Towards Faster, Better Validated Global Optimization

by
R. Baker Kearfott
Department of Mathematics
University of Louisiana at Lafayette
`rbk@louisiana.edu`

We

- point out the difference between validated and non-validated software.
- briefly review weaknesses of our GlobSol validated software;
- indicate how certain non-validated software succeeds where GlobSol fails;
- show how the non-validated techniques can be put into a validated context.

The General Validated Problem Statement

Given a box

$$\boldsymbol{x} = ([\underline{x}_1, \overline{x}_1], \ldots [\underline{x}_n, \overline{x}_n]),$$

find small boxes

$$\boldsymbol{x}^* = ([\underline{x}_1^*, \overline{x}_1^*], \ldots [\underline{x}_n^*, \overline{x}_n]^*)$$

such that any solutions of

$$\boxed{\begin{array}{l} \text{minimize } \varphi(x) \\ \text{subject to } c_i(x) = 0, i = 1, \ldots, m_1, \\ \qquad\qquad\quad g_i(x) \le 0, i = 1, \ldots, m_2, \\ \text{where } \varphi : \mathbb{R}^n \to \mathbb{R} \text{ and } c_i, g_i : \mathbb{R}^n \to \mathbb{R} \end{array}}$$

are guaranteed to be within one of the $\boldsymbol{x}^*$ that has been found.

General Deterministic Branch and Bound Algorithm

1. Begin with an initial region (box) $\boldsymbol{x}$.
2. Compute an upper bound $\overline{\varphi}$ on the global optimum of φ over $\boldsymbol{x}$ (say, by finding a feasible point $\check{x}$ and evaluating $\varphi(\check{x})$).
3. Place $\boldsymbol{x}$ on a list $\mathcal{L}$ of boxes to be processed.
4. *DO WHILE* $\mathcal{L} \neq \emptyset$:
   (a) Remove a region $\boldsymbol{x}$ from $\mathcal{L}$.
   (b) Eliminate portions of $\boldsymbol{x}$ in various ways.
       - *IF* all of $\boldsymbol{x}$ is eliminated, *THEN CYCLE*.
   (c) Compute a lower bound $\underline{\varphi}(\boldsymbol{x})$ of φ over the feasible points in $\boldsymbol{x}$.
   (d) *IF* $\underline{\varphi} > \overline{\varphi}$ *THEN CYCLE*.
   (e) Possibly update $\overline{\varphi}$.
   (f) If $\boldsymbol{x}$ is sufficiently small, then place $\boldsymbol{x}$ onto a list $\mathcal{C}$ of answer boxes, and *CYCLE*.
   (g) Subdivide $\boldsymbol{x}$ into two or more subregions; place each of these subregions onto $\mathcal{L}$.

   END DO

Validated Branch and Bound

In validated branch and bound:

- All roundoff error is taken into account when evaluating $\overline{\varphi} = \varphi(\check{x})$.
- The lower bounds $\underline{\varphi}(\boldsymbol{x})$ are evaluated taking account of all roundoff error.
- Any process that rejects a portion of a region $\boldsymbol{x}$ takes account of (i.e. compensates for) all roundoff error.

Note: GlobSol (our validated branch and bound software) was designed beginning with focus on validation, while BARON (Sahinidis et al) and αBB (Floudas et al) were designed initially without restriction on validation.

What is GlobSol?

- A Fortran 90 package
  - well-tested.
  - self-contained.
- Solves constrained and unconstrained global optimization problems
- Utility programs for interval and point evaluation, etc.
- Subroutine / module libraries for interval arithmetic, automatic differentiation, etc.
- Publicly available free of charge
  `http://interval.louisiana.edu`
  `/GlobSol/download_GlobSol.html`
- Predecessors were developed beginning in the early 1990's.
- Was put in its present, usable form as part of a SunSoft cooperative research project.

More Information on GlobSol

Additional talks and papers, including information on

- GlobSol's successes (problems successfully solved by GlobSol),
- how to use GlobSol,
- GlobSol's features

can be found at:

`http://interval.louisiana.edu`
`/preprints.html`

(Search on "GlobSol".)

Weaknesses of “Naive” Procedures in GlobSol

- The lower bound on the objective over a region does not take account of the constraints;
- In computing the upper bound on the global optimum, a simplified projection method that only computes feasible points (and not approximate optimizers) has been used (so the upper bound is not sharp).

  (The historical reason for this is because GlobSol contains only code that can be freely distributed.)

Example of Objective and Constraint Interaction

Minimax Approximation

$$\min_x \max_{1\le i\le m} |f_i(x)|, \quad f_i : \mathbb{R}^n \to \mathbb{R}, \quad x \in \mathbb{R}^n, \quad m \ge n.$$

- Non-smooth extensions in GlobSol have not been successful for this problem.
- Alternately, convert to a smooth problem:

$$\min_{x\in\mathbb{R}^n} v$$

$$\text{such that } \left\{ \begin{array}{rcl} f_i(x) & \le & v \\ -f_i(x) & \le & v \end{array} \right\}, \quad 1 \le i \le m.$$

  - Evaluation of the "objective" with interval arithmetic is useless here.
  - Constraint propagation to reduce the objective range does not take account of coupling in the inequality constraints.

Solution to the Constraint and Objective Interaction Difficulty

Use Convex Underestimators and Linear Programming

Use ideas borrowed from the BARON and αBB developers:

- Replace the original problem by a quadratic program that is a relaxation.
  - (well-developed by Floudas, Sahinidis et al)
- Solve the relaxation with standard linear programming technology.
  - (also well-developed by Floudas, Sahinidis et al)
- Validate the relaxation with a technique of Jansson.
  - (new, but hopefully straightforward; work by me and Hongtong Siriporn)

A Possible Pitfall with Convex Underestimators

1. The underestimators work with inequality constraints.
2. A standard way of solving the resulting quadratic programming problem (QP) is with linear complementarity.
3. The linear complementarity problem is degenerate when equality constraints are converted into inequality constraints.
4. Pairs of nonlinear inequality constraints corresponding to equality constraints are replaced by linear underestimators, the resulting linear complementarity problem may not be exactly degenerate but may be approximately degenerate.

We are in the process of investigating this both theoretically and practically.

Solution to GlobSol's Problem of Good Upper Bounds

- Use a good constrained local optimizer.
  - `IPOPT` has proven reliable in initial tests.
    - `IPOPT` is part of COIN, an open-source, free software project, and can be distributed with GlobSol.
    - We have interfaced `IPOPT` to GlobSol's parser.