Lectures in Applied Mathematics
Volume 26 (1990)

[redacted]

Interval Arithmetic Techniques in the Computational Solution of Nonlinear Systems of Equations: Introduction, Examples, and Comparisons

R. BAKER KEARFOTT

Abstract. Methods of interval arithmetic can be used to reliably find *all* solutions to nonlinear systems of equations and to reliably solve *global* optimization problems. Such methods can also be used in studies of the sensitivity of systems to certain parameters or in computing rigorous bounds on the range of behavior of a system as certain parameters vary. More generally, properly applied interval methods can give results that have *mathematical certainty*, since the effects of roundoff error are fully taken into account.

Here, for the nonexpert, we cite references and briefly review elementary aspects of interval arithmetic. We then describe a class of algorithms for finding all roots of nonlinear systems of equations within a box in n-space. Third, we discuss inherent differences between interval methods and alternate techniques for nonlinear systems. Finally, to illustrate the applicability of interval techniques, we show how interval arithmetic can be used to make the choice of predictor stepsize in continuation methods foolproof.

1. Introduction. Interval mathematics and computer algorithms involving interval arithmetic have been much studied. In [12] and [13], approximately 2,000 books, journal and conference proceedings articles, and technical reports on interval mathematics are listed. This interest is perhaps due to the fact that properly designed and implemented interval methods are *totally reliable* in the sense that they give results with

1980 *Mathematics Subject Classification* (1985 *Revision*). Primary 65H10; Secondary 90C30, 65G10.

[redacted]

mathematical certainty. Here, we will use this property to solve the problem:

> Find, *with certainty, approximations to all solutions of the nonlinear system*

$$f_i(x_1, x_2, \ldots, x_n) = 0, \qquad 1 \le i \le n, \tag{1.1}$$

> where bounds l_i and u_i are known such that
>
> $$l_i \le x_i \le u_i \qquad \text{for } 1 \le i \le n.$$

(We will denote the n-vector whose ith component is x_i by X and the n-vector whose ith component is f_i by $F(X)$.)

An interval algorithm will produce a list of solutions whose coordinates x_i are given as small intervals of uncertainty. If the proper algorithm (cf. Section 3) is correctly implemented with directed roundings (cf. Section 2), completion of this algorithm constitutes a computational but *mathematically rigorous proof* that all solutions of (1.1) are within the intervals given in the list.

Tasks other than (1.1) can be made totally reliable with interval arithmetic techniques. However, algorithms cannot necessarily be made both reliable and practical merely by replacing the usual arithmetic operations by intervals and interval operations. (That is, new algorithms need to be developed for interval arithmetic.)

In Section 2, we review some elementary definitions of interval arithmetic. In Section 3, we present a class of interval arithmetic algorithms for solving nonlinear algebraic systems and for nonlinear optimization. In Section 4, we briefly discuss some issues in the design of software for solving nonlinear algebraic systems with interval arithmetic, and we cite our portable package. In Section 5, we compare interval methods for the numerical analysis of nonlinear systems to alternate techniques. In Section 6, to illustrate interval mathematics as a general tool, we explain how it can be used in stepsize control for continuation methods.

2. Elementary facts about interval arithmetic. Thorough introductions to interval mathematics are given in the books [1] and [41]. (Also, Moore, who is attributed with inventing interval arithmetic, wrote [38] in 1966.) In particular, one can find details for this section in Chapters 1–3 of [41] or in Chapters 1–4 of [1]. Also see [54] and [59] if they are available. A conference on use of interval methods for scientific computing in general was recently held. This conference brought together experts in both interval techniques and non-interval techniques; in the

proceedings [42], the participants attempt to clarify the role of interval mathematics. Over the years, experts in the area have also presented technical details at conferences such as [15], [37], [48], and [49].

Here, we will give an elementary explanation of some of the most important concepts. Throughout, interval quantities will be denoted by boldface.

Interval arithmetic is based on defining the four elementary arithmetic operations on intervals. Let $\mathbf{a} = [a_l, a_u]$ and $\mathbf{b} = [b_l, b_u]$ be intervals. Then, if $\text{op} \in \{+, -, *, /\}$, we define

$$\mathbf{a} \,\text{op}\, \mathbf{b} = \{x \,\text{op}\, y \mid x \in \mathbf{a} \text{ and } y \in \mathbf{b}\}. \tag{2.1}$$

For example,

$$\mathbf{a} + \mathbf{b} = [a_l + b_l, a_u + b_u].$$

In fact, all four operations can be defined in terms of addition, subtraction, multiplication, and division of the endpoints of the intervals, although multiplication and division may require comparison of several results. The result of these operations is an interval except when we compute $\mathbf{a} / \mathbf{b}$ and $0 \in \mathbf{b}$. In that case, we use *extended interval arithmetic* (cf., e.g., [41], pp. 66–68) to get two semi-infinite intervals or else the whole real line. For example,

$$[10, 20] / [-2, 5] = (-\infty, -5] \cup [2, \infty).$$

A large part of the power of interval mathematics lies in the ability to compute *inclusion monotonic interval extensions* of functions. If f is a continuous function of a real variable, then an inclusion monotonic interval extension $\mathbf{f}$ is defined to be a function from the set of intervals to the set of intervals, such that, if $\mathbf{x}$ is an interval in the domain of $\mathbf{f}$,

$$\{f(x) \mid x \in \mathbf{x}\} \subset \mathbf{f}(\mathbf{x})$$

and such that

$$\mathbf{x} \subset \mathbf{y} \quad \text{implies} \quad \mathbf{f}(\mathbf{x}) \subset \mathbf{f}(\mathbf{y}).$$

Inclusion monotonic interval extensions of a polynomial may be obtained by simply replacing the dependent variable by an interval and by replacing the additions and multiplications by the corresponding interval operations. For example, if $p(x) = x^2 - 4$, then $\mathbf{p}([1, 2])$ may be defined by

$$\mathbf{p}([1, 2]) = ([1, 2])^2 - 4 = ([1, 4]) - [4, 4] = [-3, 0].$$

We emphasize here that the result of an elementary interval operation is precisely the range of values that the usual result attains as we

let the operands range over the two intervals. However, the value of an interval extension of a function is not precisely the range of the function over its interval operand, but only contains this range; we hope to construct interval extensions whose values differ from the range as little as possible. For example, if we write p above as $p(x) = (x - 2)(x + 2)$, then the corresponding interval extension gives

$$\mathbf{p}([1,2]) = ([1,2] - 2)([1,2] + 2) = [-1,0][3,4]$$
$$= [-4,0],$$

which is not as good as the previous extension. See [58] for a discussion of efficient ways of formulating interval extensions.

We may use the mean value theorem or Taylor's theorem with remainder formula to obtain interval extensions of transcendental functions. For example, suppose $\mathbf{x}$ is an interval and $a \in \mathbf{x}$. Then, for any $y \in \mathbf{x}$, we have

$$\sin(y) = \sin(a) + (y - a)\cos(a) - [(y - a)^2/2]\sin(c)$$

for some c between a and y. If a and y are both within a range where the sine function is nonnegative, then we obtain

$$\sin(y) \in \sin(a) + (\mathbf{x} - a)\cos(a) - (\mathbf{x} - a)^2/2.$$

The right side of this relationship gives the value of an interval extension of $\sin(x)$, albeit a somewhat crude one. We may also use rational approximations in cases where the function does not have a Taylor series; if the function is approximated in the uniform norm, then we may bound the error by a constant interval. See [58] for details.

Several computer packages (as in [5], [6], [35], and [72]) are available for interval extensions of the elementary functions.

Mathematically rigorous interval extensions can be computed in finite-precision arithmetic via the use of *directed roundings*. Let x and y be machine-representable numbers, and assume op is one of the four elementary operations $+$, $-$, $*$, or $/$. Normally, x op y is not representable in the machine's memory, and there are various schemes of rounding. For example, we may always *round down* to the nearest machine number less than x op y, or we may always *round up* to the nearest machine number greater than x op y. In interval arithmetic with directed rounding, if

$$\mathbf{x} \text{ op } \mathbf{y} = [c, d],$$

then we always round the value for c down, and we always round the value for d up. In such computations, we first apply directed rounding to the initial data in order to store it.

Machine interval arithmetic with directed rounding does not involve deep concepts, but it can be quite powerful. For example, if interval arithmetic with directed rounding is used to compute an interval extension $\mathbf{f}$ of f,

$$[c, d] = \mathbf{f}([a, b]),$$

and $[c, d]$ does not contain zero, then this is a rigorous proof (regardless of the machine wordlength, etc.) that there is no root of f in $[a, b]$.

In [34], Kulisch and Miranker carefully present concepts of directed rounding and machine interval arithmetic. They include discussion related to producing interval extensions of functions which yield intervals that are as close as possible to the range of the function.

Various precompilers and compilers exist which support the interval datatype. See [27], etc., for details.

3. Interval methods in solving nonlinear systems of equations and in nonlinear optimization. Here, we discuss the problem (1.1) of finding *all* roots of a nonlinear system of equations subject to upper and lower bounds on the variables. We also discuss the related problem:

> Find, *with certainty, the* **global** *minimum* of the nonlinear objective function

$$\Phi(x_1, x_2, \ldots, x_n) \tag{3.1}$$

> where bounds l_i and u_i are known such that

$$l_i \le x_i \le u_i \qquad \text{for } 1 \le i \le n.$$

The problem (1.1) may be solved via generalized bisection in conjunction with interval Newton methods. This general technique is described in Chapters 19 and 20 of [1] and in Chapters 5 and 6 of [41]. An early paper on the technique is [14]. Other papers include [16], [25], [26], [39], [40], [45], [47], [50], and [62].

In what follows we denote the *box* in n-space described by

$$\{X = (x_1, x_2, \ldots, x_n) \mid l_i \le x_i \le u_i \text{ for } 1 \le i \le n\}$$

by $\mathbf{B}$. Similarly, we denote vectors whose entries are intervals by capital boldface letters.

In interval Newton methods, we find a box $\overline{\mathbf{X}}_k$ that contains all solutions of the interval linear system

$$\mathbf{F}'(\mathbf{X}_k)(\overline{\mathbf{X}}_k - X_k) = -F(X_k), \tag{3.2}$$

where $\mathbf{F}'(\mathbf{X}_k)$ is a suitable interval extension of the Jacobian matrix of F over the box $\mathbf{X}_k$ (with $\mathbf{X}_0 = \mathbf{B}$), and where X_k is some point in $\mathbf{X}_k$.

(An elementwise interval extension of the Jacobian matrix is suitable.) We then define the next iterate $\mathbf{X}_{k+1}$ by

$$\mathbf{X}_{k+1} = \mathbf{X}_k \cap \overline{\mathbf{X}}_k. \tag{3.3}$$

The scheme based on solving (3.2) and performing (3.3) is termed an *interval Newton method*. The *convergence rate* of an interval Newton method is defined in terms of the ratios of the widths of the component intervals of $\mathbf{X}_{k+1}$ to the corresponding widths of $\mathbf{X}_k$.

If each row of $\mathbf{F}'$ contains all possible vector values that that row of the scalar Jacobian matrix takes on as X ranges over all vectors in $\mathbf{X}_k$, then it follows from the mean value theorem that all solutions of (1.1) in $\mathbf{X}_k$ must be in $\mathbf{X}_{k+1}$. If the coordinate intervals of $\mathbf{X}_{k+1}$ are smaller than those of $\mathbf{X}_k$, then we may iterate (3.2) and (3.3) until we obtain an interval vector the widths of whose components are smaller than a specified tolerance.

If the coordinate intervals of $\mathbf{X}_{k+1}$ are not smaller than those of $\mathbf{X}_k$, then we may *bisect* one of these intervals to form two new boxes; we then continue the iteration with one of these boxes, and push the other one onto a *stack* for later consideration. After completion of the current box, we pop a box from the stack, and apply (3.2) and (3.3) to it; we thus continue until the stack is exhausted. As is explained in [40], [25], and elsewhere, such a composite *generalized bisection algorithm* will reliably compute all solutions to (1.1) to within a specified tolerance.

The efficiency of the generalized bisection algorithm depends on

(1) the sharpness of the interval extension to the rows of the Jacobian matrix;
(2) the way we find the solution $\overline{\mathbf{X}}_k$ to (3.2); and
(3) how we select the coordinate directions in which to bisect.

In particular, iteration with formulas (3.2) and (3.3) should exhibit the quadratic local convergence properties of Newton's method, but repeated bisections are to be avoided if possible. We are thus interested in arranging the computations so that $\overline{\mathbf{X}}_k$ has coordinate intervals that are as narrow as possible.

An early method of solving (3.2) was the Krawczyk method, in which $\overline{\mathbf{X}}_k$ is given by

$$\overline{\mathbf{X}}_k = \mathbf{K}(\mathbf{X}_k) = X_k - Y_k F(X_k) + (I - Y_k \mathbf{F}'(\mathbf{X}_k))(\mathbf{X}_k - X_k), \tag{3.4}$$

where Y_k is a *preconditioner matrix* that is sometimes taken to be an approximation to $[F'(X_k)]^{-1}$. Moore observed in [39] that the Krawczyk

method converged (without bisection)

(3.5) provided $\|I - Y_k \mathbf{F}'(\mathbf{X}_k)\| < 1$,

where the norm is the usual one for interval matrices. (See [1] or [41].) Condition (3.5) also serves as part of a computational existence and uniqueness test. Various researchers have subsequently weakened this condition. In fact, for many methods of solving (3.2), researchers have shown that

(3.6) if $\overline{\mathbf{X}}_k$ is *strictly* contained in $\mathbf{X}_k$, then the system of equations in (1.1) has a unique solution in $\mathbf{X}_k$, and Newton's method starting from any point in $\mathbf{X}_k$ will converge to that solution. Conversely, if $\mathbf{X}_k \cap \overline{\mathbf{X}}_k$ is empty, then there are no solutions of the system in (1.1) in $\mathbf{X}_k$.

(See [20], [45], and [52].)

An interval version of the Gauss–Seidel method, with extended interval arithmetic, can also be used to bound the solution set to (3.2) (see, for example, [19]). In such a method, we often multiply both sides of (3.2) by the matrix Y_k before dividing by the diagonal element; this ensures local convergence, and results in a method that is often superior to the Krawczyk method. We will denote the ith component of X_k by x_i, the ith component of $Y_k F(X_k)$ by $\mathbf{k}_i$, and the entry in the ith row and jth column of $Y_k \mathbf{F}'(\mathbf{X}_k)$ by $\mathbf{G}_{i,j}$. The step for the ith row of the interval Gauss–Seidel method then becomes

$$\overline{\mathbf{x}}_i = \mathbf{x}_i - \left[\mathbf{k}_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} \mathbf{G}_{i,j}(\mathbf{x}_j - x_j) \right] \Big/ \mathbf{G}_{i,i}; \tag{3.7}$$

$$\mathbf{x}_i^+ \leftarrow \mathbf{x}_i \cap \overline{\mathbf{x}}_i.$$

(It is understood that, if $i > 1$, we replace $\mathbf{x}_j$ by $\mathbf{x}_j^+$ for $j < i$ when we define $\overline{\mathbf{x}}_i$.)

A detailed example of an iteration of the interval Gauss–Seidel method is available from the author. In that example, because of the strict containments $\overline{\mathbf{x}}_1 \subset \mathbf{x}_1$ and $\overline{\mathbf{x}}_2 \subset \mathbf{x}_2$, we may use results in [45] to conclude that there is a unique root of F in the box $\mathbf{X}$, and that the usual Newton's method will converge to that root, starting from any point in $\mathbf{X}$.

For large, banded, or sparse systems, multiplication by an inverse is impractical. In [61] and [63], Schwandt uses the interval Gauss–Seidel method without a preconditioner to solve systems like finite-difference discretizations of Poisson's equation with a nonlinear forcing term. In

such cases, an interval generalization of diagonal dominance ensures convergence of repeated application of (3.7) when Y_k is the identity. An excellent exposition of this technique appears in [67].

Techniques for computing the rows of Y_k explicitly to minimize the widths of the intervals $\mathbf{x}_i^+$ (and thus maximize convergence rate) appear in [31]. These techniques involve solving linear programming problems for the elements of Y_k; these linear programming problems express optimality conditions for the width of $\bar{\mathbf{x}}_i$. Performance results for an interval Newton method using these techniques and for an interval Newton method with $Y_k = [F'(X_k)]^{-1}$ appear in [31]. Though the linear programming step is a major contributor to computation time, the technique is applicable to ill-conditioned and singular systems, and the linear programming problem can be altered to take account of structure or sparsity. Also, further study is yielding ways of making solution of the special linear programming problems arising here more efficient.

An alternate technique for applying interval Newton methods when the Jacobian is ill-conditioned or singular near the roots appears in [30].

The global nonlinear optimization problem (3.1) can be solved by solving (1.1), where the f_i are the components of $\nabla\Phi$. However, we may use the objective function directly to increase the algorithm's efficiency. If $\mathbf{p}$ and $\mathbf{q}$ are intervals, we say that $\mathbf{p} > \mathbf{q}$ if every element of $\mathbf{p}$ is greater than every element of $\mathbf{q}$. Suppose that $\mathbf{X}$ and $\mathbf{Y}$ are interval vectors in the stack described below (3.2), and let Φ be an interval extension to Φ. Then, if $\Phi(\mathbf{Y}) > \Phi(\mathbf{X})$, we may discard $\mathbf{Y}$ from the stack.

Papers and reviews on solution of the global optimization problem include [4], [17], [21], [23], [50], [56], and [60]. Walster, Hansen, and Sengupta report performance results on their global optimization algorithm in [69]. Researchers not expert in interval mathematics also occasionally rediscover the exclusion principle just described without explicitly using the machinery of interval arithmetic.

4. Issues in implementing reliable interval Newton nonlinear equation solvers. As mentioned in Section 2, there are various language tools for working with interval arithmetic, and computer hardware supports interval arithmetic to varying degrees. Likewise, there are libraries of elementary and special functions for interval arithmetic. Also, it is possible through automatic differentiation (as in [53], [55], or [66]) or through symbolic manipulation (as in, for example, [65]) to shift the burden of programming the Jacobian matrix from the user to either the compiler or to the executable code. However, these arithmetic and

differentiation tools are not yet universally available or standardized. (Future versions of the FORTRAN standard, if implemented, will help.)

Furthermore, there is also not yet a consensus about which interval Newton method is "best," nor is it clear how best to precondition the Jacobian matrix in all cases. Also, we are still doing practical investigations related to the choice of coordinate interval to bisect.

Another issue deals with how to handle the case where roots occur on or near boundaries of boxes. This problem could degrade the efficiency of the algorithm in higher dimensions when the root occurs near a low-dimensional boundary or vertex of the box. We are presently investigating refining the "expansion" technique described in [25] not only to eliminate redundancies, but also to significantly increase the algorithm's efficiency.

We are also pursuing additional work to handle systems where the Jacobian matrix is singular or ill-conditioned near the root. In such cases, (3.6) is usually not satisfied, and the predominant mode of the algorithm can be the costly coordinate bisection process. We are looking at the preconditioner technique mentioned above and in [31], among other ideas.

Finally, as with many numerical tasks, there is an unclear interplay between the order of approximation used and the overall efficiency of the algorithm. For example, we could approximate elements of the Jacobian matrix to high order with centered forms (as in [60]). For many functions, this would allow completion of the algorithm with fewer function and Jacobian evaluations; however, we would encounter the additional complication of setting up second-derivative tensors and the additional cost of evaluating them.

Despite these problems and ambiguities, we have felt it important to make interval nonlinear equation software generally available. In [29], we describe a portable, self-contained FORTRAN 77 package for interval Newton/bisection as in Section 3; we present performance results for an early version of this code in [26], and the abstract structure of the algorithm in [25]. We can supply the FORTRAN source code, and also machine executable code for IBM PC compatibles, on MS-DOS $5\frac{1}{4}$ in. diskettes.

In this code, we have sacrificed some speed and generality for portability and ease of use. Our code is perhaps a factor of 20 slower than if interval arithmetic were available directly in the complier. However, it is still competitive with alternate techniques for many problems; see [26].

In our code, we have used the interval Gauss–Seidel method, and we have preconditioned with the inverse of the matrix of midpoints of the elements of the interval Jacobian matrix. The interval Gauss–Seidel method is competitive in many instances (cf. [18] and [19]), while our choice of preconditioner seems popular and is also relatively good in practice. We have used a special scaling technique to choose coordinate intervals to be bisected; see [29].

Our code simulates directed roundings, given a bound on the maximum number of units in the last place by which the result of an elementary machine arithmetic operation can be in error. It thus gives mathematically rigorous results.

5. Interval Newton/bisection methods and alternate nonlinear equation solvers. As the topics in this conference indicate, not only are there various techniques for solving nonlinear algebraic systems of equations, but the different techniques tackle different problems and have different goals. The class of interval methods outlined in Section 3 above is well suited to

(i) reliably finding *all* solutions within a given region of space (i.e., solving (1.1));
(ii) reliable *global* optimization;
(iii) analysis of the sensitivity of solutions to certain coefficients or parameters; and
(iv) the general study of parameter-dependent systems.

(The problem (1.1) is related to the global optimization problem since the latter can be solved by finding all critical points. However, global optimization is numerically easier since values of the objective function may also be used to eliminate some critical points early in the computation.)

Among alternate popular methods for finding all solutions are

(a) hybrid techniques (perhaps introduced with [51] and reviewed in [10], [11], or possibly notes to this conference), which include trust region algorithms with Newton's method or quasi-Newton methods;
(b) other bisection techniques, such as those involving computation of the topological degree; and
(c) continuation methods based on mathematical homotopies.

Techniques for solving the global optimization problem include, in addition to the above,

(d) schemes with a stochastic component, such as random function sampling, the tunneling method ([36]), and simulated annealing.

Below, we compare each of these four algorithmic classes with interval Newton/bisection.

5.1. *Comparison with hybrid methods.* Hybrid steepest descent/Newton-like algorithms are very efficient and reasonably (though not *rigorously*) reliable when we have sufficient knowledge of the problem. There is a large body of theory, and the methods have been used successfully in a wide range of applications; this includes very large problems and problems with a special structure. However, such methods are designed to find *just one* solution, or just a *local* optimum; the globalization techniques associated with these algorithms refer merely to an expansion of the domain of convergence of the underlying Newton-like method. Sensitivity information can be obtained indirectly from the algorithms' behavior, or local information can be obtained by decomposition of the Jacobian matrix at the solution.

5.2. *Comparison with topological degree-based bisection.* Topological degree-based bisection methods do not require interval arithmetic (and indeed do not need accurate function values), and do not require Jacobian evaluations. They are thus applicable to general nonsmooth functions (although interval methods are appropriate in some such cases). In [68] and elsewhere, Vrahatis has demonstrated that such methods can also be reasonably efficient.

For certain cases (such as for analytic functions of n complex variables), the underlying mathematics indicates that the algorithms will rigorously find all roots. However, computation of the topological degree is sometimes done heuristically, at the expense of reliability. The topological degree can be computed reliably when Lipschitz constants or moduli of continuity are used to bound the range of function components. (See other notes to this conference.) However, such range bounds can be viewed as values of an interval extension to the function, and thus are an example of application of interval techniques. Further investigation of the interplay between interval methods and the topological degree may yield discoveries of value.

In topological degree-based bisection methods, our goal should be perhaps merely to compute starting points for fast local methods. In interval Newton/bisection methods, once the box is small enough, the interval Newton method will exhibit quadratic convergence, and several

mechanisms (including the intersection in (3.3)) can decrease the size of the box before then. In contrast, pure bisection without use of some kind of higher-order information is apt to be only linearly convergent. Also, interval Newton methods have been applied to relatively large systems; in contrast, most topological degree-based algorithms exhibit either computational or storage requirements that are exponential in the number of unknowns. (We do not claim this is an inherent property, though.)

5.3. *Comparison with continuation methods.* Continuation methods can do a good job of solving (1.1) in some cases. In particular, the theory and practice of finding all solutions to moderately sized polynomial systems is well developed. See [43] for an introduction, and see other lecture notes to this conference or [44], etc., for examples of recent work.

One cannot say without additional information about the problems that continuation methods or interval Newton methods will be better suited to solve (1.1). There is abundant convergence theory for homotopy techniques only for polynomial systems; for other systems, homotopies may need to be constructed in an ad hoc fashion. (Interval Newton/bisection algorithms can be applied to any functions that have interval extensions.) In practice, the homotopy/continuation algorithms must find all complex roots to find all roots within a certain region; the number of such roots is equal to the total degree of the system, which is the product of the degrees of the individual components, but is somewhat less when the system has an m-homogeneous structure. (See other notes to this conference or [44].) Thus, the algorithms may inefficiently compute large numbers of complex roots that are not of interest. Furthermore, unless most components are linear, or the system has a special structure, the total degree and amount of work will increase exponentially with the number of variables and equations.

On the other hand, the width of an interval image of a component $\mathbf{f}_i(\mathbf{X})$ will depend both on the number of arithmetic operations and the way that these operations are arranged. This could make interval Newton/bisection algorithms impractical for certain functions that have a large number of naively arranged operations. Higher-degree polynomials generally have wider interval values than lower-degree polynomials. However, this phenomenon is different from that of total degree, and the class of problems for which it occurs is neither contained in nor contains those for which the total degree is high.

Reliability is a second question. The underlying theory guarantees that the homotopies will lead to all roots to polynomial systems of equations, except if we are unlucky enough to choose certain parameters from "bad" sets of measure zero or of finite cardinality. The continuation methods themselves, however, can involve heuristics or can suffer from the effects of rounding errors. For example, if the predictor stepsize, which is usually heuristically chosen in predictor/corrector methods (see [2] for an introduction), is too large, then the steps may jump from one homotopy path to another. This results in roots listed more times than their multiplicity, and in some roots being totally missed. We have observed this behavior with some software ([24]). However, our experience with the code in [43] has convinced us that properly tuned and implemented algorithms can be very reliable in practice. Furthermore, we have a priori knowledge of the total number of solutions to a polynomial system, and we may check this against the number actually found and against rank deficiencies. Nonetheless, such reliability is qualitatively different from the *mathematical rigor* that can be achieved with interval Newton/bisection methods.

In Section 6, we describe a new use of interval arithmetic to control the stepsize in continuation methods. With that scheme, we believe we would have a rigorous guarantee that the predictor/corrector iteration would not jump from one homotopy path to another.

Both continuation methods and interval arithmetic techniques are useful in the study of sensitivities. In interval Newton methods, the studied coefficient or parameter would be separate from the unknown variables; it would appear in the formulas for evaluation of the $\mathbf{f}_i$ as a constant *interval*. Iteration of (3.3) would then result in convergence not to a point solution to the nonlinear system, but convergence to a box in n-space. That box would contain the set of all solutions to the system as the parameter ranged over its interval; the sharpness of the containment would depend on details of the interval extension.

With continuation methods, we can treat the studied parameter as a homotopy variable. We can then observe solutions to the system as we trace the homotopy paths. Using this technique, we successfully studied the amount of information that can be obtained with certain models of evoked cerebral potentials; see [28].

Both continuation methods and interval mathematics can be used in the study of physical systems in which a parameter naturally varies. For example, Keller and others (e.g., [32]) study fluid flows as Reynolds

number (or velocity or viscosity) varies. In this application of continuation methods, discretizations of partial differential equations result in very large systems. Researchers can take advantage of structure and sparsity since the predominant computation in such instances is the analysis of linear systems of equations in standard floating-point arithmetic; techniques for bordered systems, as in [7] or [33], have also been inspired by this application.

Homotopy paths can also be resolved via interval Newton techniques; see [46]. However, large parameter-dependent systems have not yet been tackled.

5.4. *Comparison with methods with a stochastic component.* In contrast with hybrid methods, probabilistic methods make an effort to find a global optimum instead of just a single local one. However, none of these techniques can give mathematical assurance of their results, and random function evaluation can converge slowly even in a probabilistic sense. It may, however, be possible to use them on some very large or complicated optimization problems for which interval Newton methods embodying the present state-of-the-art would take too many operations to be practical.

The tunneling method is interesting since it combines a clever deterministic component with a stochastic component; see [36]. This method is able to "tunnel" under very large numbers of local minima on its way to the global minimum. However, interval Newton based methods also perform fairly well on some of these problems; see [69].

6. An interval arithmetic stepsize control for continuation methods. We describe here a criterion for a foolproof stepsize control for predictor/corrector continuation methods. We first give an abstract algorithm (which would allow various predictor and corrector schemes) within which we can embed the stepsize control.

Let $H: \mathbf{R}^{n+1} \to \mathbf{R}^n$ denote the homotopy, and let $Y_k \in \mathbf{R}^{n+1}$ be such that $H(Y_k) = 0$. We then compute an approximation Y_{k+1}^p to another point on the zero manifold of H containing Y_k by

$$Y_{k+1}^p = Y_k + \delta_k B_k. \tag{6.1}$$

The direction vector B_k ($\|B_k\|_2 = 1$) may be obtained in a number of ways, while we will adaptively adjust and reset δ_k with our stepsize control. We then correct Y_{k+1}^p to obtain a point Y_{k+1} that is more nearly on the manifold. (See [2] for an introduction.) In other words, we have

ALGORITHM 6.1. (A SINGLE PREDICTOR/CORRECTOR STEP).

(1) *Compute a predictor step direction* B_k *and an initial predictor steplength* δ_k.

(2) *Compute* Y_{k+1}^p *by* (6.1).

(3) *Choose a corrector manifold* $r_k(Y)$, *where* $r_k : \mathbf{R}^{n+1} \to \mathbf{R}^n$ *and* $r_k(Y_{k+1}^p) = 0$.

(4) *See if corrector iteration (step 5) will converge starting with the present* Y_{k+1}^p.

(a) *If it will not, then decrease* δ_k *and repeat steps 2, 3, and this step.*

(b) *If it will, then perform step 5.*

(5) *Find* Y_{k+1} *by using a convergent method to solve the* $(n+1)$ *by* $(n+1)$ *system of equations*

$$\tilde{F}(Y) = \begin{bmatrix} r_k(Y) \\ H(Y) \end{bmatrix} = 0. \tag{6.2}$$

Generally, the corrector step is checked (step 4) using heuristic criteria; see [3], [22], [64], etc. In such instances, if δ_k is chosen too large, the corrector iteration could converge to a point that is on a different manifold from that of Y_k; see Figure 1.

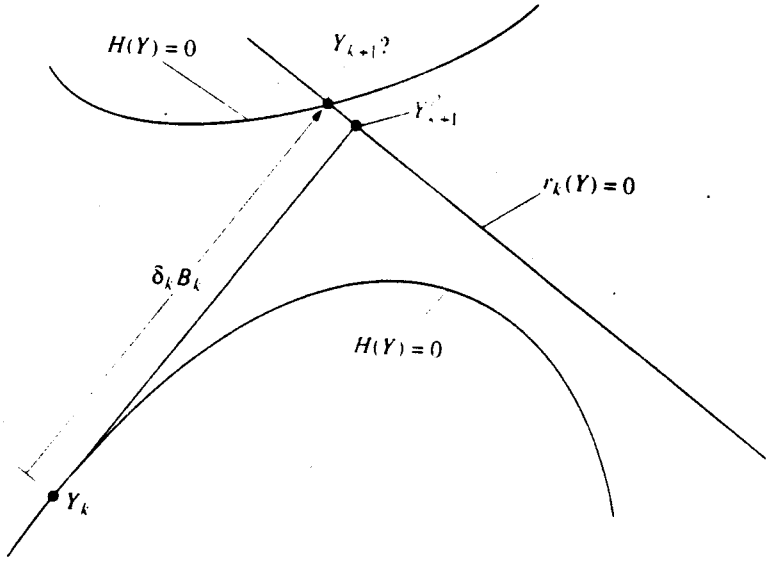

FIGURE 1. Even though the corrector iteration converges, iterates of the predictor/corrector algorithm may jump from one path to another.

In contrast, we may employ an interval Newton method in step (4), and use (3.5) or (3.6). If we include values of the Jacobian matrix over the entire line segment between Y_k and Y^p_{k+1}, then it is possible to construct an algorithm for which the phenomenon in Figure 1 cannot occur. In fact, with such interval arithmetic, the negative result in [64] is in many cases not relevant. (We do note, however, that we make some implicit assumptions when we use interval extensions.)

To construct the stepsize test, we first select a function $q(\delta)$ such that, for all sufficiently small δ with $\|Y - Y_k\|_2 = \delta$ and Y on the manifold of $H(Y) = 0$ that contains Y_k, the tangent of the angle that the line segment with Y_k and Y as endpoints makes with B_k is at most $q(\delta)$. We also require that $q(\delta) \to 0$ as $\delta \to 0$. For example, if there is a fixed $c > 0$ such that

$$\frac{B_k \circ (Y - Y_k)}{\|Y - Y_k\|_2} \geq c,$$

then $q(\delta) = \delta^{-\varepsilon}$ for any $\varepsilon > 0$ will do; see Figure 2.

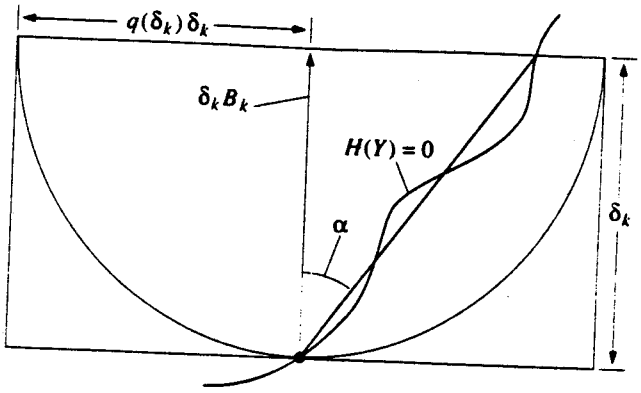

FIGURE 2. $q(\delta_k)$ is defined so that $\tan(\alpha) < q(\delta_k)$ for all sufficiently small δ_k.

For simplicity, we assume that $r_k(Y) = 0$ defines the linear manifold through Y^p_{k+1} perpendicular to B_k. Then, in the interval Newton method, we will extend the Jacobian matrix to contain values of the scalar Jacobian matrix over a box that contains the line segment between Y and Y^p_{k+1} and the portion of $r_k(Y) = 0$ within distance δ of Y^p_{k+1}; again refer to Figure 2. We may then take the point X_0 in (3.2) or in (3.7) to be Y^p_{k+1}. Our interval Newton method will operate on

the manifold defined by r_k, so that it will solve a square n-by-n system. However, in order to ensure that path-jumping as in Figure 1 will not occur, the interval iterations will, in a sense, include all interval iterations corresponding to steplengths between 0 and δ_k.

To illustrate, let us examine

$$H(Y) = H(y_1, y_2) = y_2(y_1^2 - 4) + (1 - y_2)(y_1^2 - 1).$$

Let us take $Y_0 = (1, 0)^T$, and try $B_0 = (0, 1)^T$ and $\delta_0 = 0.1$. We will use $r_k(Y) = B_k \circ (Y - Y^p_{k+1})$. We first characterize the manifold $r_k(Y) = 0$ for $\delta = \delta_0$ by

$$\{(x_1, \delta) | x_1 \in \mathbf{R}\}.$$

We may then solve $\tilde{F}(Y) = 0$ by solving $F(x_1) = 0$, where

$$F(x_1) = H(x_1, \delta) = \delta(x_1^2 - 4) + (1 - \delta)(x_1^2 - 1).$$

Simplifying, we obtain

$$F(x_1) = x_1^2 - (1 + 3\delta), \quad \text{so} \quad F'(x_1) = 2x_1.$$

We make an interval extension to F by letting δ range in the interval $[0, \delta_0] = [0, .1]$; we use the function q then to define the interval for x_1 to be

$$\mathbf{X} = [.68, 1.32];$$

we may take $X = 1$ and $\delta_0 = 0.1$ to compute Y. (Throughout this example, we use directed roundings to represent quantities to two digits of accuracy.) We use (3.7) and (3.6). We then have

$$\mathbf{F}(X) = [0, .3],$$

where we get an interval value because we use an interval for δ_0. We also have

$$\mathbf{F}'(\mathbf{X}) = [1.36, 2.64], \quad Y = \tfrac{1}{2}, \quad \text{and} \quad \mathbf{G} = [.68, 1.32].$$

Finally,

$$\mathbf{k} = [0, .15] \quad \text{and} \quad \overline{\mathbf{X}} = [1, 1.23] \subset \mathbf{X}.$$

Since the inclusion is strict and since we have used intervals for δ_0 in the values of $\mathbf{F}$ and $\mathbf{F}'$, we believe we can conclude that there is a unique solution to (6.2) for each value of δ between 0 and δ_0, and that Newton's method starting at X will converge to that solution; the analysis would be similar to that in [30].

Since we may control the stepsize, generalized bisection can be entirely avoided. Thus, the method should be applicable to problems in high-dimensional spaces.

A complete analysis, generalizations, and more involved numerical examples of this technique will appear elsewhere.

ACKNOWLEDGMENTS. The idea for the continuation method stepsize control came to Alexander Morgan and me over dinner. I am also indebted to Alec for putting the code in [43] into perspective.

REFERENCES

1. G. Alefeld and J. Herzberger, *Introduction to Interval Computations*, Academic Press, New York, 1983.

2. E. L. Allgower and K. Georg, *Simplicial and continuation methods for approximating fixed points and solutions to systems of equations*, SIAM. Rev. **22** (1) (1980), 28–55.

3. J. Avila, *The feasibility of continuation methods for nonlinear equations*, SIAM J. Numer. Anal. **11** (1974), 102–120.

4. E. Baumann, *Globale optimierung stetig differenzierbarer Functionen einer Variablen*, preprint, Freiburger Intervall-Berichte 86/6, Institut für Angewandte Mathematik der Universität Freiburg, 1986, pp. 1–89.

5. J. H. Bleher, S. M. Rump, U. Kulisch, M. Metzger, and W. Walter, *Fortran-SC—A study of a Fortran extension for engineering scientific computations with access to ACRITH*, Computing **39** (2) (1987), 93–110.

6. A. Bundy, *A generalized interval package and its use for semantic checking*, ACM Trans. Math. Software **10** (1984), 397–409.

7. T. F. Chan and Y. Saad, *Iterative methods for bordered systems with applications to continuation methods*, SIAM J. Sci. Statist. Comput. **6** (2) (1985), 438–451.

8. M. Clemmesen, *Interval arithmetic implementations using floating point arithmetic*, SIGNUM News, **19**(4), 1984, pp. 2–8.

9. F. Crary, *The AUGMENT precompiler*, Mathematics Research Center report no. 1470, the University of Wisconsin at Madison, 1976.

10. J. E. Dennis and J. J. Moré, *Quasi-Newton methods, motivation and theory*, SIAM Rev. **19** (1) (1977), 46–89.

11. J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Least Squares*, Prentice-Hall, Englewood Cliffs, New Jersey, 1983.

12. J. Garloff, *Interval mathematics: a bibliography*, preprint, Freiburger Intervall-Berichte 85/6, Institut für Angewandte Mathematik der Universität Freiburg, 1985.

13. J. Garloff, *Bibliography on interval mathematics, continuation*, preprint, Freiburger intervall-Berichte 87/2, Institut für Angewandte Mathematik der Universität Freiburg, 1987.

14. E. R. Hansen, *On solving systems of equations using interval arithmetic*, Math. Comp. **22** (1968), 374–384.

15. E. R. Hansen (ed.), *Topics in interval analysis*, Oxford University Press, London, 1969.

16. E. R. Hansen, *Interval forms of Newton's method*, Computing **20** (1978), 158–163.

17. ——, *Global optimization using interval analysis–the multidimensional case*, Numer. Math. **34** (1980), 247–270.

18. E. R. Hansen and S. Sengupta, *Bounding solutions of systems of equations using interval arithmetic*, BIT **21** (1981), 203–211.

19. E. R. Hansen and R. I. Greenberg, *An interval Newton method*, Appl. Math. Comput. **12** (1983), 89–98.

20. E. R. Hansen and G. W. Walster, *Nonlinear equations and optimization*, accepted for publication in Math. Programming.

21. E. R. Hansen, *An Overview of Global Optimization Using Interval Analysis*, Reliability in Computing, R. E. Moore (ed.), Academic Press, New York, 1988.

22. C. den Heijer and W. C. Rheinboldt, *On steplength algorithms for a class of continuation methods*, SIAM J. Numer. Anal. **18** (5) (1981), 925–948.

23. K. Ichida and Y. Fujii, *An interval arithmetic method for global optimization*, Computing **23** (1) (1979), 85–97.

24. R. B. Kearfott, *On a general technique for finding directions proceeding from bifurcation points*, Numerical Methods for Bifurcation Problems, T. Küpper, H. D. Mittelmann, and H. Weber (eds.), Birkhäuser, Basel, 1984.

25. ——, *Abstract generalized bisection and a cost bound*, Math. Comput. **49** (179) (1987), 187–202.

26. ——, *Some tests of generalized bisection*, ACM Trans. Math. Software **13** (3) (1987).

27. ——, *Interval arithmetic methods for nonlinear systems and nonlinear optimization: an introductory review*, Impact of Recent Computer Advances on Operations Research (R. Sharda, B. L. Golden, E. Wasil, O. Balci, and W. Stewart, eds.), North-Holland, New York, 1989, pp. 533–542 (Proc. Conf., Williamsburg, Virginia, January–1989).

28. ——, *The role of homotopy techniques in biomedical modelling: a case study*, Twelfth IMACS World Congress on Scientific Computation (Proc. Conf., Paris, France, July 1988) (to appear).

29. R. B. Kearfott and M. Novoa, *A program for generalized bisection*, accepted for publication as an algorithm in ACM Trans. Math. Software.

30. R. B. Kearfott, *On handling singular systems with interval Newton methods*, Twelfth IMACS World Congress on Scientific Computation (Proc. Conf., Paris, France, July 1988).

31. ——, *Preconditioners for the interval Gauss–Seidel method*, accepted for publication in SIAM J. Numer. Anal. (June, 1990).

32. H. B. Keller, *Continuation methods in computational fluid dynamics*, Numerical and Physical Aspects of Aerodynamic Flows, T. Cebeci (ed.), Springer-Verlag, New York, 1982, pp. 3–13.

33. ——, *The bordering algorithm and path following near singular points of higher nullity*, SIAM J. Sci. Statist. Comput. **4** (4) (1983), 573–582.

34. U. W. Kulisch and W. L. Miranker, *The arithmetic of the digital computer*, SIAM Rev. **28** (1) (1986), 1–40.

35. T. D. Ladner and J. M. Yohe, *An interval arithmetic package for the Univac 1108*, Technical Summary Report no. 100, Mathematics Research Center, University of Wisconsin at Madison, 1970.

36. A. V. Levy and S. Gomez, *The tunneling method applied to global optimization*, Numerical Optimization 1984, P. T. Boggs, R. H. Byrd, and R. B. Schnabel (eds.), SIAM, Philadelphia, 1985, pp. 213–244.

37. W. L. Miranker (ed.), *Accurate scientific computations*, Lecture Notes in Computer Science no. 235, Springer-Verlag, New York, 1986.

38. R. E. Moore, *Interval Analysis*, Prentice-Hall, Englewood Cliffs, New Jersey, 1966.

39. ——, *A test for existence of solutions to nonlinear systems*, SIAM J. Numer. Anal. **14** (4) (1977), 611–615.

40. R. E. Moore and S. T. Jones, *Safe starting regions for iterative methods*, SIAM J. Numer. Anal. **14** (6) (1977), 1051–1065.

41. R. E. Moore, *Methods and applications of interval analysis*, SIAM, Philadelphia, 1979.

42. R. E. Moore (ed.), *Reliability in Computing*, Academic Press, New York, 1988.

43. A. P. Morgan, *Solving Polynomial Systems Using Continuation for Engineering and Scientific Problems*, Prentice-Hall, Englewood Cliffs, New Jersey, 1987.
44. A. P. Morgan and A. Sommese, *Computing all solutions to polynomial systems using continuation*, Appl Math. Comput. **24** (2) (1987), 115–138.
45. A. Neumaier, *Interval iteration for zeros of systems of equations*, BIT **25** (1) (1985), 256–273.
46. ——, *The enclosure of solutions of parameter-dependent systems of equations*, Reliability in Computing, R. E. Moore (ed.), Academic Press, New York, 1988.
47. K. Nickel, *On the Newton method in interval analysis*, Technical Summary Report no. 1136, Mathematics Research Center, University of Wisconsin at Madison, 1971.
48. K. Nickel (ed.), *Interval mathematics 1980*, International Symposium on Interval Mathematics (Proc. Conf.), Academic Press, New York, 1980.
49. ——, *Interval Mathematics 1985*, Lecture Notes in Computer Science vol. 212, Springer-Verlag, Berlin, 1986.
50. K. Nickel, *Optimization using interval mathematics*, preprint, Freiburger Intervall-Berichte 86/7, Institut für Angewandte Mathematik der Universität Freiburg, 1986, pp. 55–83.
51. M. J. D. Powell, *A hybrid method for nonlinear equations*, Numerical Methods for Nonlinear Algebraic Equations, P. Rabinowitz (ed.), Gordon and Breach, London, 1970.
52. L. Qi, *A note on the Moore test for nonlinear systems*, SIAM J. Numer. Anal. **19** (4) (1982), 851–857.
53. L. B. Rall, *Automatic differentiation: Techniques and applications*, Lecture Notes in Computer Science, Springer-Verlag, Berlin, Heidelberg, New York, 1981.
54. ——, *Interval analysis: A new tool for applied mathematics*, Mathematics Research Center report no. 2268, the University of Wisconsin at Madison, 1981.
55. ——, *Differentiation in Pascal-SC: type Gradient*, ACM Trans. Math. Software **10** (2) (1984), 161–184.
56. ——, *Global optimization using automatic differentiation and interval iteration*, Mathematics Research Center report no. 2832, the University of Wisconsin at Madison, 1985.
57. ——, *An introduction to the scientific computing language Pascal-SC*, Computers and Mathematics with Applications **14** (1) (1987), 53–69.
58. H. Ratschek and J. G. Rokne, *Computer Methods for the Range of Functions*, Horwood, Chichester, England, 1984.
59. H. Ratschek, *Interval mathematics*, preprint, Freiburger Intervall-Berichte 87/4, Institut für Angewandte Mathematik der Universität Freiburg (to appear as an entry in *Encyclopedia of computer science and technology*, A. G. Holzman, A. Kent, and J. G. Williams (eds.), Marcel Dekker, New York).
60. H. Ratschek and J. G. Rokne, *Efficiency of a global optimization algorithm*, SIAM J. Numer. Anal. **24** (5) (1987), 1191–1201.
61. H. Schwandt, *An interval arithmetic approach for the construction of an almost globally convergent method for the solution of the nonlinear Poisson equation*, SIAM J. Sci. Statist. Comput. **5** (2) (1984), 427–452.
62. ——, *Krawczyk-like algorithms for the solution of systems of nonlinear equations*, SIAM J. Numer. Anal. **22** (4) (1985), 792–810.
63. ——, *The solution of nonlinear elliptic Dirichlet problems on rectangles by almost globally convergent interval methods*, SIAM J. Sci. Statist. Comput. **6** (3) (1985), 617–638.
64. H. Schwetlick, *On the choice of steplength in path following methods*, Z. Angew. Math. Mech. **64** (9) (1984), 391–396.
65. J. M. Shearer and M. A. Wolfe, *ALGLIB, a simple symbol manipulation package*, Comm. ACM **28** (8) (1985), 820–825.

66. B. Speelpenning, *Compiling fast partial derivatives of functions given by algorithms*, Department of Computer Science technical report no. UIUCDCSR-80-1002, University of Illinois at Urbana-Champaign, 1980.

67. S. Thiel, *Intervalliterationsverfahren für diskretisierte elliptische Differentialgleichungen*, Diplomarbeit and preprint, Freiburger Intervall-Berichte 86/8, Institut für Angewandte Mathematik der Universität Freiburg, 1986, pp. 1–72.

68. M. N. Vrahatis and K. I. Iordanidis, *A rapid generalized method of bisection for solving systems of nonlinear equations*, Numer. Math. 49 (2) (1986), 123–138.

69. G. W. Walster, E. R. Hansen, and S. Sengupta, *Test results for a global optimization algorithm*, Numerical Optimization 1984, P. T. Boggs, R. H. Byrd, and R. B. Schnabel (eds.), SIAM, Philadelphia, 1985.

70. W. Walter and M. Metzger, *Fortran-SC, a Fortran extension for engineering/scientific computation with access to ACRITH*, Reliability in Computing, R. E. Moore (ed.), Academic Press, New York, 1988.

71. J. M. Yohe, *The interval arithmetic package*, Mathematics Research Center report no. 175, the University of Wisconsin at Madison, 1977.

72. ——, *Software for interval arithmetic: A reasonably portable package*, ACM Trans. Math. Software 5 (1) (1979), 50–63.

THE UNIVERSITY OF SOUTHWESTERN LOUISIANA