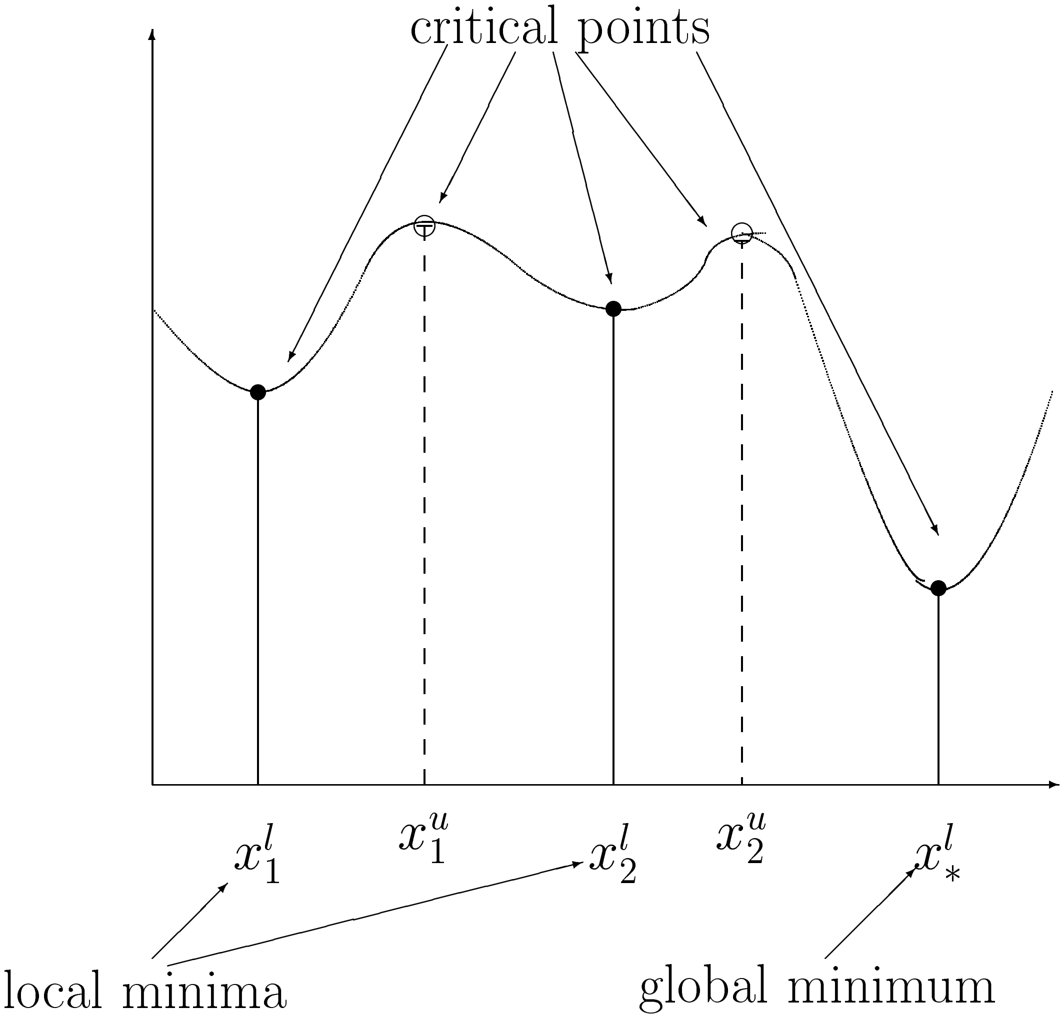
An Overview of the GlobSol Package for Verified Global Optimization

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

This talk will

- Contrast verified and non-verified global optimization
- Briefly discuss the underlying mathematics of verified global optimization
- Review capabilities of the GlobSol software package
- Review an example of how to use GlobSol
- Give an on-line demonstration.

Local Versus Global Optimization



Local Optimization Versus Global Optimization

Local Optimization

- The model is steepest descent with univariate line searches (for monotone decrease of the objective function). (Start a ball on a hill and let it roll to the bottom of the nearest valley.)
- Algorithm developers speak of "globalization," but mean only the design of algorithm variants that increase the domain of convergence. (See J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Least Squares*, Prentice–Hall, 1983.)
- Algorithms contain many heuristics, and do not always work. However, many useful implementations exist.

Local Optimization Versus Global Optimization

Global Optimization

- is a much harder problem. Progress has accelerated with increases in computing power.
- Early milestones are L. C. W. Dixon and G. P. Szego, *Towards Global Optimization* (North–Holland, 1975), and *Towards Global Optimization 2* (North–Holland, 1977).
- Two types of algorithms: <u>stochastic</u> and <u>deterministic</u>.
- Deterministic algorithms can be either <u>rigorous</u> or <u>heuristic</u>.

Global Optimization

Stochastic Algorithms

Monte-Carlo search: Random points are generated in the search space. The point with lowest objective value is taken to be the global optimum.

Simulated annealing: is similar to a local optimization method, although larger objective values are accepted with a probability that decreases as the algorithm progresses.

Genetic algorithms: Attributes, such as values of a particular coordinate, correspond to particular "genes." "Chromosomes" of these genes are recombined randomly, and only the best results are kept. Random "mutations" are introduced.

Global Optimization

Deterministic Optimization

- involves some kind of systematic global search over the domain.
- The various algorithms rely on estimates of the range of the objective function over subdomains.
- Some algorithms (due to Mladineo, Schubert, Wood, etc.) rely on <u>Lipschitz constants</u> to obtain estimates of ranges.
- Bounds on ranges are also obtained with <u>outwardly rounded interval arithmetic</u>.

Global Optimization

Hybrid Stochastic / Deterministic Algorithms

Recently, several people have combined statistical methods with deterministic methods.

- Janos Pinter uses a statistical model to estimate local approximations to Lipschitz constants for a global search.
- Donald Jones constructs a cumulative statistical model of of the objective, and uses deterministic global optimization with a simpler objective to determine optimal placement of the next sample point.
- Kaj Madsen uses multiple starts of a local optimizer to simulate a rigorous global search with interval methods.

On the State of the Art

- Minimizing a function over a compact set in $\mathbb{R}^n$ is an NP-complete problem.
- Thus, barring monumental discoveries, any *general* algorithm will fail for some high-dimensional problems.
- There are many practical problems that can be solved in low-dimensional spaces.
- Some low-dimensional problems are difficult.
- Advances in computer speed and algorithm construction have allowed many more practical problems to be solved, including high-dimensional ones.

Deterministic Global Optimization

Interval Methods

- Evaluation of an objective function $\phi(X)$ at an interval vector $\boldsymbol{X}$ gives bounds on the actual range of ϕ over $\boldsymbol{X}$.
  - If directed rounding is used, the bounds rigorously contain the mathematical range.
  - The bounds, in general, are overestimates.
- If the lower bound of $\phi(\boldsymbol{X})$ is greater than a previously computed objective value $\phi(X)$, then $\boldsymbol{X}$ can be discarded.
- Interval Newton Methods, combined with directed rounding, can *prove* existence and uniqueness of critical points, as well as reduce the size of regions $\boldsymbol{X}$.

Interval Methods

Advantages

easier to use: Obtaining bounds with interval methods involves programming the objective function, while using Lipschitz constant-based methods may require extensive preliminary analysis.

more efficient: Despite interval overestimation of ranges, the overestimation is often less than with a fixed Lipschitz constant. *(But keep in mind the success of hybrid deterministic / stochastic algorithms.)*

more capable: With directed roundings, interval methods cannot lie. Also, interval Newton iteration results in quadratic convergence effects.

Underlying Mathematics

- Classical fixed point theory implies existence.
  - Contraction Mapping Theorem
  - Brouwer Fixed Point Theorem
  - Miranda's Theorem
- Regularity (non-singularity) implies uniqueness.
- Fundamental property of interval arithmetic allows *computational* existence and uniqueness.

Underlying Mathematics

Miranda's Theorem

Theorem 1 *Suppose $\boldsymbol{X} \in \mathbb{IR}^n$, and let the faces of $\boldsymbol{X}$ be denoted by*

$$\begin{aligned} \boldsymbol{X}_{\underline{i}} &= (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, \underline{x}_i, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_n)^T \\ \boldsymbol{X}_{\overline{i}} &= (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, \overline{x}_i, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_n)^T . \end{aligned}$$

Let $F = (f_1, \ldots, f_n)^T$ be a continuous function defined on $\boldsymbol{X}$. If

$$\boldsymbol{f}_i^u(\boldsymbol{X}_{\underline{i}}) \boldsymbol{f}_i^u(\boldsymbol{X}_{\overline{i}}) \leq 0 \tag{1}$$

for each i between 1 and n, then there is an $X \in \boldsymbol{X}$ such that $F(X) = 0$.

Underlying Mathematics

Regularity

Lemma 2 *Suppose $F : \boldsymbol{X} \to \mathbb{R}^n$ and $\boldsymbol{A}$ is a Lipschitz matrix, such as $\boldsymbol{F'}(\boldsymbol{X})$. If $\boldsymbol{A}$ is regular, then any root of F in $\boldsymbol{X}$ is unique.*

Proof: Suppose $X^* \in \boldsymbol{X}$ and $X \in \boldsymbol{X}$ have $F(X^*) = 0$ and $F(X) = 0$. If $\boldsymbol{A}$ is a Lipschitz set, then there is an $A \in \boldsymbol{A}$ such that

$$\begin{aligned} F(X^*) - F(X) &= 0 \\ &= A(X^*) - A(X) \\ &= A(X^* - X). \end{aligned}$$

If $X^* \neq X$, then A would have a null vector, contradicting the regularity of $\boldsymbol{A}$. $\square$

A Computational Existence Tool

Interval Gauss–Seidel Method

Assume $F(X) = (f_1(X), \ldots, f_n(X))$ is continuously differentiable. Then the mean value theorem gives

$$f_i(x_i) = f_i(x_{\neg i}, \check{x}_i) + \frac{\partial f_i}{\partial x_i}(x_{\neg i}, \xi_i)(x_i - \check{x}_i),$$

whence $f_i(X) = 0$ provided

$$\begin{aligned} x_i &= \check{x}_i - f_i(x_{\neg i}, \check{x}_i) \Big/ \frac{\partial f_i}{\partial x_i}(x_{\neg i}, \xi_i) \\ &\subseteq \check{x}_i - f_i(\boldsymbol{x}_{\neg i}, \check{x}_i) \Big/ \frac{\partial f_i}{\partial x_i}(\boldsymbol{x}_{\neg i}, \boldsymbol{x}_i) \end{aligned}$$

The above inclusion forms the Interval Gauss–Seidel iteration equation.

Interval Gauss–Seidel Method

Verification Properties

Do sequentially for $1 = 1$ to n:

1. $\tilde{\boldsymbol{x}}_i \leftarrow \check{x}_i - f_i(\boldsymbol{x}_{\neg i}, \check{x}_i) \Big/ \frac{\partial f_i}{\partial x_i}(\boldsymbol{x}_{\neg i}, \boldsymbol{x}_i)$
2. $\boldsymbol{x}_i \leftarrow \tilde{\boldsymbol{x}}_i$

Then:

1. If $\tilde{\boldsymbol{x}}_i \subseteq \boldsymbol{x}_i$ after each step 1, then the hypotheses of Miranda's theorem are satisfied, and, hence, there exists a solution to $F(X) = 0$ within $\boldsymbol{X}$.
2. If $\tilde{\boldsymbol{x}}_i \subseteq \boldsymbol{x}_i$ after each step 1, then $\boldsymbol{F}'(\boldsymbol{X})$ must be regular, and hence, the solution in $\boldsymbol{X}$ is also unique.

Note: The system $F(X) = 0$ is usually *preconditioned* by a point matrix Y to make the Jacobi matrix $Y\boldsymbol{F}'(\boldsymbol{X})$ approximately diagonal.

Preconditioned Interval Gauss–Seidel

An Example

$$\begin{aligned} f_1(x_1, x_2) &= x_1^2 - 4x_2, \\ f_2(x_1, x_2) &= x_2^2 - 2x_1 + 4x_2, \end{aligned}$$

and

$$\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2)^T = ([-0.1, 0.1], [-0.1, 0.3])^T.$$

$X^* = (0, 0)^T$, of $F(X^* = 0$ is unique in $\boldsymbol{X}$. Take $\check{X} = (0, .1)^T$, so $F(\check{X}) = (-.4, .41)^T$, and

$$\begin{aligned} \boldsymbol{F}'(\boldsymbol{X}) &= \begin{pmatrix} 2\boldsymbol{x}_1 & -4 \\ -2 & 2\boldsymbol{x}_2 + 4 \end{pmatrix} \\ &= \begin{pmatrix} [-.2, .2] & -4 \\ -2 & [3.8, 4.6] \end{pmatrix}. \end{aligned}$$

An Example

(Continued)

Take Y to be the inverse of the midpoint matrix of $\boldsymbol{F}'(\boldsymbol{X})$:

$$Y = \{\mathrm{m}(\boldsymbol{F}'(\boldsymbol{X}))\}^{-1} = \begin{pmatrix} -.525 & -.5 \\ -.25 & 0 \end{pmatrix}.$$

Then $Y\boldsymbol{F}'(\boldsymbol{X}) = \begin{pmatrix} [0.895, 1.105] & [-.2, .2] \\ [-.05, .05] & 1 \end{pmatrix}$,
$YF(\check{X}) = (.005, .1)^T$, and

$$\begin{aligned} \tilde{\boldsymbol{x}}_1 &= 0 - \frac{-.005 + [-.2, .2][-.2, .2]}{[.895, 1.105]} \\ &\subseteq \frac{[-.035, .045]}{[.895, 1.105]} \subseteq [-.0392, .0503] \\ &\subset \boldsymbol{x}_1 = [-.1, .1]. \end{aligned}$$

Similarly, $\tilde{\boldsymbol{x}}_2 \subset \boldsymbol{x}_2$, so there is a unique root of F in $\boldsymbol{X}$.

What is GlobSol?

- A Fortran 90 package
  - well-tested.
  - self-contained.
- Solves constrained and unconstrained global optimization problems
- Separate program solves square algebraic systems of equations.
- 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`

GlobSol

Special Features

- Objective function and constraints are coded as Fortran 90 programs.
- Can use constraint propagation (substitution/iteration) on the intermediate quantities in objective function, equality, and inequality constraint evaluation.
- Can use an overestimation-reducing "peeling" process for bound-constraints.
- Uses an effective point method to find approximate feasible points.
- Has a special augmented system mode for least squares problems.

GlobSol Features

(continued)

- Has extensive error-checking (user input, internal errors, etc.)
- Has on-line web page documentation.
- The algorithm is configurable.
- Has various levels of printing, for various algorithm aspects.
- Source code and libraries for components are available.
  - Automatic differentiation access.
  - Interval arithmetic access.
  - User-modifiable, with adequate study.
- Gives performance statistics, both in report form and for input to spreadsheets.

Use of GlobSol

An Example

The following Fortran 90 program defines the objective function

$$\text{minimize} \quad \phi(X) = -2 * x_1^2 - x_2^2$$

subject to constraints

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

Use of GlobSol

An Example, continued

```
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 Example

(continued)

1. Running the above program produces an internal representation, or <u>code list</u>.
2. The code list is then symbolically differentiated.
3. The optimization code interprets the derivative code list at run time to produce floating point and interval evaluations of the objective function, gradient, and Hessian matrix.
4. A separate data file defines the initial search box, the bound constraints, and the initial guess, if any.
5. A configuration file `GlobSol.CFG` supplies algorithm options, such as which interval Newton method to use and how to precondition the linear systems.

GlobSol Example

The Data File

```
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.

GlobSol Example

Output File – abridged first part

```
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 Example

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
```

Simple Example of “Thick” Constants

$$\text{minimize } (x_1 - [1,2])^2 + (x_2 - [3,4])^2$$

```
PROGRAM THICK_PARAMETER_EXAMPLE
 USE CODELIST_CREATION
 IMPLICIT NONE

 INTEGER, PARAMETER:: NN=2
 TYPE(CDLVAR), DIMENSION(NN):: X
 TYPE(CDLLHS) :: PHI

 OUTPUT_FILE_NAME='thick_parameter_example.CDL'
 CALL INITIALIZE_CODELIST(X)

 PHI = (X(1) - INTERVAL(1,2))**2 &
       + (X(2)-INTERVAL(3,4))**2

 CALL FINISH_CODELIST
END PROGRAM THICK_PARAMETER_EXAMPLE
```

Example with Thick Constants

(continued)

The “solution” is the set of all possible minima with constants in $[1, 2]$ and $[3, 4]$. Thus, a minimum minimum and maximum minimum are obtained. The solution is

$$x_1 \in [1, 2], \quad x_2 \in [3, 4], \quad \text{and } \phi = 0.$$

With initial box ([-10,10], [-10,10]), GlobSol gives

```
 Box no.:           1
 Box coordinates:
 [ 0.1000D+01, 0.1500D+01 ] [ 0.3000D+01, 0.4000D+01 ]
PHI:
 [ 0.0000D+00, 0.2000D+01 ]

 Box no.:           2
 Box coordinates:
 [ 0.1500D+01, 0.2000D+01 ] [ 0.3000D+01, 0.4000D+01 ]
PHI:
 [ 0.0000D+00, 0.2000D+01 ]

 BEST_ESTIMATE:    0.5000D+00
 Total number of boxes processed in loop:           4
 Overall CPU time:    0.0000D+00
```

Thick Constants

A Practical Application

- Work by Claudio Rocco
- Maintenance scheduling optimization based on a model by Dekker et al
- A multi-component system has n components. Each component can be scheduled for preventive maintenance every $k_i T$ time units, where T is a fixed interval and the k_i are integers.
- The total cost of a particular schedule is written as an unconstrained minimization problem, with integer variables k_i and the base interval T.
- The expected deterioration cost of each component is not known precisely; the cost of the i-th component depends on a parameter c_i known to lie in an interval.

Maintenance Scheduling Application

GlobSol's Possibilities

1. For an objective of the form $f(x, \boldsymbol{p})$, where $\boldsymbol{p}$ is a set of parameters subject to interval uncertainty, GlobSol can compute a lower bound on
$$\min_{p \in \boldsymbol{p}} \left\{ \min_x f(x, p) \right\}$$
and an upper bound on
$$\max_{p \in \boldsymbol{p}} \left\{ \min_x f(x, p) \right\}.$$
2. This lower bound and upper bound can be made sharper by subdividing the parameter intervals $\boldsymbol{p}$, solving the subproblems, then taking the union of the solutions. Subdividing can also decrease the overall execution time.
3. Details can be found in a preprint at `http://interval.usl.edu/preprints/TOMS_thick.ps`

Maintenance Scheduling Application

GlobSol's Performance

1. GlobSol solves an 8-component model with fixed c_i very effectively.
2. If the c_i are intervals, GlobSol gives satisfactory results if only one of the c_i is assumed to be uncertain, and if the variation is within the range of 10% or so.
3. GlobSol presently takes excessive time when more than one c_i is assumed to be uncertain.
4. Certain algorithmic improvements appear promising for increasing the practicality of GlobSol on this problem when more than one parameter is uncertain.

GlobSol References

- For the source, installation instructions, user guide, etc.:
  `http://www.mscs.mu.edu/~globsol/`
- *Rigorous Global Search: Continuous Problems*, R. B. Kearfott, Kluwer Academic Publishers, 1996. Contains
  - Most of the basic ideas underlying GlobSol.
  - Structure of the research code that eventually became GlobSol.
- For various preprints related to techniques in GlobSol and applications:
  `http://interval.louisiana.edu/preprints.html`
- For these transparencies:
  `http://interval.louisiana.edu/preprints/2000_Dresden.ps`
  (Postscript)
  `http://interval.louisiana.edu/preprints/2000_Dresden.dvi`
  (TEX DVI)