Reprinted from:

IMACS

ANNALS ON COMPUTING AND APPLIED MATHEMATICS

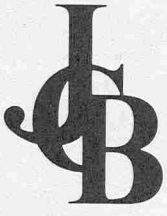
NUMERICAL AND APPLIED MATHEMATICS

Volume 1.2 (1989) 653-655

R. BAKER KEARFOTT

Department of Mathematics, University of Southwestern Louisiana, USL Box 4-1010, Lafayette, Louisiana 70504, U.S.A.

ON HANDLING SINGULAR SYSTEMS WITH INTERVAL NEWTON METHODS



J.C. BALTZER SCIENTIFIC PUBLISHING COMPANY
BASEL

NUMERICAL AND APPLIED MATHEMATICS
C. Brezinski (editor)
J.C. Baltzer AG, Scientific Publishing Co. © IMACS, 1989
pp. 653-655

ON HANDLING SINGULAR SYSTEMS WITH INTERVAL NEWTON METHODS

R. BAKER KEARFOTT
Department of Mathematics, University of Southwestern Louisiana, USL Box 4-1010, Lafayette Louisiana 70504, U.S.A.

Abstract

Interval Newton methods in conjunction with generalized bisection are successful for finding all roots of a system of nonlinear equations within given bounds on the parameters. However, such methods become inefficient when the system is ill-conditioned or singular near its roots. This paper proposes a technique consisting of selecting a parameter such that the system obtained when that parameter is held fixed is well-conditioned. We then apply the interval Newton method to this reduced system while applying generalized bisection in the direction of the selected parameter. We present the rationale for this scheme, we present an algorithm in summary form, and we give numerical results which show the effect of incorporating the algorithm into a general code based on an interval Newton method.

1. Introduction and Motivation

The general problem we address is:

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

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

where bounds a_i and b_i are known such that:

$$a_i \le x_i \le b_i \quad \text{for} \quad 1 \le i \le n.$$

We write $F(X) = \emptyset$, where $F = (f_1, f_2, \ldots, f_n)$ and $X = (x_1, x_2, \ldots, x_n)$. We denote the box given by the inequalities on the variables x_i by B.

A successful approach to this problem is generalized bisection in conjunction with interval Newton methods, as described in (2), (3), (4), (6), (7), (10), etc. Let K(B) denote the image of B upon applying the interval Newton method. For many such interval Newton methods K it can be shown that

(i) $K(B) \cap B \subset B$ (with strict containment) implies that there is a unique solution of (1.1) in B and Newton's method starting from any point in B will converge to it; (1.2)

(ii) $K(B) \cap B$ empty implies that there is no solution of (1.1) in B.

This class of methods achieves 100% reliability and computational efficiency on a wide subclass of the problems defined by (1.1). However, if the Jacobian matrix of F is ill-conditioned at a root $z \in B$, then K(B) is generally a large box, so that neither (1.2)(i) nor (1.2)(ii) will hold. As computations reported in (7) demonstrate, this degrades the efficiency of the algorithm and may also lead to redundant listing of roots and other problems. A similar problem can also arise if one or more components of F involves a large number of arithmetic operations.

The expansion and exclusion steps described in (6) and (7) aid the algorithm to solve singular problems, but alternate iteration schemes for the singular case would be desirable.

Ideas from the theory of the behavior of Newton's method in the vicinity of singularities as reviewed in (1) indicate that possibilities for development of Newton's method-based algorithms to handle general singularities are limited. However, we may decompose the domain into two orthogonal subspaces such that the null space of ∇F is approximately in the first subspace, and such that the Jacobian of F restricted to the second subspace is nonsingular. We then apply a robust bisection procedure in the (hopefully small) subspace where ∇F is approximately singular, and we apply an interval Newton method in the other subspace. We present results for this technique here.

We introduction notation in Section 2, while we present the algorithm in Section 3. Numerical test results appear in Section 4, and conclusions appear in Section 5.

2. Notation

Notation 2.1. Suppose F is as above. Define $f_j : R^n \to R^1$ to be the j-th component of F, define $f_{j,k,y} : R^{n-1} \to R^1$ to be the j-th component of F with the k-th variable held fixed at y, define $F_j : R^n \to R^{n-1}$ to be the function derived from F by deleting the j-th component, and define $F_{j,k,y} : R^{n-1} \to R^{n-1}$ to be the function derived from F by deleting the j-th component and holding the k-th variable fixed at y.

Notation 2.2. Let B be as below (1.1). Define B_k to be the box in R^{n-1} obtained by ignoring the k-th axis of the box B, and let $I = [a_k, b_k]$ be the k-th axis of B. For X a vector in R^n, we define $X_k \in R^{n-1}$ and $x_k \in R^1$ similarly.

Notation 2.3. We use the symbol "∇" to denote the vector derivative or Jacobian matrix as appropriate according to context. Vector derivatives will always mean column vectors.

Notation 2.4. If A is a matrix, then $(A)_k$ will denote the k-th column of A.

Let j and k be such that $\nabla F_{j,k}$ is nonsingular. We then form ϕ and ψ by

$$\phi(y_k) = \{(x_1, x_2, \ldots, x_{k-1}, x_{k+1}, \ldots, x_n) \mid F_j(x_1, x_2, \ldots, x_{k-1}, y_k, y_{k+1}, \ldots, x_n) = \emptyset \tag{2.1}$$

and

$$\psi(x_1, x_2, \ldots, x_{k-1}, x_{k+1}, \ldots, x_n) =$$

$$\{y_k \mid f_j(x_1, x_2, \ldots, x_{k-1}, y_k, x_{k+1}, \ldots, x_n) = \emptyset\}.$$

Also define

$$Y_k = \phi(y_k) = (y_1, y_2, \ldots, y_{k-1}, y_{k+1}, \ldots, y_n),$$

$$\overline{Y}_k = (y_1, y_2, \ldots, y_{k-1}, y_k, y_{k+1}, \ldots, y_n), \quad \text{and} \qquad (2.2)$$

$$\overline{f}_j(y) = f_j(\overline{Y}_k(y)).$$

We will apply generalized bisection to the problem $\overline{f}_j = \emptyset$ in conjunction with an interval Newton method to evaluate $\overline{Y}_k$.

3. The Algorithm

Here, we present an algorithm based on the ideas in Section 2. This algorithm is in general to be embedded in a generalized bisection algorithm as in (7), etc. In such algorithms, we continue to bisect the coordinate intervals $[a_i, b_i]$ in (1.1), replacing B by smaller B until (i) the diameter of B is small; (ii) the norm of F over B is small; or (iii) (1.2)(i) or (1.2)(ii) holds.

As is indicated in Section 1, bisecting the coordinate intervals (and reducing the diameter of B) will not always result in (1.2)(i) or (1.2)(ii). (Success or failure depends on the condition of ∇F and on how much the interval arithmetic widens the intervals.) In such instances, we choose a j and k as in Section 2, with $\nabla(F_{j,k})$ well-conditioned. We then form $K_{j,k}$ and $R_{j,k}$ based on $F_{j,k}$ instead of F. In particular (using the interval Newton method in (9)), we define

$$K_{j,k,I}(B_k) = M_k - H_{j,k}F_{j,k,I}(M_k) + [I - H_{j,k}\nabla(F_{j,k,I})(B_k)](B_k - M_k)$$

where

$$B_k = [a_i, b_i] \times \ldots \times [a_{k-1}, b_{k-1}] \times [a_{k+1}, b_{k+1}] \times \ldots \times [a_n, b_n], \qquad (3.1)$$

$M_k = (m_1, m_2, \ldots, m_{k-1}, m_{k+1}, \ldots, m_n)$ where $m_i = (a_i + b_i)/2$, and where

$H_{j,k} = (\nabla F_{j,k,m_k}(M_k))^{-1}$.

We will use $K_{j,k,I}$ to evaluate ϕ and use generalized bisection in one variable to solve $\overline{f}_j(y) = \emptyset$ only if we cannot use $K_{j,k,I}$.

We desire

$$w(K_{j,k,I}(B_k)) < w(B_k) \text{ as } w(B_k) \to \emptyset, \qquad (3.2)$$

where $w(B_k)$ is the "width" of B_k, and is the length of the largest coordinate interval defining the box B_k; this ensures convergence of

$$B_k \leftarrow B_k \cap K_{j,k,I}(B_k). \qquad (3.3)$$

If $H_{j,k}F_{j,k,I}(M_k)$ were a point vector instead of a box, then (3.2) would translate to a condition similar to the condition on $\|I - H\nabla F\|$ in (9). However, in general we have

$$w(K_{j,k,I}(B_k)) \le w(H_{j,k}F_{j,k,I}(M_k)) + \|R_{j,k}\|_\infty \frac{w(B_k)}{2}, \qquad (3.4)$$

where

$$R_{j,k} = I - H_{j,k}\nabla(F_{j,k,I})(B_k).$$

Combining (3.2) and (3.4) gives Condition (3.1)

$$\|R_{j,k}\|_\infty < 2 \qquad (3.5)$$

and

$$\omega = \frac{w(H_{j,k}F_{j,k,I}(M_k))}{1 - \frac{\|R_{j,k}\|_\infty}{2}} < w(B_k). \qquad (3.6)$$

In fact, if (3.5) holds, then (3.6) can be made to hold by bisecting $[a_k, b_k]$ a sufficient number of times.

In our experiments, we thus safely employed the scheme when (3.5) held and when none of the diagonal elements of $H_{j,k}\nabla(F_{j,k,I})(B_k)$ contained zero. (The latter condition was desirable since our actual interval Newton method involved an analogue of the one in (3), which is superior to the one in (9), but does not have an analogue to (3.5).)

We may choose j and k from the permutation information in the L-U decomposition used to compute $H = (\nabla F(M))^{-1}$. In particular, if the forward phase of Gaussian elimination with partial pivoting completed (except for possibly elimination in the last row), then we choose j to be the index of the row which was not used as a pivot row, and we choose k to be the index of the element in the last (i.e., the (n-1)-st) pivot row which has minimum absolute value. Reasoning roughly, we may expect the matrix $\nabla F_{j,k}$ to be as well-conditioned as the partial pivoting process.

The following algorithm summarizes this section.

Algorithm 3.1.

1. Assume that the box $B \in R^n$ is given such that iteration of $B \leftarrow K(B) \cap B$ is unsuccessful, but $\emptyset \in F(B)$ and $K(B) \cap B$ is non-empty.
2. Compute H, j, k, $K_{j,k}$, and $R_{j,k}$.
3. If $\|R_{j,k}\| \ge 2$ or a diagonal element of $H_{j,k}$ times $\nabla(F_{j,k,I})(B_k)$ contains $\emptyset$, return to the main algorithm (which will do further bisections in R^n).
4. Bisect I. Put one of the subintervals on a stack for later consideration, and replace I by the other one.
5. Iterate (3.3) until one of the following holds.
   (a) $B_k \cap K_{j,k,I}(B_k)$ is empty.
   (b) $\emptyset \notin F(B)$, where B is defined by the present I and B_k.
   (c) $w(B_k)$ is within a specified tolerance of $\emptyset$.
   (d) $\|F(B)\|$ is within a specified tolerance of $\emptyset$.
   (e) The process fails to reduce $w(B_k)$.
6. If 5(a) or 5(b) occurred, go to step 10.
7. If 5(c) occurred, check $|b_k - a_k|$; if $|b_k - a_k|$ is small enough, store B on the list of root containing boxes and go to step 10. Otherwise, return to Step 4.
8. If 5(d) occurred, store B on the list of root-containing boxes, then go to step 10.

9. If 5(e) occurred, return to step 4.

10. If the stack of I produced in step 4 is empty, return with the list of root-containing boxes. Otherwise, pop an I from the stack, restore B_k to its value when that I was formed, and recommence execution of step 5.

Finally, we note that Algorithm 3.1 handles systems where simple singularities occur at the roots. This limitation is merely for simplicity of exposition; I could represent a box in R^p for some $p > 1$, where the corresponding coordinate indices could also be obtained during the Gaussian elimination process.

4. Numerical Tests

We compared an improved version of the code in (7) ((8)) without Algorithm 3.1 to the same code with Algorithm 3.1. (Implementation details will appear elsewhere.) We used the test set from (7) and an additional simple singular problem (problem "T") with $f_1 = .5(x_1)^2 - x_2$, $f_2 = .5(x_1)^2 + x_2$. The code called Algorithm 3.1 only for problems T, 2, 3, 4, and 14 (where the problem numbers are as in (7)). The results appear in Table 1.

Table 1. Comparison of a Code Which Does Not Include the Technique for Handling Singular and Ill-Conditioned Systems to a Code Which Does.

#	n	meth.	NFUN	NJAC	NFNM1	NJNM1	est. total work
T	2	Usual	55	27	0	0	218
		Sing.	122	51	86	34	448
2	2	Usual	118	50	0	0	436
		Sing.	237	95	213	85	854
3	4	Usual	3239	1301	0	0	33772
		Sing.	3225	1296	8	2	33636
4	5	Usual	15048	6440	0	0	236240
		Sing.	1143	501	959	409	18240
14	2	Usual	59	28	0	0	230
		Sing.	223	92	200	82	814
TOTALS:							
		Usual	18519	7846	0	0	270896
		Sing.	4950	2035	1466	612	53992
Ratio:							
Usual/Sing.			3.74	3.86	0.00	0.00	5.02

Percentage of work in (n-1)-space: 23%

In Table 1, NFUN is the total number of interval function evaluations, NJAC is the number of interval Jacobian evaluations, NFM1 is the number of interval function evaluations in Algorithm 3.1, and NJM1 is the number of interval Jacobian evaluations in Algorithm 3.1. The estimated total work is

n^2 * NJAC + n * NFUN where n is the number of variables and functions.

5. Conclusions

Table 1 shows that Algorithm 3.1 may degrade the efficiency in some cases but may improve it dramatically in others. Problem 4 is Brown's almost linear function, which is linear in the first n - 1 components and highly nonlinear in the last; that structure is ideal for Algorithm 3.1. Problem 3 is Powell's singular function, for which the Jacobian matrix is null at the root. A recursive version of Algorithm 3.1 may work better on it.

The overall performance of our scheme may be improved by adjusting certain tuning parameters.

BIBLIOGRAPHY

(1) A. Griewank, "On solving nonlinear equations with simple singularities or nearly singular solutions," SIAM Rev., Vol. 27, pp. 537-564, Dec., 1985.

(2) E. R. Hansen, "On solving systems of equations using interval arithmetic," Math. Comp., Vol. 22, pp. 374-384, 1968

(3) E. R. Hansen and S. Sengupta, "Bounding solutions of systems of equations using interval analysis," BIT, Vol. 21, pp. 203-211, 1981.

(4) E. R. Hansen and R. I. Greenberg, "An interval Newton method," Appl. Math. Comput., Vol. 12, pp. 89-98, 1983.

(5) E. R. Hansen and G. W. Walster, "Nonlinear equations and optimization," submitted to SIAM Rev.

(6) R. B. Kearfott, "Abstract generalized bisection and a cost bound," Math. Comp., Vol. 49, 179 pp. 187-202, July 1987.

(7) R. B. Kearfott, "Some tests of generalized bisection," ACM Trans. Math. Software, Vol. 13, 3, pp. 197-220, Sept. 1987.

(8) R. B. Kearfott and M. Novoa, "A program for generalized bisection," manuscript.

(9) R. E. Moore, "A test for existence of solutions to nonlinear systems," SIAM J. Numer. Anal., Vol. 14, 4, pp. 611-615, Sept. 1977.

(10) R. E. Moore and S. T. Jones, "Safe starting regions for iterative methods," SIAM J. Numer. Anal., Vol. 14, 6, pp. 1051-1065, Dec. 1977.

(11) R. E. Moore and L. Qi, "A successive interval test for nonlinear systems," SIAM J. Numer. Anal., Vol. 19, 4, pp. 845-850, Aug. 1982.

(12) A. Neumaier, "Interval iteration for zeros of systems of equations," BIT, Vol. 25, 1, pp. 256-273, 1985.

(13) J. M. Shearer and M. A. Wolfe, "Some computable existence, uniqueness, and convergence tests for nonlinear systems," SIAM J. Numer. Anal. Vol. 22, 6, pp. 1200-1207, Dec. 1985.