Modelling and solution of Nonlinear Programs

Leo Liberti

LIX, École Polytechnique, France
The story so far

- **Mathematical program**: problem model consisting of parameters, variables, objective function, constraints

- **Parameters**: the problem input

- **Variables**: the problem output

- Variables may be continuous (∈ \( \mathbb{R} \)), integer (∈ \( \mathbb{Z} \)) or binary (∈ \{0, 1\}); they may also be bounded (∈ [\( L, U \)])

- **Objective and constraints** are expressed as mathematical functions of parameters and variables

- **Assumption**: objective and constraints are linear forms

- **Modelling software**: AMPL

- **Solution software**: CPLEX

- Many application examples
Nonlinear Programming

- Mathematical methods for modelling and solving nonlinear problems

⇒ NonLinear Programming (NLP)

- **Nonconvex NLPs** (NLPs with at least one nonconvex objective and/or constraint)

- **Mixed-Integer NLPs** (MINLPs — with at least one integer variable)

In practice, it is much more difficult to solve (MI)NLPs than (MI)LPs

- No truly standard software

- In general, no guarantee of optimality for nonconvex MINLPs

- Few successful general-purpose algorithms

  *Can still use AMPL, though*
Nonlinear Modelling

Linear assumption is not always valid

Logical “and” condition:
1. cost associated to conjunctive occurrence of two conditions \( \text{if } x_i \text{ is 1 and } x_j \text{ is 1 then add a cost } c_{ij} \)
2. a constraint is valid iff a certain binary variable has value 1 \( \text{if } y \text{ is 1 then } g(x) \leq 0 \)

Percentages and quantities: variables expressing percentage and variables expressing quantity must be multiplied together

Economies of scale: unit costs decrease with quantity

Problems involving 1-, 2- and \( \infty \)-norms

Nonlinear models of natural phenomena expressed in constraints
Canonical MINLP formulation

\[
\begin{align*}
\min_x & \quad f(x) \\
\text{s.t.} & \quad l \leq g(x) \leq u \\
& \quad x^L \leq x \leq x^U \\
\forall i & \in Z \subseteq \{1, \ldots, n\} \quad x_i \in \mathbb{Z}
\end{align*}
\]

\[ [P] \quad (1) \]

where \( x, x^L, x^U \in \mathbb{R}^n; l, u \in \mathbb{R}^m; f: \mathbb{R}^n \to \mathbb{R}; g: \mathbb{R}^n \to \mathbb{R}^m \)

- \( F(P) = \) feasible region of \( P \), \( L(P) = \) set of local optima, \( G(P) = \) set of global optima
- Nonconvexity \( \Rightarrow G(P) \subsetneq L(P) \)

\[
\min_{x \in [-3,6]} \frac{1}{4}x + \sin(x)
\]
Reformulations

Defn.

Given a formulation $P$ and a formulation $Q$, $Q$ is a reformulation of $P$ if there is a mapping $\varphi : F(Q) \rightarrow F(P)$ such that $\varphi(L(Q)) = L(P)$ and $\varphi(G(Q)) = G(P)$

This means: $\varphi$ restricted to $L(Q)$ is onto $L(P)$ and $\varphi$ restricted to $G(Q)$ is onto $G(P)$

- Reformulations are used to transform problems into equivalent forms
- “Equivalence” here means a precise correspondence between local and global optima via the same transformation

Basic reformulation operations:

1. adding / deleting variables / constraints
2. replacing a term with another term (e.g. a product $xy$ with a new variable $w$)
Product of binary variables

Consider binary variables $x, y$ and a cost $c$ to be added to the objective function only of $xy = 1$

⇒ Add term $cxy$ to objective

Problem becomes mixed-integer (some variables are binary) and nonlinear

Reformulate “$xy$” to MILP form ($\text{PRODBIN}$ reform.):

- replace $xy$ by $z$
- add $z \leq y$, $z \leq x$
- $z \geq 0$, $z \geq x + y - 1$
- $x, y \in \{0, 1\} \Rightarrow z = xy$
Product of bin. and cont. vars.

**PRODBinCont reformulation**

Consider a binary variable \( x \) and a continuous variable \( y \in [y^L, y^U] \), and assume product \( xy \) is in the problem.

Replace \( xy \) by an added variable \( w \).

Add constraints:

\[
\begin{align*}
  w &\leq y^U x \\
  w &\geq y^L x \\
  w &\leq y + y^L (1 - x) \\
  w &\geq y - y^U (1 - x)
\end{align*}
\]

**Exercise 1**: show that \( \text{PRODBinCont} \) is indeed a reformulation

**Exercise 2**: show that if \( y \in \{0, 1\} \) then \( \text{PRODBinCont} \) is equivalent to \( \text{PRODBin} \).
Product of continuous variables

Suppose a flow is composed by \( m \) different materials

Let \( x_i \in [0, 1] \) indicate the unknown fraction of material \( i \leq m \) in the flow

Let \( y \) be the unknown total flow

Get terms \( x_i y \) in the problem to indicate the amount of each material \( i \leq m \) in the flow

Constraint \( \sum_{i \leq m} x_i = 1 \): all fractions sum up to 1

\[ \Rightarrow \text{Nonconvex NLP} \]

No exact linear reformulation possible, but can be approximated by discretization

Best way to solve it directly is by dedicated algorithm (e.g. SLP or SQP)
Prod. cont. vars.: approximation

- **BILINAPPROX** approximation
- Consider $x \in [x^L, x^U], y \in [y^L, y^U]$ and product $xy$
- Suppose $x^U - x^L \leq y^U - y^L$, consider an integer $d > 0$
- Replace $[x^L, x^U]$ by a finite set
  
  $D = \{x^L + (i - 1)\gamma \mid 1 \leq i \leq d\}$, where $\gamma = \frac{x^U - x^L}{d-1}$
BILINAPPROX

- Replace the product $xy$ by a variable $w$
- Add binary variables $z_i$ for $i \leq d$
- Add assignment constraint for $z_i$'s

$$\sum_{i \leq d} z_i = 1$$

- Add definition constraint for $x$:

$$x = \sum_{i \leq d} (x^L + (i - 1)\gamma)z_i$$

($x$ takes exactly one value in $D$)

- Add definition constraint for $w$

$$w = \sum_{i \leq d} (x^L + (i - 1)\gamma)z_i y$$

(2)

- Reformulate the products $z_i y$ via PROD_BINCONT
Conditional constraints

- Suppose $\exists$ a binary variable $y$ and a constraint $g(x) \leq 0$ in the problem
- We want $g(x) \leq 0$ to be active iff $y = 1$
- Compute maximum value that $g(x)$ can take over all $x$, call this $M$
- Write the constraint as:
  \[ g(x) \leq M(1 - y) \]
- This sometimes called the “big $M$” modelling technique

Example:

Can replace constraint (2) in BILINAPPROX as follows:

\[ \forall i \leq d \quad -M(1 - z_i) \leq w - (x^L + (i - 1)\gamma)y \leq M(1 - z_i) \]

where $M$ s.t. $w - (x^L + (i - 1)\gamma)y \in [-M, M]$ for all $w, x, y$
Graph Partitioning Problem I

**GPP**: Given an undirected graph $G = (V, E)$ and an integer $k \leq |V|$, find a partition of $V$ in $k$ disjoint subsets $V_1, \ldots, V_k$ (called clusters) of minimal given cardinality $M$ s.t. the number (weight) of edges with adjacent vertices in different clusters is minimized.

### Applications:
- telecom network planning
- sparse matrix factorization
- parallel computing
- VLSI circuit placement

### Minimal bibliography:
- Battiti & Bertossi, *IEEE Trans. Comp.*, 1999 (heuristics);
- Boulle, *Opt. Eng.*, 2004 (formulations);
- Liberti *4OR*, 2007 (reformulations)
Graph Partitioning Problem II

- For all vertices $i \in V$, $h \leq k$:
  
  $x_{ih} = 1$ if vertex $i$ in cluster $h$ and 0 otherwise

- **Objective function:** $\min \frac{1}{2} \sum_{h \neq l \leq k} \sum_{\{i,j\} \in E} x_{ih} x_{jl}$

- **Assignment:** $\forall i \in V \sum_{h \leq k} x_{ih} = 1$

- **Cluster cardinality:** $\forall h \leq k \sum_{i \in V} x_{ih} \leq M$

- **nonconvex BQP:** reformulate or linearize to MILP, then solve with CPLEX
Pooling and blending I

Given an oil routing network with pools and blenders, unit prices, demands and quality requirements:

Find the input quantities minimizing the costs and satisfying the constraints: mass balance, sulphur balance, quantity and quality demands.
Pooling and blending II

- Variables: input quantities $x$, routed quantities $y$, percentage $p$ of sulphur in pool
- Bilinear terms arise to express sulphur quantities in terms of $p, y$
- Sulphur balance constraint: $3x_{11} + x_{21} = p(y_{11} + y_{12})$
- Quality demands:
  \[ py_{11} + 2y_{21} \leq 2.5(y_{11} + y_{21}) \]
  \[ py_{12} + 2y_{22} \leq 1.5(y_{12} + y_{22}) \]
- Continuous bilinear formulation $\Rightarrow$ nonconvex NLP
Haverly’s pooling problem

\[
\begin{align*}
\min_{x,y,p} & \quad 6x_{11} + 16x_{21} + 10x_{12} - 9(y_{11} + y_{21}) - 15(y_{12} + y_{22}) \\
\text{s.t.} & \quad x_{11} + x_{21} - y_{11} - y_{12} = 0 \\
& \quad x_{12} - y_{21} - y_{22} = 0 \\
& \quad y_{11} + y_{21} \leq 100 \\
& \quad y_{12} + y_{22} \leq 200 \\
& \quad 3x_{11} + x_{21} - p(y_{11} + y_{12}) = 0 \\
& \quad py_{11} + 2y_{21} \leq 2.5(y_{11} + y_{21}) \\
& \quad py_{12} + 2y_{22} \leq 1.5(y_{12} + y_{22})
\end{align*}
\]
Successive Linear Programming

- Heuristic for solving bilinear programming problems
- Formulation includes bilinear terms $x_i y_j$ where $i \in I, j \in J$
- Problem is nonconvex $\Rightarrow$ many local optima
- Fact: fix $x_i, i \in I$, get LP$_1$; fix $y_j, j \in J$, get LP$_2$
- Algorithm: solve LP$_1$, get values for $y$, update and solve LP$_2$, get values for $x$, update and solve LP$_1$, and so on
- Iterate until no more improvement
- **Warning**: no convergence may be attained, and no guarantee to obtain global optimum
SLP applied to HPP

Problem LP₁: fixing $p$

$$\begin{align*}
\min_{x,y} & \quad 6x_{11} + 16x_{21} + 10x_{12} - 9y_{11} - 9y_{21} - 15y_{12} - 15y_{22} \\
\text{s.t.} & \quad x_{11} + x_{21} - y_{11} - y_{12} = 0 \\
& \quad x_{12} - y_{21} - y_{22} = 0 \\
& \quad y_{11} + y_{21} \leq 100 \\
& \quad y_{12} + y_{22} \leq 200 \\
& \quad 3x_{11} + x_{21} - py_{11} - py_{12} = 0 \\
& \quad (p - 2.5)y_{11} - 0.5y_{21} \leq 0 \\
& \quad (p - 1.5)y_{12} + 0.5y_{22} \leq 0
\end{align*}$$

SLP Algorithm:

1. Solve LP₁, find value for $y_{11}, y_{12}$, update LP₂
2. Solve LP₂, find value for $p$, update LP₁
3. Repeat until solution does not change / iteration limit exceeded

Problem LP₂: fixing $y_{11}, y_{12}$

$$\begin{align*}
\min_{x,y_{21},y_{22},p} & \quad 6x_{11} + 16x_{21} + 10x_{12} - (9(y_{11} + y_{21}) + 15(y_{12} + y_{22})) \\
\text{s.t.} & \quad x_{11} + x_{21} = y_{11} + y_{12} \\
& \quad x_{12} - y_{21} - y_{22} = 0 \\
& \quad y_{21} \leq 100 - y_{11} \\
& \quad y_{22} \leq 200 - y_{12} \\
& \quad 3x_{11} + x_{21} - (y_{11} + y_{12})p = 0 \\
& \quad y_{11}p - 0.5y_{21} \leq 2.5y_{11} \\
& \quad y_{12}p + 0.5y_{22} \leq 1.5y_{12}
\end{align*}$$
Problem proposed by Newton

Determine maximum number $K$ of non-overlapping balls of radius 1 adjacent to a central ball of radius 1 in $\mathbb{R}^D$

- In $\mathbb{R}^2$: $K = 6$
- In $\mathbb{R}^3$: $K = 12$ (13 spheres prob.)
- In $\mathbb{R}^4$: $K = 24$ (recent result)
- Next open case: $D = 5$ ($40 \leq K \leq 45$)
Reduce to a decision problem (can \( N \) spheres be arranged in a kissing configuration?)

- Variables: let \( x^i \in \mathbb{R}^D \) be the center of the \( i \)-th ball

- Continuous quadratic formulation:

\[
\begin{align*}
\text{max} & \quad \alpha \\
\forall i \leq N & \quad ||x^i||^2 = 4 \\
\forall i < j \leq N & \quad ||x^i - x^j||^2 \geq 4\alpha \\
\alpha & \geq 0 \\
\forall i \leq N & \quad x^i \in \mathbb{R}^D,
\end{align*}
\]

- If global optimum has \( \alpha \geq 1 \), then \( N \) balls can be arranged, otherwise they cannot

- [Kucherenko et al., DAM 2007]
Consider the time-independent non-relativistic Schrödinger equation $H_{el} \Psi = E_{el} \Psi$ for the electrons in a molecule.

Solution to Schrödinger equation are products of $n$ molecular orbitals $\psi_i$.

Each $\psi_i$ is composed of a spatial orbital $\varphi_i$ and a spin orbital $\vartheta_i$.

Spatial orbitals approximated by suitable bases $\{\chi_s\}_{s=1}^b$:

$$\varphi_i = \sum_{s=1}^b c_{si} \chi_s \quad \forall i \leq n$$

where $\varphi_i$ is the approximation of $\varphi_i$. 
The Hartree-Fock problem II

- Given \( b \) and \( \{\chi_s\}_{s=1}^{b} \), determine the coefficients \( c_{si} \) such that the approximation is “best”
- Approximation is “best” when the energy \( E(c) \) (quartic polynomial in \( c \)) of approximated spatial orbitals \( \varphi_i \) is minimum
- Orthogonality constraints on \( \varphi_i \) (to enforce lin. ind.)
- Coefficients \( c \) vary over a known range \( c^L \leq c \leq c^U \)
- Continuous quartic formulation:

\[
\begin{align*}
\min_c & \quad E(c) \\
\text{s.t.} & \quad \langle \varphi_i | \varphi_j \rangle = \delta_{ij} \quad \forall i \leq j \leq n \\
& \quad c^L \leq c \leq c^U
\end{align*}
\]

- [Lavor et al., EPL 2007]
Molecular Distance Geometry

Known set of atoms $V$, determine 3D structure

Some inter-atomic distances $d_{ij}$ known (NMR)

Find atomic positions $x^i \in \mathbb{R}^3$ which preserve distances

$\Rightarrow$ given weighted graph $G = (V, E, d)$, find immersion in $\mathbb{R}^3$

Continuous quartic formulation:

$$\min_{x} \sum_{\{i,j\} \in E} (||x^i - x^j||^2 - d_{ij}^2)^2$$

[1] Lavor et al. 2006
Scheduling with delays I

- **T**: tasks of length $L_i$ with precedences given by DAG $G = (V, A, c)$, where $c_{ij}$ = amount of data passed from $i$ to $j$

- **P**: homogeneous processors with distance $d_{kl}$ between processors $k, l$ in architecture

- Delays $\gamma_{ij}^{kl}$ occur if dependent tasks $i, j$ are executed on different processors $k, l$
Scheduling with delays II

- Idea: pack $L_j \times 1$ “task rectangles” into a $T_{\text{max}} \times |P|$ “total time” rectangle
- Use binary assignment variables $z_{jk} = 1$ if task $j \in T$ is executed on processor $k \in P$
- Use continuous scheduling variables $t_j =$ starting time of task $j$
- Model communication delays with quadratic constraints:
  \[
  t_j \geq t_i + L_i + \sum_{k,l \in P} \gamma_{ij}^k z_{ik} z_{jl} \quad \forall j \in V, i : (i, j) \in A
  \]
- Mixed-integer quadratic formulation
- [Davidović et al., MISTA Proc. 2007]
Variable Neighbourhood Search

- Applicable to discrete and continuous problems
- Uses any local search as a black-box
- In its basic form, easy to implement
- Few configurable parameters
- Structure of the problem dealt with by local search
- Few lines of code around LS black-box
VNS algorithm I

- Random 1
  - Local search 1
    - Local minimum 1,2
      - Random 2
        - Local search 2
        - Random 3
          - K = 1
          - K = 2
    - K = Kmax
- K = 1
  - Local minimum 3
VNS algorithm II

Input: max no. $k_{\text{max}}$ of neighbourhoods

loop

Set $k \leftarrow 1$, pick random point $\tilde{x}$, perform a local search to find a local minimum $x^*$.

while $k \leq k_{\text{max}}$ do

Let $N_k(x^*)$ neigh. of $x^*$ s.t. $N_k(x^*) \supset N_{k-1}(x^*)$

Sample a random point $\tilde{x}$ from $N_k(x^*)$

Perform a local search from $\tilde{x}$ to find a local minimum $x'$

If $x'$ is better than $x^*$, set $x^* \leftarrow x'$ and $k \leftarrow 0$

Set $k \leftarrow k + 1$

Verify termination condition; if true, exit

end while

end loop
Neighbourhoods in continuous space

- Use hyper-rectangular neighbourhoods $N_k(x')$ proportional to the region delimited by the variable ranges.
- May also employ hyper-rectangular “shells” of size $k/k_{\text{max}}$ of the original domain.