Approximation, Optimization and Computing:
Theory and Applications, A.G. Law and C.L. Wang (eds.)
Elsevier Science Publishers B.V. (North-Holland)

A PIVOTING SCHEME FOR THE INTERVAL GAUSS-SEIDEL METHOD: NUMERICAL EXPERIMENTS

Chen-Yi Hu and R. Baker Kearfott

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

Abstract. Interval Newton methods in conjunction with generalized bisection form the basis of algorithms which find all real roots within a specified box $\mathbf{X} \subseteq \mathbf{R}^n$ of a system of nonlinear equations $F(X) = 0$ *with mathematical certainty*, even in finite precision arithmetic. The practicality and efficiency of such methods is, in general, dependent upon *preconditioning* a certain interval linear system of equations. Here, we present the results of numerical experiments for such a *pivoting* preconditioner we are developing.

1. INTRODUCTION AND ALGORITHMS

The general problem we address is:

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

$$F(X) = (f_1(x_1, x_2, \ldots, x_n), \ldots, f_n(x_1, x_2, \ldots, x_n)) = 0, \tag{1.1}$$

> where bounds $\underline{x}_i$ and $\bar{x}_i$ are known such that

$$\underline{x}_i \le x_i \le \bar{x}_i \text{ for } 1 \le i \le n.$$

We write $X = (x_1, x_2, \ldots, x_n)$, and we denote the box given by the inequalities on the variables x_i by $\mathbf{B}$.

A successful approach to this problem is generalized bisection in conjunction with interval Newton methods, as described in [4], [8] or numerous other works. For an introduction to the interval arithmetic underlying these methods, see [1], [9], the recent review [6], etc. Also, the book [11] will contain an overview of interval methods for linear and nonlinear systems of equations.

In these methods, we first transform $F(X) = 0$ to the linear interval system

$$\mathbf{F}'(\mathbf{X}_k)(\tilde{\mathbf{X}}_k - X_k) = -F(X_k), \tag{1.2}$$

where $\mathbf{F}'(\mathbf{X}_k)$ is a suitable (such as an elementwise) interval extension of the Jacobian matrix over the box $\mathbf{X}_k$ (with $\mathbf{X}_0 = \mathbf{B}$), and where $X_k \in \mathbf{X}_k$ represents a *predictor* or *initial guess* point. (Consult [1], [6], [9], [11], [12], etc. for information on interval extensions.) If we formally solve (1.2) using interval arithmetic, the resulting box $\tilde{\mathbf{X}}_k$, which actually just satisfies

$$\mathbf{F}'(\mathbf{X}_k)(\tilde{\mathbf{X}}_k - X_k) \supset -F(X_k), \tag{1.2(b)}$$

will contain all solutions to all systems

$$A(X - X_k) = -F(X_k),$$

for $A \in \mathbf{F}'(\mathbf{X}_k)$. Also, for suitable interval extensions $\mathbf{F}'(\mathbf{X}_k)$, the mean value theorem implies that $\mathbf{X}_k$ will contain all solutions to $F(X) = 0$. We then define the next iterate $\mathbf{X}_{k+1}$ by

$$\mathbf{X}_{k+1} = \mathbf{X}_k \cap \tilde{\mathbf{X}}_k. \tag{1.3}$$

This scheme is termed an *interval Newton method*.

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 put the other one on a stack for later consideration. The following fact (from [10]) allows such a composite generalized bisection algorithm to compute all solutions to (1.1) *with mathematical certainty*. For many methods of solving (1.2),

> if $\tilde{\mathbf{X}}_k \subseteq \mathbf{X}_k$, then the system of equations in (1.1) has a unique solution in $\mathbf{X}_k$. Conversely, if $\tilde{\mathbf{X}}_k \cap \mathbf{X}_k = \emptyset$ then there are no solutions of the system in (1.1) in $\mathbf{X}_k$. (1.4)

We give complete details of the overall generalized bisection algorithm in [5]. Here, we are interested in the fact that the efficiency of generalized bisection depends on the way we find the solution bounds $\tilde{\mathbf{X}}_k$ to (1.2).

In [2], we derived a pivoting scheme for the *interval Gauss-Seidel* method for computing $\tilde{\mathbf{X}}_k$. We review that scheme and report computational results here. For proofs and additional details, see [2].

We use the following notation. We write

$$\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$$

for $\mathbf{X}_k$ and we let $\mathbf{A}_{i,j}$ be the interval in the i-th row and j-th column of $\mathbf{A} = \mathbf{F}'(\mathbf{X})$. We denote the components of F as boldface intervals, since they must be evaluated in interval arithmetic with directed roundings, so that we have

$F(X_k) = F = (\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n)$, and $X_k = (x_1, x_2, \ldots, x_n)$. In this notation, (1.2) becomes

$$\mathbf{A}(\tilde{\mathbf{X}}_k - X_k) = -F. \tag{1.5}$$

We generally *precondition* (1.5) by a scalar (i.e. non-interval) matrix Y to obtain

$$Y\mathbf{A}(\tilde{\mathbf{X}}_k - X_k) = -YF. \tag{1.6}$$

Let $Y_i = (y_1, y_2, \ldots, y_n)$ denote the i-th row of the preconditioner, let

$$\mathbf{k}_i = Y_i F,$$

and let

$$Y_i\mathbf{A} = \mathbf{G}_i = (\mathbf{G}_{i,1}, \mathbf{G}_{i,2}, \ldots, \mathbf{G}_{i,n}) = (\underline{g}_{i,1}, \bar{g}_{i,1}, \underline{g}_{i,2}, \bar{g}_{i,2}, \ldots, \underline{g}_{i,n}, \bar{g}_{i,n}).$$

Then the *preconditioned interval Gauss-Seidel algorithm* is

ALGORITHM 1. *(Preconditioned version of interval Gauss-Seidel) Do the following for $i = 1$ to n.*

1. *(Update a coordinate.)*
   (a) *Compute Y_i.*
   (b) *Compute $\mathbf{k}_i$ and $\mathbf{G}_i$.*
   (c) *Compute*

$$\tilde{\mathbf{x}}_i = x_i - \frac{\mathbf{k}_i + \sum_{\substack{j=1 \\ j \neq i}}^{n} \mathbf{G}_{i,j}(\mathbf{x}_j - x_j)}{\mathbf{G}_{i,i}} \tag{1.7}$$

   using interval arithmetic.
2. *(The new box is empty.) If $\tilde{\mathbf{x}}_i \cap \mathbf{x}_i = \emptyset$, then signal that there is no root of F in $\mathbf{X}$, and continue the generalized bisection algorithm.*
3. *(The new box is non-empty; prepare for the next coordinate.)*
   (a) *Replace $\mathbf{x}_i$ by $\mathbf{x}_i \cap \tilde{\mathbf{x}}_i$.*
   (b) *Possibly re-evaluate $\mathbf{F}'(\mathbf{X}_k)$ to replace $\mathbf{A}$ by an interval matrix whose corresponding widths are smaller.*

We will denote the *width* of the interval $\mathbf{x}_i = [\underline{x}_i, \bar{x}_i]$ by $w(\mathbf{x}_i) = \bar{x}_i - \underline{x}_i$. Similarly, we will denote the *lower bound* for the box

$$\mathbf{X} = ([\underline{x}_1, \bar{x}_1], [\underline{x}_2, \bar{x}_2], \ldots, [\underline{x}_n, \bar{x}_n]).$$

by $\mathbf{X} \downarrow = (\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_n)$ and the *upper bound* for the box by $\mathbf{X} \uparrow = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)$. We will speak of the *diameter* of the box $\mathbf{X}$ as $\|\mathbf{X} \uparrow - \mathbf{X} \downarrow\|_2$.

The following theorem shows us that preconditioning does not affect the reliability of the overall generalized bisection scheme.

THEOREM 2. *Let $\mathbf{X}^+ = (\mathbf{x}_1^+, \mathbf{x}_2^+, \ldots, \mathbf{x}_n^+)$ denote the new (possibly altered and possibly empty) box which Algorithm 1 returns, and refer to the $\mathbf{X}$ entering Algorithm 1 as simply $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$. Then any roots of F in $\mathbf{X}$ must also be in the new $\mathbf{X}^+$.*

See [5] for a proof of Theorem 2.

The pivoting preconditioner is based on the following characterization.

THEOREM 3. *In (1.7), if $\mathbf{k}_i \in -\sum_{\substack{j=1 \\ j \neq i}}^{n} \mathbf{G}_{i,j}(\mathbf{x}_j - x_j)$ and $0 \notin \mathbf{G}_{i,i}$, then*

$$w(\tilde{\mathbf{x}}_i) = \sum_{\substack{j=1 \\ j \neq i \\ \sigma_j = 1}}^{n} \max\{\, \underline{g}_{i,j}, \bar{g}_{i,j} \,\} w(\mathbf{x}_j^1) - \sum_{\substack{j=1 \\ j \neq i \\ \sigma_j = 0}}^{n} \lambda_j^* w(\mathbf{G}_{i,j}) w(\mathbf{x}_j^1),$$

where

$$\lambda_j^* = \begin{cases} \dfrac{\max\{(x_j - \underline{x}_j), (\bar{x}_j - x_j)\}}{w(\mathbf{x}_j^1)} & \text{if } w(\mathbf{x}_j^1) \neq 0, \\ 0 & \text{otherwise,} \end{cases}$$

where

$$\sigma_j = \begin{cases} 0 & \text{if } \underline{g}_{i,j} \dfrac{1}{\lambda_j^*} - 1 < |\bar{g}_{i,j}| < \underline{g}_{i,j} \dfrac{\lambda_j^*}{1 - \lambda_j^*} \\ & \text{and } 0 \in \mathbf{G}_{i,j} = [\underline{g}_{i,j}, \bar{g}_{i,j}], \\ 1 & \text{otherwise,} \end{cases}$$

and where

$$\mathbf{x}_j^1 = \begin{cases} \mathbf{x}_j \cap \tilde{\mathbf{x}}_j & \text{if } j \leq i, \\ \mathbf{x}_j & \text{if } j > i. \end{cases}$$

The idea in [2] and here is to develop preconditioners which, though not necessarily giving a minimal $w(\mathbf{x}_i)$, require no numerical linear algebra and only $O(n^2)$ arithmetic operations, and are potentially effective on many problems. See [2] and [5] for *rationale* and details. We review the resulting preconditioner here.

Noting that terms in the sums in Theorem 3 for which $w(\mathbf{x}_j^1) = 0$ are irrelevant, we make

DEFINITION 4.

$$\Lambda_{\text{col}} = \{j \in \{1, 2, \ldots, i-1, i-1, \ldots, n\} \mid w(\mathbf{x}_j^1) \neq 0\}.$$

Also, because of considerations explained in [2], our preconditioner algorithm makes use of

DEFINITION 5. $\Lambda_{\text{row}} = \{j \in \{1, 2, \ldots, n\} \mid 0 \notin \mathbf{A}_{j,i}\}$.

Our pivoting preconditioner is based on selecting a row of the interval Jacobian matrix $\mathbf{A}$ to use as the preconditioned row $\mathbf{G}_i$; i.e., on selecting Y_i to be a unit vector. It is optimal in the sense of

DEFINITION 3.5. *We say that Y_i is pivoting first optimal if Y_i yields a $\mathbf{G}_i$ which minimizes $w(\tilde{\mathbf{x}}_i)$, where $\tilde{\mathbf{x}}_i$ is as in (1.7), subject to the condition that Y_i be a unit vector.*

We use the following lemma to determine which column indices m give pivoting first optimal preconditioner.

LEMMA 6. *Suppose there exists a pivoting first optimal preconditioner row $Y_i = e_m^T$. Suppose further that, as in Theorem 3, $\mathbf{k}_i \in -\sum_{\substack{j=1 \\ j \neq i}}^{n} \mathbf{A}_{m,j}(\mathbf{x}_j - x_j)$ and $0 \notin \mathbf{A}_{m,i}$, where $\mathbf{k}_i = \mathbf{f}_m$ is the m-th component of the function value $F(X_k)$.*

Then

$$w(\tilde{\mathbf{x}}_i) = \left[\sum_{\substack{j \in \Lambda_{\text{col}} \\ \sigma_j = 1}} \max\{|\underline{a}_{m,j}|, |\bar{a}_{m,j}|\} w(\mathbf{x}_j^1) + \sum_{\substack{j \in \Lambda_{\text{col}} \\ \sigma_j = 0}} \lambda_j^* w(\mathbf{A}_{m,j}) w(\mathbf{x}_j^1) \right] / \min\{|\underline{a}_{m,i}|, |\bar{a}_{m,i}|\},$$

where

$$\lambda_j^* = \begin{cases} \dfrac{\max\{(x_j - \underline{x}_j), (\bar{x}_j - x_j)\}}{w(\mathbf{x}_j^1)} & \text{if } w(\mathbf{x}_j^1) \neq 0 \\ 0 & \text{otherwise,} \end{cases}$$

where

$$\sigma_j = \begin{cases} 0 & \text{if } |\underline{a}_{m,j}|(1/\lambda_j^* - 1) < \bar{a}_{m,j} < \dfrac{|\underline{a}_{m,j}|\lambda_j^*}{1 - \lambda_j^*} \\ & \text{and } 0 \in \mathbf{A}_{m,j} = [\underline{a}_{m,j}, \bar{a}_{m,j}] \\ 1 & \text{otherwise,} \end{cases}$$

and where

$$\mathbf{x}_j^1 = \begin{cases} \mathbf{x}_j \cap \tilde{\mathbf{x}}_j & \text{if } j \leq i \\ \mathbf{x}_j & \text{if } j > i. \end{cases}$$

Lemma 6 follows from Theorem 3; see [2], and leads to

ALGORITHM 7. *(Determination of the column index for the i-th first optimal pivoting preconditioner row)*

1. *Check* $w(\mathbf{x}_j^1)$ *for* $j = 1, 2, \ldots, i-1, i-1, \ldots, n$ *to find* Λ_{col}.
2. *Check the i-th column of the interval Jacobian matrix* $\mathbf{A}$ *to determine* Λ_{row}.
3. *Pick* $m_0 \in \Lambda_{\text{row}}$ *which minimizes* $w(\tilde{\mathbf{x}}_i)$ *in Lemma 3.8 over all* $m \in \Lambda_{\text{row}}$.
4. *Choose* $Y_i = e_{m_0}^T$ *in Step 1(a) of Algorithm 1.2; where* $e_{m_0}^T$ *is that unit row vector with 1 in the* m_0*-th coordinate and 0 in the other coordinates.*

2. ALTERNATE ALGORITHMS AND COST

The pivoting preconditioner may not result in an optimally small $w(\tilde{\mathbf{x}}_i \cap \mathbf{x}_i)$ in Algorithm 1; see [2]. However, it can sometimes result in reduction of $w(\mathbf{x}_i)$ with less arithmetic operations than other preconditioners. Here, we compare the number of arithmetic operations to complete Algorithm 1, when various different procedures are used to compute Y_i. In particular, we will compare the following schemes:

1) the pivoting preconditioner (Algorithm 7);
2) the minimum width linear programming preconditioner;
3) the inverse midpoint preconditioner;
4) solving for each variable in each equation.

The linear programming preconditioner is explained in [5]. It gives a *minimal* $w(\tilde{\mathbf{x}}_i)$ for each i, but is expensive to obtain. We are unsure of the exact operations count, but on two variable dimension test problems, the total work for the generalized bisection method and possibly also for Algorithm 1.2) seems to increase like $O(n^3)$.

The inverse midpoint preconditioner is commonly used in Algorithm 1. In it, we take Y_i to be the i-th row of the inverse of the n by n non-interval matrix formed by taking the midpoint of each entry of the n by n interval matrix $\mathbf{F}'(\mathbf{X}_k)$. This preconditioner has some nice properties, but is not always appropriate, as explained in [5].

We use the following modification of the interval Gauss-Seidel algorithm when we solve for each coordinate.

ALGORITHM 8. *(Solve for each variable in each equation.) Let* $\mathbf{A} = \mathbf{F}'(\mathbf{X})$. *Do the following for* **for** $m = 1$ **to** n *and for* $i = 1$ *to* n.

1. *Compute*

$$\tilde{\mathbf{x}}_i = x_i - \left[\mathbf{f}_m + \sum_{\substack{j=1 \\ j \neq m}}^{n} \mathbf{A}_{m,j}(\mathbf{x}_j - x_j) \right] \Big/ \mathbf{A}_{m,i}$$

   using interval arithmetic.
2. *If* $\tilde{\mathbf{x}}_i \cap \mathbf{x}_i = \emptyset$, *then signal that there is no root of F in* $\mathbf{X}$, *and continue the generalized bisection algorithm.*
3. *(Prepare for the next coordinate.)*
   (a) *Replace* $\mathbf{x}_i$ *by* $\mathbf{x}_i \cap \tilde{\mathbf{x}}_i$.
   (b) *Possibly re-evaluate* $\mathbf{F}'(\mathbf{X}_k)$ *to replace* $\mathbf{A}$ *by an interval matrix whose corresponding widths are smaller.*

In our operations counts here, we assume a dense interval Jacobian matrix.

We give summary of the arithmetic complexity of the above algorithms, in terms of both non-interval and interval operations, in Table 1. Precise values appear in [2], and are derived in [3]. Values for sparse Jacobian matrices also appear in those two places. We emphasize that the values in Table 1 represent the order of operations to complete an entire n steps of Algorithm 1.

Table 1

Type of op.	Pivoting	Each variable	inverse midpoint
interval $\times$	$O(n^2)$	$O(n^2)$	$O(n^2)$
interval $\div$	$O(n)$	$O(n^2)$	$O(n)$
interval $-$	$O(n^2)$	$O(n^3)$	$O(n^3)$
usual $\times$	$O(n^2)$	0	$O(n^3)$
usual $\div$	$O(n^2)$	0	$O(n^2)$
usual $-$	$O(n^2)$	$O(n^2)$	$O(n^3)$

An alternative to the above three preconditioners is to apply no preconditioner at all, i.e. to always take $m_0 = i$. In that case, there are no scalar arithmetic operations, and the numbers of interval arithmetic operations are the same as for the pivoting preconditioner.

3. EXPERIMENTAL RESULTS

In this section, we report results of some preliminary

experiments on the three preconditioners mentioned in Section 2 and on the interval Gauss-Seidel method with no preconditioner at all (i.e. with Y equal to the identity matrix). The computation involves repeated execution of Algorithm 1 for each box $\mathbf{X}$ produced from the initial box via bisection, until either one of the possibilities in (1.4) happens or else until the maximum width $w(\mathbf{x}_i)$ is smaller than a fixed tolerance (10^{-5} here). See [4], [5], and [7] for details of the overall algorithm and of the way the interval Gauss-Seidel method is combined with generalized bisection.

Here, we examine a variable dimension test problem, which can be solved with no preconditioner at all. This allows us to measure the overhead costs of the preconditioner schemes in a relatively simple way. A more thorough analysis of the types of problems for which the pivoting preconditioner is appropriate appears in [2].

We report total virtual CPU times on an IBM 3090, with a code similar to that in [7], in Table 2. We report the total number of boxes considered in Table 3 and the total number of interval Jacobian evaluations in Table 4; these are measures of the effectiveness of a preconditioner which are independent of the cost to obtain the preconditioner. (For this test problem, no preconditioner and the pivoting preconditioner seem to give the same row indices.) In each case, the first column gives the order of the problem.

The algorithm was unable to finish in eleven CPU hours for the entries in the table marked with asterisks.

Table 2 - CPU times

n	Pivoting	Each variable	inverse midpoint	none
5	0.61	0.50	1.13	0.40
10	4.79	4.51	121.74	1.75
15	21.39	21.73	*	6.46
20	48.45	69.80	*	17.79
25	91.23	144.63	*	34.84
30	149.46	264.20	*	61.21
35	225.47	441.42	*	98.47
40	320.51	680.28	*	148.62

Table 3 - Number of boxes

n	Pivoting	Each variable	inverse midpoint	none
5	45	47	56	43
10	71	73	589	67
15	113	103	*	97
20	141	139	*	133
25	171	169	*	163
30	201	199	*	193
35	231	229	*	223
40	261	259	*	253

Table 4 - Number Jacobian matrix evaluations

n	Pivoting	Each variable	inverse midpoint	none	
5		52	52	29	56
10		82	82	294	86
15		126	116	*	120
20		154	154	*	158
25		184	184	*	188
30		214	214	*	218
35		244	244	*	248
40		274	274	*	278

REFERENCES

[1] Alefeld, G., and Herzberger, J., "Introduction to Interval Computations", (Academic Press, New York, 1983).

[2] Hu, C.-Y., and Kearfott, R. B., *A Width Characterization, Efficiency, and a Pivoting Strategy for the Interval Gauss-Seidel Method*, Math. Comp., submitted (1989).

[3] Hu, C.-Y., "Preconditioners for Interval Newton Methods", Ph.D. dissertation, The University of Southwestern Louisiana (expected in 1990).

[4] Kearfott, R. B., *Some Tests of Generalized Bisection*, ACM Trans. Math. Software **13** 3, 197-220 (1987).

[5] Kearfott, R. B., *Preconditioners for the interval Gauss-Seidel method*, SIAM J. Numer. Anal., to appear.

[6] Kearfott, R. B. *Interval Arithmetic Techniques in the Computational Solution of Nonlinear Systems of Equations: Introduction, Examples, and Comparisons*, to appear in the proceedings of the 1988 AMS SIAM Summer Seminar.

[7] Kearfott, R. B., and Novoa, M, *INTBIS, an Portable Interval Newton / Bisection Package*, to appear in ACM Trans. Math. Software.

[8] Moore, R. E., and Jones, S. T. *Safe Starting Regions for Iterative Methods*, SIAM J. Numer. Anal. **14** 6, 1051-1065 (1977).

[9] Moore, R. E., "Methods and Applications of Interval Analysis" (SIAM, Philadelphia, 1979).

[10] Neumaier, A. *Interval iteration for zeros of systems of equations*, BIT **25** 1, 256-273 (1985).

[11] Neumaier, A., "Interval Methods for Systems of Equations", Cambridge University Press, in press.

[12] Ratschek, H., and Rokne, J. "Computer Methods for the Range of Functions", (Horwood, Chichester, England, 1984).