Reliable Computing **5:** 103–111, 1999.

An Improved Method for Global Solution of Non-Linear Systems

LUBOMIR V. KOLEV
Dept. of Theoretical Electrotechnics, Faculty of Automatica, Technical University of Sofia, 1756 Sofia, Bulgaria, e-mail: LKOLEV@vmei.acad.bg

(Received: 13 July 1998; accepted: 2 October 1998)

Abstract. In this paper, two modifications of an interval method for global solution of systems of non-linear equations are suggested. The first consists in reducing the size of the linear interval system that is to be solved at each iteration of the method. The second incorporates the constraint propagation approach in solving the reduced linear system. The modifications introduced result in a method of improved numerical efficiency.

1. Introduction

Let $\psi : D \subset R^n \to R^n$ be a (at least) continuous function and let $X^0 = (X_1^0, \ldots, X_n^0) \subset D$ be a given interval vector (a box). Consider the following global solution problem (GS problem).

THE GS PROBLEM. *Given the function $\psi : D \subset R^n \to R^n$ and the interval vector (box) $X^{(0)} = (X_1^{(0)}, \ldots, X_n^{(0)}) \subset D$, find the set $S(\psi, X^{(0)}) = (x^{(1)}, \ldots, x^{(p)})$ of all real isolated solutions (zeros) to the system of equations*

$$\psi(x) = 0 \tag{1.1a}$$

which are contained in $X^{(0)}$, i.e. when

$$x \in X^{(0)}. \tag{1.1b}$$

In a recently published paper [3], a new method for tackling the above GS problem was suggested. It is based on the following approach. First, the original system (1.1) is transformed into a larger system of $n' = n + m$ non-linear equations

$$f(x) = 0, \tag{1.2a}$$

$$x \in X^{(0)} \in R^{n'} \tag{1.2b}$$

by introducing m auxiliary variables. Each component $f_i(x)$ of $f(x)$ is in the so-called semiseparable form [3]:

$$f_i(x) = \sum_{j=1}^{n'} f_{ij}(x_j) + \sum_{\substack{k=1 \\ k \neq l}}^{n'} \sum_{l=1}^{n'} \alpha_{kl}^{(i)} x_k x_l \tag{1.2c}$$

(some of the terms $f_{ij}(x_j)$ or/and some of the products $x_k x_l$ may be missing). The GS problem associated with (1.2) is then solved by an iterative method which exploits the specific semiseparable form of (1.2). The method reduces, essentially, to setting up and solving the following linear interval system

$$\tilde{A}^{(v)}x = \tilde{B}^{(v)} \tag{1.3}$$

at each iteration v of the method. Here $\tilde{A}^{(v)}$ is a $(n+m) \times (n+m)$ real matrix while $\tilde{B}^{(v)}$ is a $(n+m)$-dimensional interval vector. The elements of $\tilde{A}^{(v)}$ and $\tilde{B}^{(v)}$ are computed using two specific procedures. Finally, the solution of the original GS problem (1.1) is found through the solution of the augmented system (1.2). Thus, if $x^s \in R^{n'}$ is a solution to (1.2), then the first n components of x^s determine a solution to (1.1).

In this paper, two modifications will be introduced into the computational scheme of the method from [3]. The first is associated with the elimination of all m auxiliary variables from the linear system (1.3). Thus, (1.3) is transformed to a system

$$A^{(v)}y = B^{(v)} \tag{1.4}$$

of reduced $n \times n$ size. The second modification consists of applying the constraint propagation approach to the reduced system (1.4). These modifications result in a considerable improvement of the numerical efficiency of the original method from [3]. Numerical examples illustrating the improved version are provided.

2. Improvements

2.1. Elimination of the Auxiliary Variables

To present this modification, system (1.3) will be rewritten in partioned form

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \tag{2.1}$$

where the vector y_1 corresponds to the original vector $x \in R^n$ while the components of y_2 correspond to the auxiliary variables. For simplicity, A_{22} will be assumed to be the unit matrix E (which, typically, is the case although the example in Section 2 of [3] shows that, in general, $A_{22} \neq E$).

Eliminating y_2 in (2.1), we get

$$Cy_1 = B_1 - A_{12}B_2 \tag{2.2a}$$

where

$$C = A_{11} - A_{12}A_{21}. \tag{2.2b}$$

Hence

$$Y_1 = C^{-1}B_1 - C^{-1}A_{12}B_2. \tag{2.3}$$

Now the following $n \times n'$ real matrix

$$D = [D_1, D_2] \tag{2.4a}$$

is computed with

$$D_1 = C^{-1}, \quad D_2 = C^{-1}A_{12} \tag{2.4b}$$

whose partitioning corresponds to the partitioning of the interval vector B into B_1 and B_2. Thus, from (2.3) and (2.4) we finally obtain

$$Y_1 = DB. \tag{2.5}$$

The above formula seems to be preferable to the simpler expression

$$Y_1' = C^{-1}F \tag{2.6a}$$

where the interval vector F has been computed from (2.2a) as

$$F = B_1 - A_{12}B_2 \tag{2.6b}$$

Indeed, because of the subdistributivity property of interval arithmetic, the interval vector Y_1 will be, in general, narrower than the interval vector Y_1'.

Once Y_1 is computed, the remaining part Y_2 of the interval solution Y to (2.1) could be easily found (for $A_{22} = E$) as

$$Y_2 = B_2 - A_{21}Y_1 \tag{2.7}$$

or

$$Y_2 = -A_{21}D_1B_1 + [E + A_{21}D_2]B_2. \tag{2.8}$$

However, there is a better possibility. It is based on replacing Y_1 in (2.7) by the interval vector

$$Z_1 = Y_1 \cap X_1 \tag{2.9}$$

where X_1 is that part of X which corresponds to the original variables x_i, $i = 1, \ldots, n$. Since

$$Z_1 \subset Y_1 \tag{2.10}$$

it follows that

$$Y_2' = B_2 - A_2Z_1 \tag{2.11}$$

is a better choice because

$$Y_2' \subset Y_2. \tag{2.12}$$

The incorporation of the above modification in the computational scheme of the method from [3] (denoted further as method M1) results in a better performance of the method as illustrated by the following example.

EXAMPLE 2.1. The problem is to find all real solutions to the system

$$\begin{aligned} x_1 + 2x_2 + x_3 - 6 &= 0, \\ 2x_1x_2 + x_2x_3 - 6 &= 0, \\ x_1x_2x_3 - 3 &= 0 \end{aligned} \tag{2.13a}$$

contained in the box $X^{(0)}$ with components

$$X_i^{(0)} = [0,4], \quad i = 1, \dots, 3. \tag{2.13b}$$

The last equation in (2.13a) is not in semiseparable form. It can, however, be readily transformed into such form by introducing an auxiliary variable

$$x_4 = x_2 x_3. \tag{2.14}$$

Thus, we get the augmented system

$$\begin{aligned} x_1 + 2x_2 + x_3 - 6 &= 0, \\ 2x_1 x_2 + x_4 - 6 &= 0, \\ x_1 x_4 - 3 &= 0, \\ x_2 x_3 - x_4 &= 0. \end{aligned} \tag{2.15a}$$

System (2.15a) is now in semiseparable form. It corresponds to system (1.2a) for the example considered with $n = 3$ and $m = 1$.

To solve (2.15a), we have to introduce bounds on the auxiliary variables x_4. From (2.14) and (2.13b)

$$X_4^{(0)} = [0, 16]. \tag{2.15b}$$

Now system (2.15) can be solved using method M1. The linear system approximating (2.15a) in a current box X is

$$\begin{aligned} x_1 + 2x_2 + x_3 &= 6, \\ \tilde{a}_{21}x_1 + \tilde{a}_{22}x_2 + \tilde{a}_{23}x_3 &= \tilde{B}_2, \\ \tilde{a}_{31}x_1 + \tilde{a}_{34}x_4 &= \tilde{B}_3, \\ \tilde{a}_{42}x_2 + \tilde{a}_{43}x_3 + \tilde{a}_{44}x_4 &= \tilde{B}_4. \end{aligned} \tag{2.16}$$

By eliminating x_4 in (2.16) we get the reduced system

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= b_1, \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 &= B_2, \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 &= B_3 \end{aligned} \tag{2.17a}$$

where

$$a_{ij} = \tilde{a}_{ij}, \quad B_i = \tilde{B}_i \tag{2.17b}$$

for the second equation while for the last equation

$$a_{31} = \tilde{a}_{31}, \quad a_{32} = -\tilde{a}_{34}\tilde{a}_{42} / \tilde{a}_{44}, \quad a_{33} = -\tilde{a}_{33}\tilde{a}_{43} / \tilde{a}_{44}, \tag{2.17c}$$

$$B_3 = \tilde{B}_3 + \tilde{a}_{34}\tilde{B}_4 / \tilde{a}_{44}. \tag{2.17d}$$

The above example was solved using Algorithms A1 and A2 from [3] (which are based on setting up and solving system (2.16) at each iteration). It has also been solved by a new algorithm denoted as A3 which incorporates the approach of

Table 1.

Algorithm	A1	A2	A3
N_i	206	123	88
t (sec)	0.514	0.327	0.141
N_{cl}	6	5	3

eliminating the auxiliary variables and is, therefore, based on solving the reduced system (2.17). All three algorithms locate infallibly the two solutions of (2.13)

$$x^{(1)} = (1.0, 1.5, 2.0), \quad x^{(2)} = (3.261, 0.779, 1.181) \tag{2.18}$$

(where the solution components are recorded to three decimal places). The desired accuracy ε was chosen to be 10^{-4} (ε is the width of the box enclosing each solution).

Data illustrating the improved numerical efficiency of Algorithm A3 as compared to A1 and A2 are given in Table 1.

In Table 1, N_i is the number of iterations required to solve the problem considered within the given accuracy, t (in seconds) is the corresponding execution time for a Pentium 166 MHz computer and N_{cl} is the number of cluster boxes (boxes generated additionally to the two solution boxes by the respective algorithm).

For comparison, the same example has been solved by Krawczyk's method also (its improved componentwise version [1], [2], [4]). It is worthwhile noting that the latter method required $N_i = 1156$ iterations to solve the GS problem considered.

2.2. Use of Constraint Propagation

The second modification is related to the application of the constraint propagation approach in solving the reduced system (2.2).

System (2.2) is written for simplicity of notation in the form

$$Cy = B. \tag{2.19}$$

Actually, (2.19) stands for

$$Cy = b, \quad b \in B. \tag{2.20a}$$

Taking into account the fact that y must remain in the current box X, (2.20a) is to be completed with the condition

$$y \in X. \tag{2.20b}$$

The problem is to find an interval solution to (2.20), that is an interval vector Y which contains the solution set of (2.20)

$$S(C, B, X) = \{y : Cy = b,\ b \in B,\ y \in X\}. \tag{2.21}$$

The optimal interval solution Y^* will be the smallest interval solution still containing $S(C, B, X)$. It is readily seen that each component $Y_i^* = [\underline{y}_i^*, \overline{y}_i^*]$, $i = 1, 2, \ldots, n$ can be determined by solving two linear programming problems:

$$
\begin{aligned}
&y_i = \min, \\
&Cy - b = 0, \quad y \in X, \quad b \in B
\end{aligned}
\tag{2.22}
$$

and

$$
\begin{aligned}
&y_i = \max, \\
&Cy - b = 0, \quad y \in X, \quad b \in B
\end{aligned}
\tag{2.23}
$$

to find the endpoints $\underline{y}_i^*$ and $\overline{y}_i^*$, respectively. Thus, computing Y^* would require the solution of $2n$ linear programming problems. Such an approach to tackling the GS problem considered seems to be rather costly since Y^* is to be computed at each iteration v. Therefore, a simpler approach will be adopted here which is based on computing a tight interval solution Y in a cheap manner. This is made possible by resorting to the constraint propagation method as a preliminary stage in solving (2.20). Several algorithms implementing the latter approach will be presented now.

ALGORITHM A4. In this algorithm, Y is determined as follows. First, the optimal solution Y' of (2.20a) is computed

$$Y' = C^{-1}B. \tag{2.24}$$

Then

$$Y = Y' \cap X. \tag{2.25}$$

Now Y is remained X and the iterative process continues until convergence criteria are met. As is seen, the above algorithm does not resort to the constraint propagation approach and is included here only for the purpose of comparing its numerical efficiency with that of the algorithms to be presented below.

ALGORITHM A5. This is an algorithm that is based on the following procedure involving two stages.

Stage A. For $i = 1$ to n do

$$Y_i = \frac{1}{c_{ii}}\left[B_i - \sum_{\substack{j \neq i \\ j=1}}^{n} c_{ij}X_j\right], \tag{2.26a}$$

$$X_i := Y_i \cap X_i. \tag{2.26b}$$

It is seen that this stage implements the known interval Gauss-Saidel scheme [1], [2], [4] (in fact, (2.26) is a simpler version since unlike other interval methods now all the coefficients c_{ij} are real numbers rather than intervals).

Stage B. Now procedure (2.24), (2.25) from Algorithm A4 is applied to the box X obtained on exit from Stage A.

ALGORITHM A6. This is an extended version of the previous algorithm, in which the first stage is modified as follows.

Stage A. For $i = 1$ to n do

For $j = 1$ to n do

$$Y_i = \frac{1}{c_{ij}} \left[B_i - \sum_{\substack{k \neq j \\ k=1}}^{n} c_{ik} X_k \right], \tag{2.27a}$$

$$X_j := Y_j \cap X_j. \tag{2.27b}$$

(In actual computation, (2.27a) is implemented in a more efficient manner by first computing

$$S_1 = B_i - \sum_{\substack{k \neq j \\ k=1}}^{n} c_{ik} X_k = [\underline{S}_1, \overline{S}_1]. \tag{2.28}$$

Then

$$\underline{S}_2 = \underline{S}_1 + \underline{c_{i1} X_1} - \underline{c_{i2} X_2} \tag{2.29a}$$

and

$$\overline{S}_2 = \overline{S}_1 + \overline{c_{i1} X_1} - \overline{c_{i2} X_2}. \tag{2.29b}$$

The next sums S_j, $j > 2$ are computed in a similar way.)

Stage B. The same as in Algorithm A5.

In the algorithms presented so far, all elements of the real matrix C and the interval vector B are computed at the start of the current iteration and remain unchanged during the iteration. A better, equationwise (row by row) computation of C and B is implemented in the next algorithm.

ALGORITHM 7. In this algorithm, stage A is modified in the following manner. Initially for $i = 1$, we compute the first row of C and the first element of B using the current box X. We then apply (2.27) to (hopefully) reduce X to a new box X'. Now X' is renamed X and the second row of C and the second element of B are determined. Now (2.27) is applied with $i = 2$. This process continues until $i = n$.

Stage B. The same as in Algorithm A6.

To illustrate the efficiency of the above algorithms, a numerical example will be considered.

EXAMPLE 2.2. The system to be solved is

$$\begin{aligned} x_3(6x_1^5 - 25 \cdot 2x_1^3 + 24x_1 + 6x_2) - 32x_1x_4 + x_5(39x_1^2 - 145) + x_2x_6 &= 0, \\ x_3(6x_1 + 12x_2) - 50x_2x_4 + 85x_5 + x_1x_6 &= 0, \\ x_4(16x_1^2 + 25x_2^2 - 1) &= 0, \\ x_5(13x_1^3 - 145x_1 + 85x_2 - 400) &= 0, \\ x_6(x_1x_2 - 4) &= 0, \\ x_3 + x_4 + x_5 + x_6 - 1 &= 0 \end{aligned} \tag{2.30a}$$

Table 2.

Algorithm	A4	A5	A6	A7
N_i	3233	2780	1354	473
t (sec)	6.86	5.68	2.97	1.18

and stems from a global optimisation problem [2]. The initial box $X^{(0)}$ has the following components

$$X_1^{(0)} = X_2^{(0)} = [-2, 4], \quad X_i^{(0)} = [0, 1], \quad i = 3, \ldots, 6. \tag{2.30b}$$

The GS problem considered has 9 solutions:

$$\begin{aligned}
x^1 &= (-1.7475, & 0.8738, && 1, && 0, && 0, \ 0),\\
x^2 &= (-1.075, & 0.5353, && 1, && 0, && 0, \ 0),\\
x^3 &= (-0.2398, & -0.05648, && 0.5716, && 0.4284, && 0, \ 0),\\
x^4 &= (0.06604, & -0.1929, && 0.8341, && 0.1659, && 0, \ 0),\\
x^5 &= (0.2398, & 0.05648, && 0.5716, && 0.4284, && 0, \ 0),\\
x^6 &= (0, & 0, && 1, && 0, && 0, \ 0)\\
x^7 &= (-0.06604, & 0.1929, && 0.8341, && 0.1659, && 0, \ 0),\\
x^8 &= (1.075, & -0.5353, && 1, && 0, && 0, \ 0),\\
x^9 &= (1.7475, & -0.8738, && 1, && 0, && 0, \ 0).
\end{aligned}$$

They have been located by Algorithms A4 to A7 with $\varepsilon = 10^{-4}$. No cluster effect has been observed. Data about number of iterations and execution time (for a Pentium 166 MHz) required by each algorithm are given in Table 2.

It is seen that Algorithm A7 has relatively the best performance. It is interesting to compare the above results with the result obtained by Krawczyk's method. For the same accuracy the latter method required $N_i = 33089$ iterations to solve the GS problem considered.

3. Conclusion

Two modifications in the computational scheme of the interval methods proposed in [3] for global solution of systems of n non-linear equations have been suggested. The first consists in eliminating the m auxiliary variables from the augmented linear interval system (1.3) of size $(n+m) \times (n+m)$. This leads to a reduced $n \times n$ system (2.2) and results in speeding up the computational process. The second improvement is associated with the application of the constraint propagation approach in reducing the size of the box corresponding to the original variables at each iteration of the method as implemented in Stage A of the Algorithm A7 suggested in this paper. The combined effect of the two modifications results in a method of improved efficiency which is confirmed by a numerical example.

Acknowledgement

This work was supported by the Bulgarian Research Foundation under Contract No TN-554/97.

References

1. Alefeld, G. and Herzberger, J.: *Introduction to Interval Computations*, Academic Press, New York, 1983.
2. Hansen, E.: *Global Optimization Using Interval Analysis*, Marcel Dekker Inc., New York, 1992.
3. Kolev, L.: A New Method for Global Solution of Systems of Non-Linear Equations, *Reliable Computing* 4 (2) (1998), pp. 125–146.
4. Neumaier, A.: *Interval Methods for Systems of Equations*, Cambridge University Press, London, 1990.