On Stopping Criteria in Verified Nonlinear Systems or Optimization Algorithms

R. B. KEARFOTT
University of Louisiana at Lafayette
and
G. W. WALSTER
Sun Microsystems, Inc.

Traditionally, iterative methods for nonlinear systems use heuristic domain and range stopping criteria to determine when accuracy tolerances have been met. However, such heuristics can cause stopping at points far from actual solutions, and can be unreliable due to the effects of round-off error or inaccuracies in data. In verified computations, rigorous determination of when a set of bounds has met a tolerance can be done analogously to the traditional approximate setting. Nonetheless, the range tolerance possibly cannot be met. If the criteria are used to determine when to stop subdivision of n-dimensional bounds into subregions, then failure of a range tolerance results in excessive, unnecessary subdivision, and could make the algorithm impractical. On the other hand, interval techniques can detect when inaccuracies or round-off will not permit residual bounds to be narrowed. These techniques can be incorporated into *range thickness* stopping criteria that complement the range stopping criteria. In this note, the issue is first introduced and illustrated with a simple example. The thickness stopping criterion is then formally introduced and analyzed. Third, inclusion of the criterion within a general verified global optimization algorithm is studied. An industrial example is presented. Finally, consequences and implications are discussed.

Categories and Subject Descriptors: G.1.5 [**Numerical Analysis**]: Roots of Nonlinear Equations—*Error analysis*; *Iterative methods*; *Systems of equations*; G.1.6 [**Numerical Analysis**]: Optimization—*Unconstrained optimization*; *Constrained optimization*; *Global optimization*

General Terms: Algorithms, Performance, Reliability, Verification

Additional Key Words and Phrases: Stopping criteria, verified computations, GlobSol

This work was partially funded by National Science Foundation grant DMS-9701540 and by a Sun Microsystems Cooperative Research grant.
Authors' addresses: R. B. Kearfott, Department of Mathematics, University of Louisiana at Lafayette, Box 4-1010, Lafayette, LA 70504-1010; email: rbk@louisiana.edu; G. W. Walster, MTV12-40, Sun Microsystems, Inc., 2550 Garcia Avenue, Mountain View, CA 94043; email: Bill.Walster@Eng.Sun.COM.

1. INTRODUCTION: BACKGROUND AND SIMPLE EXAMPLES

In a general implementation of Newton's method, the steepest descent method, or a hybrid method to compute solutions of $f(x) = 0$, where $f : \mathbb{R}^n \to \mathbb{R}^n$, a *domain tolerance* of a form similar to

$$\delta_k = \frac{\|x^{(k+1)} - x^{(k)}\|}{\max\{\|x^{(k+1)}\|, 1\}} < \delta \text{ (point domain tolerance condition)} \tag{1}$$

is used, for some domain stopping tolerance δ. The well-known idea is that if $x^{(*)}$ solves $f(x^{(*)}) = 0$ exactly, then

$$\|x^{(k+1)} - x^{(*)}\| \sim C\|x^{(k)} - x^{(*)}\|^2 \tag{2}$$

for some constant C and for $x^{(k)}$ sufficiently close to $x^{(*)}$. The quadratic convergence assumption represented in (2) combined with the triangle inequality leads to

$$\|x^{(k+1)} - x^{(*)}\| \leq\cdot C\left(\frac{\|x^{(k+1)} - x^{(k)}\|}{1 - C\|x^{(k)} - x^{(*)}\|}\right)^2 \leq 2C\|x^{(k+1)} - x^{(k)}\|^2 \tag{3}$$

for $x^{(k)}$ sufficiently close to $x^{(*)}$. The approximation (3) justifies the domain heuristic (1).

Similarly, f may be flat (with unknown flatness) near a solution $x^{(*)}$, or else merely a point $x^{(k)}$ with a small residual $\|f(x^{(k)})\|$ is desired. In such instances, the following *range tolerance* is appropriate,

$$\rho_k = \|f(x^{(k+1)})\| \leq \rho \text{ (point range tolerance condition)} \tag{4}$$

for some range tolerance ρ.

Example 1. Consider

$$f(x) = x^2 - 2,$$

with $x \in [1, 2]$ and $\check{x} = 1.5$.

Newton's method for Example 1 converges rapidly for any starting point $x^{(0)} > \sqrt{2}$, and in particular for $x^{(0)} = \check{x} = 1.5$. Since $f'(x^{(*)}) = 2\sqrt{2}$ is nonsingular, the domain and range tolerances are roughly equivalent, and the iterations in Table I are obtained. Here, IEEE arithmetic was used. In lines with errors of 0, the machine-representable approximations being subtracted were identical.

It is seen in Table I that the domain tolerance (1) and the range tolerance (4) reflect actual error behavior well until the errors become small enough for round-off error to dominate the computation. For this example, a serious problem could occur only if δ were set to be $\mathcal{O}(\epsilon_m^2)$, or ρ were set to be $\mathcal{O}(\epsilon_m)$, where ϵ_m is the *machine epsilon*, defined to be the smallest

Table I. Stopping Tolerance Behavior for the Point Newton Method with Function $x^2 - 2$

k	$x^{(k+1)}$	δ_k^2	ρ_k	$\|x^{(k+1)} - x^{(*)}\|$
0	1.50000000000000			
1	1.41666666666667	3.46×10^{-3}	6.94×10^{-3}	2.45×10^{-3}
2	1.41421568627451	3.00×10^{-6}	6.01×10^{-6}	2.12×10^{-6}
3	1.41421356237469	2.26×10^{-12}	4.51×10^{-12}	1.59×10^{-12}
4	1.41421356237310	1.27×10^{-24}	4.44×10^{-16}	0
5	1.41421356237309	2.47×10^{-32}	4.44×10^{-16}	2.22×10^{-16}
6	1.41421356237310	2.47×10^{-32}	4.44×10^{-16}	0

number that, when added to one, gives a number different from one. (For double-precision IEEE arithmetic machines, that number is $\epsilon_m \approx 2.220446049250313 \times 10^{-16}$.)

However, uncertainties in data or computational errors can lead to loss of accuracy orders of magnitude greater than the machine epsilon. Under these conditions, the maximum attainable accuracy in a Newton iteration can be difficult to predict with floating-point iterations. In such instances, the domain and range tolerances will not terminate the algorithm, and behavior as in iterations 4 through 6 in Table I will occur, although the iteration may be more erratic for more complicated functions and with more than one variable, and the actual magnitudes of the values δ_k^2 and ρ_k may not be predictable.

Such uncertainties can be automatically handled with multidimensional interval Newton methods, as described in the texts on interval computations, such as Moore [1979], Alefeld and Herzberger [1983], Neumaier [1990], Hansen [1992], and Kearfott [1996b]. In one dimension, interval Newton methods are of the form

$$\mathbf{x}^{(k+1)} = \left\{ \check{x}^{(k)} - \frac{f(\check{x}^{(k)})}{\mathbf{f}'(\mathbf{x}^{(k)})} \right\} \cap \mathbf{x}^{(k)}. \tag{5}$$

(Here, interval quantities are denoted by boldface, so that $\mathbf{x}^{(k)}$ is an interval; $\check{x}^{(k)}$ is a representative point, often taken to be the midpoint of $\mathbf{x}^{(k)}$; and $\mathbf{f}'(\mathbf{x}^{(k)})$ is an interval extension of the derivative of f over $\mathbf{x}^{(k)}$.) A domain tolerance condition corresponding to (1) is

$$\bar{\delta}_k = \max_{1 \le i \le n} \left\{ \frac{\mathrm{w}(\mathbf{x}_i^{(k)})}{\max\{|\mathbf{x}_i^{(k)}|, 1\}} \right\} < \delta \text{ (interval domain tolerance condition)}, \tag{6}$$

where $\mathrm{w}(\mathbf{x}_i^{(k)})$ represents the width of the ith coordinate interval. A range tolerance condition corresponding to (4) is

$$0 \in \mathbf{f}_i(\mathbf{x}^{(k)}) \text{ and } \mathrm{w}(\mathbf{f}_i(\mathbf{x}^{(k)})) < \rho, \ 1 \le i \le n,$$

or, equivalently,

Table II. Stopping Tolerance Behavior for the Interval Newton Method with Function $x^2 - 2$

k	$\mathbf{x}^{(k+1)}$	$\bar{\delta}_k$	$\bar{\rho}_k$
0	[1.00000000000000, 2.00000000000000]	5.00×10^{-1}	2.00×10^{0}
1	[1.37499999999999, 1.43750000000001]	4.35×10^{-2}	1.09×10^{-1}
2	[1.41406249999999, 1.41441761363637]	2.51×10^{-4}	5.77×10^{-4}
3	[1.41421355929452, 1.41421356594718]	4.70×10^{-9}	1.01×10^{-8}
4	[1.41421356237309, 1.41421356237310]	4.71×10^{-16}	1.33×10^{-15}
5	[1.41421356237309, 1.41421356237310]	4.71×10^{-16}	1.33×10^{-15}
6	[1.41421356237309, 1.41421356237310]	4.71×10^{-16}	1.33×10^{-15}

$$\bar{\rho}_k = \max_{1 \le i \le n} |\mathbf{f}_i(\mathbf{x}^{(k)})| < \rho \text{ (interval range tolerance condition)}$$

Application of the interval Newton method (5) to Example 1 with starting bounds $\mathbf{x}^{(0)} = [1, 2]$ gives Table II. In Table II, the library INTLIB [Kearfott et al. 1994] and Fortran 90 module `INTERVAL_ARITHMETIC` [Kearfott 1996a] were used for simulated directed roundings; the intervals in the table were rounded outward for display. The actual iterates became exactly stationary at iteration number 4.

Analogously to the floating-point case, $\bar{\delta}_k$ and $\bar{\rho}_k$ are reasonable quantities to check for stopping tolerances, except that the interval versions represent mathematically rigorous domain and range error bounds, rather than heuristic values. However, as in the floating-point case, the tolerance conditions (6) and (7) possibly cannot be met if δ or ρ are set too small. With inclusion-monotonic interval arithmetic, the interval Newton iteration (5) must eventually become stationary. However, multidimensional versions of (5) often become stationary far before reaching an accurate enclosure for a solution, and many iterations may be spent with only small progress toward the limiting endpoints.

Example 2. Take

$$f_1(x_1, x_2) = x_1^3 - x_2$$

$$f_2(x_1, x_2) = x_2^3 + x_1,$$

with $\mathbf{x}_1^{(0)} = \mathbf{x}_2^{(0)} = [-1, 1]$.

In Example 2, the interval extension of the Jacobi matrix over $\mathbf{x}^{(0)}$ is

$$\begin{pmatrix} [0, 3] & -1 \\ 1 & [0, 3] \end{pmatrix},$$

and it does not contain any singular matrices. Furthermore, the system resulting from applying the inverse midpoint preconditioner also does not contain any singular matrices, and there is a unique solution within $\mathbf{x}^{(0)}$. However, both the interval Gauss-Seidel method and interval Gaussian

elimination are stationary at $\mathbf{x}^{(0)}$. In such instances, subdividing the initial bounds and applying interval techniques to subregions lead to success. In this case, using a starting box of $\mathbf{x}_1^{(0)} = \mathbf{x}_2^{(0)} = [-0.2, 0.1]$ (and perhaps larger starting boxes) results in convergence to narrow bounds around the true solution $x^{(*)} = (0, 0)$, while other subregions, such as say $\mathbf{x}_1 = [0.1, 1]$, $\mathbf{x}_2 = [0.1, 1]$, can be rejected with other interval techniques. (For an explanation of these terms and algorithms, see Kearfott [1996b].)

On the other hand, the interval Newton method may become stationary at some $\mathbf{x}^{(k)}$ due to round-out error (downward rounding for the left endpoint and upward rounding for the right endpoint), as in Table II, or due to uncertainty in the data. In such cases, it is not appropriate to subdivide $\mathbf{x}^{(k)}$ further. To illustrate the situation where uncertainties in the data lead to questions about stopping tolerances, consider the following.

Example 3. Suppose the task is to

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

Here, the intervals $[1, 2]$ and $[3, 4]$ represent uncertainty in the problem specification. Subject to this uncertainty, find maximum possible and minimum possible values of x_1 and x_2.

If an interval Newton method starting with initial coordinate bounds

$$x^{(0)} = ([-10, 10], [-10, 10])$$

is applied to the gradient system

$$\nabla \phi = (x_1 - [1, 2], x_2 - [3, 4])^T = (0, 0)^T,$$

then (1) the first iteration of an interval Gauss-Seidel method gives $\mathbf{x}^{(1)} = ([1, 2], [3, 4])^T$ (to within round-out error) and (2) all subsequent iterates are stationary at $\mathbf{x}^{(k)} = \mathbf{x}^{(1)}$, $k \geq 1$. Thus, $\overline{\delta}_k = 1/2$ and $\overline{\rho}_k = 2$ for $k \geq 1$. In some algorithms, lack of tolerance satisfaction causes the box $([1, 2], [3, 4])^T$ to be bisected, say, into the boxes $([1, 1.5], [3, 4])^T$ and $([1.5, 2], [3, 4])^T$. However, since the box $([1, 2], [3, 4])$ represents the sharpest possible bounds on the answer in this case, such bisection is not appropriate.

Verified global optimization algorithms also sometimes use the width of the interval extension of the objective function to determine when a box should be accepted as small enough. With this criterion, the width is

$$\begin{aligned} \mathrm{w}(\phi(\mathbf{x}^{(1)})) &= \mathrm{w}(([1, 2] - [1, 2])^2 + ([3, 4] - [3, 4]^2)) \\ &= \mathrm{w}([0, 1] + [0, 1]) = 2. \end{aligned}$$

This value cannot be made small.

Thus, stopping due to domain or range tolerance conditions, or stopping when the interval Newton method becomes stationary, is not sufficient. A stopping criterion that determines when further subdivision will not enable convergence will be useful.

2. THE THICKNESS STOPPING CRITERION AND CONSEQUENCES OF ITS USE

In rigorous global optimization algorithms with objective function $\phi(x)$, progress is made either by comparing bounds for the objective function over a region (interval vector) $\mathbf{x}^{(0)}$ or by successful iteration of an interval Newton method applied to $\nabla \phi(x) = 0$, with $\mathbf{x}^{(0)}$ as initial iterate.[1] Subdivision of $x^{(0)}$ is done either to enable sharper bounds on the objective ϕ or to enable the interval Newton method to converge to narrower bounds, and such subdivision leads to unnecessary inefficiency if it does not accomplish either of these things. The thickness stopping criterion explained here detects when further bisection will not lead to sharper bounds on the objective function, and can also be applied similarly to constraints or gradients.

Applied to the objective function ϕ, the criterion is simply

$$\mathrm{w}(\phi(\check{x}^{(k)})) \geq \eta_\phi \mathrm{w}(\phi(\mathbf{x}^{(k)})) \text{ (objective thickness condition)} \tag{8}$$

where $\eta_\phi \in [0, 1]$ is termed the "objective function thickness factor." Smaller values of η_ϕ give a looser tolerance, but also allow more overestimation in the range of ϕ.

In Example 3 with $\mathbf{x}^{(k)} = ([1, 2], [3, 4])^T$ and $\check{x}^{(k)} = (1.5, 3.5)^T$,

$$\mathrm{w}(\phi(\check{x}^{(k)})) = \mathrm{w}([-0.5, 0.5]^2 + [-0.5, 0.5]^2) = 0.5$$

and

$$\mathrm{w}(\phi(\mathbf{x}^{(k)})) = 2.$$

Thus, the optimally small box $\mathbf{x}^{(k)} = ([1, 2], [3, 4])^T$ will be accepted without further subdivision if η_ϕ is chosen so $\eta_\phi < 0.25$.

The following observations aid in determining appropriate values of η_ϕ.

THEOREM 1 (UPPER BOUND ON THE RANGE OVERESTIMATION). *Suppose that the objective thickness condition (8) is satisfied for some* $\eta_\phi \in (0, 1)$. *Let* $\phi^\sharp(\mathbf{x}^{(k)})$ *represent the actual range of* ϕ *over* $\mathbf{x}^{(k)}$. *Then the range overestimation satisfies*

[1]In nonlinear systems algorithms, just a nonlinear system is considered, and in constrained algorithms, both the constraints by themselves and the Lagrange multiplier or Fritz-John system are considered. This will be discussed in the next section.

$$\mathrm{w}(\phi^{\sharp}(\mathbf{x}^{(k)})) - \mathrm{w}(\phi(\mathbf{x}^{(k)})) \leq \frac{1 - \eta_\phi}{\eta_\phi}\mathrm{w}(\phi(\check{x}^{(k)})).$$

PROOF.

$$\begin{aligned}\mathrm{w}(\phi^{\sharp}(\mathbf{x}^{(k)})) - \mathrm{w}(\phi(\mathbf{x}^{(k)})) &\leq \mathrm{w}(\phi(\mathbf{x}^{(k)})) - \mathrm{w}(\phi(\check{x}^{(k)})) \\ &\leq \frac{\mathrm{w}(\phi(\check{x}^{(k)}))}{\eta_\phi} - \mathrm{w}(\phi(\check{x}^{(k)})) \\ &= \frac{1 - \eta_\phi}{\eta_\phi}\mathrm{w}(\phi(\check{x}^{(k)}))\end{aligned}$$

Furthermore, for constant $\phi(\mathbf{x}) \equiv [a, b]$, $\mathrm{w}(\phi(\check{x}^{(k)})) = \mathrm{w}(\phi(\mathbf{x}^{(k)})) = \mathrm{w}(\phi^{\sharp}(\mathbf{x}^{(k)})) = b - a$, and the first inequality becomes an equation. □

Theorem 1 gives guidance on how near η_ϕ should be to 1 to ensure a given accuracy in the range of the objective function and, hence, a given accuracy in the global optimum. Conversely, η_ϕ should be chosen closer to 0 for the algorithm to be more efficient. It has been seen that, in Example 3, choosing $\eta_\phi < 0.25$ will totally prevent unnecessary subdivision. Using a larger value of η_ϕ for this example will result in some extra subdivision, but it will not be excessive for η_ϕ around, say, 0.5. In general, an appropriate value of η_ϕ to avoid unnecessary subdivision depends both on the amount of interval uncertainty in the objective and specific characteristics of the interval evaluation. (The width overestimation can, in principle, be arbitrarily large.) When the objective ϕ is an uncertain constant, any $\eta_\phi \in [0, 1]$ will do. Otherwise, the value $\eta_\phi = 0.5$ seems to work well in practice.

3. STOPPING CRITERIA IN GLOBAL OPTIMIZATION ALGORITHMS

Philosophy and details of incorporation into actual global optimization algorithms are considered here.

3.1 "Thick" Parameters: What is a Solution?

Considering Example 3, one may ask why the uncertain constants $[1, 2]$ and $[3, 4]$ should not be considered as variables, i.e., why not consider the four-dimensional bound-constrained problem

$$\text{minimize } \phi(x) = \frac{1}{2}\{(x_1 - x_3)^2 + (x_2 - x_4)^2\},\ x_3 \in [1, 2],\ x_4 \in [3, 4] \quad (9)$$

instead? In fact, in this particular example, the GlobSol package [Corliss 1998], starting with $x_1 \in [-10, 10]$, $x_2 \in [-10, 10]$, considers 21 boxes

to give an answer of four boxes whose union is the box $\mathbf{x}^{(*)} = ([1, 2], [3, 4], [1, 2], [3, 4])^T$, exactly the solution that is obtained in the two variable problem with uncertain constants. But this is due only to the special form of Example 3.

Example 4. Minimize

$$\phi(x) = (x - a)^2 + 10a^2$$

with

$$a \in [-1, 1], \text{ for } x \in [-10, 10].$$

With $\eta_\phi = 0.2$, GlobSol processes exactly two subintervals to obtain a solution list consisting of two subintervals whose union is the exact solution set $\mathbf{x}^{(*)} = [-1, 1]$. However, if a is considered as a variable, i.e., if GlobSol solves the problem

$$\text{minimize } \phi(x) = (x_1 - x_2)^2 + 10x_2^2 \text{ for } x \in ([-10, 10], [-1, 1])^T, \quad (10)$$

then GlobSol processes a total of two boxes, and returns a small box centered on the unique solution $(0, 0)$, a very different solution from that to the original formulation in Example 4. The solution set for Example 4 contains the set of *all* solutions to the univariate problem for a any fixed point in $[1, 2]$, while the solution to the variant in (10) contains only the solution for that a which gives the minimum value of ϕ. One can think of the interval value of ϕ, $[0, 14]$ in the case of GlobSol's solution to Example 4 with $\eta_\phi = 0.2$, as bounds on the minimum value of the minimum and the maximum value of the minimum, with some overestimation if the constant appears in more than one place. (In Example 4, the actual range of ϕ over the minimum values is $[0, 10]$. See Section 3.4 below.)

Efficiency considerations may also play a role in how the solution set is viewed.

Example 5. Minimize

$$\phi(x) = a(x_1 - a)^2 + b(x_2 - b)^2$$

with

$$a \in [1, 2], b \in [3, 4], \text{ for } x_1 \in [-10, 10] \text{ and } x_2 \in [-10, 10].$$

With $\eta_\phi = 0.2$, GlobSol processed a total of four boxes, and, with $\eta_\phi = 0.5$, GlobSol processed a total of 10 boxes. GlobSol was not able to complete without considering more than 20,000 subregions when $\eta_\phi = 0.9$. In both of the successful cases, the union of the output boxes consisted of the set

$$\{(x_1, x_2) \mid x_1 \in [0.8408, 2.1592], x_2 \in [2.8333, 4.1667]\}$$

to within round-out. (Note that this is an overestimate of the actual solution set $\mathbf{x}^{(*)} = ([1, 2], [3, 4])^T$.) If Example 5 is reformulated as

$$\begin{aligned} &\text{minimize } \phi(x) = x_3(x_1 - x_3)^2 + x_4(x_2 - x_4)^2 \\ &\text{with } x_3 \in [1, 2], x_4 \in [3, 4], \\ &\text{for } x_1 \in [-10, 10] \text{ and } x_2 \in [-10, 10], \end{aligned} \tag{11}$$

then GlobSol was unable to complete a search in an hour of processor time, corresponding to 89,954 boxes processed. Although Problem (11) is similar to Problem (9) over the region in question, both problems have a two-dimensional set of solutions in $\mathbb{R}^4$. The small but significant overestimation in Problem (11), due to x_3 and x_4 appearing more than once, is enough to cause GlobSol to subdivide the region into too many small subboxes surrounding the solution set.

Recapitulating:

—The solution set for a problem with thick constants is different from the solution set of the corresponding problem with the constants considered as variables.

—The solution set to a problem with nonzero width ("thick") constants contains information about the solution to a kind of minimax problem. The corresponding interval values of the objective function approximate the minimum minimum and maximum minimum and are, in general, better approximations when the widths of the constants are smaller.

—Combined with the stopping criteria described here, formulating the problem with thick constants can be more efficient than formulating the problem with variables only.

3.2 Structuring the Stopping Criteria

Global optimization problems are generally of the form

$$\begin{aligned} &\text{minimize } \phi(x) \\ &\text{subject to } c(x) = 0, g(x) \leq 0. \end{aligned} \tag{12}$$

Here $\phi : \mathbf{x} \subset \mathbb{R}^n \to \mathbb{R}$, $c : \mathbf{x} \to \mathbb{R}^{m_1}$, and $g : \mathbf{x} \to \mathbb{R}^{m_2}$, where $\mathbf{x}$ is an interval vector $\mathbf{x} = ([\underline{x}_1, \overline{x}_1], \ldots, [\underline{x}_n, \overline{x}_n])^T$. In a step of a global optimization code such as GlobSol, a *current box* $\mathbf{x}^{(k)}$ is processed, roughly with the following steps.

Algorithm 1 (General Steps in Verified Global Search).

(1) The stopping tolerances are checked.

(2) Constraint propagation techniques are applied to narrow the coordinate bounds $\mathbf{x}_i^{(k)}$, $1 \le i \le n$. (See Kearfott [1996b] and the references therein.)

(3) It is checked whether the lower bound $\underline{\phi}(\mathbf{x}^{(k)}) > \bar{f}$, where $\bar{f}$ is a previously computed rigorous upper bound on the global optimum. If it is then reject $\mathbf{x}^{(k)}$, and proceed to step (9).

(4) If consideration of $\mathbf{x}^{(k)}$ leads to a smaller value of $\bar{f}$, then step (3) is applied with the boxes $\mathbf{x}^{(*)}$ from the list of previously stored boxes.

(5) It is checked whether $0 \in \mathbf{c}_i(\mathbf{x}^{(k)})$, $1 \le i \le m_1$ and whether $\overline{\mathbf{g}}_i \le 0$, $1 \le i \le m_2$.

(6) It is checked whether $0 \in L$, where L is a generalized Lagrange multiplier system or Fritz-John system corresponding to Problem (12).

(7) An interval Newton method is applied to the generalized Lagrange system.

(8) The stopping tolerances are checked again, and the box is either discarded, stored on a solution list, or subdivided, so the components can be stored for further processing.

(9) A new $\mathbf{x}^{(k)}$ is selected from the list of boxes yet to be processed. If there is no such $\mathbf{x}^{(k)}$, then the algorithm exits.

End Algorithm 1

In steps (3) and (5), interval values of the objective, equality, and inequality constraints are used, while interval extensions of their gradients are needed in step (6). The sharpness of these interval bounds can possibly be improved through subdivision. Thus, application of the stopping criteria should take account of the objective ϕ, the equality constraints c, the inequality constraints g, and their gradients.

Different problems may benefit from different sharpnesses with which the objective and two kinds of constraints are enclosed. Also, it may be unknown what the exact effects of round-out error or thick tolerances are on the minimum achievable widths in the ranges of the objective and two kinds of constraints. Thus, a rational way of combining range tolerances, thickness conditions, gradient range tolerances, and domain tolerances is needed.

The range stopping test is structured into three parallel parts, corresponding to the objective, the equality constraints, and the inequality constraints. For simplicity, the internal structure will be explained for the objective; the internal structure of the other two is similar.

Algorithm 2 (Range and Thickness Criteria for the Objective Function).

INPUT : The objective range width tolerance ρ_ϕ and the objective thickness factor η_ϕ, the present box $\mathbf{x}^{(k)}$, its midpoint $\check{x}^{(k)}$, and the machine epsilon ϵ_m.

OUTPUT : Either "$\mathbf{x}^{(k)}$ meets the criteria" or "$\mathbf{x}^{(k)}$ needs to be subdivided further."

if $\mathrm{w}(\phi(\mathbf{x}^{(k)})) < \max\{\rho_\phi, 100\epsilon_{\mathrm{m}}\}$ **or** $\mathrm{w}(\phi(\check{x}^{(k)}) > \eta_\phi \mathrm{w}(\phi(\mathbf{x}^{(k)})))$

then signal "$\mathbf{x}^{(k)}$ meets the criteria,"

else signal "$\mathbf{x}^{(k)}$ needs to be subdivided further."

end if

End Algorithm 2

If $\phi(\mathbf{x}^{(k)})$ is thick, it cannot be made narrow, since $\mathrm{w}(\phi(\mathbf{x}^{(k)})) \geq \mathrm{w}(\phi(\check{x}^{(k)}))$. Conversely, if $\phi(\mathbf{x}^{(k)})$ is narrow to within tolerance ρ_ϕ, further subdivision will not improve the chances that $\underline{\phi}(\mathbf{x}^{(k)}) > \bar{f}$ (step (1) described above in the global optimization algorithm process) can be verified. Thus, disjunction as illustrated, rather than conjunction or a user-controlled choice, is appropriate in the conditional in Algorithm 2.

In the complete stopping test, the analogue of Algorithm 2 is applied to the constraints c and the inequality constraints g, in addition to the objective ϕ. Since the importance of accuracy in the objective and constraints, as well as the efficiency with which such accuracy can be achieved, varies from problem to problem, user control of the importance of each of conditions on the objective and two types of constraints is useful. This results in the following scheme.

Algorithm 3 (Combined Range Width and Thickness Conditions).

INPUT : (1) flags $\mathcal{F}_\phi$, $\mathcal{F}_c$, and $\mathcal{F}_g$ indicating whether it is necessary for the range or width condition on the objective, equality, or inequality constraints, respectively, to be satisfied for convergence to be signalled,

(2) the width tolerances ρ_ϕ, ρ_c, and ρ_g for the objective, equality, and inequality constraints, respectively,

(3) the thickness factors η_ϕ, η_c, and η_g for the objective, equality, and inequality constraints, respectively,

(4) the present box $\mathbf{x}^{(k)}$, its midpoint $\check{x}^{(k)}$, and the machine epsilon ϵ_{m}.

OUTPUT : Either "$\mathbf{x}^{(k)}$ meets the criteria" or "$\mathbf{x}^{(k)}$ needs to be subdivided further."

(1) Define

(a) $\mathcal{P}$ = "true" if and only if Algorithm 2 returns "$\mathbf{x}^{(k)}$ meets the criteria."
(b) $\mathcal{C}$ = "true" if and only if Algorithm 2, applied with ρ_c replacing ρ_ϕ and with *each* c_i, $1 \le i \le m_1$, replacing ϕ, returns "$\mathbf{x}^{(k)}$ meets the criteria."
(c) $\mathcal{G}$ = "true" if and only if Algorithm 2, applied with ρ_g replacing ρ_ϕ and with *each* g_i, $1 \le i \le m_2$, replacing ϕ, returns "$\mathbf{x}^{(k)}$ meets the criteria."

(2) **if** $\{\{(\mathcal{F}_\phi \vee \mathcal{F}_c \vee \mathcal{F}_g)\} \wedge (\mathcal{P} \vee \neg\mathcal{F}_\phi) \wedge (\mathcal{C} \vee \neg\mathcal{F}_c) \wedge (\mathcal{G} \vee \neg\mathcal{F}_g)\}$
$\vee \{\{\neg(\mathcal{F}_\phi \vee \mathcal{F}_c \vee \mathcal{F}_g)\} \wedge (\mathcal{P} \vee \mathcal{C} \vee \mathcal{G})\}$
then signal "$\mathbf{x}^{(k)}$ meets the criteria,"
else signal "$\mathbf{x}^{(k)}$ needs to be subdivided further."

end if

End Algorithm 3

In the conditional in Algorithm 3, the user-set flags $\mathcal{F}_\phi$, $\mathcal{F}_c$, and $\mathcal{F}_g$ define whether or not it is necessary for the objective, the equality constraints, or the inequality constraints, respectively, to meet the width or thickness criterion, unless all three flags are set to "false." If all three flags are set to "false," then the box is considered subdivided enough if at least one of the width/thickness criteria are met. This scheme, although not the only one, provides substantial flexibility with a small number of parameters. The second line of logic in step (2), handling the case when all three flags are "false," prevents vacuously signalling "$\mathbf{x}^{(k)}$ meets the criteria," and allows stopping the subdivision if *any* of the criteria is met. In this case, i.e., when $(\neg\mathcal{F}_\phi \wedge \neg\mathcal{F}_c \wedge \neg\mathcal{F}_g)$, individual tolerances can be turned off completely by adjusting the width tolerances and the thickness factors. For example, in the case $(\neg\mathcal{F}_\phi \wedge \neg\mathcal{F}_c \wedge \neg\mathcal{F}_g)$, $\mathcal{P}$ can be turned off by setting $\rho_\phi = 0$ and $\eta_\phi = 1$.

Algorithm 3 includes all conditions on the ranges of the objective and constraints, but does not include either the domain conditions or the condition on the gradient. Also, it is useful to include the domain condition (6) for cases for which the domain tolerance δ is easier to specify than the range width tolerances and range thickness factors. Finally, a condition to ensure that the gradient of the Lagrange function is narrow is useful. This condition, which can also be viewed as a scaled domain condition, has been termed *maximum smear* in Kearfott [1996b]. The condition, justified in Ratz and Csendes [1995] and elsewhere, consists of the following:

$$\sigma_i < \delta\sigma,\ 1 \le i \le n, \text{ where}$$

$$\sigma_i = \left|\frac{\partial\phi}{\partial\mathbf{x_i}}(\mathbf{x}^{(k)})\right| + \sum_{j=1}^{m_1}\left|\frac{\partial\mathbf{c_j}}{\partial\mathbf{x_i}}(\mathbf{x}^{(k)})\right| + \sum_{j=1}^{m_2}\left|\frac{\partial\mathbf{g_j}}{\partial\mathbf{x_i}}(\mathbf{x}^{(k)})\right| \text{ and where}$$

$$\sigma = \max_{1 \le i \le n} \sigma_i. \tag{13}$$

Here, $|[a, b]| = \max\{|a|, |b|\}$. In the sums for σ_i, it would be advantageous for the partial derivatives of the c_j and g_j to be multiplied by corresponding generalized Lagrange multipliers. However, such multipliers are in general not known when the stopping test is applied.

In GlobSol, the range/thickness test, scaled domain tolerance (6), and maximum smear (13) are combined alternatively. The reasoning is that it is hard to predict, in general, which conditions are easily satisfiable, and alternative satisfaction causes subdivision to stop under the weakest possible conditions, thus avoiding impractical amounts of subdivision. (If the answers given by such an algorithm are not narrow enough, it is relatively easy to refine them.) This alternative formulation results in the following overall stopping criterion implemented in GlobSol.

Algorithm 4 (Overall Stopping Criterion).

INPUT : (1) the domain tolerance δ,

(2) flags $\mathcal{F}_\phi$, $\mathcal{F}_c$, and $\mathcal{F}_g$ indicating whether it is necessary for the range or width condition on the objective, equality, or inequality constraints, respectively, to be satisfied for convergence to be signalled,

(3) the width tolerances ρ_ϕ, ρ_c, and ρ_g for the objective, equality, and inequality constraints, respectively,

(4) the thickness factors η_ϕ, η_c, and η_g for the objective, equality, and inequality constraints, respectively,

(5) the present box $\mathbf{x}^{(k)}$, its midpoint $\check{x}^{(k)}$, and the machine epsilon ϵ_m.

OUTPUT : Either "$\mathbf{x}^{(k)}$ meets the criteria" or "$\mathbf{x}^{(k)}$ needs to be subdivided further."

if the scaled diameter is small according to (6),

then signal "$\mathbf{x}^{(k)}$ meets the criteria," and **return**

else if the range conditions (Algorithm 3) are satisfied,

then signal "$\mathbf{x}^{(k)}$ meets the criteria," and **return**

else if the smear diameter according to (13) is satisfied,

then signal "$\mathbf{x}^{(k)}$ meets the criteria," and **return**

else

signal "$\mathbf{x}^{(k)}$ needs to be subdivided further."

end if

End Algorithm 4

3.3 Which Coordinate Should Be Subdivided?

A simple and common way of subdividing a box $\mathbf{x}^{(k)}$ is to bisect a coordinate to form $\underline{\mathbf{x}}^{(k)}$ and $\overline{\mathbf{x}}^{(k)}$. For example, if $\mathbf{x}^{(k)} = ([-1, 1], [-2, 2])^T$, then the second coordinate may be bisected to form $\underline{\mathbf{x}}^{(k)} = ([-1, 1], [-2, 0])^T$ and $\overline{\mathbf{x}}^{(k)} = ([-1, 1], [0, 2])^T$, or the first coordinate may be bisected to form $\underline{\mathbf{x}}^{(k)} = ([-1, 0], [-2, 2])^T$ and $\overline{\mathbf{x}}^{(k)} = ([0, 1], [-2, 2])^T$. Ideally, the coordinate to be bisected will be chosen to maximize the chance that the overall stopping criterion as defined by Algorithm 4 will be satisfied by the resulting boxes. A good heuristic for this is to order the coordinates according to the smears σ_i of (13), after removing those terms for which ϕ, $\mathbf{c}_j$, or $\mathbf{g}_j$ is thick or narrow according to Algorithm 2. This is what is presently done in GlobSol.

3.4 Is it Advantageous to Subdivide Thick Constants?

In Example 5, due to interval dependence in the constants a and b, the estimate for ϕ was an overestimate of the actual range. Subdividing the constants, such as considering two problems, one with $a \in [1, 1.5]$ and one with $a \in [1.5, 2]$, will reduce the overestimation. (The amount by which the overestimation is reduced can be estimated by Theorem 1.) Furthermore, especially in more complicated problems, subdivision can reduce significantly the total amount of work involved. See Section 3.5 below.

There are differences between subdivision of thick constants and subdivision of variables. Typically, subdivision of the constants will not lead to rapid rejection of one half of the original region, as happens when subdividing a variable interval. Thus, subdividing a multidimensional array of thick constants potentially produces a very large number of subproblems to be considered. Furthermore, supplying different subintervals is logically equivalent to supplying a different objective function, and the midpoint test, i.e., step (4) of the general optimization scheme (Algorithm 1), will reject some boxes $\mathbf{x}^{(*)}$ that were stored when a previous subinterval was being processed. Similarly, the value of $\overline{f}$ from steps (3) and (4) is different for different subintervals. Thus, a more sophisticated algorithm would need to be supplied if thick constants were subdivided automatically.

For these reasons, GlobSol only allows for subdivision of a maximum of 10 thick constants, and an example driver is supplied that only subdivides one thick constant. Nonetheless, it is possible to treat constants as variables, as explained in Section 3.3, when deciding how to subdivide. Further implementation effort and research are required to determine how valuable this is.

To illustrate the advantages of subdivision of a constant, a one-dimensional version of Example 5 was tried:

$$\text{minimize } \phi(x) = a(x_1 - a)^2, \text{ with } a \in [1, 2], \text{ for } x_1 \in [-10, 10]. \tag{14}$$

With no subdivision, GlobSol gave two answer boxes whose union was enclosed in $\mathbf{x}^{(*)} = [0.9481, 1.4996]$, with objective value enclosed in $\phi(x^{(*)}) \in [-2.23 \times 10^{-308}, 2.213]$. Subdividing a into five pieces gave 10 solution boxes whose union was enclosed in $\mathbf{x}^{(*)} = [0.9992, 2.0004]$, with $\phi(x^{(*)}) \in [-2.23 \times 10^{-308}, 4.840 \times 10^{-2}]$.

3.5 An Industrial Example

In Rocco et al. [1999], GlobSol and the thick parameter scheme are used to solve a maintenance-scheduling problem. The first case treated there is a nine-variable mixed integer programming problem.

Example 6 (A Maintenance Scheduling Problem). Minimize

$$\left\{ s_0 + \sum_{i=1}^{8} s_i / x_{i+1} \right\} \Bigg/ x_1 + \left\{ \sum_{i=1}^{8} \left(\frac{\mathbf{c}_i}{l_i^{\beta_i}} \right) (x_1 x_{i+1})^{\beta_i - 1} \right\},$$

where

$$l = (8, 7, 9, 14, 6, 15, 3, 5)^T,$$

$$s = (100, 105, 225, 345, 165, 500, 345, 105, 345)^T,$$

$$\beta = (1.7, 1.7, 2.0, 2.0, 1.7, 2.0, 1.25, 1.75)^T,$$

the $\mathbf{c}_i$ are interval constants, the variables x_2 through x_9 are integers, and the solution is to be found within the bounds

$$\mathbf{x}^{(0)} = ([0.5, 20.5], [0.5, 2.5], [0.5, 2.5], [0.5, 4.5], [0.5, 4.5],$$

$$[0.5, 2.5], [1.5, 5.5], [0.5, 2.5], [0.5, 3.5])^T.$$

One experiment had

$$\mathbf{c} = (92, [163.8, 200.2], 28, 30, 172, 30, 90, 50),$$

i.e., only c_2 was an interval for this particular experiment.

The integer conditions on variables x_2 through x_9 were enforced by introducing equality constraints of the form

$$\sin(\pi x_i) = 0, \quad 2 \le i \le 9.$$

(This technique was first suggested in Hansen [1992].) As a consequence of the special form of these constraints, it was logical to set $\rho_c = 0.5$ (a value that assures accuracy to within the nearest integer). Also, η_ϕ and η_c were both set to 0.5.

It was not possible for GlobSol to complete this problem within several days of processor time without the thickness stopping conditions described

Table III.

	5 c_2 Subdivisions	10 c_2 Subdivisions	20 c_2 Subdivisions
Number of answer boxes	411	194	82
Number of boxes tried	18,994	28,996	50,073
CPU seconds	42,310	19,010	27,090
objective range	[317, 327]	[317, 326]	[317, 325]
$\mathbf{x}_1$ range	[10.9, 14.4]	[11.1, 14.3]	[11.1, 14.3]
$\mathbf{x}_2$ range	1	1	1
$\mathbf{x}_3$ range	1	1	1
$\mathbf{x}_4$ range	[2, 4]	[2, 3]	[2, 3]
$\mathbf{x}_5$ range	[2, 4]	[2, 4]	[2, 3]
$\mathbf{x}_6$ range	1	1	1
$\mathbf{x}_7$ range	[3, 5]	[3, 5]	[3, 5]
$\mathbf{x}_8$ range	1	1	1
$\mathbf{x}_9$ range	[1, 2]	[1, 2]	[1, 2]

above. It also took an inordinate amount of time when $\mathbf{c}_2$ was not subdivided. When $\mathbf{c}_2$ was subdivided into varying numbers of intervals, the results were as in Table III.

Also of interest in this problem is the range on the objective ϕ associated with particular integer values of x_2 through x_9. This information is also available from the GlobSol output boxes.

4. SUMMARY

A criterion for determining when to stop subdividing interval bounds into subregions has been introduced. This criterion is based on comparing the width of interval evaluations at points with interval evaluations over the entire bounds. The criterion enables verified global optimization algorithms to handle problems with significant uncertainty in the original data. Inclusion of the criterion into a verified global optimization code has been studied, and experiments on a significant problem have been carried out.

ACKNOWLEDGMENTS

We wish to thank Michael Saunders who, as editor, made many helpful style suggestions after a careful reading.

REFERENCES

ALEFELD, G. AND HERZBERGER, J. 1983. *Introduction to Interval Computation*. Academic Press, Inc., New York, NY.

CORLISS, G. F. 1998. Globsol entry page. http://www.mscs.mu.edu/~globsol/

HANSEN, E. R. 1992. *Global Optimization Using Interval Analysis*. Marcel Dekker, Inc., New York, NY.

KEARFOTT, R. B. 1996a. Algorithm 763: INTERVAL_ARITHMETIC: A Fortran 90 module for an interval data type. *ACM Trans. Math. Softw. 22*, 4, 385–392.

KEARFOTT, R. B. 1996b. *Rigorous Global Search: Continuous Problems*. Kluwer Academic, Dordrecht, Netherlands.

KEARFOTT, R. B., DAWANDE, M., DU, K., AND HU, C. 1994. Algorithm 737; INTLIB: a portable Fortran 77 interval standard-function library. *ACM Trans. Math. Softw. 20*, 4 (Dec.), 447–459.

MOORE, R. E. 1979. Methods and applications of interval analysis. In *Studies in Applied Mathematics*. SIAM, Philadelphia, PA.

NEUMAIER, A. 1990. *Interval Methods for Systems of Equations*. Cambridge University Press, New York, NY.

RATZ, D. AND CSENDES, T. 1995. On the selection of subdivision directions in interval branch-and-bound methods for global optimization. *J. Global Optim. 7*, 183–207.

ROCCO, C. M., MILLER, A. J., AND KEARFOTT, R. B. 1999. Uncertainty analysis of indirect-grouping maintenance strategy using an interval arithmetic approach. Preprint. Universidad Central de Venezuela, Miam, FL. Available via: POBA INTERNATIONAL No. 100, P.O. Box 02-5255, Universidad Central de Venezuela, Miami, FL 33102.

Received: July 1999; revised: March 2000; accepted: March 2000