Implementing a Global Solver in a General Purpose Callable Library

by Tony Gau
Linus Schrage
LINDO Systems

http://www.lindo.com

at Argonne Global Optimization Theory Institute

10 Sept 2003
LINDO API library is an LP, NLP, IP solver used by LINGO and What’sBest spreadsheet add-in.

LINDO API contains a global solver that finds a guaranteed (disclaimer: assuming infinite precision) global optimum to an arbitrary optimization problem;

Fully supports all common math functions:

\[ x \times y, \frac{x}{y}, x^y, \log(x), \exp(x), \sqrt{x}, \sin(x), \cos(x), \tan(x), \]

\[ \text{floor}(x), \]

\[ \text{abs}(x), \max(x,y), \min(x,y), \]

\[ \text{if}(x,y,z), \ AND, \ OR, \ [\text{where } x \text{ is a logical expression}] \]

\[ \text{psn}(z), \text{psl}(z) \ [\text{Normal distribution}] \]
Global Solver in LINDO API: Outline

1) Getting a good solution quickly, multistart and other ideas;
2) Guaranteed solutions: a) convex relaxation, b) split/branch;
3) Constraint propagation, bound tightening, interval arithmetic;
4) Constructing convex relaxations for wide range of functions:
   - continuous and smooth: \( x+y, x-y, x*y, \sin(x), \cos(x) \), etc.
   - continuous, nonsmooth: \( \text{abs}(x), \max(x,y), \min(x,y) \),
   - smooth not quite continuous, \( x/y, x^y, \tan(x), \text{floor}(x) \),
   - logical functions: \( \text{if}( ), \text{and}, \text{or}, \text{not} \geq, \leq, = =, \neq \),
   - application specific functions: Normal cdf & linear loss function;
5) Using linearization + linear MIP only for functions such as:
   - \( \text{abs}( ), \min( ), \text{if}( ) \), special cases of \( x*y \);
6) Choosing an algebraic representation, reformulation,
   - e.g., \( x*(y-x) \) vs. \( x*y - x^2 \);
7) Choosing a machine representation with some vector functions,
8) Choosing a good branching;
9) Numerical stability issues in cut management, branch selection;
10) Computational testing.

Sahinidis (1996): first general implementation of Relax, and Branch-if-necessary.


Ugray, Lasdon, et. al. (2002) Multi-start to find good solution.

Gau: Implementation in LINDO API

Atlihan: Multi-start in LINDO API
Getting a Good Initial Solution, Multi-start

Why?  
   a) User wants a good solution quickly, 
   b) Do not waste time adding cuts far from optimum, 
   c) B&B has minimum number of nodes.

Basic Reference for multi-start: Ugray, Lasdon et. al.

For $i = 1$ to $n_{trials}$:

   Randomly select a point, $s_i$, in $n$-space so that it is not in the neighborhood of any of preceding points.

   Call conventional hill-climbing solver with point $s_i$ as initial solution, giving a final solution $f_i$.

   If solution $f_i$ is best yet, store it.

   Set the neighborhood of point $f_i$ big enough to include $s_i$. 
_General Global Optimization Methodology_

Uses the branch and bound approach popularized by McCormick, Sahinidis.

Two ideas:

1) For each (arbitrary) nonlinear function, given current bounds on variables, automatically generate a convex relaxation of the function. Solve the relaxed convexified model.

2) If solution to the relaxed problem is not feasible to the original model, then branch, i.e., partition the feasible region into two subregions. Calculate new implied bounds on the variables for each subproblem. Go back to (1).
Why? Relaxations are tighter if bounds on variables are tighter.

Example for operators + and -:

Round 0: Given:
\[ 2x - y \geq 3; \quad -x + 2y \geq 3; \quad x, y \geq 0; \]

Round 1: Implies:
\[ x \geq (3+0)/2 = 1.5; \quad y \geq (3+0)/2 = 1.5; \]

Round 2:
\[ x \geq (3+1.5)/2 = 2.25; \quad y \geq (3+1.5)/2 = 2.25; \]

etc.

Need rules for stopping,

generalize for every operator supported.
Creating a Convex Relaxation/Bound

Example: \( \text{Min} = \sin(x) + 0.5\cdot\text{abs}(x-9.5); \)
\[ s.t. \quad 0 \leq x \leq 12; \]

A Nonconvex Function: \( \sin(x) + 0.5\cdot\text{abs}(x-9.5) \)
We replace $\sin(\cdot)$ by its convex bound. Solve, get $x = 9.5$. 
Branching

We branch on $x \leq 9.5$ vs. $x \geq 9.5$ and re-bound.

The branch $x \geq 9.5$ is convex with solution $x = 10.47197$.

Bound on $\sin(x)$ with branch at $x = 9.5$

Bound discards $x \leq 9.5$ branch, and we are done.
Some functions can be recognized and linearized exactly.

Let \( \delta \) be a 0/1 variable. \( M \) = a big number.

Given:

a) \( r = \max(x, y) \);

 Linearization:

\[
\begin{align*}
    r & \geq x; \quad r \geq y; \quad r \leq x + \delta M; \quad r \leq y + (1 - \delta)M;
\end{align*}
\]

b) \( r = \max(x, -x) \);

c) \( r = \min(x, y) = - \max(-x, -y) \);
d) $r = \text{IF}(\delta, x, y)$;

\[ x - (1 - \delta) M \leq r \leq x + (1 - \delta) M; \]
\[ y - \delta M \leq r \leq y + \delta M; \]

\[ r \leq \delta M; \]

e) $r = \delta y$;

\[ y - (1 - \delta) M \leq r \leq y + (1 - \delta) M; \]
\[ r \leq \delta M; \]

f) $xy = 0; \ (\text{Complementarity})$

\[-(1 - \delta) M \leq x \leq (1 - \delta) M; \]
\[-\delta M \leq y \leq \delta M; \]
A small text book example:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EOQ Inventory with Quantity Discount</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>All Units Case, C and M, Chapter 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Parameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>120000 = D = demand/year</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>100 = K = setup cost</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.2 = i = interest charge</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Discount schedule</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Breakpoint</td>
<td>Cost/unit at or above this level</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>5000</td>
<td>2.96</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>10000</td>
<td>2.92</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>100000 = Q = amount to order</td>
<td>$354,520.00 = (K<em>D/Q)+(i</em>Q/2+D)*IF(Q&lt;A10,B9,IF(Q&lt;A11,B10,B11))</td>
<td></td>
</tr>
</tbody>
</table>
IF( , , ) is convenient for representing quantity discount price schedules, using nested IF’s.

A customer example:

- 7 discount levels,
- 13 suppliers,
- 361 SKU’s

Resulted in model with

- 4646 rows and 7790 variables.
Choosing an Algebraic Representation/Reformulation

1) $x \times x$ is converted to $x^2$ to get tighter convex relaxation;

2) More generally: $f_1(x \times y) \geq 0; \quad f_2(y \times x) \geq 0$;

   is converted to: $f_1(w) \geq 0; \quad f_2(w) \geq 0; \quad w = x \times y$;

3) $x \times (y-x)$ vs. $x \times y - x^2$;

   One may be better for tight intervals, the other for a tight relaxation.
“Careful”, though not rigorous rounding is used in LINGO/LINDO API.

Example: Arnold Neumaier’s problem, may be difficult to solve accurately for some solvers. LINGO solves to optimality in 0 secs.

\[
\begin{align*}
n & = 20; \\
\text{min} & = -x(n);
\end{align*}
\]

\[
\begin{align*}
(s+1)x(1) - x(2) & \geq s-1; \\
-sx(n-2) - (3s-1)x(n-1) + 3x(n) & \geq -(5s-7);
\end{align*}
\]

\[
\begin{align*}
@for(\text{point}(i)| i \geq 1 \text{ and} i \lt n: \\
-sx(i-1) + (s+1)x(i) - x(i+1) & \geq ((-1)^i)(s+1)
\end{align*}
\]

\[
\begin{align*}
@for(\text{point}(i)| i \leq 13: \\
& @bnd(0, x(i), 10)
\end{align*}
\]

\[
\begin{align*}
@for(\text{point}(i)| i \gt 13: \\
& @bnd(0, x(i), 1000000)
\end{align*}
\]

\[
\begin{align*}
@for(\text{point}(i): \\
& @gin(x(i))
\end{align*}
\]
Some solvers have difficulty finding a correct solution to this problem with 6 variables and 1 constraint;

! (bigsum01) Obj = -540564, LINGO time = .2 secs.;
MIN = - 81 * X_1 - 221 * X_2 - 219 * X_3
    - 317 * X_4 - 385 * X_5 - 413 * X_6;

12228 * X_1 + 36679 * X_2 + 36682 * X_3
+ 48908 * X_4 + 61139 * X_5 + 73365 * X_6 = 89716837;

@GIN( X_1); @GIN( X_2); @GIN( X_3);
@GIN( X_4); @GIN( X_5); @GIN( X_6);
@BND( 0, X_1, 99999); @BND( 0, X_2, 99999);
@BND( 0, X_3, 99999); @BND( 0, X_4, 99999);
@BND( 0, X_5, 99999); @BND( 0, X_6, 99999);
How to Input Nonlinear Programs?

A) Through a file:
   1) LINGO Script:
      Execute runlingo scriptfile.lng
   2) Low level RPN notation:
      Execute runlindo modelfile.mpi.

B) Through memory:
   1) LINGO Script:
      \[ nError = \text{LSexecuteScriptLng}(\ pLINGO, \ cScript) ; \]
   2) Low level RPN notation:
      \[ nError = \text{LSloadInstruct}(\ pModel,...,codelist,...) ; \]
What Does an RPN Codelist(.mpi) Look Like?

! minimize = x*sin(x*pi) + 10
! subject to
! x - 10 <= 0;
BEGINMODEL XSINXPI
VARIABLES
  X0001 8.0 0.0 10.0 C
OBJECTIVES
  XSINXPI LS_MIN
    EP_PUSH_VAR X0001
    EP_PUSH_NUM 3.1415926
    EP_MULTIPLY
    EP_SIN
    EP_PUSH_VAR X0001
    EP_MULTIPLY
    EP_PUSH_NUM 10.0
    EP_PLUS
CONSTRAINTS
  ROW1 L
    EP_PUSH_VAR X0001
    EP_PUSH_NUM 10.0
    EP_MINUS
ENDMODEL
A suite of 60 continuous NLPs arising in different applications

- Nonlinear Least Squares Regression
- Inventory Management and Network Flows
- Chemical Processes
- Engineering Design (constrained polynomials etc...)

NLP Model Sizes

- (Min - Max) Constraints: (0 - 576)
- (Min - Max) Variables: (1 - 518)
Performances on Continuous NLPs (Cont.)

- Server Specs (P4, 1.4 GHz, 2G RAM, NT4)
- Seconds required to solve the entire suite
  - Global solver: 1789 secs
  - Multi-Start solver: 333 secs
  - Single-Start solver: 11 secs
- Proving global optimality takes more time.
- Multi-starts help finding improved solutions
- Single-start is the fastest but solution quality is compromised.
The Global Solver
found provably optimal solutions for 58 problems
proved infeasibility for 2 problems

The Multi-Start Solver (with 5 multi-starts)
obtained the global optima in 39 out of 58 problems.
failed to find a feasible solution in 4 out of 58 problems.
found better solution than single-start in 11 out of 19 problems.

Single-Start Solver
obtained the global solution in 30 out of 58 problems.
failed to find a feasible solution in 5 out of 58.
Performances on Mixed-Integer NLPs

A suite of 50 NLPs with integer variables.

Model Sizes
- (Min-Max) Constraints: (1 - 113)
- (Min-Max) Variables: (1 - 131)

Global Solver
- found provably global optima for 50 out of 50 problems. (Total time: 2475 secs.)

Multi-Start Solver
- performed 2 multi-starts at every node in B&B tree.
- obtained global optima in 42 out of 50 problems (Total time: 404 secs).
- no feasible solutions for 3 out of 50 problems.
### Some Recent Example Problems:

<table>
<thead>
<tr>
<th>Problem</th>
<th>Constraints</th>
<th>Vars</th>
<th>NLvars</th>
<th>Intvars</th>
</tr>
</thead>
<tbody>
<tr>
<td>test15-global</td>
<td>959</td>
<td>2492</td>
<td>764</td>
<td>192</td>
</tr>
</tbody>
</table>

*Application: power plant operation. Originally took 15 hours, now takes 2 hours to global optimum. Types of nonlinearities: $x^y$, $x^k$, $\text{abs}(x)$, $\text{IF}()$*
References


