeasible **then**

    Solve the infeasibility subproblem for y^K.

  else

    if $f(x^K, y^K) < f^*$ **then**

      $f^* = f(x^K, y^K)$, $(x^*, y^*) = (x^K, y^K)$.

    end if

  end if

  Generate linearization cuts, update MILP relax.

  $K = K + 1$.

end while

return (x^*, y^*)

Number of subproblems solved

	# MILP	# NLP	note
BB	0	# nodes	
OA	# iterations	# iterations	
GBD	# iterations	# iterations	[1]
ECP	# iterations	0	
QG	1	1 + # explored MILP solutions	
Hyb ALL10	1	1 + # explored MILP solutions	[2]
Hyb CMUIBM	1	[# explored MILP solutions,# nodes]	

Table: Number of MILP and NLP subproblems solved by each algorithm.

[1] weaker lower bound w.r.t. OA, MILP with less constraints than the one of OA
[2] stronger lower bound w.r.t. QG ,MILP with more constraints than the one of QG

References

- C. D'Ambrosio, A. Lodi. Mixed Integer Non-Linear Programming Tools: a Practical Overview, **4OR**: A Quarterly Journal of Operations Research, 9 (4), pp. 329-349, 2011.
- P. Bonami, M. Kilinç, J. Linderoth, Algorithms and software for convex mixed integer nonlinear programs. In: Lee J, Leyffer S (eds) **Mixed integer nonlinear programming**. Springer, pp. 1–39, 2012.
- C. D'Ambrosio, A. Lodi. Mixed integer nonlinear programming tools: an updated practical overview, **Annals of Operations Research**, 204, pp. 301–320, 2013.
- P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, A. Mahajan, Mixed-integer nonlinear optimization. **Acta Numerica**, 22, pp. 1–131, 2013.
- J. Kronqvist, D. E. Bernal, A. Lundell, I. E. Grossmann, A review and comparison of solvers for convex MINLP, **Optimization and Engineering**, to appear.

From convex to nonconvex...

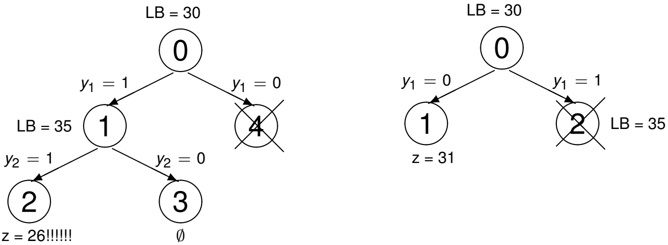
In fact the great watershed in optimization isn't between linearity and nonlinearity, but convexity and nonconvexity.

R. T. Rockafellar. Lagrange multipliers and optimality. SIAM Review, 35:183–238, 1993.

Outline

MINLP branch-and-bound with local NLP solver

Branch-and-bound algorithm: solve continuous (NLP) relaxation at each node of the search tree and branch on variables.
NLP solver used:
Local NLP solvers $\rightarrow$ local optimum
No valid bound for nonconvex MINLPs.

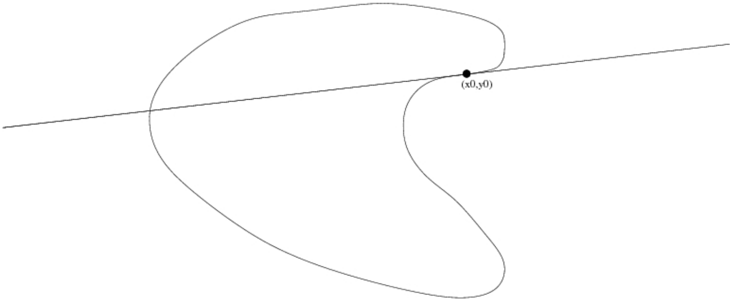


Outer Approximation and nonconvex MINLPs

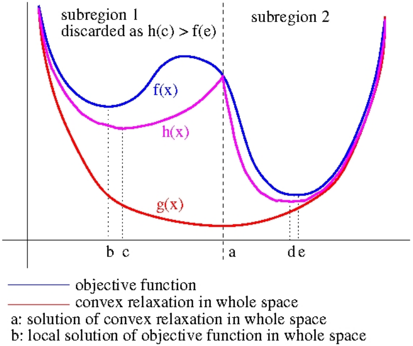
Several methods for convex MINLPs use **Outer Approximation** cuts (Duran and Grossman, 1986) which are not exact for nonconvex MINLPs.

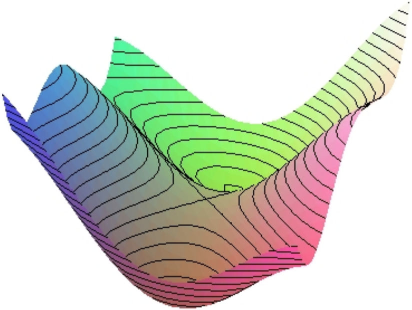
$$g_i(x) \leq 0 \quad \rightarrow \quad g_i(x^k) + \nabla g_i(x^k)^T \left(\ x - x^k \ \right) \leq 0$$

where $\nabla g(x^k)$ is the Jacobian of $g(x)$ evaluated at point (x^k).



Outline





Exact

- “Exact” in continuous space: ε-approximate (*find solution within pre-determined* ε *distance from optimum in obj. fun. value*)
- For some problems, finite convergence to optimum ($\varepsilon = 0$)

Heuristic

- Find solution with probability 1 in infinite time

Outline

Multistart

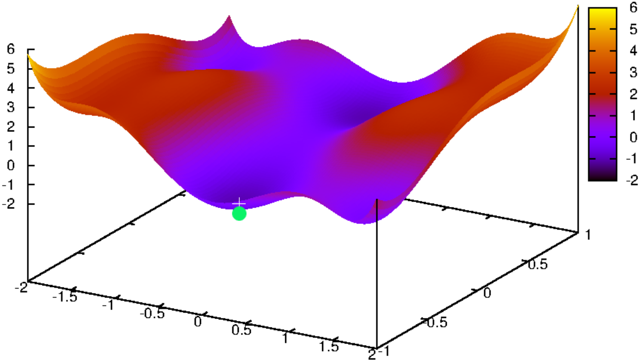
- The easiest GO method

$f^* = \infty$
$x^* = (\infty, \ldots, \infty)$
while $\neg$ termination **do**
  $x' = (\text{random}(), \ldots, \text{random}())$
  $x = \text{localSolve}(P, x')$
  if $f_P(x) < f^*$ **then**
    $f^* \leftarrow f_P(x)$
    $x^* \leftarrow x$
  end if
end while

- Termination condition: e.g. *repeat k times*

Six-hump camelback function

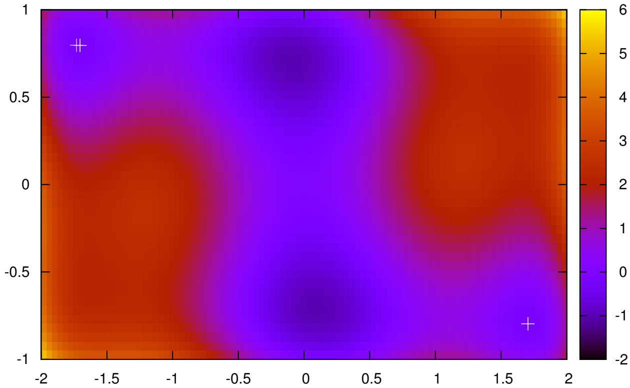
$$f(x_1, x_2) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$$



Global optimum (COUENNE)

Six-hump camelback function

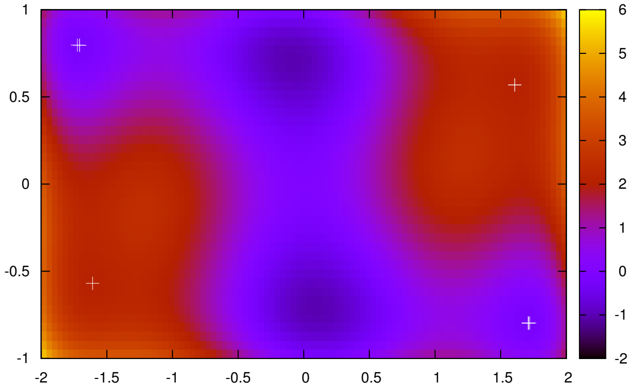
$$f(x_1, x_2) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$$



Multistart with IPOPT, $k = 5$

Six-hump camelback function

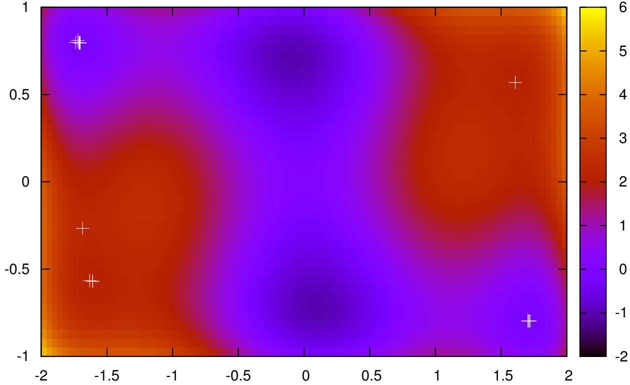
$$f(x_1, x_2) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$$



Multistart with IPOPT, $k = 10$

Six-hump camelback function

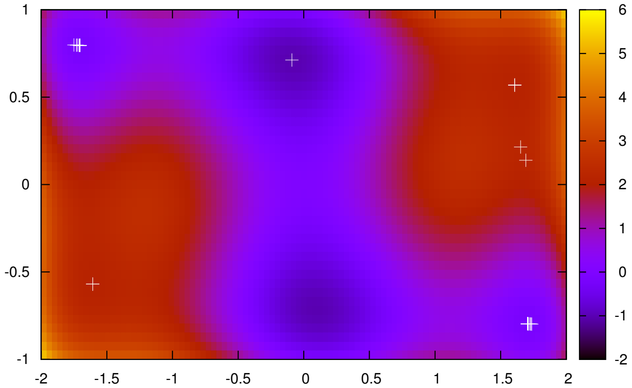
$$f(x_1, x_2) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$$



Multistart with IPOPT, $k = 20$

Six-hump camelback function

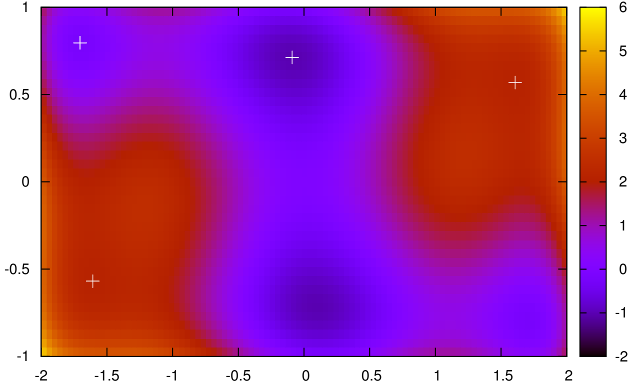
$$f(x_1, x_2) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$$



Multistart with IPOPT, $k = 50$

Six-hump camelback function

$$f(x_1, x_2) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$$



Multistart with SNOPT, $k = 20$

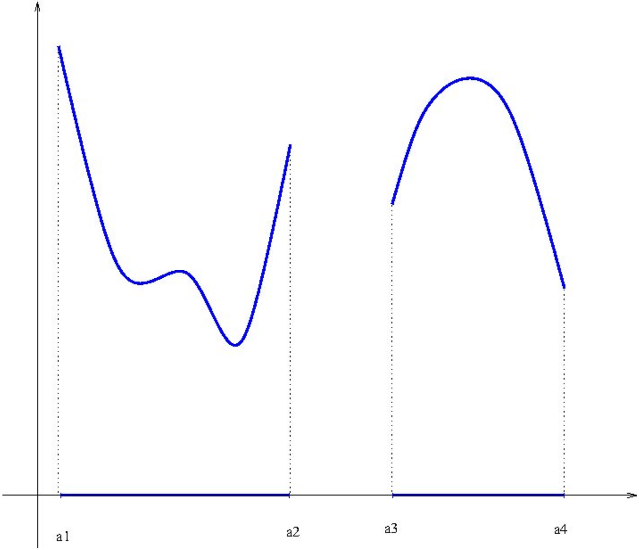
Outline

Spatial Branch-and-Bound

Falk and Soland (1969) "An algorithm for separable nonconvex programming problems".
20 years ago: first general-purpose "exact" algorithms for nonconvex MINLP.

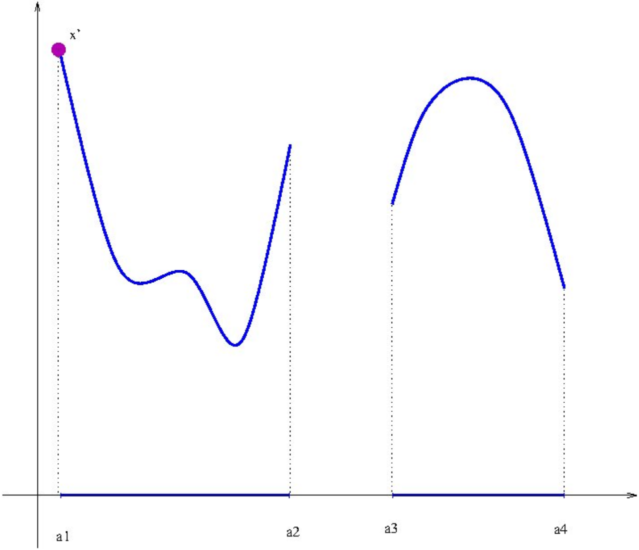
- Tree-like search
- Explores search space exhaustively but implicitly
- Builds a sequence of decreasing upper bounds and increasing lower bounds to the global optimum
- Exponential worst-case
- Only general-purpose "exact" algorithm for MINLP
  Since continuous vars are involved, should say "ε-approximate"
- Like BB for MILP, but may branch on continuous vars
  Done whenever one is involved in a nonconvex term

Spatial B&B: Example



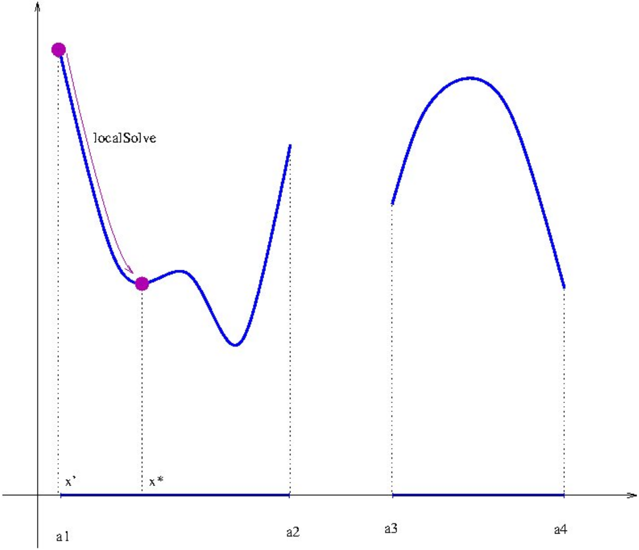
Original problem P

Spatial B&B: Example



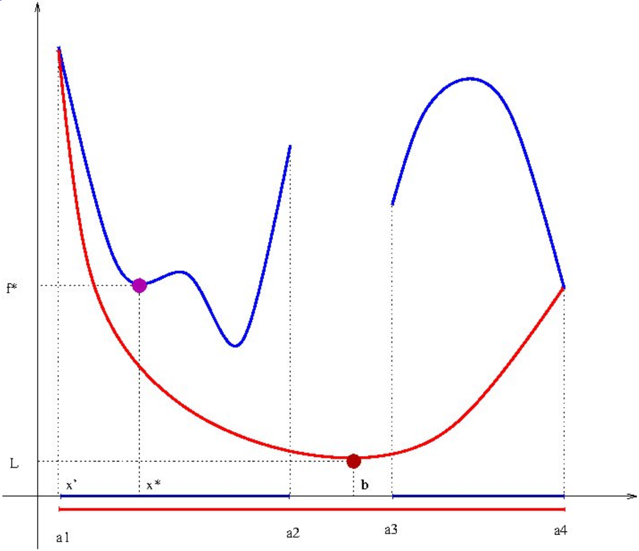
Starting point x'

Spatial B&B: Example



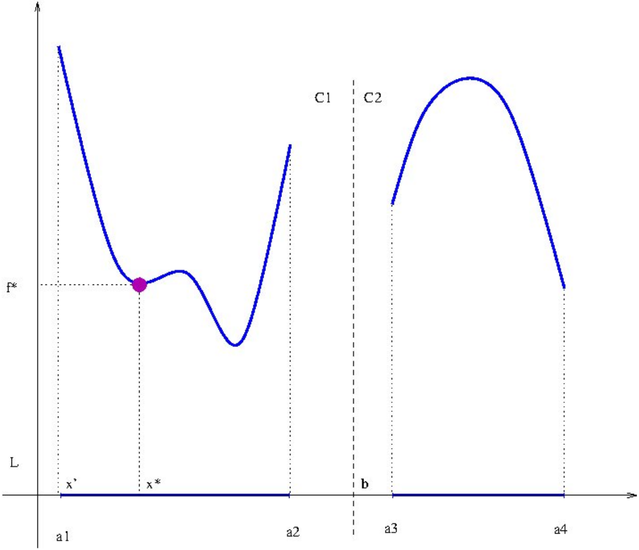
Local (upper bounding) solution x^*

Spatial B&B: Example



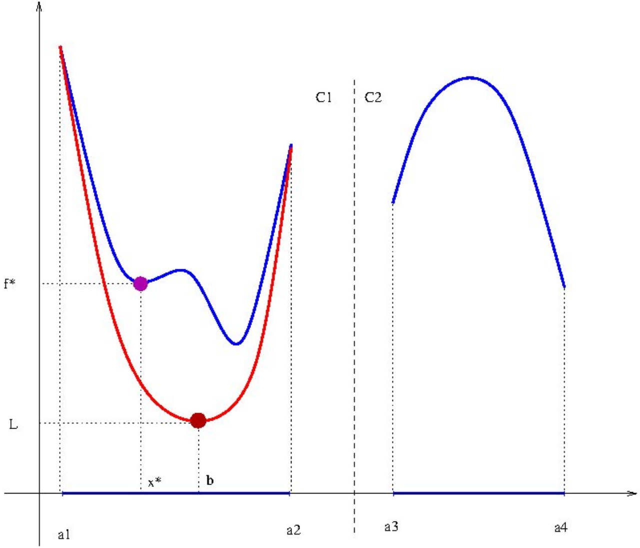
Convex relaxation (lower) bound $\bar{f}$ *with* $|f^* - \bar{f}| > \varepsilon$

Spatial B&B: Example



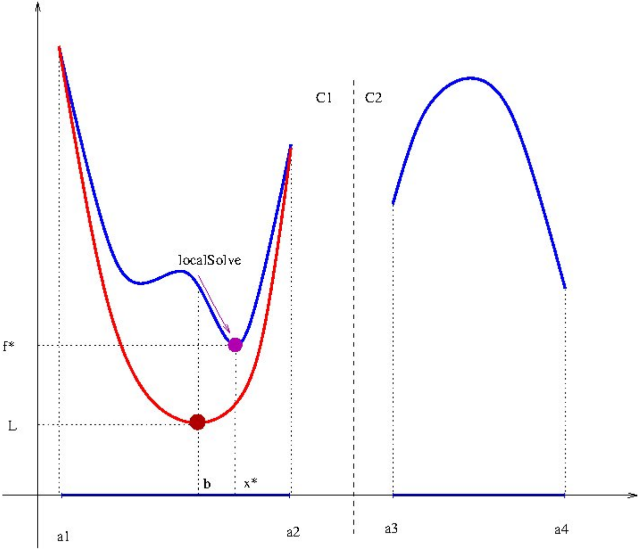
Branch at $x = \bar{x}$ *into* C_1, C_2

Spatial B&B: Example



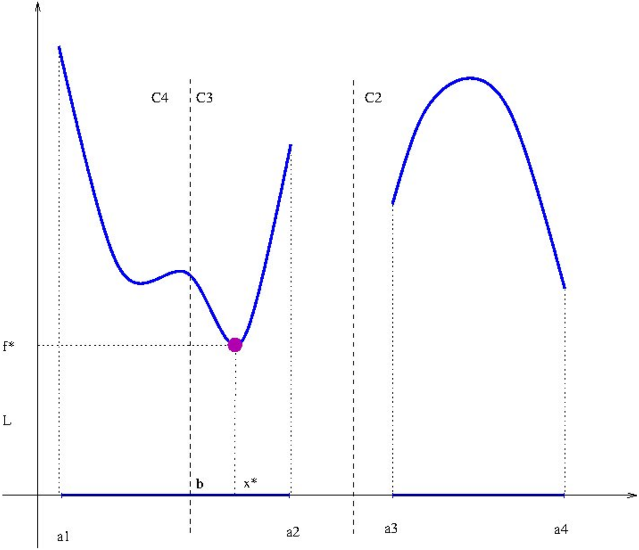
Convex relaxation on C_1: lower bounding solution $\bar{x}$

Spatial B&B: Example



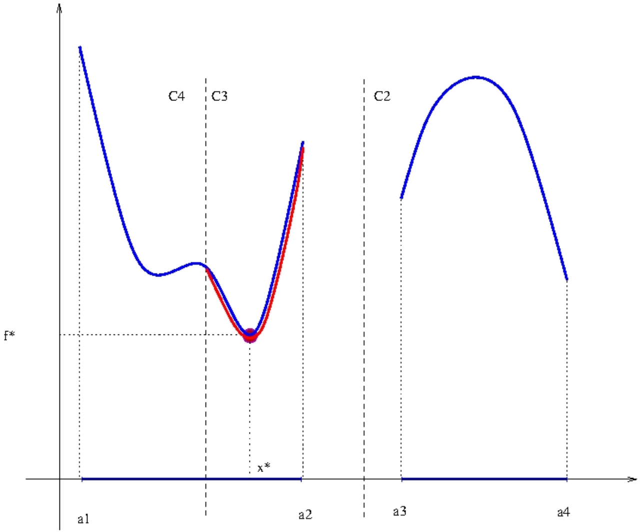
localSolve. *from $\bar{x}$: new upper bounding solution x^**

Spatial B&B: Example



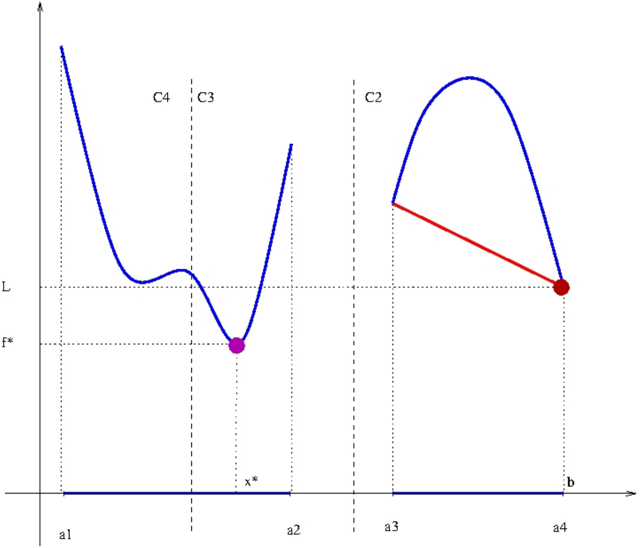
$$|f^* - \bar{f}| > \varepsilon\text{: branch at } x = \bar{x}$$

Spatial B&B: Example

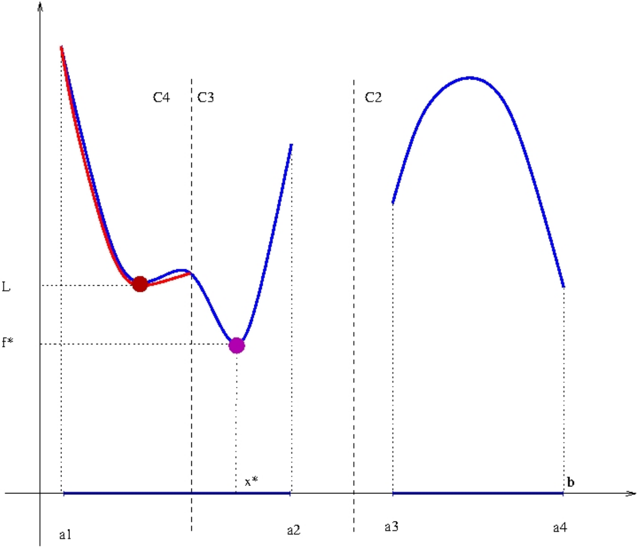


Repeat on C_3: get $\bar{x} = x^*$ and $|f^* - \bar{f}| < \varepsilon$, no more branching

Spatial B&B: Example

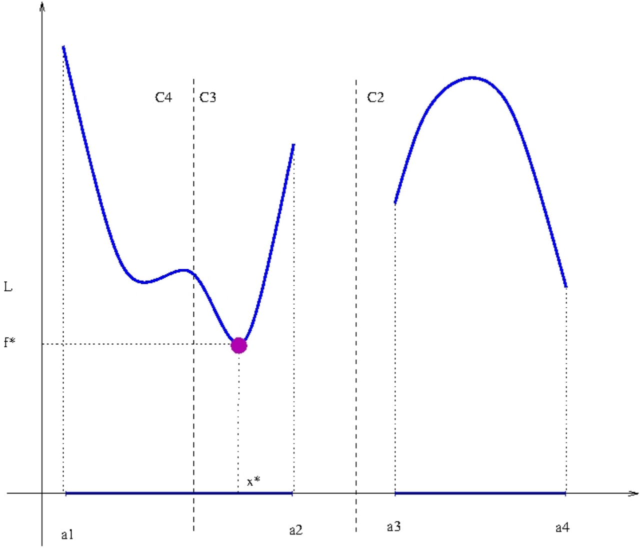


Repeat on C_2: $\bar{f} > f^*$ (can't improve x^* in C_2)



Repeat on C_4: $\bar{f} > f^*$ (can't improve x^* in C_4)

Spatial B&B: Example



No more subproblems left, return x^* and terminate

Spatial B&B: Pruning

1. P was branched into C_1, C_2
2. C_1 was branched into C_3, C_4
3. C_3 was **pruned by optimality**
   ($x^* \in \mathcal{G}(C_3)$ was found)
4. C_2, C_4 were **pruned by bound**
   (lower bound for C_2 worse than f^*)
5. No more nodes: whole space explored, $x^* \in \mathcal{G}(P)$

- Search generates a tree
- Subproblems are nodes
- Nodes can be pruned by optimality, bound or **infeasibility** (when subproblem is infeasible)
- Otherwise, they are branched

Spatial B&B: General idea

Aimed at solving "factorable functions", i.e., f and g of the form:

$$\sum_h \prod_k f_{hk}(x, y)$$

where $f_{hk}(x, y)$ are univariate functions $\forall h, k$.

- Exact reformulation of MINLP so as to have "isolated basic nonlinear functions" (additional variables and constraints).
- Relax (linear/convex) the basic nonlinear terms (library of envelopes/underestimators).
- Relaxation depends on variable bounds, thus branching potentially strengthen it.

The Standard form

Spatial B&B: exact reformulation to standard form

Consider a NLP for simplicity. Transform it in a standard form like:

$$\begin{aligned} \min c^\mathsf{T}(x, w) & & \\ A(x, w) & \leq & b \\ w_{ij} & = & x_i \bigotimes x_j \quad \text{for suitable } i, j \\ x & \in & X \\ w & \in & W \end{aligned}$$

where, for example, $\bigotimes \in$ {sum, product, quotient, power, exp, log, sin, cos, abs} (Couenne).

Convexification

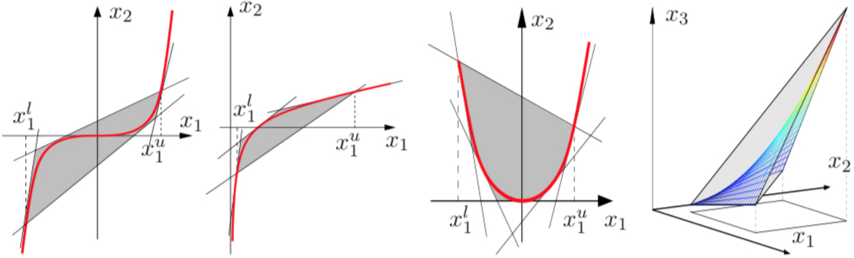
Spatial B&B: convexification

Relax $w_{ij} = x_i \bigotimes x_j$ $\forall$ suitable i, j where $\bigotimes \in$ {sum, product, quotient, power, exp, log, sin, cos, abs} such that:

$$\begin{aligned} w_{ij} &\leq \text{overestimator}(x_i \bigotimes x_j) \\ w_{ij} &\geq \text{underestimator}(x_i \bigotimes x_j) \end{aligned}$$

Convex relaxation is not the tightest possible, but built automatically.

- Underestimator/overestimator of convex/concave function: tangent cuts (OA)
- Odd powers or Trigonometric functions: separate intervals in which function is convex or concave and do as for convex/concave functions
- Product or Quotient: Mc Cormick relaxation



(a) $x_2 = x_1^3$ (b) $x_2 = \log x_1$ (c) $x_2 = x_1^2$ (d) $x_3 = x_1 x_2$

P. Belotti, J. Lee, L. Liberti, F. Margot, A. Wächter, “Branching and bounds tightening techniques for non-convex MINLP”. Optimization Methods and Software 24(4-5): 597-634 (2009).

Example: Standard Form Reformulation

$$
\begin{aligned}
\min x_1^2 + x_1 x_2 & \\
x_1 + x_2 &\geq 1 \\
x_1 &\in [0, 1] \\
x_2 &\in [0, 1]
\end{aligned}
$$

becomes

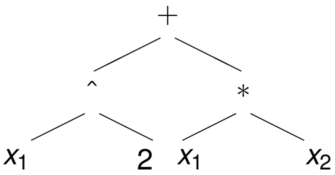
$$
\begin{aligned}
\min w_1 + w_2 & \\
w_1 &= x_1^2 \\
w_2 &= x_1 x_2 \\
x_1 + x_2 &\geq 1 \\
x_1 &\in [0, 1] \\
x_2 &\in [0, 1]
\end{aligned}
$$

Expression trees

Representation of objective f and constraints g

Encode mathematical expressions in trees or DAGs

E.g. $x_1^2 + x_1 x_2$:

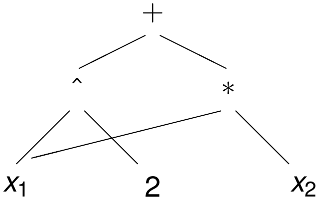


Expression trees

Representation of objective f and constraints g

Encode mathematical expressions in trees or DAGs

E.g. $x_1^2 + x_1 x_2$:



Variable ranges

Variable ranges

- Crucial property for sBB convergence: **convex relaxation tightens as variable range widths decrease**
- convex/concave under/over-estimator constraints are (convex) functions of x^L, x^U
- it makes sense to **tighten** x^L, x^U at the sBB root node (trading off speed for efficiency) and at each other node (trading off efficiency for speed)

Bounds tightening

Bounds Tightening

- In sBB we need to tighten variable bounds at each node
- Two methods:
  - Optimization Based Bounds Tightening (OBBT)
  - Feasibility Based Bounds Tightening (FBBT)
- OBBT:
  for each variable x in P compute
  - $\underline{x} = \min\{x \mid \text{conv. rel. constr.}\}$
  - $\overline{x} = \max\{x \mid \text{conv. rel. constr.}\}$

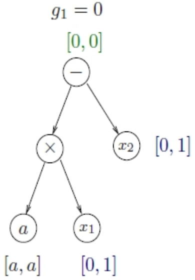
  Set $\underline{x} \leq x \leq \overline{x}$

Bounds Tightening

- In sBB we need to tighten variable bounds at each node
- Two methods:
  - Optimization Based Bounds Tightening (OBBT)
  - Feasibility Based Bounds Tightening (FBBT)
- FBBT:

  propagation of intervals up and down constraint expression trees, with tightening at the root node

  Example: $5x_1 - x_2 = 0$.



Bounds Tightening

- In sBB we need to tighten variable bounds at each node
- Two methods:
  - Optimization Based Bounds Tightening (OBBT)
  - Feasibility Based Bounds Tightening (FBBT)
- FBBT: **propagation of intervals up and down constraint expression trees, with tightening at the root node**

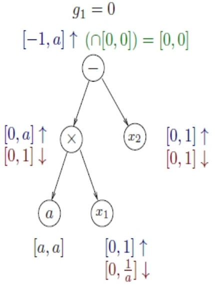
  Example: $5x_1 - x_2 = 0$.

  Up: $\otimes$:$[5,5] \times [0,1] = [0,5]$; $\ominus$:$[0,5] - [0,1] = [-1,5]$.

  Root node tightening: $[-1,5] \cap [0,0] = [0,0]$.

  Downwards: $\otimes$:$[0,0] + [0,1] = [0,1]$;

  x_1:$[0,1]/[5,5] = [0, \frac{1}{5}]$



Citations

- Sherali, Alameddine, *A new reformulation-linearization technique for bilinear programming problems*, JOGO, 1991
- Falk, Soland, *An algorithm for separable nonconvex programming problems*, Manag. Sci. 1969.
- Horst, Tuy, *Global Optimization*, Springer 1990.
- Ryoo, Sahinidis, *Global optimization of nonconvex NLPs and MINLPs with applications in process design*, Comp. Chem. Eng. 1995.
- Adjiman, Floudas et al., *A global optimization method, αBB, for general twice-differentiable nonconvex NLPs*, Comp. Chem. Eng. 1998.
- Smith, Pantelides, *A symbolic reformulation/spatial branch-and-bound algorithm for the global optimisation of nonconvex MINLPs*, Comp. Chem. Eng. 1999.
- Nowak, *Relaxation and decomposition methods for Mixed Integer Nonlinear Programming*, Birkhäuser, 2005.
- Belotti, et al., *Branching and bounds tightening techniques for nonconvex MINLP*, Opt. Meth. Softw., 2009.
- Vigerske, PhD Thesis: Decomposition of Multistage Stochastic Programs and a Constraint Integer Programming Approach to Mixed-Integer Nonlinear Programming, Humboldt-University Berlin, 2013.

Outline

The class of MINLP problems

$$\min \sum_{j \in N} C_j x_j$$

$$f_i(x) + \sum_{k \in H_i} g_{ik}(x_k) \leq 0 \qquad \forall i \in M$$

$$L_j \leq x_j \leq U_j \qquad \forall j \in N$$

$$x_j \text{ integer} \qquad \forall j \in I$$

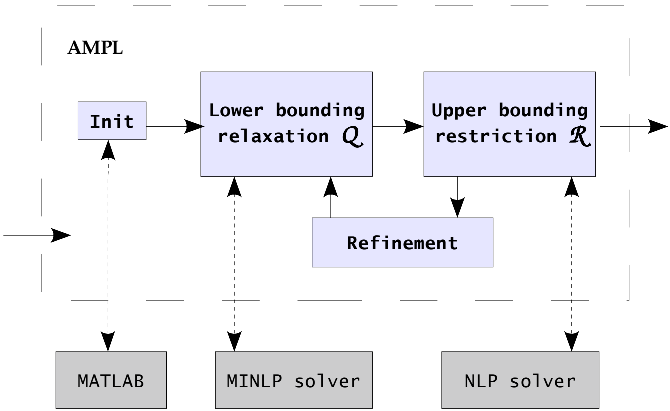
where:

- $f_i : \mathbb{R}^n \to \mathbb{R}$ are convex functions $\forall i \in M$,
- $g_{ik} : \mathbb{R} \to \mathbb{R}$ are non convex univariate function $\forall i \in M, \forall k \in H_i$,
- $H_i \subseteq N \ \ \forall i \in M$,
- $I \subseteq N$, and
- L_j and U_j are finite $\forall i \in M, j \in H_i$

The General Framework

General Framework

Global optimization algorithm proposed in D'Ambrosio, Lee, and Wächter (2009, 2012).



The Upper Bounding problem

The Upper Bounding problem

Upper Bound of the original problem:

1. The integer variables are fixed;
2. We solve the resulting non convex NLP problem to local optimality;

$$\min \sum_{j \in N} C_j x_j$$

$$f_i(x) + \sum_{k \in H_i} g_{ik}(x_k) \leq 0 \qquad \forall i \in M$$

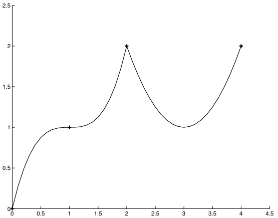
$$L_j \leq x_j \leq U_j \qquad \forall j \in N$$

$$x_j = \underline{x}_j \qquad \forall j \in I$$

The Lower Bounding problem

The Lower Bounding problem: step 1

For simplicity, let us consider a term of the form $g(x_k) := g_{ik}(x_k)$:
$g : \mathbb{R} \to \mathbb{R}$ is a univariate non convex function of x_k , for some k $(1 \leq k \leq n)$.



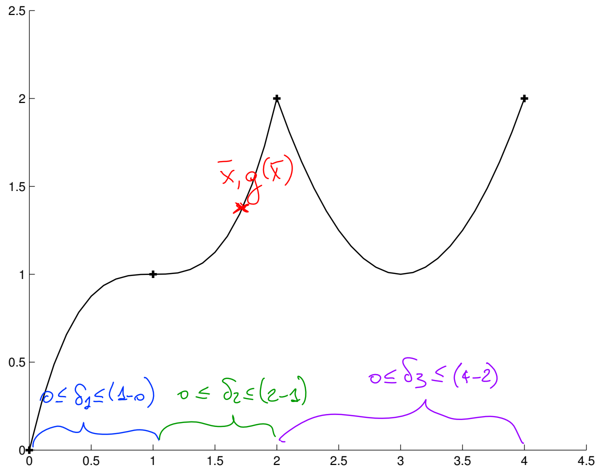
Automatically detect the concavity/convexity intervals or piecewise definition:

$[P_{p-1}, P_p] :=$ the p-th subinterval of the domain of g ($p \in \{1 \dots \overline{p}\}$);

$\check{H} :=$ the set of indices of subintervals on which g is convex;

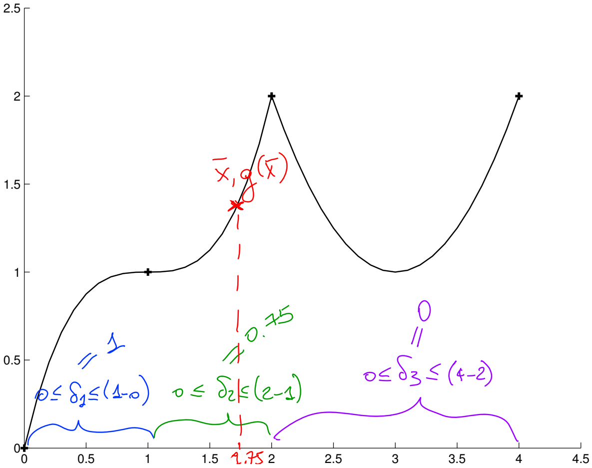
$\hat{H} :=$ the set of indices of subintervals on which g is concave.

Introduction of additional variables $\delta_p \in [0, P_p - P_{p-1}]$ such that $x_k = P_0 + \sum_{p=1}^{\bar{p}} \delta_p$



The Lower Bounding problem: step 2

Introduction of additional variables $\delta_p \in [0, P_p - P_{p-1}]$ such that $x_k = P_0 + \sum_{p=1}^{\bar{p}} \delta_p = 0 + 1 + 0.75 + 0$



The Lower Bounding problem: step 2

- All the δ's but at most 1 take either the lower or the upper bound value
- To model such behavior additional binary variables are needed: $z_p \in \{0, 1\}\ \forall p$
- $z_1 \geq z_2 \geq \cdots \geq z_p$
- $\delta_p = \begin{cases} 0 & z_{p-1} = 0 \\ [0, P_p - P_{p-1}] & z_{p-1} = 1 \text{ and } z_p = 0 \\ P_p - P_{p-1} & z_p = 1 \end{cases}$

δ	$P_1 - P_0$	$P_2 - P_1$	$\dots$	$P_{p-1} - P_{p-2}$	$[0, P_p - P_{p-1}]$	0	$\dots$	0
z	1	1	$\dots$	1	0	0	$\dots$	0

The Lower Bounding problem: step 2

Replace the term $g(x_k)$ with:

$$\sum_{p=1}^{\bar{p}} g(P_{p-1} + \delta_p) - \sum_{p=1}^{\bar{p}-1} g(P_p) ,$$

and we include the following set of new constraints:

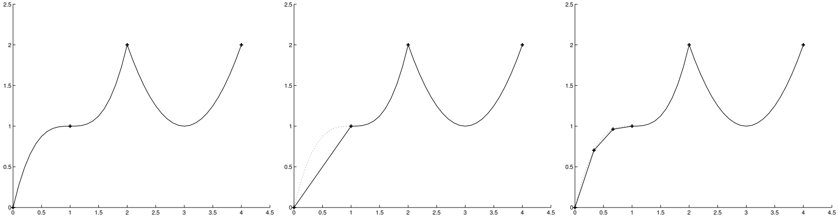
$$x_k = P_0 + \sum_{p=1}^{\bar{p}} \delta_p ;$$
$$\delta_p \geq (P_p - P_{p-1}) z_p , \ \forall p \in \check{H} \cup \hat{H} ;$$
$$\delta_p \leq (P_p - P_{p-1}) z_{p-1} , \ \forall p \in \check{H} \cup \hat{H} ;$$
$$0 \leq \delta_p \leq P_p - P_{p-1}, \ \forall p \in \{1, \ldots, \bar{p}\};$$

with two dummy variables $z_0 := 1$ and $z_{\bar{p}} := 0$ and two new sets of variables z_p (binary) and δ_p (continuous).

The Lower Bounding problem: step 3

Still non convex;

Use piece-wise linear approximation for the concave intervals:



The Lower Bounding problem: the convex MINLP model

Replace the term $g(x_k)$ with:

$$\sum_{p\in\check{H}} g(P_{p-1}+\delta_p) + \sum_{p\in\hat{H}}\sum_{b\in B_p} g(X_{p,b})\,\alpha_{p,b} - \sum_{p=1}^{\overline{p}-1} g(P_p)\ ,$$

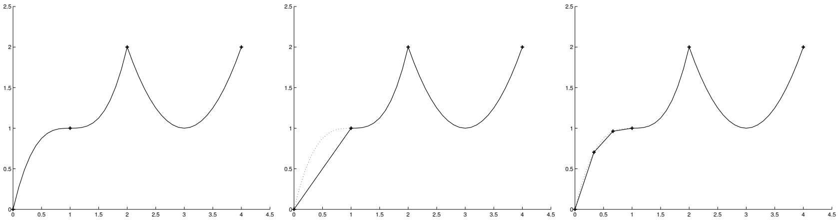
and we include the following set of new constraints:

$$\begin{aligned}
& P_0 + \sum_{p=1}^{\overline{p}} \delta_p - x_k = 0\ ; \\
& \delta_p - (P_p - P_{p-1})z_p \geq 0\ ,\ \forall p \in \check{H}\cup\hat{H}\ ; \\
& \delta_p - (P_p - P_{p-1})z_{p-1} \leq 0\ ,\ \forall p \in \check{H}\cup\hat{H}\ ; \\
& P_{p-1} + \delta_p - \sum_{b\in B_p} X_{p,b}\,\alpha_{p,b} = 0\ ,\ \forall p \in \hat{H}\ ; \\
& \sum_{b\in B_p} \alpha_{p,b} = 1\ ,\ \forall p \in \hat{H}\ ; \\
& \{\alpha_{p,b} : b \in B_p\} := \text{SOS2}\ ,\qquad \forall p \in \hat{H}\ .
\end{aligned}$$

with two dummy variables $z_0 := 1$, $z_{\overline{p}} := 0$ and the new set of variables $\alpha_{p,b}$.

The Refinement phase

Refining the Lower Bounding problem



- Add a breakpoint where the solution of problem $\mathcal{Q}$ of the previous iteration lies (global convergence);
- Add a breakpoint where the solution of problem $\mathcal{R}$ of the previous iteration lies (speed up the convergence).

The Lower Bounding problem tightening

Lower Bounding problem tightening

Let us consider the convex pieces:

$$g(P_{p-1} + \delta_p) - g(P_{p-1})$$

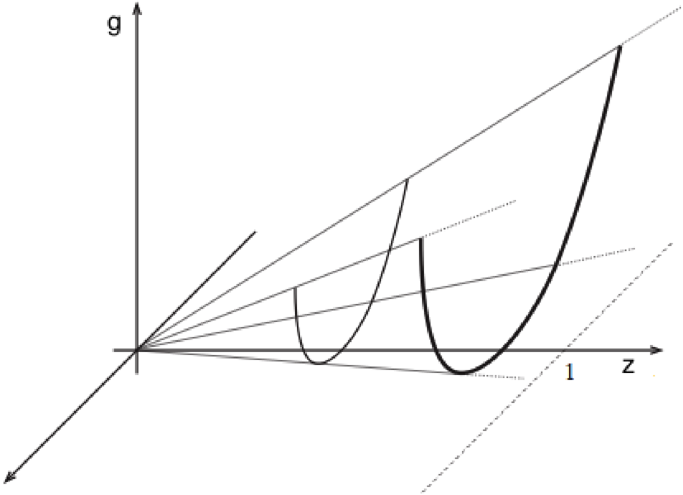
with

- $0 \leq \delta_p \leq (P_p - P_{p-1})z_{p-1}$
- $z_{p-1} \in \{0, 1\}$

Its **convex envelope** is:

$$z_{p-1}(g(P_{p-1} + \delta_p / z_{p-1}) - g(P_{p-1}))$$

Perspective function



Where can we exploit it?

Use it to solve the Lower Bounding problem:

- Reformulate the convex MINLP
- Stronger the convex continuous relaxation
- Generate stronger linear cuts
- Solve the convex MINLP with cutting plane

Computational Results

Computational Results

- PC: linearization of PR of LB problem
- STD: linearization of original LB problem
- Bonmin
- Minotaur
- SCIP

Tests on non linear knapsack problem and uncapacitated facility location problem.

10,000 seconds time limit.

The non linear knapsack problem

$$
\text{(NCK)} \quad
\begin{array}{lll}
\max & \sum_{j \in N} p_j & \\
 & p_j - \frac{c_j}{1 + b_j \exp(-a_j(x_j + d_j))} \leq 0 & j \in N \\
 & \sum_{j \in N} x_j \leq C & \\
 & 0 \leq x_j \leq U & j \in N
\end{array}
$$

- $|N| \in \{10, 20, 50, 100, 200, 500\}$ (10 instances for each)
- Random Uniformly:
  $a_j \in [\,0.1\,,\,0.2\,]$, $b_j \in [\,0\,,\,100\,]$, $c_j \in [\,0\,,\,100\,]$, and $d_j \in [\,-100\,,\,0\,]$.

Results on the non linear knapsack problem

	Bonmin							Minotaur					
size	B-BB		B-OA		B-Hyb		B-OA-C	BNB-I		QG-I		QPD-I	
	time	gap	time	gap	time	gap	time	time	gap	time	gap	time	gap
10	1.06	-	0.25	-	0.59	-	0.27	0.22	-	0.11	-	0.09	-
20	2.99	-	0.34	-	2.12	-	0.32	0.53	-	0.22	-	0.16	-
50	13.8	-	0.65	-	8.05	-	0.62	2.97	-	1.07	-	0.63	-
100	78.9	-	9.16	-	7936	1.00	1.07	13.0	-	4.25	-	3.44	-
200	1000	-	5035	0.62	4019	0.88	2.24	88.5	-	37.8	-	28.6	-
500	tl	0.12	8035	0.62	9027	1.49	8.41	8621	0.07	7080	0.15	7692	0.16

Table: NCK: Bonmin and Minotaur options comparison

Results on the non linear knapsack problem

size	PC		STD		Bonmin	MINOTAUR			SCIP
	time	cuts	time	cuts	time	time	gap	bgap	time
10	0.014	96	0.015	102	0.267	0.09	-	-	0.07
20	0.021	155	0.019	195	0.324	0.16	-	-	0.10
50	0.048	431	0.085	678	0.617	0.63	-	-	0.21
100	0.072	947	0.183	1182	1.067	3.44	-	-	0.66
200	0.105	1780	0.565	2461	2.237	28.6	-	-	131.2
500	0.380	4681	3.593	7821	8.406	7080	0.15	0.05	181.4

Table: NCK: comparison among the different algorithms

The uncapacitated facility location problem

$$
\text{(UFL)} \quad
\begin{array}{lll}
\min & \sum_{k\in K} C_k y_k + \sum_{t\in T} \sum_{k\in K} s_{kt} & \\
 & g_{kt}(w_{kt}) - s_{kt} \leq 0 & t \in T \ , \ k \in K \\
 & \sum_{k\in K} w_{kt} = 1 & t \in T \\
 & 0 \leq w_{kt} \leq y_k & t \in T \ , \ k \in K \\
 & y_k \in \{0,1\} & k \in K
\end{array}
$$

For each combination $(|K|,|T|) \in \{\,(6,12)\,,\,(12,24)\,,\,(24,48)\,\}$ we generated 3 instances of increasing difficulty.

Results on the uncapacitated facility location problem

	Bonmin						Minotaur					
instance	B-BB		B-OA-C		B-Hy-C		BNB		QPD		QG-I	
	time	gap	time	gap	time	gap	time	gap	time	gap	time	gap
6x12x1	176	-	1.76	-	1.37	-	538	-	24.8	-	4.66	-
6x12x2	tl	1.16	7.25	-	5.64	-	tl	29.17	tl	51.08	65.5	-
6x12x3	tl	657.6	tl	∞	tl	∞	tl	∞	tl	315.5	tl	260.3
12x24x1	1592	-	9.68	-	7.14	-	tl	8.07	tl	66.57	57.4	-
12x24x2	tl	18.77	93.8	-	57.9	-	tl	∞	tl	∞	tl	17.40
12x24x3	tl	∞	tl	∞	tl	∞	tl	∞	tl	∞	tl	271.6
24x48x1	tl	84.70	116	-	132	-	tl	∞	tl	∞	2844	-
24x48x2	tl	73.44	tl	∞	tl	∞	tl	∞	tl	∞	tl	31.49

Table: UFL: Comparison among Bonmin and Minotaur options

Results on the uncapacitated facility location problem

instance	PC				STD				Bonmin			Minotaur		
	time	gap	bgap	cuts	time	gap	bgap	cuts	time	gap	bgap	time	gap	bgap
6x12x1	0.35	-	-	1673	0.26	-	-	1531	1.37	-	-	4.66	-	-
6x12x2	0.45	-	-	1842	0.42	-	-	1796	5.64	-	-	65.6	-	-
6x12x3	7921	-	-	33417	tl	54.3	52.4	180561	tl	657	796	tl	260	615
12x24x1	3.36	-	-	9565	2.55	-	-	8971	7.14	-	-	57.4	-	-
12x24x2	46.1	-	-	19653	27.3	-	-	17384	57.9	-	-	tl	17.4	10.5
12x24x3	tl	23.9	23.9	127380	tl	121	134	284557	tl	∞	1524	tl	272	1447
24x48x1	261	-	-	81372	316	-	-	102160	116	-	-	2844	-	-
24x48x2	tl	5.93	5.67	164809	tl	9.66	9.66	409177	tl	73.4	26.4	tl	31.5	24.6

Table: UFL: Comparison among different algorithms

Conclusions and Future Directions

- Flexible framework that guarantees convergence to global solution for relevant class of MINLPs
- Perspective reformulation of the convex Lower Bounding problem

With C. Artigues, A. Frangioni, C. Gentile, R. Trindade, S. Ulrich Ngueveu

- Integration the perspective reformulation (reuse the cuts at each iterations)
- Is the LB problem formulation the tightest?
- Use the “Piecewise linear bounding of univariate functions” for the concave part

With J. Lee, D. Skipper, D. Thomopulos

- Use disjunctive cuts to tighten the formulation instead of adding breakpoints

Thanks!