Mixed Integer Nonlinear Programming (MINLP)

Sven Leyffer

MCS Division
Argonne National Lab
leyffer@mcs.anl.gov

Jeff Linderoth

ISE Department
Lehigh University
jtl3@lehigh.edu

Overview

Part I

Introduction, Applications, and Formulations

The Problem of the Day

Mixed Integer Nonlinear Program (MINLP)

$$\begin{cases} \underset{x,y}{\text{minimize}} & f(x, y) \\ \text{subject to} & c(x, y) \leq 0 \\ & x \in X, \; y \in Y \text{ integer} \end{cases}$$

The Problem of the Day

Mixed Integer Nonlinear Program (MINLP)

$$\begin{cases} \underset{x,y}{\text{minimize}} & f(x, y) \\ \text{subject to} & c(x, y) \leq 0 \\ & x \in X, \ y \in Y \text{ integer} \end{cases}$$

- f, c smooth (convex) functions
- X, Y polyhedral sets, e.g. $Y = \{y \in [0, 1]^p \mid Ay \leq b\}$

The Problem of the Day

Mixed Integer Nonlinear Program (MINLP)

$$\begin{cases} \underset{x,y}{\text{minimize}} & f(x, y) \\ \text{subject to} & c(x, y) \leq 0 \\ & x \in X, \; y \in Y \text{ integer} \end{cases}$$

- f, c smooth (convex) functions
- X, Y polyhedral sets, e.g. $Y = \{y \in [0,1]^p \mid Ay \leq b\}$
- $y \in Y$ integer $\Rightarrow$ hard problem

The Problem of the Day

Mixed Integer Nonlinear Program (MINLP)

$$\begin{cases} \underset{x,y}{\text{minimize}} & f(x,y) \\ \text{subject to} & c(x,y) \leq 0 \\ & x \in X,\ y \in Y \text{ integer} \end{cases}$$

- f, c smooth (convex) functions
- X, Y polyhedral sets, e.g. $Y = \{y \in [0,1]^p \mid Ay \leq b\}$
- $y \in Y$ integer $\Rightarrow$ hard problem
- f, c not convex $\Rightarrow$ very hard problem

Why the MI?

- We can use 0-1 (binary) variables for a variety of purposes
  - Modeling yes/no decisions
  - Enforcing disjunctions
  - Enforcing logical conditions
  - Modeling fixed costs
  - Modeling piecewise linear functions

Why the MI?

- We can use 0-1 (binary) variables for a variety of purposes
  - Modeling yes/no decisions
  - Enforcing disjunctions
  - Enforcing logical conditions
  - Modeling fixed costs
  - Modeling piecewise linear functions
- If the variable is associated with a physical entity that is indivisible, then it must be integer
  - Number of aircraft carriers to to produce. Gomory's Initial Motivation

A Popular MINLP Method

Dantzig's Two-Phase Method for MINLP

Adapted by Leyffer and Linderoth

A Popular MINLP Method

Dantzig's Two-Phase Method for MINLP

Adapted by Leyffer and Linderoth

1. Convince the user that he or she does not wish to solve a mixed integer nonlinear programming problem at all!

A Popular MINLP Method

Dantzig's Two-Phase Method for MINLP

Adapted by Leyffer and Linderoth

1. Convince the user that he or she does not wish to solve a mixed integer nonlinear programming problem at all!
2. Otherwise, solve the continuous relaxation (NLP) and round off the minimizer to the nearest integer.

A Popular MINLP Method

Dantzig's Two-Phase Method for MINLP

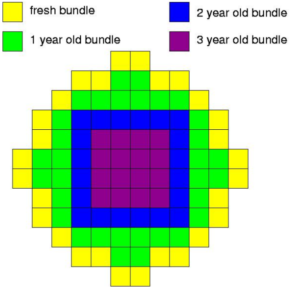
Adapted by Leyffer and Linderoth

1. Convince the user that he or she does not wish to solve a mixed integer nonlinear programming problem at all!
2. Otherwise, solve the continuous relaxation (NLP) and round off the minimizer to the nearest integer.

- For $0-1$ problems, or those in which the $|y|$ is "small", the continuous approximation to the discrete decision is not accurate enough for practical purposes.
- Conclusion: MINLP methods must be studied!

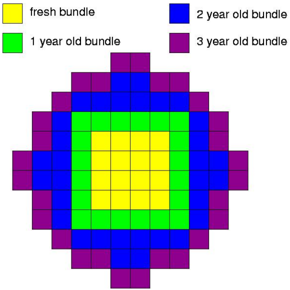
Example: Core Reload Operation (Quist, A.J., 2000)

- max. reactor efficiency after reload subject to diffusion PDE & safety
- diffusion PDE $\simeq$ nonlinear equation $\Rightarrow$ integer & nonlinear model
- avoid reactor becoming sub-critical



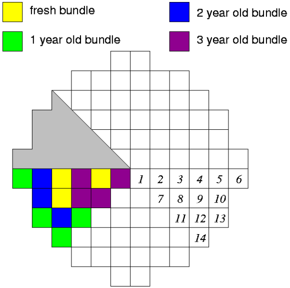
Example: Core Reload Operation (Quist, A.J., 2000)

- max. reactor efficiency after reload subject to diffusion PDE & safety
- diffusion PDE $\simeq$ nonlinear equation $\Rightarrow$ integer & nonlinear model
- avoid reactor becoming overheated



Example: Core Reload Operation (Quist, A.J., 2000)

- look for cycles for moving bundles:
  e.g. $4 \to 6 \to 8 \to 10$
  i.e. bundle moved from 4 to 6 ...
- model with binary $x_{ilm} \in \{0, 1\}$
  $x_{ilm} = 1$
  $\Leftrightarrow$ node i has bundle l of cycle m



AMPL Model of Core Reload Operation

Exactly one bundle per node:

$$\sum_{l=1}^{L} \sum_{m=1}^{M} x_{ilm} = 1 \qquad \forall i \in I$$

AMPL model:

```
var x {I,L,M} binary ;
Bundle {i in I}:  sum{l in L, m in M} x[i,l,m] = 1 ;
```

- Multiple Choice: One of the most common uses of IP
- Full AMPL model `c-reload.mod` at `www.mcs.anl.gov/~leyffer/MacMINLP/`

Gas Transmission Problem (De Wolf and Smeers, 2000)

- Belgium has no gas!

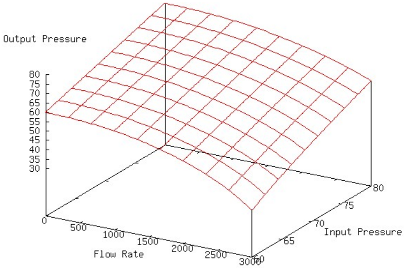
Gas Transmission Problem (De Wolf and Smeers, 2000)

- Belgium has no gas!
- All natural gas is imported from Norway, Holland, or Algeria.
- Supply gas to all demand points in a network in a minimum cost fashion.

Gas Transmission Problem (De Wolf and Smeers, 2000)

- Belgium has no gas!
- All natural gas is imported from Norway, Holland, or Algeria.
- Supply gas to all demand points in a network in a minimum cost fashion.
- Gas is pumped through the network with a series of compressors
- There are constraints on the pressure of the gas within the pipe

Pressure Loss is Nonlinear



- Assume horizontal pipes and steady state flows
- Pressure loss p across a pipe is related to the flow rate f as

$$p_{in}^2 - p_{out}^2 = \frac{1}{\Psi} \operatorname{sign}(f) f^2$$

- Ψ: "Friction Factor"

Gas Transmission: Problem Input

- Network (N, A). $A = A_p \cup A_a$
  - A_a: active arcs have compressor. Flow rate can increase on arc
  - A_p: passive arcs simply conserve flow rate
- $N_s \subseteq N$: set of supply nodes
- $c_i, i \in N_s$: Purchase cost of gas
- $\underline{s}_i, \overline{s}_i$: Lower and upper bounds on gas “supply” at node i
- $\underline{p}_i, \overline{p}_i$: Lower and upper bounds on gas pressure at node i

Gas Transmission: Problem Input

- Network (N, A). $A = A_p \cup A_a$
  - A_a: active arcs have compressor. Flow rate can increase on arc
  - A_p: passive arcs simply conserve flow rate
- $N_s \subseteq N$: set of supply nodes
- $c_i, i \in N_s$: Purchase cost of gas
- $\underline{s}_i, \overline{s}_i$: Lower and upper bounds on gas "supply" at node i
- $\underline{p}_i, \overline{p}_i$: Lower and upper bounds on gas pressure at node i
- $s_i, i \in N$: supply at node i.
  - $s_i > 0 \Rightarrow$ gas added to the network at node i
  - $s_i < 0 \Rightarrow$ gas removed from the network at node i to meet demand
- $f_{ij}, (i, j) \in A$: flow along arc (i, j)
  - $f(i, j) > 0 \Rightarrow$ gas flows $i \rightarrow j$
  - $f(i, j) < 0 \Rightarrow$ gas flows $j \rightarrow i$

Gas Transmission Model

$$\min \sum_{j \in N_s} c_j s_j$$

subject to

$$\begin{aligned}
\sum_{j|(i,j)\in A} f_{ij} &= s_i & \forall i \in N \\
\text{sign}(f_{ij}) f_{ij}^2 - \Psi_{ij}(p_i^2 - p_j^2) &= 0 & \forall (i,j) \in A_p \\
\text{sign}(f_{ij}) f_{ij}^2 - \Psi_{ij}(p_i^2 - p_j^2) &\geq 0 & \forall (i,j) \in A_a \\
s_i &\in [\underline{s}_i, \overline{s}_i] & \forall i \in N \\
p_i &\in [\underline{p}_i, \overline{p}_i] & \forall i \in N \\
f_{ij} &\geq 0 & \forall (i,j) \in A_a
\end{aligned}$$

Your First Modeling Trick

- Don't include nonlinearities or nonconvexities unless necessary!
- Replace $p_i^2 \leftarrow \rho_i$

Your First Modeling Trick

- Don't include nonlinearities or nonconvexities unless necessary!
- Replace $p_i^2 \leftarrow \rho_i$

$$
\begin{aligned}
\text{sign}(f_{ij})f_{ij}^2 - \Psi_{ij}(\rho_i - \rho_j) &= 0 \qquad \forall (i,j) \in A_p \\
f_{ij}^2 - \Psi_{ij}(\rho_i - \rho_j) &\geq 0 \qquad \forall (i,j) \in A_a \\
\rho_i &\in [\sqrt{\underline{p}_i}, \sqrt{\overline{p}_i}] \qquad \forall i \in N
\end{aligned}
$$

Your First Modeling Trick

- Don't include nonlinearities or nonconvexities unless necessary!
- Replace $p_i^2 \leftarrow \rho_i$

$$\begin{aligned} \text{sign}(f_{ij})f_{ij}^2 - \Psi_{ij}(\rho_i - \rho_j) &= 0 \qquad \forall (i,j) \in A_p \\ f_{ij}^2 - \Psi_{ij}(\rho_i - \rho_j) &\geq 0 \qquad \forall (i,j) \in A_a \\ \rho_i &\in [\sqrt{\underline{p}_i}, \sqrt{\overline{p}_i}] \qquad \forall i \in N \end{aligned}$$

- This trick only works because
  1. p_i^2 terms appear only in the bound constraints
  2. Also $f_{ij} \geq 0 \ \forall (i,j) \in A_a$
- This model is nonconvex: $\text{sign}(f_{ij})f_{ij}^2$ is a nonconvex function
- Some solvers do not like sign

Dealing with sign(·): The NLP Way

- Use auxiliary binary variables to indicate direction of flow
- Let $|f_{ij}| \leq F \; \forall (i,j) \in A_p$

$$z_{ij} = \begin{cases} 1 & f_{ij} \geq 0 \\ 0 & f_{ij} \leq 0 \end{cases} \qquad \begin{aligned} f_{ij} &\geq -F(1 - z_{ij}) \\ f_{ij} &\leq F z_{ij} \end{aligned}$$

- Note that

$$\text{sign}(f_{ij}) = 2z_{ij} - 1$$

- Write constraint as

$$(2z_{ij} - 1)f_{ij}^2 - \Psi_{ij}(\rho_i - \rho_j) = 0.$$

Special Ordered Sets

- Sven thinks this ’NLP trick’ is pretty cool

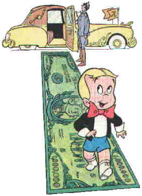
Special Ordered Sets

- Sven thinks this 'NLP trick' is pretty cool
- It is not how it is done in De Wolf and Smeers (2000).

Special Ordered Sets

- Sven thinks this 'NLP trick' is pretty cool
- It is not how it is done in De Wolf and Smeers (2000).
- Heuristic for finding a good starting solution, then a local optimization approach based on a piecewise-linear simplex method
- Another (similar) approach involves approximating the nonlinear function by piecewise linear segments, but searching for the globally optimal solution: Special Ordered Sets of Type 2
- If the "multidimensional" nonlinearity cannot be removed, resort to Special Ordered Sets of Type 3

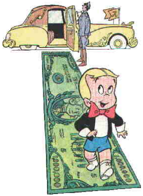
Portfolio Management



- N: Universe of asset to purchase
- x_i: Amount of asset i to hold
- B: Budget

$$\min_{x \in \mathbb{R}_+^{|N|}} \left\{ u(x) \mid \sum_{i \in N} x_i = B \right\}$$

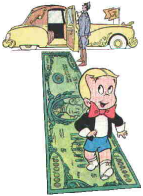
Portfolio Management



- N: Universe of asset to purchase
- x_i: Amount of asset i to hold
- B: Budget

$$\min_{x \in \mathbb{R}_+^{|N|}} \left\{ u(x) \mid \sum_{i \in N} x_i = B \right\}$$

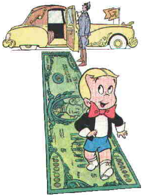
- Markowitz: $u(x) \stackrel{\text{def}}{=} -\alpha^T x + \lambda x^T Q x$
  - α: Expected returns
  - Q: Variance-covariance matrix of expected returns
  - λ: Risk aversion parameter



More Realistic Models

- $b \in \mathbb{R}^{|N|}$ of "benchmark" holdings
- Benchmark Tracking: $u(x) \stackrel{\text{def}}{=} (x - b)^T Q(x - b)$
  - Constraint on $\mathbb{E}$[Return]: $\alpha^T x \geq r$

More Realistic Models

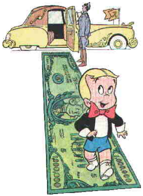


- $b \in \mathbb{R}^{|N|}$ of "benchmark" holdings
- Benchmark Tracking: $u(x) \stackrel{\text{def}}{=} (x-b)^T Q(x-b)$
  - Constraint on $\mathbb{E}$[Return]: $\alpha^T x \geq r$
- Limit Names: $|i \in N \;:\; x_i > 0| \leq K$
  - Use binary indicator variables to model the implication $x_i > 0 \Rightarrow y_i = 1$
  - Implication modeled with variable upper bounds:

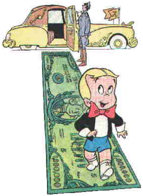
$$x_i \leq By_i \qquad \forall i \in N$$

  - $\sum_{i \in N} y_i \leq K$

Even More Models

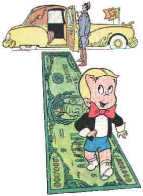


- Min Holdings: $(x_i = 0) \vee (x_i \geq m)$
  - Model implication: $x_i > 0 \Rightarrow x_i \geq m$
  - $x_i > 0 \Rightarrow y_i = 1 \Rightarrow x_i \geq m$
  - $x_i \leq By_i, x_i \geq my_i \ \forall i \in N$



Even More Models

- Min Holdings: $(x_i = 0) \vee (x_i \geq m)$
  - Model implication: $x_i > 0 \Rightarrow x_i \geq m$
  - $x_i > 0 \Rightarrow y_i = 1 \Rightarrow x_i \geq m$
  - $x_i \leq By_i, x_i \geq my_i \; \forall i \in N$
- Round Lots: $x_i \in \{kL_i, k = 1, 2, \ldots\}$
  - $x_i - z_i L_i = 0, z_i \in \mathbb{Z}_+ \; \forall i \in N$



Even More Models

- Min Holdings: $(x_i = 0) \vee (x_i \geq m)$
  - Model implication: $x_i > 0 \Rightarrow x_i \geq m$
  - $x_i > 0 \Rightarrow y_i = 1 \Rightarrow x_i \geq m$
  - $x_i \leq By_i, x_i \geq my_i \ \forall i \in N$
- Round Lots: $x_i \in \{kL_i, k = 1, 2, \ldots\}$
  - $x_i - z_i L_i = 0, z_i \in \mathbb{Z}_+ \ \forall i \in N$
- Vector h of initial holdings
- Transactions: $t_i = |x_i - h_i|$
- Turnover: $\sum_{i \in N} t_i \leq \Delta$
- Transaction Costs: $\sum_{i \in N} c_i t_i$ in objective
- Market Impact: $\sum_{i \in N} \gamma_i t_i^2$ in objective

Multiproduct Batch Plants (Kocis and Grossmann, 1988)

- M: Batch Processing Stages
- N: Different Products
- H: Horizon Time
- Q_i: Required quantity of product i
- t_{ij}: Processing time product i stage j
- S_{ij}: "Size Factor" product i stage j

Multiproduct Batch Plants (Kocis and Grossmann, 1988)

- M: Batch Processing Stages
- N: Different Products
- H: Horizon Time
- Q_i: Required quantity of product i
- t_{ij}: Processing time product i stage j
- S_{ij}: "Size Factor" product i stage j

- B_i: Batch size of product $i \in N$
- V_j: Stage j size: $V_j \geq S_{ij} B_i \; \forall i, j$
- N_j: Number of machines at stage j
- C_i: Longest stage time for product i: $C_i \geq t_{ij} / N_j \; \forall i, j$

Multiproduct Batch Plants

$$\min \sum_{j \in M} \alpha_j N_j V_j^{\beta_j}$$

s.t.

$$\begin{array}{rcll} V_j - S_{ij} B_i & \geq & 0 & \forall i \in N, \forall j \in M \\ C_i N_j & \geq & t_{ij} & \forall i \in N, \forall j \in M \\ \sum_{i \in N} \frac{Q_i}{B_i} C_i & \leq & H & \\ \text{Bound Constraints} & \text{on} & V_j, C_i, B_i, N_j & \\ N_j & \in & \mathbb{Z} & \forall j \in M \end{array}$$

Modeling Trick #2

- Horizon Time and Objective Function Nonconvex. :-(

Modeling Trick #2

- Horizon Time and Objective Function Nonconvex. :-(
- Sometimes variable transformations work!

$$v_j = \ln(V_j), n_j = \ln(N_j), b_i = \ln(B_i), c_i = \ln C_i$$

Modeling Trick #2

- Horizon Time and Objective Function Nonconvex. :-(
- Sometimes variable transformations work!

$$v_j = \ln(V_j), n_j = \ln(N_j), b_i = \ln(B_i), c_i = \ln C_i$$

$$\min \sum_{j \in M} \alpha_j e^{N_j + \beta_j V_j}$$

$$\begin{array}{rcll} \text{s.t. } v_j - \ln(S_{ij}) b_i & \geq & 0 & \forall i \in N, \forall j \in M \\ c_i + n_j & \geq & \ln(\tau_{ij}) & \forall i \in N, \forall j \in M \\ \sum_{i \in N} Q_i e^{C_i - B_i} & \leq & H & \\ \text{(Transformed) Bound Constraints} & \text{on} & V_j, C_i, B_i & \end{array}$$

How to Handle the Integrality?

- But what to do about the integrality?

$$1 \leq N_j \leq \overline{N}_j \qquad \forall j \in M, N_j \in \mathbb{Z} \qquad \forall j \in M$$

- $n_j \in \{0, \ln(2), \ln(3), \ldots \ldots\}$

$$Y_{kj} = \begin{cases} 1 & n_j \text{ takes value } \ln(k) \\ 0 & \text{Otherwise} \end{cases}$$

$$\begin{aligned} n_j - \sum_{k=1}^{K} \ln(k) Y_{kj} &= 0 \qquad \forall j \in M \\ \sum_{k=1}^{K} Y_{kj} &= 1 \qquad \forall j \in M \end{aligned}$$

- This model is available at `http://www-unix.mcs.anl.gov/~leyffer/macminlp/problems/batch.mod`

A Small Smattering of Other Applications

- Chemical Engineering Applications:
  - process synthesis (Kocis and Grossmann, 1988)
  - batch plant design (Grossmann and Sargent, 1979)
  - cyclic scheduling (Jain, V. and Grossmann, I.E., 1998)
  - design of distillation columns (Viswanathan and Grossmann, 1993)
  - pump configuration optimization (Westerlund, T., Pettersson, F. and Grossmann, I.E., 1994)
- Forestry/Paper
  - production (Westerlund, T., Isaksson, J. and Harjunkoski, I., 1995)
  - trimloss minimization (Harjunkoski, I., Westerlund, T., Pörn, R. and Skrifvars, H., 1998)
- Topology Optimization (Sigmund, 2001)

Part II

Classical Solution Methods

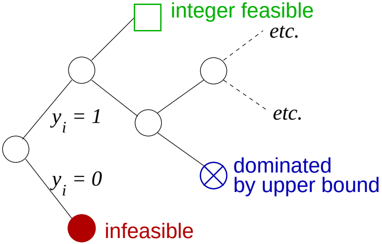
Classical Solution Methods for MINLP

1. Classical Branch-and-Bound
2. Outer Approximation & Benders Decomposition
3. Hybrid Methods
   - LP/NLP Based Branch-and-Bound
   - Integrating SQP with Branch-and-Bound

Branch-and-Bound

Solve relaxed NLP ($0 \leq y \leq 1$ continuous relaxation)
...solution value provides lower bound

- Branch on y_i non-integral
- Solve NLPs & branch until
  1. Node infeasible ... ●
  2. Node integer feasible ... □
     $\Rightarrow$ get upper bound (U)
  3. Lower bound $\geq U$ ... $\otimes$



Search until no unexplored nodes on tree

Variable Selection for Branch-and-Bound

Assume $y_i \in \{0,1\}$ for simplicity ...
$(\hat{x},\hat{y})$ fractional solution to parent node; $\hat{f} = f(\hat{x},\hat{y})$

1. **maximal fractional branching:** choose $\hat{y}_i$ closest to $\frac{1}{2}$

$$\max_i \{\min(1-\hat{y}_i, \hat{y}_i)\}$$

Variable Selection for Branch-and-Bound

Assume $y_i \in \{0, 1\}$ for simplicity ...
$(\hat{x}, \hat{y})$ fractional solution to parent node; $\hat{f} = f(\hat{x}, \hat{y})$

1. **maximal fractional branching:** choose $\hat{y}_i$ closest to $\frac{1}{2}$

$$\max_i \{\min(1 - \hat{y}_i, \hat{y}_i)\}$$

2. **strong branching**: (approx) solve *all* NLP children:

$$f_i^{+/-} \leftarrow \left\{ \begin{array}{ll} \underset{x,y}{\text{minimize}} & f(x, y) \\ \text{subject to} & c(x, y) \leq 0 \\ & x \in X, \; y \in Y, \; y_i = 1/0 \end{array} \right.$$

branching variable y_i that changes objective the most:

$$\max_i \left\{\min(f_i^+, f_i^-)\right\}$$

Node Selection for Branch-and-Bound

Which node n on tree $\mathcal{T}$ should be solved next?

1. **depth-first search:** select deepest node in tree
   - minimizes number of NLP nodes stored
   - exploit warm-starts (MILP/MIQP only)

Node Selection for Branch-and-Bound

Which node n on tree $\mathcal{T}$ should be solved next?

1. **depth-first search:** select deepest node in tree
   - minimizes number of NLP nodes stored
   - exploit warm-starts (MILP/MIQP only)
2. **best estimate:** choose node with best expected integer soln

$$\min_{n \in \mathcal{T}} \left\{ f_{p(n)} + \sum_{i: y_i \text{fractional}} \min\left\{ e_i^+(1 - y_i), e_i^- y_i \right\} \right\}$$

where $f_{p(n)}$ = value of parent node, $e_i^{+/-}$ = pseudo-costs
summing pseudo-cost estimates for all integers in subtree

Outer Approximation (Duran and Grossmann, 1986)

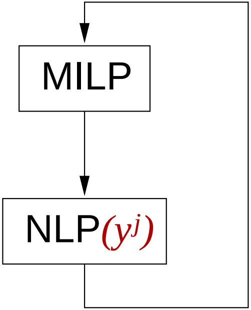
Motivation: avoid solving huge number of NLPs
- Exploit MILP/NLP solvers: decompose integer/nonlinear part

Key idea: reformulate MINLP as MILP (implicit)
- Solve alternating sequence of MILP & NLP

NLP subproblem y_j fixed:

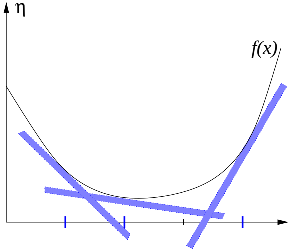
$$\text{NLP}(y_j) \begin{cases} \underset{x}{\text{minimize}} & f(x, y_j) \\ \text{subject to} & c(x, y_j) \leq 0 \\ & x \in X \end{cases}$$

Main Assumption: f, c are convex



Outer Approximation (Duran and Grossmann, 1986)

- let (x_j, y_j) solve NLP(y_j)
- linearize f, c about $(x_j, y_j) =: z_j$
- new objective variable $\eta \geq f(x, y)$
- MINLP $(P) \equiv$ MILP (M)

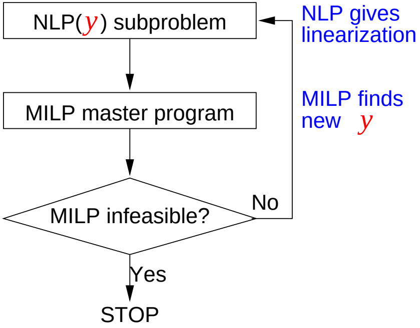


$$
(M) \begin{cases} \underset{z=(x,y),\eta}{\text{minimize}} & \eta \\ \text{subject to} & \eta \geq f_j + \nabla f_j^T (z - z_j) \quad \forall y_j \in Y \\ & 0 \geq c_j + \nabla c_j^T (z - z_j) \quad \forall y_j \in Y \\ & x \in X, \; y \in Y \text{ integer} \end{cases}
$$

SNAG: need all $y_j \in Y$ linearizations

Outer Approximation (Duran and Grossmann, 1986)

(M_k): lower bound (underestimate convex f, c)
NLP(y_j): upper bound U (fixed y_j)



$\Rightarrow$ stop, if lower bound $\geq$ upper bound

Outer Approximation & Benders Decomposition

Take OA cuts for $z_j := (x_j, y_j)$ ... wlog $X = \mathbb{R}^n$

$$\eta \geq f_j + \nabla f_j^T (z - z_j) \quad \& \quad 0 \geq c_j + \nabla c_j^T (z - z_j)$$

sum with $(1, \lambda_j)$ ... λ_j multipliers of NLP(y_j)

$$\eta \geq f_j + \lambda_j^T c_j + (\nabla f_j + \nabla c_j \lambda_j)^T (z - z_j)$$

KKT conditions of NLP(y_j) $\Rightarrow \quad \nabla_x f_j + \nabla_x c_j \lambda_j = 0$
... eliminate x components from valid inequality in y

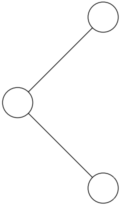
$$\Rightarrow \quad \eta \geq f_j + (\nabla_y f_j + \nabla_y c_j \lambda_j)^T (y - y_j)$$

NB: $\mu_j = \nabla_y f_j + \nabla_y c_j \lambda_j$ multiplier of $y = y_j$ in NLP(y_j)
References: (Geoffrion, 1972)

LP/NLP Based Branch-and-Bound

AIM: avoid re-solving MILP master (M)

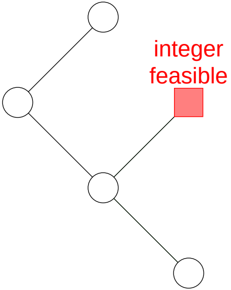
- Consider MILP branch-and-bound



LP/NLP Based Branch-and-Bound

AIM: avoid re-solving MILP master (M)

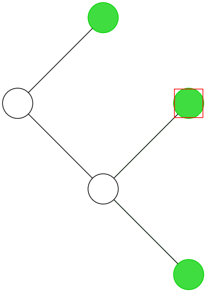
- Consider MILP branch-and-bound
- interrupt MILP, when y_j found
  $\Rightarrow$ solve NLP(y_j) get x_j



LP/NLP Based Branch-and-Bound

AIM: avoid re-solving MILP master (M)

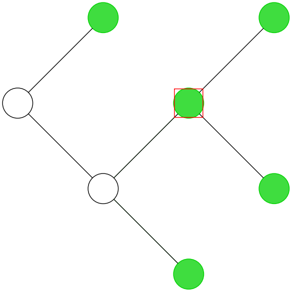
- Consider MILP branch-and-bound
- interrupt MILP, when y_j found
  $\Rightarrow$ solve NLP(y_j) get x_j
- linearize f, c about (x_j, y_j)
  $\Rightarrow$ add linearization to tree



LP/NLP Based Branch-and-Bound

AIM: avoid re-solving MILP master (M)

- Consider MILP branch-and-bound
- interrupt MILP, when y_j found
  $\Rightarrow$ solve NLP(y_j) get x_j
- linearize f, c about (x_j, y_j)
  $\Rightarrow$ add linearization to tree
- continue MILP tree-search



... until lower bound $\geq$ upper bound

LP/NLP Based Branch-and-Bound

- need access to MILP solver ... call back
  - exploit good MILP (branch-cut-price) solver
  - (Akrotirianakis et al., 2001) use Gomory cuts in tree-search

- preliminary results: order of magnitude faster than OA
  - same number of NLPs, but only one MILP
- similar ideas for Benders & Extended Cutting Plane methods
- recent implementation by CMU/IBM group

References: (Quesada and Grossmann, 1992)

Integrating SQP & Branch-and-Bound

AIM: Avoid solving NLP node to convergence.

Sequential Quadratic Programming (SQP)
$\rightarrow$ solve sequence (QP_k) at every node

$$
(QP_k) \begin{cases} \underset{d}{\text{minimize}} & f_k + \nabla f_k^T d + \frac{1}{2} d^T H_k d \\ \text{subject to} & c_k + \nabla c_k^T d \leq 0 \\ & x_k + d_x \in X \\ & y_k + d_y \in \hat{Y}. \end{cases}
$$

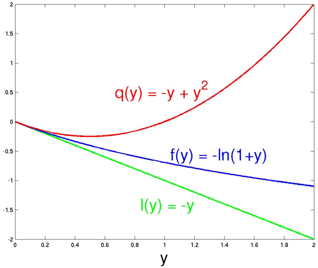
Early branching:
After QP step choose non-integral y_i^{k+1}, branch & continue SQP
References: (Borchers and Mitchell, 1994; Leyffer, 2001)

Integrating SQP & Branch-and-Bound

SNAG: (QP_k) not lower bound

$\Rightarrow$ no fathoming from upper bound

$$\begin{array}{ll} \underset{d}{\text{minimize}} & f_k + \nabla f_k^T d + \frac{1}{2} d^T H_k d \\ \text{subject to} & c_k + \nabla c_k^T d \leq 0 \\ & x_k + d_x \in X \\ & y_k + d_y \in \hat{Y}. \end{array}$$

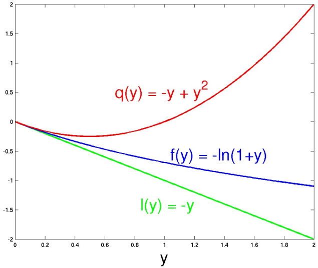


NB: (QP_k) inconsistent and trust-region active $\Rightarrow$ do not fathom

Integrating SQP & Branch-and-Bound

SNAG: (QP_k) not lower bound
$\Rightarrow$ no fathoming from upper bound

$$
\begin{array}{ll}
\underset{d}{\text{minimize}} & f_k + \nabla f_k^T d + \frac{1}{2} d^T H_k d \\
\text{subject to} & c_k + \nabla c_k^T d \leq 0 \\
& x_k + d_x \in X \\
& y_k + d_y \in \hat{Y}.
\end{array}
$$



Remedy: Exploit OA underestimating property (Leyffer, 2001):

- add objective cut $f_k + \nabla f_k^T d \leq U - \epsilon$ to (QP_k)
- fathom node, if (QP_k) inconsistent

NB: (QP_k) inconsistent and trust-region active $\Rightarrow$ do not fathom

Comparison of Classical MINLP Techniques

Summary of numerical experience

1. Quadratic OA master: usually fewer iteration
   MIQP harder to solve
2. NLP branch-and-bound faster than OA
   ... depends on MIP solver
3. LP/NLP-based-BB order of magnitude faster than OA
   . . . also faster than B&B
4. Integrated SQP-B&B up to $3\times$ faster than B&B
   $\simeq$ number of QPs per node
5. ECP works well, if function/gradient evals expensive

Part III

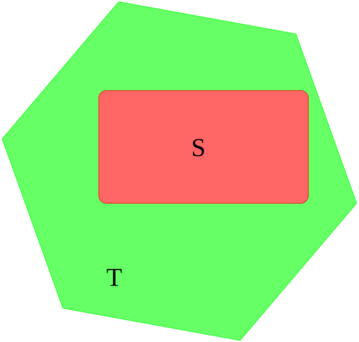
Modern Developments in MINLP

Modern Methods for MINLP

1. Formulations
   - Relaxations
   - Good formulations: big $M's$ and disaggregation
2. Cutting Planes
   - Cuts from relaxations and special structures
   - Cuts from integrality
3. Handling Nonconvexity
   - Envelopes
   - Methods

Relaxations

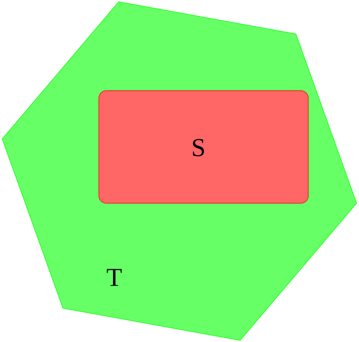
- $z(S) \stackrel{\text{def}}{=} \min_{x \in S} f(x)$
- $z(T) \stackrel{\text{def}}{=} \min_{x \in T} f(x)$



- Independent of f, S, T:
  $z(T) \leq z(S)$

Relaxations

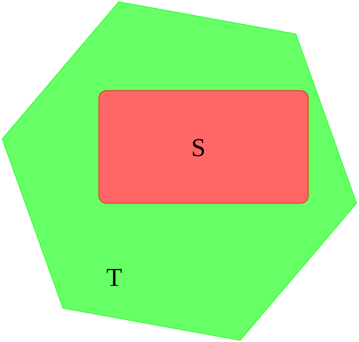
- $z(S) \stackrel{\text{def}}{=} \min_{x \in S} f(x)$
- $z(T) \stackrel{\text{def}}{=} \min_{x \in T} f(x)$



- Independent of f, S, T: $z(T) \leq z(S)$
- If $x^*_T = \arg\min_{x \in T} f(x)$
- And $x^*_T \in S$, then

Relaxations

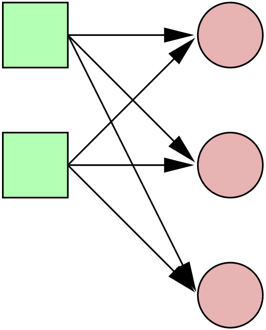
- $z(S) \stackrel{\text{def}}{=} \min_{x \in S} f(x)$
- $z(T) \stackrel{\text{def}}{=} \min_{x \in T} f(x)$



- Independent of f, S, T: $z(T) \leq z(S)$
- If $x_T^* = \arg\min_{x \in T} f(x)$
- And $x_T^* \in S$, then
- $x_T^* = \arg\min_{x \in S} f(x)$

UFL: Uncapacitated Facility Location

- Facilities: J
- Customers: I



$$\min \sum_{j \in J} f_j x_j + \sum_{i \in I} \sum_{j \in J} f_{ij} y_{ij}$$

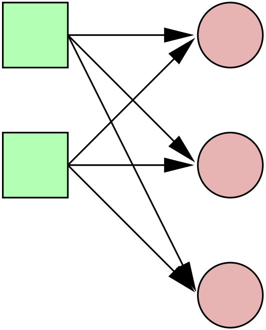
$$\sum_{j \in J} y_{ij} = 1 \quad \forall i \in I$$

$$\sum_{i \in I} y_{ij} \leq |I| x_j \quad \forall j \in J \qquad (1)$$

OR $y_{ij} \leq x_j \quad \forall i \in I, \; j \in J \qquad (2)$

UFL: Uncapacitated Facility Location

- Facilities: J
- Customers: I



$$\min \sum_{j \in J} f_j x_j + \sum_{i \in I} \sum_{j \in J} f_{ij} y_{ij}$$

$$\sum_{j \in J} y_{ij} = 1 \quad \forall i \in I$$

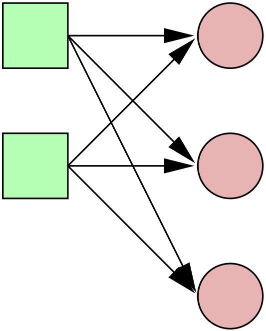
$$\sum_{i \in I} y_{ij} \leq |I| x_j \quad \forall j \in J \qquad (1)$$

$$\text{OR } y_{ij} \leq x_j \quad \forall i \in I,\ j \in J \qquad (2)$$

- Which formulation is to be preferred?

UFL: Uncapacitated Facility Location

- Facilities: J
- Customers: I



$$\min \sum_{j \in J} f_j x_j + \sum_{i \in I} \sum_{j \in J} f_{ij} y_{ij}$$

$$\sum_{j \in J} y_{ij} = 1 \quad \forall i \in I$$

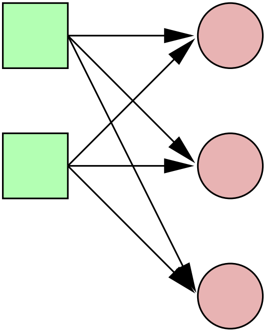
$$\sum_{i \in I} y_{ij} \leq |I| x_j \quad \forall j \in J \qquad (1)$$

$$\text{OR } y_{ij} \leq x_j \quad \forall i \in I,\ j \in J \qquad (2)$$

- Which formulation is to be preferred?
- $I = J = 40$. Costs random.

UFL: Uncapacitated Facility Location

- Facilities: J
- Customers: I



$$\min \sum_{j \in J} f_j x_j + \sum_{i \in I} \sum_{j \in J} f_{ij} y_{ij}$$

$$\sum_{j \in J} y_{ij} = 1 \quad \forall i \in I$$

$$\sum_{i \in I} y_{ij} \leq |I| x_j \quad \forall j \in J \qquad (1)$$

OR $$y_{ij} \leq x_j \quad \forall i \in I,\ j \in J \qquad (2)$$

- Which formulation is to be preferred?
- $I = J = 40$. Costs random.
  - Formulation 1. 53,121 seconds, optimal solution.
  - Formulation 2. 2 seconds, optimal solution.

Valid Inequalities

- Sometimes we can get a better formulation by dynamically improving it.

- An inequality $\pi^T x \leq \pi_0$ is a valid inequality for S if $\pi^T x \leq \pi_0 \; \forall x \in S$
- Alternatively: $\max_{x \in S}\{\pi^T x\} \leq \pi_0$

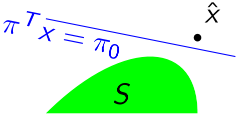
Valid Inequalities

- Sometimes we can get a better formulation by dynamically improving it.

- An inequality $\pi^T x \leq \pi_0$ is a valid inequality for S if $\pi^T x \leq \pi_0 \ \forall x \in S$
- Alternatively: $\max_{x \in S}\{\pi^T x\} \leq \pi_0$

- **Thm:** (Hahn-Banach). Let $S \subset \mathbb{R}^n$ be a closed, convex set, and let $\hat{x} \notin S$. Then there exists $\pi \in \mathbb{R}^n$ such that

$$\pi^T \hat{x} > \max_{x \in S}\{\pi^T x\}$$



Nonlinear Branch-and-Cut

Consider MINLP

$$\begin{cases} \underset{x,y}{\text{minimize}} & f_x^T x + f_y^T y \\ \text{subject to} & c(x,y) \leq 0 \\ & y \in \{0,1\}^p, \; 0 \leq x \leq U \end{cases}$$

- Note the Linear objective
- This is WLOG:

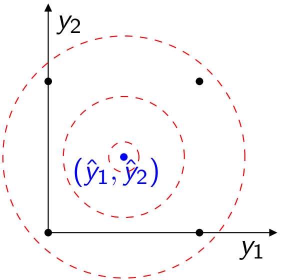
$$\min \; f(x,y) \qquad \Leftrightarrow \qquad \min \; \eta \;\; \text{s.t.} \;\; \eta \geq f(x,y)$$

It's Actually Important!

- We want to approximate the convex hull of integer solutions, but without a linear objective function, the solution to the relaxation might occur in the interior.
- No Separating Hyperplane! :-(

$$\min (y_1 - 1/2)^2 + (y_2 - 1/2)^2$$

$$\text{s.t. } y_1 \in \{0, 1\}, y_2 \in \{0, 1\}$$



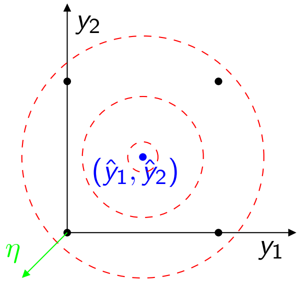
It's Actually Important!

- We want to approximate the convex hull of integer solutions, but without a linear objective function, the solution to the relaxation might occur in the interior.
- No Separating Hyperplane! :-(

$$\min (y_1 - 1/2)^2 + (y_2 - 1/2)^2$$

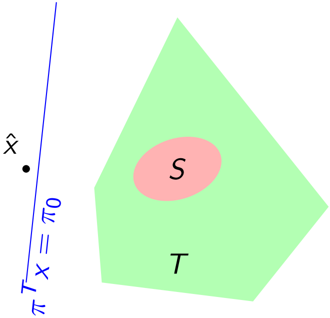
$$\text{s.t. } y_1 \in \{0, 1\}, y_2 \in \{0, 1\}$$

$$\eta \geq (y_1 - 1/2)^2 + (y_2 - 1/2)^2$$



Valid Inequalities From Relaxations

- Idea: Inequalities valid for a relaxation are valid for original
- Generating valid inequalities for a relaxation is often easier.



- Separation Problem over T: Given $\hat{x}$, T find (π, π_0) such that $\pi^T \hat{x} > \pi_0$, $\pi^T x \leq \pi_0 \forall x \in T$

Simple Relaxations

- Idea: Consider one row relaxations

Simple Relaxations

- Idea: Consider one row relaxations
- If $P = \{x \in \{0,1\}^n \mid Ax \leq b\}$, then for any row i, $P_i = \{x \in \{0,1\}^n \mid a_i^T x \leq b_i\}$ is a relaxation of P.

Simple Relaxations

- Idea: Consider one row relaxations
- If $P = \{x \in \{0,1\}^n \mid Ax \leq b\}$, then for any row i, $P_i = \{x \in \{0,1\}^n \mid a_i^T x \leq b_i\}$ is a relaxation of P.
- If the intersection of the relaxations is a good approximation to the true problem, then the inequalities will be quite useful.

Simple Relaxations

- Idea: Consider one row relaxations
- If $P = \{x \in \{0,1\}^n \mid Ax \leq b\}$, then for any row i, $P_i = \{x \in \{0,1\}^n \mid a_i^T x \leq b_i\}$ is a relaxation of P.
- If the intersection of the relaxations is a good approximation to the true problem, then the inequalities will be quite useful.
- Crowder et al. (1983) is the seminal paper that shows this to be true for IP.

Simple Relaxations

- Idea: Consider one row relaxations
- If $P = \{x \in \{0,1\}^n \mid Ax \leq b\}$, then for any row i, $P_i = \{x \in \{0,1\}^n \mid a_i^T x \leq b_i\}$ is a relaxation of P.
- If the intersection of the relaxations is a good approximation to the true problem, then the inequalities will be quite useful.
- Crowder et al. (1983) is the seminal paper that shows this to be true for IP.
- MINLP: Single (linear) row relaxations are also valid $\Rightarrow$ same inequalities can also be used

Knapsack Covers

$$K = \{x \in \{0,1\}^n \mid a^T x \leq b\}$$

- A set $C \subseteq N$ is a cover if $\sum_{j \in C} a_j > b$
- A cover C is a minimal cover if $C \setminus j$ is not a cover $\forall j \in C$

Knapsack Covers

$$K = \{x \in \{0,1\}^n \mid a^T x \leq b\}$$

- A set $C \subseteq N$ is a cover if $\sum_{j \in C} a_j > b$
- A cover C is a minimal cover if $C \setminus j$ is not a cover $\forall j \in C$
- If $C \subseteq N$ is a cover, then the cover inequality

$$\sum_{j \in C} x_j \leq |C| - 1$$

  is a valid inequality for S
- Sometimes (minimal) cover inequalities are facets of $\text{conv}(K)$

Other Substructures

- Single node flow: (Padberg et al., 1985)

$$S = \left\{ x \in \mathbb{R}_+^{|N|}, y \in \{0,1\}^{|N|} \mid \sum_{j \in N} x_j \leq b, x_j \leq u_j y_j \; \forall \; j \in N \right\}$$

- Knapsack with single continuous variable: (Marchand and Wolsey, 1999)

$$S = \left\{ x \in \mathbb{R}_+, y \in \{0,1\}^{|N|} \mid \sum_{j \in N} a_j y_j \leq b + x \right\}$$

- Set Packing: (Borndörfer and Weismantel, 2000)

$$S = \left\{ y \in \{0,1\}^{|N|} \mid Ay \leq e \right\}$$

$A \in \{0,1\}^{|M| \times |N|}, e = (1, 1, \ldots, 1)^T$

The Chvátal-Gomory Procedure

- A general procedure for generating valid inequalities for integer programs

The Chvátal-Gomory Procedure

- A general procedure for generating valid inequalities for integer programs
- Let the columns of $A \in \mathbb{R}^{m \times n}$ be denoted by $\{a_1, a_2, \ldots a_n\}$
- $S = \{y \in \mathbb{Z}^n_+ \mid Ay \leq b\}$.
  1. Choose nonnegative multipliers $u \in \mathbb{R}^m_+$
  2. $u^T Ay \leq u^T b$ is a valid inequality ($\sum_{j \in N} u^T a_j y_j \leq u^T b$).
  3. $\sum_{j \in N} \lfloor u^T a_j \rfloor y_j \leq u^T b$ (Since $y \geq 0$).
  4. $\sum_{j \in N} \lfloor u^T a_j \rfloor y_j \leq \lfloor u^T b \rfloor$ is valid for S since $\lfloor u^T a_j \rfloor y_j$ is an integer

The Chvátal-Gomory Procedure

- A general procedure for generating valid inequalities for integer programs
- Let the columns of $A \in \mathbb{R}^{m \times n}$ be denoted by $\{a_1, a_2, \ldots a_n\}$
- $S = \{y \in \mathbb{Z}^n_+ \mid Ay \leq b\}$.
    1. Choose nonnegative multipliers $u \in \mathbb{R}^m_+$
    2. $u^T Ay \leq u^T b$ is a valid inequality ($\sum_{j \in N} u^T a_j y_j \leq u^T b$).
    3. $\sum_{j \in N} \lfloor u^T a_j \rfloor y_j \leq u^T b$ (Since $y \geq 0$).
    4. $\sum_{j \in N} \lfloor u^T a_j \rfloor y_j \leq \lfloor u^T b \rfloor$ is valid for S since $\lfloor u^T a_j \rfloor y_j$ is an integer
- Simply Amazing: This simple procedure suffices to generate every valid inequality for an integer program

Extension to MINLP (Çezik and Iyengar, 2005)

- This simple idea also extends to mixed 0-1 conic programming

$$\left\{\begin{array}{ll} \underset{z \stackrel{\text{def}}{=} (x,y)}{\text{minimize}} & f^T z \\ \text{subject to} & Az \succeq_{\mathcal{K}} b \\ & y \in \{0,1\}^p, \; 0 \le x \le U \end{array}\right.$$

- $\mathcal{K}$: Homogeneous, self-dual, proper, convex cone
- $x \succeq_{\mathcal{K}} y \Leftrightarrow (x - y) \in \mathcal{K}$

Gomory On Cones (Çezik and Iyengar, 2005)

- LP: $\mathcal{K}_l = \mathbb{R}^n_+$
- SOCP: $\mathcal{K}_q = \{(x_0, \bar{x}) \mid x_0 \geq \|\bar{x}\|\}$
- SDP: $\mathcal{K}_s = \{x = \text{vec}(X) \mid X = X^T, X \text{ p.s.d}\}$

Gomory On Cones (Çezik and Iyengar, 2005)

- LP: $\mathcal{K}_l = \mathbb{R}^n_+$
- SOCP: $\mathcal{K}_q = \{(x_0, \bar{x}) \mid x_0 \geq \|\bar{x}\|\}$
- SDP: $\mathcal{K}_s = \{x = \text{vec}(X) \mid X = X^T, X \text{ p.s.d}\}$
- Dual Cone: $\mathcal{K}^* \stackrel{\text{def}}{=} \{u \mid u^T z \geq 0 \; \forall z \in \mathcal{K}\}$
- Extension is clear from the following equivalence:

$$Az \succeq_{\mathcal{K}} b \quad \Leftrightarrow \quad u^T Az \geq u^T b \; \forall u \succeq_{\mathcal{K}^*} 0$$

Gomory On Cones (Çezik and Iyengar, 2005)

- LP: $\mathcal{K}_l = \mathbb{R}^n_+$
- SOCP: $\mathcal{K}_q = \{(x_0, \bar{x}) \mid x_0 \geq \|\bar{x}\|\}$
- SDP: $\mathcal{K}_s = \{x = \text{vec}(X) \mid X = X^T, X \text{ p.s.d}\}$
- Dual Cone: $\mathcal{K}^* \stackrel{\text{def}}{=} \{u \mid u^T z \geq 0 \ \forall z \in \mathcal{K}\}$
- Extension is clear from the following equivalence:

$$Az \succeq_{\mathcal{K}} b \quad \Leftrightarrow \quad u^T Az \geq u^T b \ \forall u \succeq_{\mathcal{K}^*} 0$$

- Many classes of nonlinear inequalities can be represented as

$$Ax \succeq_{\mathcal{K}_q} b \text{ or } Ax \succeq_{\mathcal{K}_s} b$$

Using Gomory Cuts in MINLP (Akrotirianakis et al., 2001)

- LP/NLP Based Branch-and-Bound solves MILP instances:

$$\begin{cases} \underset{z \stackrel{\text{def}}{=} (x,y),\eta}{\text{minimize}} & \eta \\ \text{subject to} & \eta \geq f_j + \nabla f_j^T (z - z_j) \quad \forall y_j \in Y^k \\ & 0 \geq c_j + \nabla c_j^T (z - z_j) \quad \forall y_j \in Y^k \\ & x \in X, \; y \in Y \text{ integer} \end{cases}$$

Using Gomory Cuts in MINLP (Akrotirianakis et al., 2001)

- LP/NLP Based Branch-and-Bound solves MILP instances:

$$\left\{ \begin{array}{lll} \underset{z \stackrel{\mathrm{def}}{=} (x,y), \eta}{\text{minimize}} & \eta & \\ \text{subject to} & \eta \geq f_j + \nabla f_j^T (z - z_j) & \forall y_j \in Y^k \\ & 0 \geq c_j + \nabla c_j^T (z - z_j) & \forall y_j \in Y^k \\ & x \in X, \; y \in Y \text{ integer} & \end{array} \right.$$

- Create Gomory mixed integer cuts from

$$\begin{array}{rcl} \eta & \geq & f_j + \nabla f_j^T (z - z_j) \\ 0 & \geq & c_j + \nabla c_j^T (z - z_j) \end{array}$$

- Akrotirianakis et al. (2001) shows modest improvements

Using Gomory Cuts in MINLP (Akrotirianakis et al., 2001)

- LP/NLP Based Branch-and-Bound solves MILP instances:

$$\begin{cases} \underset{z \stackrel{\text{def}}{=} (x,y),\eta}{\text{minimize}} & \eta \\ \text{subject to} & \eta \geq f_j + \nabla f_j^T (z - z_j) \quad \forall y_j \in Y^k \\ & 0 \geq c_j + \nabla c_j^T (z - z_j) \quad \forall y_j \in Y^k \\ & x \in X,\ y \in Y \text{ integer} \end{cases}$$

- Create Gomory mixed integer cuts from

$$\begin{aligned} \eta & \geq f_j + \nabla f_j^T (z - z_j) \\ 0 & \geq c_j + \nabla c_j^T (z - z_j) \end{aligned}$$

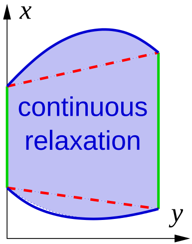
- Akrotirianakis et al. (2001) shows modest improvements
- Research Question: Other cut classes?
- Research Question: Exploit "outer approximation" property?

Disjunctive Cuts for MINLP (Stubbs and Mehrotra, 1999)

Extension of Disjunctive Cuts for MILP: (Balas, 1979; Balas et al., 1993)

Continuous relaxation ($z \stackrel{\text{def}}{=} (x, y)$)

- $C \stackrel{\text{def}}{=} \{z | c(z) \leq 0,\ 0 \leq y \leq 1,\ 0 \leq x \leq U\}$

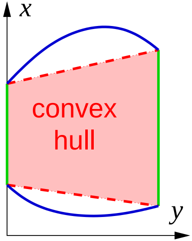


Disjunctive Cuts for MINLP (Stubbs and Mehrotra, 1999)

Extension of Disjunctive Cuts for MILP: (Balas, 1979; Balas et al., 1993)

Continuous relaxation ($z \stackrel{\text{def}}{=} (x, y)$)

- $C \stackrel{\text{def}}{=} \{z | c(z) \leq 0,\ 0 \leq y \leq 1,\ 0 \leq x \leq U\}$
- $\mathcal{C} \stackrel{\text{def}}{=} \text{conv}(\{x \in C \mid y \in \{0,1\}^p\})$

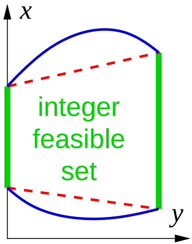


Disjunctive Cuts for MINLP (Stubbs and Mehrotra, 1999)

Extension of Disjunctive Cuts for MILP: (Balas, 1979; Balas et al., 1993)

Continuous relaxation ($z \stackrel{\text{def}}{=} (x, y)$)

- $C \stackrel{\text{def}}{=} \{z | c(z) \leq 0,\ 0 \leq y \leq 1,\ 0 \leq x \leq U\}$
- $\mathcal{C} \stackrel{\text{def}}{=} \text{conv}(\{x \in C \mid y \in \{0,1\}^p\})$
- $C_j^{0/1} \stackrel{\text{def}}{=} \{z \in C | y_j = 0/1\}$



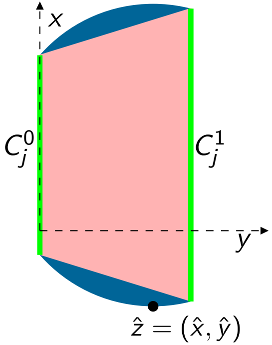
$$\text{let } \mathcal{M}_j(C) \stackrel{\text{def}}{=} \left\{ \begin{array}{l} z = \lambda_0 u_0 + \lambda_1 u_1 \\ \lambda_0 + \lambda_1 = 1,\ \lambda_0, \lambda_1 \geq 0 \\ u_0 \in C_j^0,\ u_1 \in C_j^1 \end{array} \right\}$$

$\Rightarrow \mathcal{P}_j(C) :=$ projection of $\mathcal{M}_j(C)$ onto z

$\Rightarrow \mathcal{P}_j(C) = \text{conv}\,(C \cap y_j \in \{0,1\})$ and $\mathcal{P}_{1\ldots p}(C) = \mathcal{C}$

Disjunctive Cuts: Example

$$\underset{x,y}{\text{minimize}} \left\{ x \mid (x-1/2)^2 + (y-3/4)^2 \leq 1, -2 \leq x \leq 2, y \in \{0,1\} \right\}$$

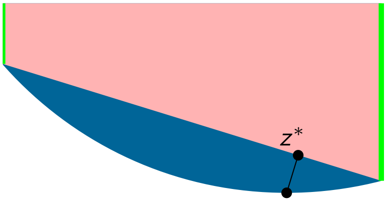


Given $\hat{z}$ with $\hat{y}_j \notin \{0,1\}$ find separating hyperplane

$$\Rightarrow \begin{cases} \underset{z}{\text{minimize}} & \|z - \hat{z}\| \\ \text{subject to} & z \in \mathcal{P}_j(C) \end{cases}$$

Disjunctive Cuts Example

$$z^* \stackrel{\text{def}}{=} \arg\min \|z - \hat{z}\|$$



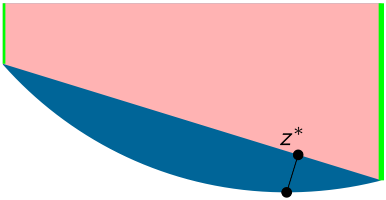
$\hat{z} = (\hat{x}, \hat{y})$

$$\begin{array}{rcl} \text{s.t.}\ \lambda_0 u_0 + \lambda_1 u_1 & = & z \\ \lambda_0 + \lambda_1 & = & 1 \\ \begin{pmatrix} -0.16 \\ 0 \end{pmatrix} \leq u_0 & \leq & \begin{pmatrix} 0.66 \\ 1 \end{pmatrix} \\ \begin{pmatrix} -0.47 \\ 0 \end{pmatrix} \leq u_1 & \leq & \begin{pmatrix} 1.47 \\ 1 \end{pmatrix} \\ \lambda_0, \lambda_1 & \geq & 0 \end{array}$$

Disjunctive Cuts Example

$$z^* \stackrel{\text{def}}{=} \arg\min \|z - \hat{z}\|_2^2$$

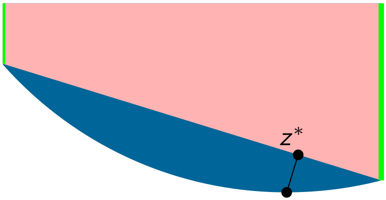
$$\begin{aligned} \text{s.t. } \lambda_0 u_0 + \lambda_1 u_1 &= z \\ \lambda_0 + \lambda_1 &= 1 \\ \begin{pmatrix} -0.16 \\ 0 \end{pmatrix} \leq u_0 &\leq \begin{pmatrix} 0.66 \\ 1 \end{pmatrix} \\ \begin{pmatrix} -0.47 \\ 0 \end{pmatrix} \leq u_1 &\leq \begin{pmatrix} 1.47 \\ 1 \end{pmatrix} \\ \lambda_0, \lambda_1 &\geq 0 \end{aligned}$$



$\hat{z} = (\hat{x}, \hat{y})$

Disjunctive Cuts Example

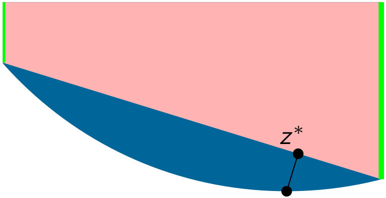
$$z^* \stackrel{\text{def}}{=} \arg\min \|z - \hat{z}\|_\infty$$



$\hat{z} = (\hat{x}, \hat{y})$

$$\begin{aligned}
\text{s.t. } \lambda_0 u_0 + \lambda_1 u_1 &= z \\
\lambda_0 + \lambda_1 &= 1 \\
\begin{pmatrix} -0.16 \\ 0 \end{pmatrix} \leq u_0 &\leq \begin{pmatrix} 0.66 \\ 1 \end{pmatrix} \\
\begin{pmatrix} -0.47 \\ 0 \end{pmatrix} \leq u_1 &\leq \begin{pmatrix} 1.47 \\ 1 \end{pmatrix} \\
\lambda_0, \lambda_1 &\geq 0
\end{aligned}$$

Disjunctive Cuts Example



$\hat{z} = (\hat{x}, \hat{y})$

$$z^* \stackrel{\text{def}}{=} \arg\min \|z - \hat{z}\|$$

$$\begin{aligned} \text{s.t. } \lambda_0 u_0 + \lambda_1 u_1 &= z \\ \lambda_0 + \lambda_1 &= 1 \\ \begin{pmatrix} -0.16 \\ 0 \end{pmatrix} \le u_0 &\le \begin{pmatrix} 0.66 \\ 1 \end{pmatrix} \\ \begin{pmatrix} -0.47 \\ 0 \end{pmatrix} \le u_1 &\le \begin{pmatrix} 1.47 \\ 1 \end{pmatrix} \\ \lambda_0, \lambda_1 &\ge 0 \end{aligned}$$

NONCONVEX

What to do? (Stubbs and Mehrotra, 1999)

- Look at the perspective of $c(z)$

$$\mathcal{P}(c(\tilde{z}), \mu) = \mu c(\tilde{z}/\mu)$$

- Think of $\tilde{z} = \mu z$

What to do? (Stubbs and Mehrotra, 1999)

- Look at the perspective of $c(z)$

$$\mathcal{P}(c(\tilde{z}), \mu) = \mu c(\tilde{z}/\mu)$$

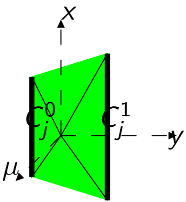
- Think of $\tilde{z} = \mu z$
- Perspective gives a convex reformulation of $\mathcal{M}_j(C)$: $\mathcal{M}_j(\tilde{C})$, where

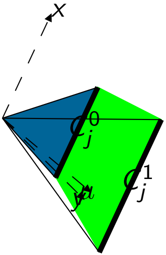
$$\tilde{C} := \left\{ (z, \mu) \left| \begin{array}{l} \mu c_i(z/\mu) \leq 0 \\ 0 \leq \mu \leq 1 \\ 0 \leq x \leq \mu U, \; 0 \leq y \leq \mu \end{array} \right. \right\}$$

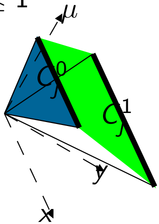
- $c(0/0) = 0 \Rightarrow$ convex representation

Disjunctive Cuts Example

$$\tilde{C} = \left\{ \begin{pmatrix} x \\ y \\ \mu \end{pmatrix} \middle| \begin{array}{c} \mu \left[(x/\mu - 1/2)^2 + (y/\mu - 3/4)^2 - 1 \right] \leq 0 \\ -2\mu \leq x \leq 2\mu \\ 0 \leq y \leq \mu \\ 0 \leq \mu \leq 1 \end{array} \right\}$$







Example, cont.

$$\tilde{C}_j^0 = \{(z, \mu) \mid y_j = 0\} \quad \tilde{C}_j^1 = \{(z, \mu) \mid y_j = \mu\}$$

Example, cont.

$$\tilde{C}_j^0 = \{(z,\mu) \mid y_j = 0\} \quad \tilde{C}_j^1 = \{(z,\mu) \mid y_j = \mu\}$$

- Take $v_0 \leftarrow \mu_0 u_0 \;\; v_1 \leftarrow \mu_1 u_1$

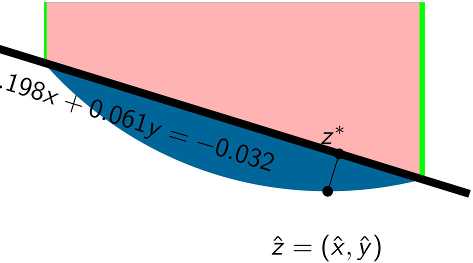
$$\begin{array}{rcl} \min \|z - \hat{z}\| & & \\ \text{s.t.} \;\; v_0 + v_1 & = & z \\ \mu_0 + \mu_1 & = & 1 \\ (v_0, \mu_0) & \in & \tilde{C}_j^0 \\ (v_1, \mu_1) & \in & \tilde{C}_j^1 \\ \mu_0, \mu_1 & \geq & 0 \end{array}$$

Solution to example:

$$\begin{pmatrix} x^* \\ y^* \end{pmatrix} = \begin{pmatrix} -0.401 \\ 0.780 \end{pmatrix}$$

- separating hyperplane: $\psi^T(z - \hat{z})$, where $\psi \in \partial\|z - \hat{z}\|$

Example, Cont.



$$\psi = \begin{pmatrix} 2x^* + 0.5 \\ 2y^* - 0.75 \end{pmatrix}$$

$$0.198x + 0.061y \geq -0.032$$

Nonlinear Branch-and-Cut (Stubbs and Mehrotra, 1999)

- Can do this at all nodes of the branch-and-bound tree
- Generalize disjunctive approach from MILP
  - solve one convex NLP per cut
- Generalizes Sherali and Adams (1990) and Lovász and Schrijver (1991)
  - tighten cuts by adding semi-definite constraint
- Stubbs and Mehrohtra (2002) also show how to generate convex quadratic inequalities, but computational results are not that promising

Generalized Disjunctive Programming (Raman and Grossmann, 1994; Lee and Grossmann, 2000)

Consider disjunctive NLP

$$\begin{cases} \underset{x,Y}{\text{minimize}} & \sum f_i \; + \; f(x) \\ \text{subject to} & \begin{bmatrix} Y_i \\ c_i(x) \leq 0 \\ f_i = \gamma_i \end{bmatrix} \bigvee \begin{bmatrix} \neg Y_i \\ B_i x = 0 \\ f_i = 0 \end{bmatrix} \forall i \in I \\ & 0 \leq x \leq U, \; \Omega(Y) = \text{true}, \; Y \in \{\text{true}, \text{false}\}^p \end{cases}$$

Application: process synthesis

- Y_i represents presence/absence of units
- $B_i x = 0$ eliminates variables if unit absent

Exploit disjunctive structure

- special branching ... OA/GBD algorithms

Generalized Disjunctive Programming (Raman and Grossmann, 1994; Lee and Grossmann, 2000)

Consider disjunctive NLP

$$\begin{cases} \underset{x,Y}{\text{minimize}} & \sum f_i + f(x) \\ \text{subject to} & \begin{bmatrix} Y_i \\ c_i(x) \le 0 \\ f_i = \gamma_i \end{bmatrix} \bigvee \begin{bmatrix} \neg Y_i \\ B_i x = 0 \\ f_i = 0 \end{bmatrix} \forall i \in I \\ & 0 \le x \le U, \ \Omega(Y) = \text{true}, \ Y \in \{\text{true}, \text{false}\}^p \end{cases}$$

Big-M formulation (notoriously bad), $M > 0$:

$$c_i(x) \le M(1 - y_i)$$
$$-My_i \le B_i x \le My_i$$
$$f_i = y_i \gamma_i \qquad \Omega(Y) \text{ converted to linear inequalities}$$

Generalized Disjunctive Programming (Raman and Grossmann, 1994; Lee and Grossmann, 2000)

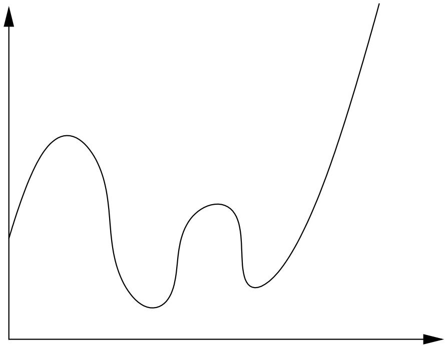
Consider disjunctive NLP

$$\begin{cases} \underset{x,Y}{\text{minimize}} & \sum f_i \;+\; f(x) \\ \text{subject to} & \begin{bmatrix} Y_i \\ c_i(x) \le 0 \\ f_i = \gamma_i \end{bmatrix} \bigvee \begin{bmatrix} \neg Y_i \\ B_i x = 0 \\ f_i = 0 \end{bmatrix} \forall i \in I \\ & 0 \le x \le U, \; \Omega(Y) = \text{true}, \; Y \in \{\text{true}, \text{false}\}^p \end{cases}$$

convex hull representation ...

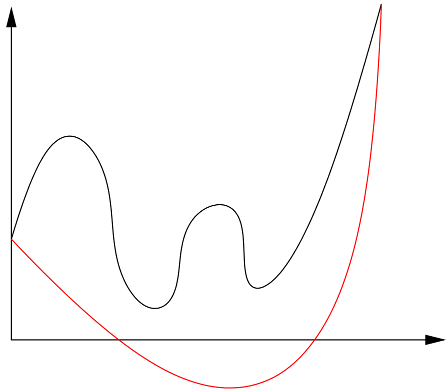
$$\begin{array}{lll} x = v_{i1} + v_{i0}, & \lambda_{i1} + \lambda_{i0} = 1 & \\ \lambda_{i1} c_i(v_{i1}/\lambda_{i1}) \le 0, & B_i v_{i0} = 0 & \\ 0 \le v_{ij} \le \lambda_{ij} U, & 0 \le \lambda_{ij} \le 1, & f_i = \lambda_{i1}\gamma_i \end{array}$$

Dealing with Nonconvexities



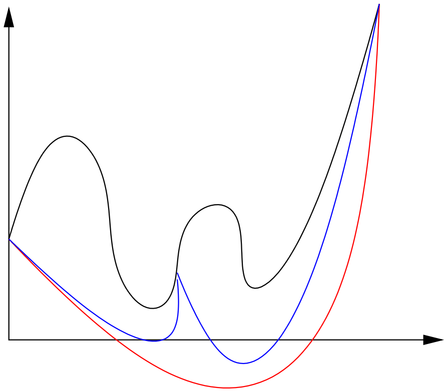
- Functional nonconvexity causes serious problems.
    - Branch and bound must have true lower bound (global solution)

Dealing with Nonconvexities



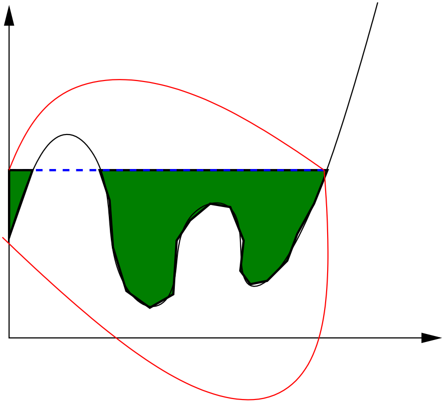
- Functional nonconvexity causes serious problems.
  - Branch and bound must have true lower bound (global solution)
- Underestimate nonconvex functions. Solve relaxation. Provides lower bound.

Dealing with Nonconvexities



- Functional nonconvexity causes serious problems.
    - Branch and bound must have true lower bound (global solution)
- Underestimate nonconvex functions. Solve relaxation. Provides lower bound.
- If relaxation is not exact, then branch

Dealing with Nonconvex Constraints

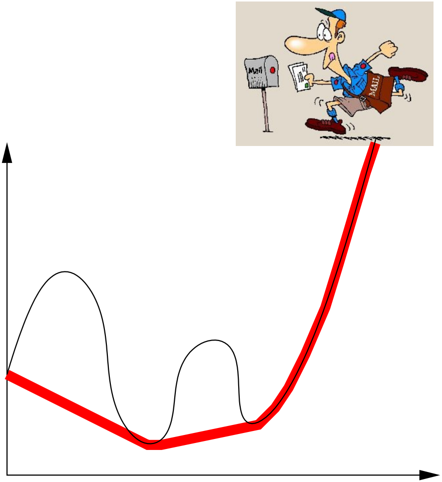


- If nonconvexity in constraints, may need to overestimate and underestimate the function to get a convex region

Envelopes

$$f : \Omega \to \mathbb{R}$$

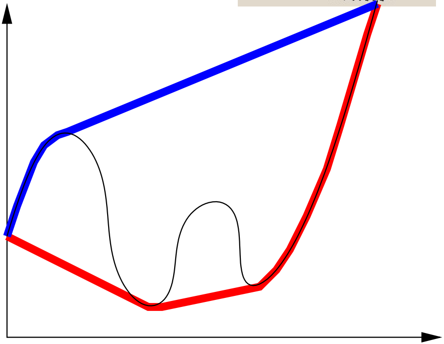
- Convex Envelope ($\text{vex}_{\Omega}(f)$): Pointwise supremum of convex underestimators of f over Ω.



Envelopes

$$f : \Omega \to \mathbb{R}$$

- Convex Envelope ($\text{vex}_\Omega(f)$): Pointwise supremum of convex underestimators of f over Ω.
- Concave Envelope ($\text{cav}_\Omega(f)$): Pointwise infimum of concave overestimators of f over Ω.



Branch-and-Bound Global Optimization Methods

- Under/Overestimate “simple” parts of (Factorable) Functions individually
  - Bilinear Terms
  - Trilinear Terms
  - Fractional Terms
  - Univariate convex/concave terms

Branch-and-Bound Global Optimization Methods

- Under/Overestimate "simple" parts of (Factorable) Functions individually
  - Bilinear Terms
  - Trilinear Terms
  - Fractional Terms
  - Univariate convex/concave terms
- General nonconvex functions $f(x)$ can be underestimated over a region $[l, u]$ "overpowering" the function with a quadratic function that is ≤ 0 on the region of interest

$$\mathcal{L}(x) = f(x) + \sum_{i=1}^{n} \alpha_i (l_i - x_i)(u_i - x_i)$$

Refs: (McCormick, 1976; Adjiman et al., 1998; Tawarmalani and Sahinidis, 2002)

Bilinear Terms

The convex and concave envelopes of the bilinear function xy over a rectangular region

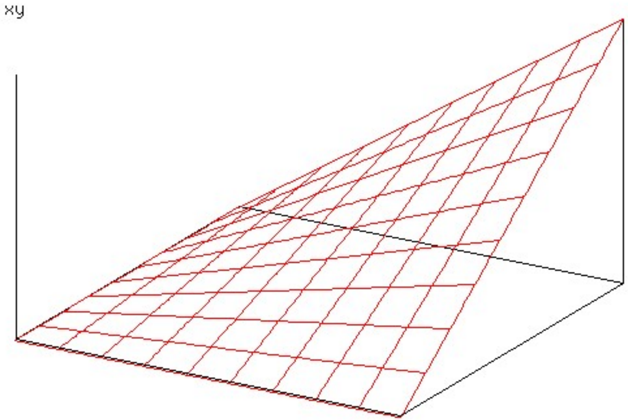
$$R \stackrel{\text{def}}{=} \{(x,y) \in \mathbb{R}^2 \mid l_x \le x \le u_x,\ \ l_y \le y \le u_y\}$$

are given by the expressions

$$\begin{aligned} \text{vexxy}_R(x,y) &= \max\{l_y x + l_x y - l_x l_y,\, u_y x + u_x y - u_x u_y\} \\ \text{cavxy}_R(x,y) &= \min\{u_y x + l_x y - l_x u_y,\, l_y x + u_x y - u_x l_y\} \end{aligned}$$

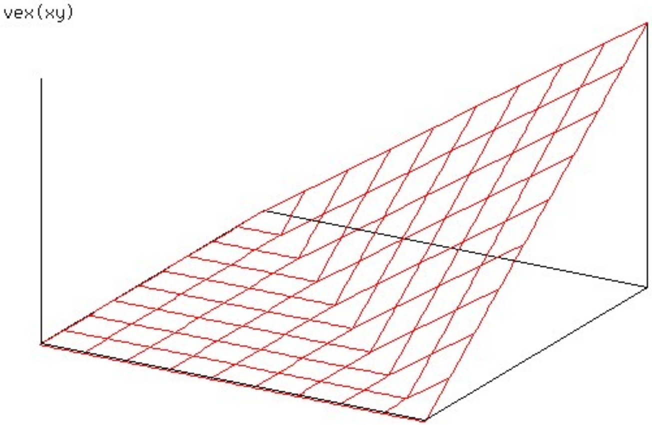
Worth 1000 Words?

xy



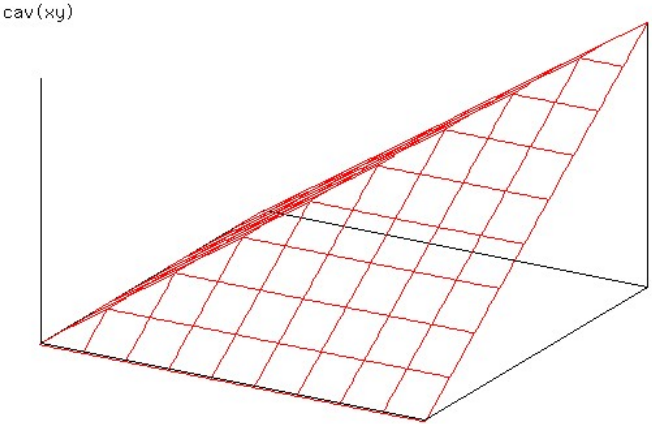
Worth 1000 Words?

$$\text{vex}_R(xy)$$



Worth 1000 Words?

$\text{cav}_R(xy)$



Summary

- MINLP: Good relaxations are important
- Relaxations can be improved
  - Statically: Better formulation/preprocessing
  - Dynamically: Cutting planes
- Nonconvex MINLP:
  - Methods exist, again based on relaxations
- Tight relaxations is an active area of research
- Lots of empirical questions remain

Part IV

Implementation and Software

Implementation and Software for MINLP

1. Special Ordered Sets
2. Implementation & Software Issues

Special Ordered Sets of Type 1

SOS1: $\sum \lambda_i = 1$ & at most one λ_i is nonzero

Example 1: $d \in \{d_1, \dots, d_p\}$ discrete diameters

$\Leftrightarrow d = \sum \lambda_i d_i$ and $\{\lambda_1, \dots, \lambda_p\}$ is SOS1

$\Leftrightarrow d = \sum \lambda_i d_i$ and $\sum \lambda_i = 1$ and $\lambda_i \in \{0, 1\}$

... d is convex combination with coefficients λ_i

Example 2: nonlinear function $c(y)$ of single integer

$\Leftrightarrow y = \sum i \lambda_i$ and $c = \sum c(i) \lambda_i$ and $\{\lambda_1, \dots, \lambda_p\}$ is SOS1

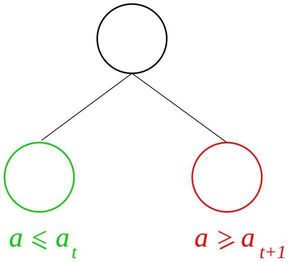
References: (Beale, 1979; Nemhauser, G.L. and Wolsey, L.A., 1988; Williams, 1993) ...

Special Ordered Sets of Type 1

SOS1: $\sum \lambda_i = 1$ & at most one λ_i is nonzero

Branching on SOS1

1. reference row $a_1 < \ldots < a_p$
   e.g. diameters
2. fractionality: $a := \sum a_i \lambda_i$

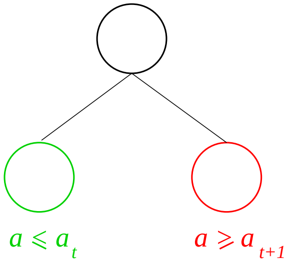


Special Ordered Sets of Type 1

SOS1: $\sum \lambda_i = 1$ & at most one λ_i is nonzero

Branching on SOS1

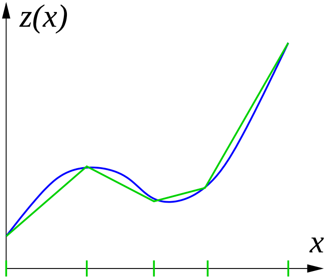
1. reference row $a_1 < \ldots < a_p$
   e.g. diameters
2. fractionality: $a := \sum a_i \lambda_i$
3. find t : $a_t < a \leq a_{t+1}$
4. branch: $\{\lambda_{t+1}, \ldots, \lambda_p\} = 0$
   or $\{\lambda_1, \ldots, \lambda_t\} = 0$



Special Ordered Sets of Type 2

SOS2: $\sum \lambda_i = 1$ & at most two adjacent λ_i nonzero

Example: Approximation of nonlinear function $z = z(x)$

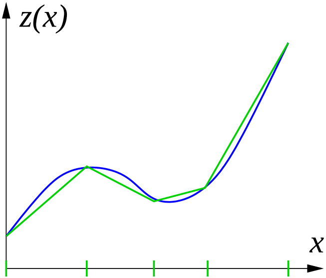


- breakpoints $x_1 < \ldots < x_p$
- function values $z_i = z(x_i)$
- piece-wise linear

Special Ordered Sets of Type 2

SOS2: $\sum \lambda_i = 1$ & at most two adjacent λ_i nonzero

Example: Approximation of nonlinear function $z = z(x)$



- breakpoints $x_1 < \ldots < x_p$
- function values $z_i = z(x_i)$
- piece-wise linear
- $x = \sum \lambda_i x_i$
- $z = \sum \lambda_i z_i$
- $\{\lambda_1, \ldots, \lambda_p\}$ is SOS2

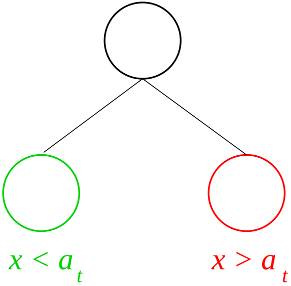
. . . convex combination of two breakpoints . . .

Special Ordered Sets of Type 2

SOS2: $\sum \lambda_i = 1$ & at most two adjacent λ_i nonzero

Branching on SOS2

1. reference row $a_1 < \ldots < a_p$
   e.g. $a_i = x_i$
2. fractionality: $a := \sum a_i \lambda_i$

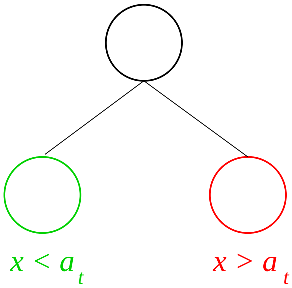


Special Ordered Sets of Type 2

SOS2: $\sum \lambda_i = 1$ & at most two adjacent λ_i nonzero

Branching on SOS2

1. reference row $a_1 < \ldots < a_p$
   e.g. $a_i = x_i$
2. fractionality: $a := \sum a_i \lambda_i$
3. find t : $a_t < a \leq a_{t+1}$
4. branch: $\{\lambda_{t+1}, \ldots, \lambda_p\} = 0$
   or $\{\lambda_1, \ldots, \lambda_{t-1}\}$

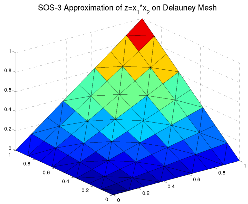


Special Ordered Sets of Type 3

Example: Approximation of 2D function $u = g(v, w)$

Triangularization of $[v_L, v_U] \times [w_L, w_U]$ domain

1. $v_L = v_1 < \ldots < v_k = v_U$
2. $w_L = w_1 < \ldots < w_l = w_U$
3. function $u_{ij} := g(v_i, w_j)$
4. λ_{ij} weight of vertex (i, j)



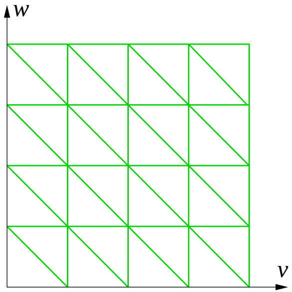
Special Ordered Sets of Type 3

Example: Approximation of 2D function $u = g(v, w)$

Triangularization of $[v_L, v_U] \times [w_L, w_U]$ domain

1. $v_L = v_1 < \ldots < v_k = v_U$
2. $w_L = w_1 < \ldots < w_l = w_U$
3. function $u_{ij} := g(v_i, w_j)$
4. λ_{ij} weight of vertex (i, j)

- $v = \sum \lambda_{ij} v_i$
- $w = \sum \lambda_{ij} w_j$
- $u = \sum \lambda_{ij} u_{ij}$

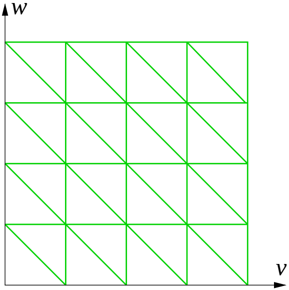


Special Ordered Sets of Type 3

Example: Approximation of 2D function $u = g(v, w)$

Triangularization of $[v_L, v_U] \times [w_L, w_U]$ domain

1. $v_L = v_1 < \ldots < v_k = v_U$
2. $w_L = w_1 < \ldots < w_l = w_U$
3. function $u_{ij} := g(v_i, w_j)$
4. λ_{ij} weight of vertex (i, j)



- $v = \sum \lambda_{ij} v_i$
- $w = \sum \lambda_{ij} w_j$
- $u = \sum \lambda_{ij} u_{ij}$

$1 = \sum \lambda_{ij}$ is SOS3 ...

Special Ordered Sets of Type 3

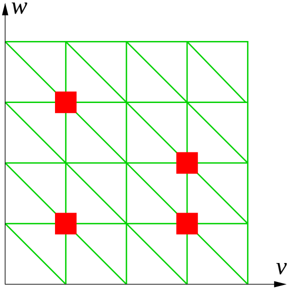
SOS3: $\sum \lambda_{ij} = 1$ & set condition holds

1. $v = \sum \lambda_{ij} v_i$ ... convex combinations
2. $w = \sum \lambda_{ij} w_j$
3. $u = \sum \lambda_{ij} u_{ij}$

$\{\lambda_{11}, \ldots, \lambda_{kl}\}$ satisfies set condition

$\Leftrightarrow\ \exists$ trangle $\Delta : \{(i,j) : \lambda_{ij} > 0\} \subset \Delta$

i.e. nonzeros in single triangle Δ

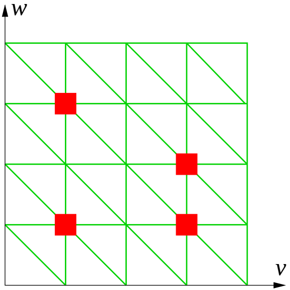


violates set condn

Branching on SOS3

λ violates set condition

- compute centers:
  $\hat{v} = \sum \lambda_{ij} v_i$ &
  $\hat{w} = \sum \lambda_{ij} w_i$
- find s, t such that
  $v_s \leq \hat{v} < v_{s+1}$ &
  $w_s \leq \hat{w} < w_{s+1}$
- branch on v or w

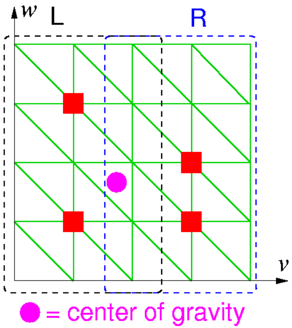


violates set condn

Branching on SOS3

λ violates set condition

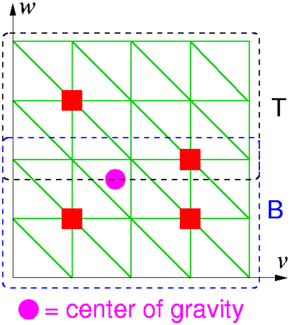
- compute centers: $\hat{v} = \sum \lambda_{ij} v_i$ & $\hat{w} = \sum \lambda_{ij} w_i$
- find s, t such that $v_s \leq \hat{v} < v_{s+1}$ & $w_s \leq \hat{w} < w_{s+1}$
- branch on v or w



vertical branching: $\sum_L \lambda_{ij} = 1$ $\sum_R \lambda_{ij} = 1$

Branching on SOS3

λ violates set condition



- compute centers:
  $\hat{v} = \sum \lambda_{ij} v_i$ &
  $\hat{w} = \sum \lambda_{ij} w_i$
- find s, t such that
  $v_s \leq \hat{v} < v_{s+1}$ &
  $w_s \leq \hat{w} < w_{s+1}$
- branch on v or w

horizontal branching: $\sum_T \lambda_{ij} = 1$ $\sum_B \lambda_{ij} = 1$

Extension to SOS-k

Example: electricity transmission network:

$$c(x) = 4x_1 - x_2^2 - 0.2 \cdot x_2 x_4 \sin(x_3)$$

(Martin et al., 2005) extend SOS3 to SOSk models for any k
$\Rightarrow$ function with p variables on N grid needs N^p λ's

Extension to SOS-k

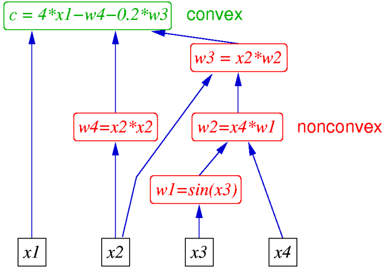
Example: electricity transmission network:

$$c(x) = 4x_1 - x_2^2 - 0.2 \cdot x_2 x_4 \sin(x_3)$$

(Martin et al., 2005) extend SOS3 to SOSk models for any k
$\Rightarrow$ function with p variables on N grid needs N^p λ's

Alternative (Gatzke, 2005):

- exploit computational graph
  $\simeq$ automatic differentiation



Extension to SOS-k

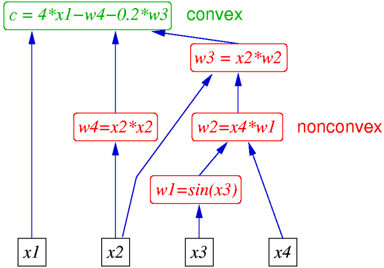
Example: electricity transmission network:

$$c(x) = 4x_1 - x_2^2 - 0.2 \cdot x_2 x_4 \sin(x_3)$$

(Martin et al., 2005) extend SOS3 to SOSk models for any k
$\Rightarrow$ function with p variables on N grid needs N^p λ's

Alternative (Gatzke, 2005):

- exploit computational graph $\simeq$ automatic differentiation
- only need SOS2 & SOS3 … replace nonconvex parts



Extension to SOS-k

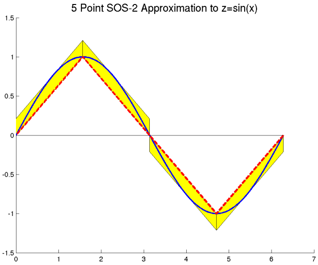
Example: electricity transmission network:

$$c(x) = 4x_1 - x_2^2 - 0.2 \cdot x_2 x_4 \sin(x_3)$$

(Martin et al., 2005) extend SOS3 to SOSk models for any k
$\Rightarrow$ function with p variables on N grid needs N^p λ's

Alternative (Gatzke, 2005):

- exploit computational graph $\simeq$ automatic differentiation
- only need SOS2 & SOS3 ... replace nonconvex parts
- piece-wise polyhedral approx.



Software for MINLP

- **Outer Approximation**: DICOPT++ (& AIMMS)
  NLP solvers: CONOPT, MINOS, SNOPT
  MILP solvers: CPLEX, OSL2
- **Branch-and-Bound Solvers**: SBB & MINLP
  NLP solvers: CONOPT, MINOS, SNOPT & FilterSQP
  variable & node selection; SOS1 & SOS2 support
- **Global MINLP**: BARON & MINOPT underestimators & branching
  CPLEX, MINOS, SNOPT, OSL
- **Online Tools**: MINLP World, MacMINLP & NEOS MINLP World
  `www.gamsworld.org/minlp/`
  NEOS server `www-neos.mcs.anl.gov/`

COIN-OR

http://www.coin-or.org

- COmputational INfrastructure for Operations Research
- A library of (interoperable) software tools for optimization
- A development platform for open source projects in the OR community
- Possibly Relevant Modules:
  - OSI: Open Solver Interface
  - CGL: Cut Generation Library
  - CLP: Coin Linear Programming Toolkit
  - CBC: Coin Branch and Cut
  - IPOPT: Interior Point OPTimizer for NLP
  - NLPAPI: NonLinear Programming API

MINLP with COIN-OR

New implementation of LP/NLP based BB

- MIP branch-and-cut: CBC & CGL
- NLPs: IPOPT interior point ... OK for NLP(y_i)
- New hybrid method:
  - solve more NLPs at non-integer y_i
    $\Rightarrow$ better outer approximation
  - allow complete MIP at some nodes
    $\Rightarrow$ generate new integer assignment

  ... faster than DICOPT++, SBB
- simplifies to OA and BB at extremes ... less efficient

... see Bonami et al. (2005) ... coming in 2006.

Conclusions

MINLP rich modeling paradigm
◦ most popular solver on NEOS

Algorithms for MINLP:
◦ Branch-and-bound (branch-and-cut)
◦ Outer approximation et al.

Conclusions

MINLP rich modeling paradigm
◦ most popular solver on NEOS

Algorithms for MINLP:
◦ Branch-and-bound (branch-and-cut)
◦ Outer approximation et al.

"MINLP solvers lag 15 years behind MIP solvers"

$\Rightarrow$ many research opportunities!!!

Part V

References

C. Adjiman, S. Dallwig, C. A. Floudas, and A. Neumaier. A global optimization method, aBB, for general twice-differentiable constrained NLPs - I. Theoretical advances. **Computers and Chemical Engineering**, 22:1137–1158, 1998.

I. Akrotirianakis, I. Maros, and B. Rustem. An outer approximation based branch-and-cut algorithm for convex 0-1 MINLP problems. **Optimization Methods and Software**, 16: 21–47, 2001.

E. Balas. Disjunctive programming. In **Annals of Discrete Mathematics 5: Discrete Optimization**, pages 3–51. North Holland, 1979.

E. Balas, S. Ceria, and G. Corneujols. A lift-and-project cutting plane algorithm for mixed 0-1 programs. **Mathematical Programming**, 58:295–324, 1993.

E. M. L. Beale. Branch-and-bound methods for mathematical programming systems. **Annals of Discrete Mathematics**, 5:201–219, 1979.

P. Bonami, L. Biegler, A. Conn, G. Cornuéjols, I. Grossmann, C. Laird, J. Lee, A. Lodi, F. Margot, N. Saaya, and A. Wächter. An algorithmic framework for convex mixed integer nonlinear programs. Technical report, IBM Research Division, Thomas J. Watson Research Center, 2005.

B. Borchers and J. E. Mitchell. An improved branch and bound algorithm for Mixed Integer Nonlinear Programming. **Computers and Operations Research**, 21(4):359–367, 1994.

R. Borndörfer and R. Weismantel. Set packing relaxations of some integer programs. **Mathematical Programming**, 88:425 – 450, 2000.

M. T. Çezik and G. Iyengar. Cuts for mixed 0-1 conic programming. **Mathematical Programming**, 2005. to appear.

H. Crowder, E. L. Johnson, and M. W. Padberg. Solving large scale zero-one linear programming problems. **Operations Research**, 31:803–834, 1983.

D. De Wolf and Y. Smeers. The gas transmission problem solved by an extension of the simplex algorithm. **Management Science**, 46:1454–1465, 2000.

M. Duran and I. E. Grossmann. An outer-approximation algorithm for a class of mixed–integer nonlinear programs. **Mathematical Programming**, 36:307–339, 1986.

A. M. Geoffrion. Generalized Benders decomposition. **Journal of Optimization Theory and Applications**, 10:237–260, 1972.

I. E. Grossmann and R. W. H. Sargent. Optimal design of multipurpose batch plants. **Ind. Engng. Chem. Process Des. Dev.**, 18:343–348, 1979.

Harjunkoski, I., Westerlund, T., Pörn, R. and Skrifvars, H. Different transformations for solving non-convex trim-loss problems by MINLP. **European Journal of Opertational Research**, 105:594–603, 1998.

Jain, V. and Grossmann, I.E. Cyclic scheduling of continuous parallel-process units with decaying performance. **AIChE Journal**, 44:1623–1636, 1998.

G. R. Kocis and I. E. Grossmann. Global optimization of nonconvex mixed–integer nonlinear programming (MINLP) problems in process synthesis. **Industrial Engineering Chemistry Research**, 27:1407–1421, 1988.

S. Lee and I. Grossmann. New algorithms for nonlinear disjunctive programming. **Computers and Chemical Engineering**, 24:2125–2141, 2000.

S. Leyffer. Integrating SQP and branch-and-bound for mixed integer nonlinear programming. **Computational Optimization & Applications**, 18:295–309, 2001.

L. Lovász and A. Schrijver. Cones of matrices and setfunctions, and 0-1 optimization. **SIAM Journal on Optimization**, 1, 1991.

H. Marchand and L. Wolsey. The 0-1 knapsack problem with a single continuous variable. **Mathematical Programming**, 85:15–33, 1999.

A. Martin, M. Möller, and S. Moritz. Mixed integer models for the stationary case of gas network optimization. Technical report, Darmstadt University of Technology, 2005.

G. P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems. **Mathematical Programming**, 10:147–175, 1976.

Nemhauser, G.L. and Wolsey, L.A. **Integer and Combinatorial Optimization**. John Wiley, New York, 1988.

M. Padberg, T. J. Van Roy, and L. Wolsey. Valid linear inequalities for fixed charge problems. **Operations Research**, 33:842–861, 1985.

I. Quesada and I. E. Grossmann. An LP/NLP based branch–and–bound algorithm for convex MINLP optimization problems. **Computers and Chemical Engineering**, 16:937–947, 1992.

Quist, A.J. **Application of Mathematical Optimization Techniques to Nuclear Reactor Reload Pattern Design**. PhD thesis, Technische Universiteit Delft, Thomas Stieltjes Institute for Mathematics, The Netherlands, 2000.

R. Raman and I. E. Grossmann. Modeling and computational techniques for logic based integer programming. **Computers and Chemical Engineering**, 18:563–578, 1994.

H. D. Sherali and W. P. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. **SIAM Journal on Discrete Mathematics**, 3:411–430, 1990.

O. Sigmund. A 99 line topology optimization code written in matlab. **Structural Multidisciplinary Optimization**, 21:120–127, 2001.

R. Stubbs and S. Mehrohtra. Generating convex polynomial inequalities for mixed 0-1 programs. **Journal of Global Optimization**, 24:311–332, 2002.

R. A. Stubbs and S. Mehrotra. A branch–and–cut method for 0–1 mixed convex programming. **Mathematical Programming**, 86:515–532, 1999.

M. Tawarmalani and N. V. Sahinidis. **Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software, and Applications**. Kluwer Academic Publishers, Boston MA, 2002.

J. Viswanathan and I. E. Grossmann. Optimal feed location and number of trays for distillation columns with multiple feeds. **I&EC Research**, 32:2942–2949, 1993.

Westerlund, T., Isaksson, J. and Harjunkoski, I. Solving a production optimization problem in the paper industry. Report 95–146–A, Department of Chemical Engineering, Abo Akademi, Abo, Finland, 1995.

Westerlund, T., Pettersson, F. and Grossmann, I.E. Optimization of pump configurations as MINLP problem. **Computers & Chemical Engineering**, 18(9):845–858, 1994.

H. P. Williams. **Model Solving in Mathematical Programming**. John Wiley & Sons Ltd., Chichester, 1993.