Affine Lower Bound Functions for Polynomials and Their Use in Global Optimisation (Part 2)

Jürgen Garloff Andrew P. Smith

University of Applied Sciences /
Fachhochschule Konstanz
Department of Computer Science Institute for Applied Research
and
University of Konstanz
Department of Mathematics and Statistics

Christian Jansson

Institute of Computer Science III
Technical University Hamburg–Harburg

Affine Lower Bound Functions for Polynomials and Their Use in Global Optimisation (Part 2)

Outline

- Global optimisation problems with polynomial constraints
- Relaxations of global optimisation problems
- A branch and bound method incorporating these relaxations
- Example GOPs and analysis
- Lower Bound Functions vs. Upper Bound Functions
- Constant vs. Linear Affine Bound Functions
- Future work and improvements

Constrained Global Optimization Problem (GOP)

$$\min_{x \in M} f(x),$$

where the set M of feasible solutions is given by inequalities and equations

$$g_i(x) \leq 0,\ i = 1, \ldots, m,\ x \in X,$$

$$h_j(x) = 0,\ j = 1, \ldots, l,\ x \in X,$$

with $D \subseteq \mathbf{R}^n$, X being a box in D, and f, g_i, h_j are real-valued functions defined on D.

A frequently used approach is the generation of *relaxations* and their use in a branch and bound framework. Generally speaking, a relaxation of the given problem has the properties that

1. each feasible point of the given problem is feasible for the relaxation,

2. the relaxation is easier to solve than the original problem, and

3. the solutions of the relaxation converge to the solutions of the original problem, provided the maximal width of the set of feasible points converges to zero.

We obtain a relaxation for the GOP if we proceed as follows:

1. Replace the objective function f and the functions g_i by affine lower bound functions $\underline{f}$ and $\underline{g_i}$, respectively.
2. If the function h_j defining the j-th equality constraint is affine, then this equation is added to the constraints that define the relaxation. The remaining equations are written as two inequalities and these are treated according to (1).

Then we obtain the following optimization problem: $\min_{x \in N} \underline{f}(x)$
with the set N of feasible solutions given by

$$\begin{aligned} \underline{g_i}(x) &\leq 0,\ i = 1, \ldots, m',\ x \in X, \\ h_j(x) &= 0,\ j = 1, \ldots, l',\ x \in X, \end{aligned}$$

References for recent relaxation techniques: Adjiman and Floudas ('96)
I.P. Androulakis, C.D. Maranas, and C.A. Floudas ('95)
C. Jansson ('00, '01) C.D. Maranas and C.A. Floudas ('94)

Branch and Bound Algorithm

1. *Initialisation*: Set a tolerance ε, set the lower and upper bounds of the global minimum value $\underline{f}^* := -\infty$, $\overline{f}^* := \infty$, $X^0 := X$, and initialise a list $L := \{(\underline{f}^*, X^0)\}$ which contains the subproblems.

2. *Subproblem selection*: If the list L is empty, then the algorithm is terminated. Otherwise select and remove a subproblem $(\underline{f}^*, X^0)$ from L.

3. *Feasibility test*: Test using the Bernstein coefficients of the constraint functions. If infeasible, discard subproblem and return to (2).

4. *Computing an upper bound*: Compute a solution $\tilde{\tilde{x}}$ of the upper-relaxed problem w.r.t X^0, using affine upper bound functions. If $f(\tilde{\tilde{x}}) < \overline{f}^*$, then the overall upper bound $\overline{f}^* := f(\tilde{\tilde{x}})$ is updated and $\overline{f}^*$ is stored in a list A.

5. *Computing a lower bound*: Compute a solution $\underline{\tilde{x}}$ of the lower-relaxed problem w.r.t X^0. If $f(\underline{\tilde{x}}) > \overline{f}^*$, then the current subproblem cannot contain a global minimum point of the original problem. Discard and return to (2). If $f(\underline{\tilde{x}}) < \underline{f}^*$, then the overall lower bound $\underline{f}^* := f(\underline{\tilde{x}})$ is updated.

6. *Branching*: The current box X^0 is bisected into two new boxes X^1, X^2. The new subproblems $(f(\underline{\tilde{x}}), X^1)$, $(f(\underline{\tilde{x}}), X^2)$ are added to L.

7. *Termination*: If $(\overline{f}^* - \underline{f}^*) > \varepsilon$, then goto (2). Otherwise stop.

Example: Nonconvex Test Problem (Jansson 2000)

minimize

$f(x) = \frac{1}{2}(x_2 - x_1^2)^2 + (1 - x_1)^2 + \sum_{i=1}^{2n} (x_i - 1)^2$

subject to

$$\sum_{i=1}^{2n} x_i - 1 = 0$$

$$\sum_{i=1}^{2n} x_i^2 - 0.75 \leq 0 \qquad \sum_{i=1}^{2n} x_i x_{n+i} - \frac{1}{5n} \leq 0$$

$x_i \in [-1, 1], \quad i = 1, \ldots, 2n$

	Constant Bound Functions		**Linear Bound Functions**	
$\mathbf{n} = 1$ (2 variables)				
ε	10^{-6}		10^{-6}	
subdivision depth	46		24	
time (s)	0.05		0.59	
subproblems	547		179	
$\mathbf{n} = 2$ (4 variables)				
ε	10^{-1}	10^{-2}	10^{-1}	10^{-2}
subdivision depth	24	$>$25	16	19
time (s)	25.44	$>$500	122.32	279.05
subproblems	24620	$>$50k	592	1384

Example: Weight Minimization of a Speed Reducer
(Golinski 1970)
Design of a speed reducer for small aircraft engines.
minimize
$f(x) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.5080x_1(x_6^2 + x_7^2) + 7.4770(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2)$
subject to (11 polynomial constraints).

Global minimum value $f^* = 2994$ computed (to 4 s.f. and with constant bound functions) in 138 seconds. (40071 subproblems checked at a maximum subdivision depth of 80.)

Constraints:

$x_1 x_2^2 x_3 \geq 27 \qquad x_1 x_2^2 x_3^2 \geq 397.5 \qquad x_2 x_3 x_6^4 / x_4^3 \geq 1.93 \qquad x_2 x_3 x_7^4 / x_5^3 \geq 1.93$

$$\left(\left(\frac{745 x_4}{x_2 x_3}\right)^2 + 16910000\right)^{\frac{1}{2}} / \, 0.1 x_6^3 \leq 1100$$

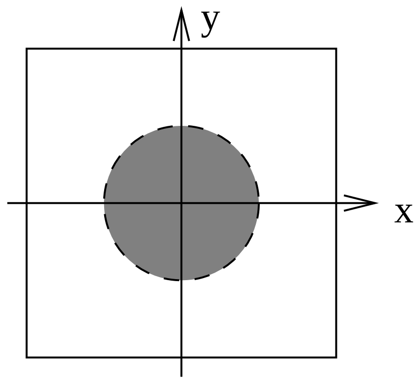
$$\left(\left(\frac{745 x_5}{x_2 x_3}\right)^2 + 157500000\right)^{\frac{1}{2}} / \, 0.1 x_7^3 \leq 850$$

$x_2 x_3 \leq 40 \qquad 5 < x_1/x_2 \leq 12 \qquad 1.5 x_6 + 1.9 \leq x_4 \qquad 1.1 x_7 + 1.9 \leq x_5$

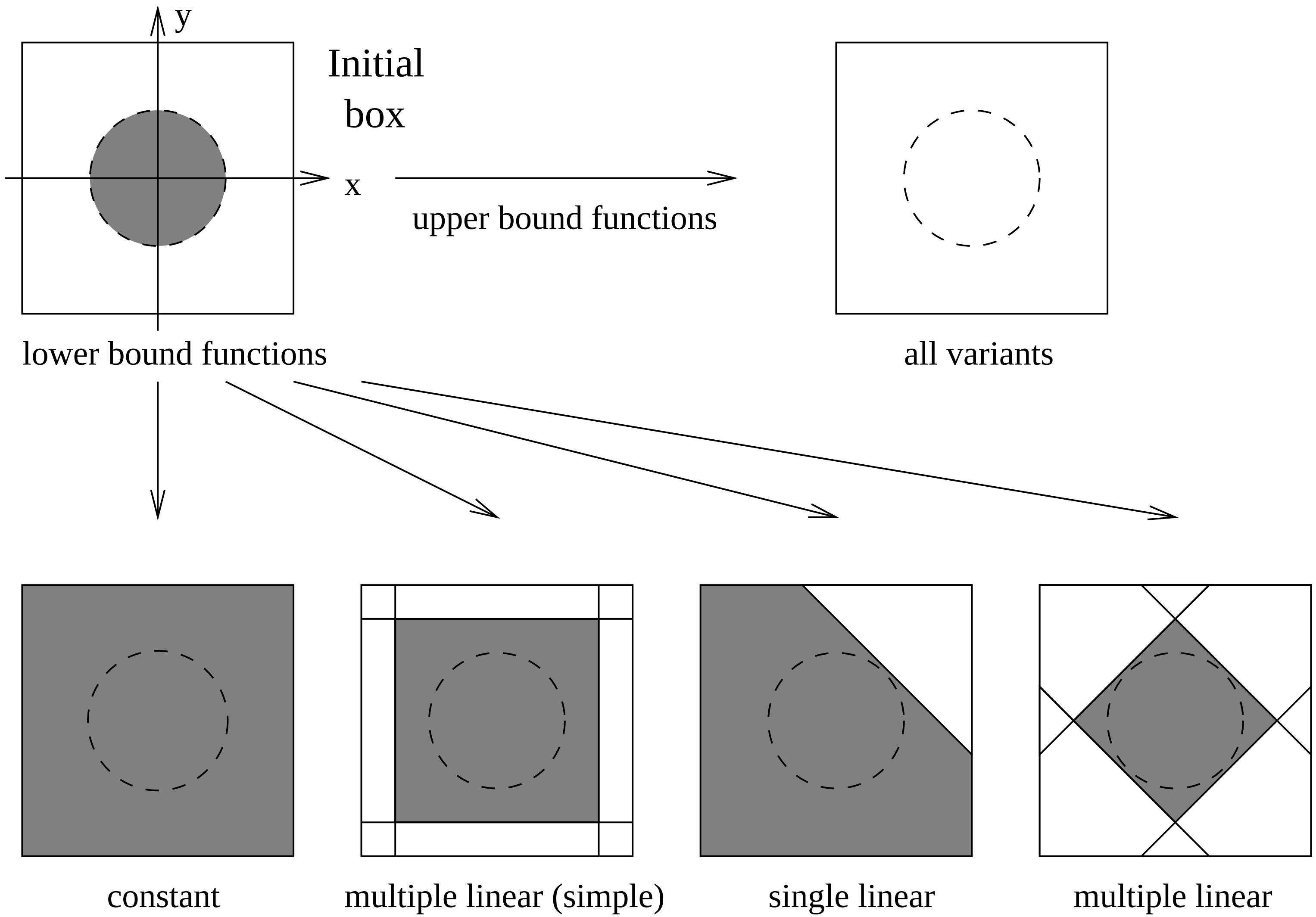
$\mathbf{x} \in ([2.6, 3.6], [0.7, 0.8], [17, 28], [7.3, 8.3], [7.3, 8.3], [2.9, 3.9], [5.0, 5.5])$

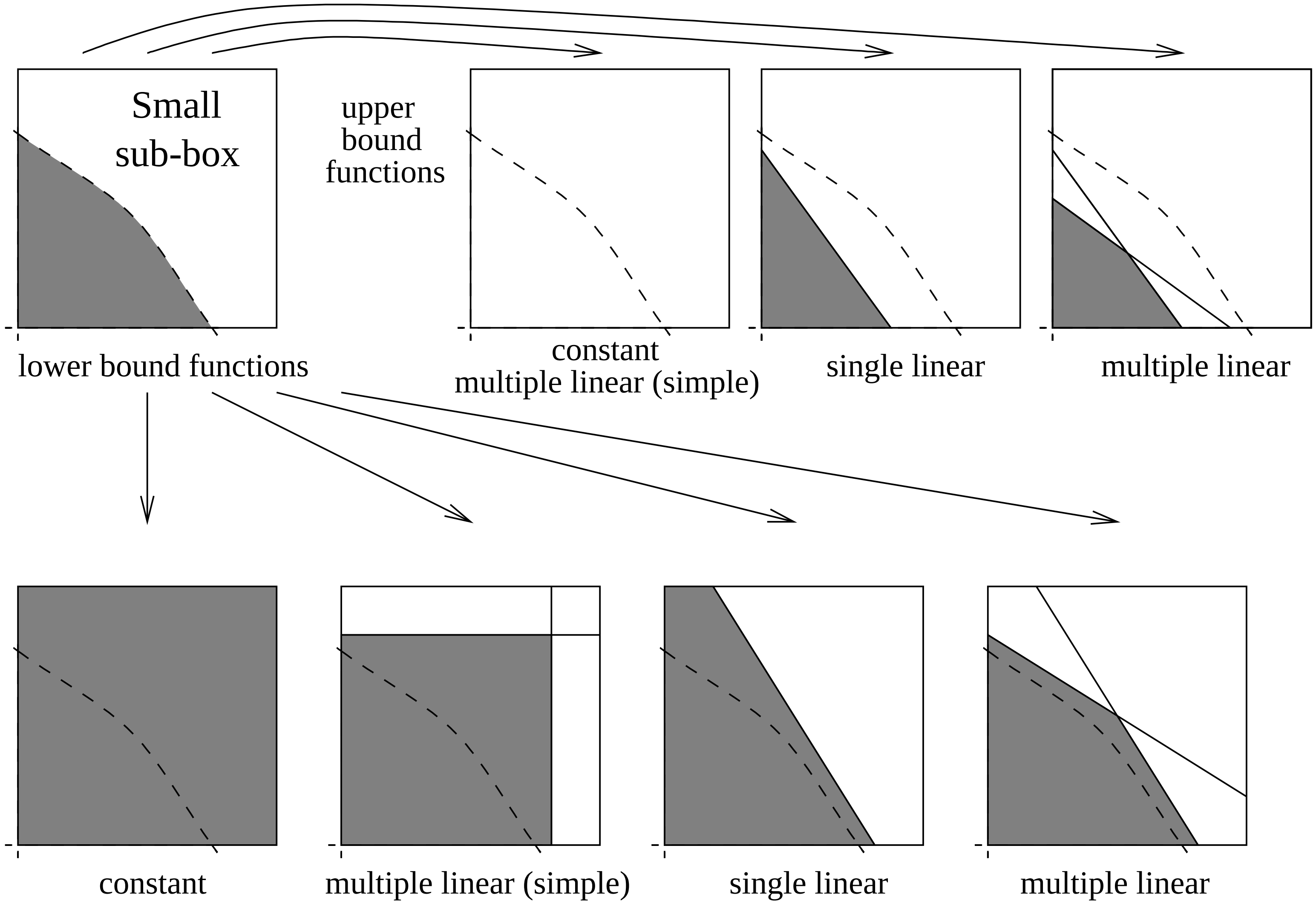
Lower Bound Functions vs. Upper Bound Functions

Consider the inequality constraint $x^2 + y^2 - 1 \leq 0$ over the box $[-2, 2]^2$:



Computing lower bound function(s) corresponds to an outer estimation of the set of feasible points. Upper bound function(s) correspond to an inner estimation.


y
Initial
box
x
upper bound functions
lower bound functions
all variants
constant
multiple linear (simple)
single linear
multiple linear


Small
sub-box
upper
bound
functions
lower bound functions
constant
multiple linear (simple)
single linear
multiple linear
constant
multiple linear (simple)
single linear
multiple linear

Constant vs. Linear Affine Bound Functions

- The use of constant bound functions equates to the use of interval enclosures for the functions, based on the Bernstein expansion. This - by itself - is crude, and requires many subproblems to be processed, but each can be computed extremely quickly. This may be applicable for problems with strongly nonlinear polynomials.

- Multiple cheaply-computed linear bound functions work well over large boxes, however the zero sets are co-ordinate aligned. They often perform as badly as constant functions over small sub-boxes or for upper bounding functions (inner estimations).

- With the best linear bound functions, the branch and bound algorithm terminates with many fewer subproblems, and with smaller subdivision depth. However the computation is not as cheap - and can involve the solution of numerous LP problems with hundreds of constraints for each subproblem. This approach may be best suited to problems with many (multi)linear constraints.

- A future implementation could switch between the use of different bound function types depending on sub-box size, current branching factor, or degree or behaviour of the polynomials.

Future Work

- (Current work) Compute linear lower bound functions more cheaply. One or more functions coinciding with facet(s) of the lower convex hull of the control points of the Bernstein expansion could be computed, requiring the solution of systems of linear equations instead of LP problems.

- Assemble a catalogue of bound functions based on Bernstein expansion, as a tool for global optimisation problems, to be integrated into the COCONUT framework.

- Verified bound functions can be obtained, either by using interval arithmetic, or by the computation of error terms and adjustment.

- Bound functions for derivatives can also be computed - first- and second-order information is easily obtainable from the Bernstein coefficients.

- Taylor expansion can be used to compute bound functions for problems involving general functions. A high-degree Taylor polynomial is computed (introduction of higher degree polynomial terms is not problematic for our approach), for which Bernstein coefficients and a bound function can be computed. The remainder of the Taylor expansion can be enclosed in an interval by interval computation methods. Subtracting this interval from the lower bound function of the Taylor polynomial provides the lower bound function for the given function.