

GlobSol: History, Composition, Advice on Use, and Future

by

R. Baker Kearfott

Department of Mathematics

University of Louisiana at Lafayette

`rbk@louisiana.edu`

This talk will

- discuss underlying elements of the GlobSol package;
- give a brief history of GlobSol's development;
- mention some strengths and weaknesses of GlobSol's approaches;
- give advice on GlobSol's use;
- outline possible future development of GlobSol.

Interval Global Optimization: Goals

Given a box

$$\mathbf{x} = ([\underline{x}_1, \bar{x}_1], \dots, [\underline{x}_n, \bar{x}_n]),$$

find small boxes

$$\mathbf{x}^* = ([\underline{x}_1^*, \bar{x}_1^*], \dots, [\underline{x}_n^*, \bar{x}_n^*])$$

such that any solutions of

minimize $\phi(x)$
subject to $c_i(x) = 0, i = 1, \dots, m_1,$
 $g_i(x) \leq 0, i = 1, \dots, m_2,$
where $\phi : \mathbb{R}^n \rightarrow \mathbb{R}$ and $c_i, g_i : \mathbb{R}^n \rightarrow \mathbb{R}$

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

Elements of GlobSol

- automatic differentiation,
- constraint propagation,
- interval Newton methods,
- additional, specialized techniques.

Automatic Differentiation in GlobSol

- Designed originally for small problems.
- Implemented simply, to aid understanding of the code and minimize implementation effort.
- Serves its purpose, but certain inefficiencies are present.

Constraint Propagation in GlobSol

GlobSol applies constraint propagation at the level of individual operations.

- Allows simple, automatic constraint propagation connected with the automatic differentiation.
- Allows decoupling of interval dependencies in the “big system.”
- *Usually* results in some savings in the overall algorithm, relative to not doing the propagation.
- *Sometimes* results in the difference between being able to solve a problem and not being able to solve it.

GlobSol Constraint Propagation

An Example

$$\begin{aligned}f_1(x_1, x_2) &= x_1^3 + x_1^2x_2 + x_2^2 + 1 \\f_2(x_1, x_2) &= x_1^3 - 3x_1^2x_2 + x_2^2 + 1.\end{aligned}$$

The automatic differentiation process produces a code list (non-unique) such as

$$\begin{aligned}v_1 &= x_1 & v_2 &= x_2 \\v_3 &= v_1^2 & v_4 &= v_2^2 \\v_5 &= v_3v_2 & v_6 &= v_1^3 \\v_7 &= v_6 + v_4 + 1 \\v_7 + v_5 &= 0 \\v_7 - 3v_5 &= 0,\end{aligned}$$

Constraint propagation is done on these equations (using upper bounds for the objective, conditions on the gradient, etc.).

Interval Newton Methods in GlobSol

- Used with generalized Lagrange multiplier equations.
 - to narrow bounds;
 - *not* to verify critical points.
- Used to verify feasibility:
 - works with inequality and active equality constraints only;
 - works well, but only when the number of equality constraints is small.
- We have adopted this scheme from experimental observation of efficiencies on a number of problems.
- By not verifying critical points, we give up some strength in the statements we make about the answers.

Use of Already Found Approximate Optima

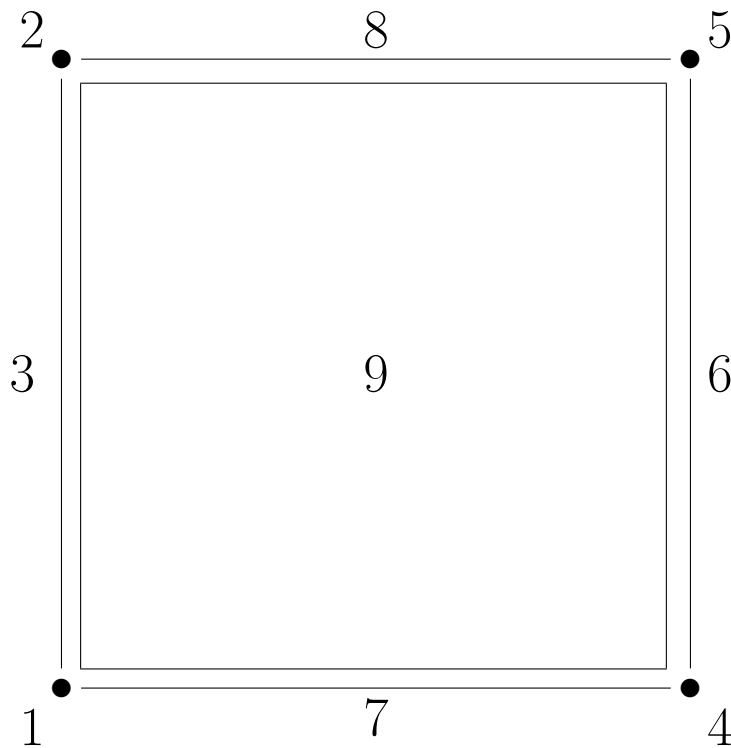
- Approximate feasible points are found with a generalized Newton method based on the Moore–Penrose pseudo-inverse.
- Boxes are constructed with diameters proportional to the square root of the tolerance to which the approximate feasible points are found.
- These boxes are placed on a list, then removed from the search space with a set complementation process.
- Boxes are later culled from the list as better upper bounds for the optimizer are found.
- This scheme avoids excessive clusters, even near singular roots;

Bound Constraints and the Peeling Process

- Bound constraints can be handled by considering the boundary boxes as usual boxes, with some coordinates degenerate.
- Boxes can be rejected (from infeasibility, etc.), and the upper bound on the optimum can be adjusted during the boundary box generation process.
- This works well for small numbers of bound constraints, since more “thin” intervals are used in range computations.
- For larger numbers of bound constraints, the number of boxes increases exponentially with the number of bound constraints. Thus, handling the bound constraints as general inequality constraints is more practical for large numbers of bound constraints.

An Illustration of “Peeling”

(handling bound constraints)



There are 1 box in \mathbb{R}^2 , 4 boxes in \mathbb{R}^1 and 4 boxes in \mathbb{R}^0 .

History of GlobSol

GlobSol began as a research test bed, and grew out of the following earlier packages.

INTBIS: A FORTRAN-77 package for verified solution of nonlinear algebraic systems (ACM TOMS Algorithm 681, 1990)

INTLIB: A FORTRAN-77 library for interval evaluation of standard functions (ACM TOMS Algorithm 737, 1994)

Constraint propagation experiments: (Article, *Computing* **47** (2) (1991), pp. 169–191)

INTERVAL_ARITHMETIC: A Fortran-90 module providing an interval data type and an interface to **INTLIB** (ACM TOMS Algorithm 763, 1996).

History of GlobSol

(continued)

AD, algorithm improvements:

INTOPT_90, *SIAM. J. Sci. Comput.* **18**
(1997) and the book *Rigorous Global
Search: Continuous Problems*, Kluwer,
1996.

The Sun Microsystems project: User
interface, testing and bug removal,
improvements, applications; first called
“GlobSol”

<http://www.mscs.mu.edu/~globalsol/>

INTBIS and INTLIB

- Emphasis is on simplicity and ability of user to modify the code and to port the code.
- There is no constraint propagation other than interval Newton methods, and algorithms are simple.
- Except for polynomial systems, the user inputs the equations by assembly-language-like programming.
- Nonetheless:
 - Evaluation of the equations (constraints) is significantly more efficient than in GlobSol.
 - Stadtherr et al have modified **INTBIS** / **INTLIB** for some of the most impressive application successes in the field.

Our Motivations for INTOPT_90 and Further Development

Motivation for AD: We were hampered by having to program many lengthy subroutines for even simple test functions for algorithm research.

Motivation for constraint propagation: This was one possible way of dealing with interval dependency and speeding up the overall global search process.

Motivation for restructuring, etc.: To make it easier

- for students, etc. to learn how to use the package, to pursue the applications in the SunSoft cooperative research project;
- to deliver and install;
- to support.

The Sun Microsystems Project

- GlobSol grew from INTOPT_90.
- Collaborative effort with George Corliss, under Bill Walster, with participation of graduate students
- Much improved distribution packaging (the build is automated)
- Improved package structuring (subroutine calling sequences, source directory structure, etc.)
- Removal of many bugs
- Improved capabilities:
 - both inequality and equality constraints
 - “thick” constants
 - subdivision stopping criteria
- Improved understanding of how to use in applications and of advice to give users

GlobSol's Overall Algorithm

- The GlobSol script
- The overall GlobSol optimization process
 - Initial I/O
 - The “peeling” process.
 - The actual global search routine
 - Final I/O.

The GlobSol Script

```
globsol <source file> <data number>
```

1. Compiles and runs the user-provided Fortran-90 program to produce the internal representation, or *codelist* for the objective, constraints, and gradients.
2. Runs an optimizing program that removes redundant operations from the code list.
3. Runs the overall GlobSol optimization process.
4. Cleans the directory and temporary files.

GlobSol's Global Search Routine

`rigorous_global_search.f90`

1. Remove a box \mathbf{x} from a list of not-yet-examined boxes \mathcal{L} .
2. *IF* \mathbf{x} is sufficiently small *THEN* store \mathbf{x} and *CYCLE*.
3. Use constraint propagation to possibly narrow the coordinate widths of the box \mathbf{x} and possibly reject the box \mathbf{x} .
4. Perform an interval Newton method to possibly narrow the coordinate widths of \mathbf{x} and to possibly reject \mathbf{x} .
5. *IF* the coordinate widths of \mathbf{x} are now sufficiently narrow *THEN* store \mathbf{x} and *CYCLE*.
6. (Subdivide) Bisect \mathbf{x} along a chosen coordinate, forming two new boxes; place these boxes on the list \mathcal{L} .

GlobSol's Algorithms

Some Complications

“Storing” a box \mathbf{x} presently involves

- an attempt to verify existence of feasible points within the box, and to adjust the best upper bound for the global optimum;
- expansion to avoid clustering of solution boxes;
- examination of intersections with not-yet-processed boxes and already-stored boxes to avoid redundancy.

Use of GlobSol: Required Files

(How the user controls GlobSol)

GlobSol.CFG: The GlobSol configuration file.

This is used to

- control printing,
- set tolerances,
- select algorithm components, such as choice of preconditioner, choice of interval Newton method, form of problem (nonlinear system optimization), etc.

<name>.f90: The user-supplied Fortran 90 source file defining the objective and constraints.

<name>.DT#: The “box data file.” where the user supplies search limits, which coordinates are considered as bound constraints, and an initial guess for a global optimizer, if available.

The Configuration File

GlobSol.CFG

- Is largely self-documenting.
- Users should seldom need to change items other than level of printing, tolerances, and type of problem, and things such as the total CPU time to be allowed to solve a particular problem.
- GlobSol contains several algorithm variants of an experimental nature. Some variants are not presently supported at the level that more promising variants are; users should be wary of changing the configuration file to use these variants.

An Example Problem and Box Data file

minimize $\phi(X) = -2 * x_1^2 - x_2^2$
subject to constraints

$$\begin{aligned}x_1^2 + x_2^2 - 1 &\leq 0 \\x_1^2 - x_2 &\leq 0 \\x_1^2 - x_2^2 &= 0\end{aligned}$$

mixed.DT1

```
1D-5      ! General domain tolerance
  0  1     ! Bounds on the first variable
  0  1     ! Bounds on the second variable
F F       ! X(1) has no bound constraints
F F       ! X(2) has no bound constraints
```

Subsequent optional lines can give an initial guess point.

A Simple User-Supplied Function file mixed.f90

```
PROGRAM SIMPLE_MIXED_CONSTRAINTS
  USE CODELIST_CREATION
  PARAMETER (NN = 2)
  TYPE(CDLVAR), DIMENSION(NN) :: X
  TYPE(CDLLHS), DIMENSION(1):: PHI
  TYPE(CDLINEQ), DIMENSION(2) :: G
  TYPE(CDLEQ), DIMENSION(1)   :: C

  CALL INITIALIZE_CODELIST(X)

  PHI(1) = -2*X(1)**2 - X(2)**2
  G(1) = X(1)**2 + X(2)**2 - 1
  G(2) = X(1)**2 - X(2)
  C(1) = X(1)**2 - X(2)**2

  CALL FINISH_CODELIST
END PROGRAM SIMPLE_MIXED_CONSTRAINTS
```

GlobSol Output File

abridged first part

Typing “`globalsol mixed 1`” gives
`mixed.OT1`, containing:

Output from FIND_GLOBAL_MIN on 04/06/1999 at 08:03:52.
Version for the system is: March 20, 1999

Codelist file name is: MIXEDG.CDL
Box data file name is: MIXED.DT1

Initial box:
[0.0000E+00, 0.1000E+01] [0.0000E+00, 0.1000E+01]

BOUND_CONSTRAINT:
F F F F

CONFIGURATION VALUES:

EPS_DOMAIN: 0.1000E-04 MAXITR: 60000
DO_INTERVAL_NEWTON: T QUADRATIC: T FULL_SPACE: F
VERY_GOOD_INITIAL_GUESS: F
USE_SUBSIT: T
OUTPUT UNIT: 7 PRINT_LENGTH: 1
Default point optimizer was used.

Globsol Output File

abridged second part

THERE WERE NO BOXES IN COMPLETED_LIST.

LIST OF BOXES CONTAINING VERIFIED FEASIBLE POINTS:

Box no.:1
Box coordinates:
[0.7071E+00, 0.7071E+00] [0.7071E+00, 0.7071E+00]
PHI:
[-0.1500E+01, -0.1500E+01]
Level: 3
Box contains the following approximate root:
0.7071E+00 0.7071E+00
OBJECTIVE ENCLOSURE AT APPROXIMATE ROOT:
[-0.1500E+01, -0.1500E+01]
U0:
[0.3852E+00, 0.3852E+00]
U:
[0.5777E+00, 0.5777E+00] [0.0000E+00, 0.1000E+01]
V:
[0.1926E+00, 0.1926E+00]
INEQ_CERT_FEASIBLE:
F T
NIN_POSS_BINDING:1

Number of bisections: 1
BEST_ESTIMATE: -0.1500E+01
Total number of boxes processed in loop: 4
Overall CPU time: 0.5000D-01

Advice on GlobSol's Use

Interpretation of Results

- The only guarantees (at present): Any global minimizers must lie within the boxes in the output list.
- A non-empty output list does not guarantee existence of global minima.
- An empty output list does not necessarily mean a bug. It usually means either an overconstrained or an underconstrained problem. (The search box does not correspond to bound constraints unless explicitly set.)

Advice on GlobSol's Use

Turning On and Off Constraint Propagation

- On average, and for most problems, constraint propagation gives a small but significant improvement in GlobSol's overall performance.
- Constraint propagation is indispensable for some problems.
- Because inverses are not implemented in extended arithmetic in GlobSol, constraint propagation does not work for some problems.
- Users can experiment with the switch `USE_SUBSIT` (use “substitution-iteration”) in `GlobSol.CFG` to turn on and off constraint propagation.

Advice on Bound Constraints

- GlobSol solves unconstrained problems more efficiently than constrained problems.
 - Large numbers of bound constraints lead to excessive “peeling.”
 - In practice, interval Newton methods do not seem to function well with the Fritz–John system for larger boxes.
- Suggestion:
 1. Start with an unconstrained system.
 2. Selectively add constraints until the problem has a solution.
 3. Experiment with which constraints are important.

But does this correspond to optimal use of constraint technology???

On Equality Constraints, Data Fitting

- We have tried GlobSol with numerous test sets with nonlinear least squares problems and nonlinear minimax problems (which we reformulated as smooth problems with constraints using Lemaréchal's technique).
- GlobSol almost uniformly failed to solve these problems efficiently.
- Although the Fritz–John matrix is not necessarily singular at critical points, we believe the problems to be related to intrinsic properties of the Fritz–John matrix, in the case of minimax problems.
- We have also tried several variants for least squares, without success.
- Thus, GlobSol cannot presently handle overdetermined systems well.

Possibilities for Data Fitting Problems

- For the interval Newton method, reformulate the minimax problem as an uncertain linear programming problem, and solve with Jansson's techniques.
- Reformulate as a system of equations with tolerances (as Hansen / Walster explain).
 - We have experimented in some depth with this technique last summer.
 - Although hopeful, we don't have a final answer yet.
 - This technique will involve getting the user to accept a different interpretation of "solution". The solution will not be least-squares or minimax, but a set over which all residuals are within a tolerance. (See my talk on the linear case from Validated Computing 2002.)

GlobSol's Future

- Totally rewrite to simplify the structure more.
- Improve the user interface (better I/O) and low-level (more efficient code list interpretation, etc.)
- Include of additional good algorithms of others (the LP rejection technique of Nenov, van Hentenryck's univariate search, etc.)
- Do additional research on new algorithms (such as for nonlinear data fitting).

These items need to be prioritized, and time resources need to be found.

GlobSol's Future

Ideas from COCOS'02

- Make GlobSol available through a web interface (probably NEOS)
- Implement affine arithmetic, convex underestimators and overestimators
- Implement capabilities for mixed integer programming.