AN INTERVAL METHOD FOR GLOBAL NON-LINEAR ANALYSIS

L. Kolev

Abstract - In this paper, the problem of finding the set of all real solutions to a system of $n$ non-linear general form algebraic equations contained in a given $n$-dimensional box (the global analysis problem) is considered. A new iterative interval method for solving the global analysis problem is suggested. It is based on the following techniques: (i) transformation of the original system into an augmented system of $n' = n + m$ equations of $n'$ variables by introducing $m$ auxiliary variables, the augmented system being of the so-called semiseparable form; (ii) enclosure of the nonlinear augmented system at each iteration by a specific linear interval system of size $n' \times n'$; (iii) elimination of the auxiliary variables; (iv) solution of the resulting reduced size $n \times n$ linear system, using the so-called constraint propagation approach. The overall efficiency of the method suggested is illustrated by several numerical examples.

I. INTRODUCTION

The problem of finding the set of all real isolated solutions to a system of $n$ non-linear general form algebraic equations of $n$ variables is one of the fundamental problems in science and technology. In the field of circuit and system analysis and design, typical applications are global analysis of resistive non-linear circuits, synthesis of linear passive and active circuits, fault diagnosis of linear circuits, determination of equilibria in neural networks, load flow in power systems and computer graphics. Solving systems of algebraic equations is also an integral part of many algorithms for global optimization.

Over the last few decades, numerous methods have been proposed for tackling the above global non-linear analysis problem. They can be categorized in the following two major groups:

i). continuation (homotopy) methods [1]-[4];
ii) interval methods [5]-[13].
The methods of the first group solve the global analysis problem solely in the special case where the non-linear functions in the system considered are multipolynomials [4]; in the general case of arbitrary functions they only are globally convergent to one or more solutions without guaranteeing localization of all solutions.

Presently, interval methods (methods based on interval analysis techniques [5]-[7]) seem to be the only methods which are capable of infallibly solving the global analysis problem (GA problem) for arbitrary functions. The problem can be formulated as follows (cf.[13] and the references therein cited).

The GA problem. Given the function \( f: D \subset \mathbb{R}^n \rightarrow \mathbb{R}^n \) and the interval vector (box) \( X^{(0)} = (X_1^{(0)}, ..., X_n^{(0)}) \subset D \), find the set \( S(f, X^{(0)}) = (x^{(1)}, ..., x^{(p)}) \) of all real isolated solutions (zeros) to the system of equations

\[
f(x) = 0
\]

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

\[
x \in X^{(0)}
\]

where each component \( X_i^{(0)} \) of \( X^{(0)} \) is an interval

\[
X_i^{(0)} = [\underline{x}_i^{(0)}, \overline{x}_i^{(0)}]
\]

The initial box (a \( n \)-dimensional rectangular region in \( \mathbb{R}^n \) with sides parallel to the co-ordinate axes) \( X^{(0)} \) is chosen large enough to enclose all solutions to (1a) in \( \mathbb{R}^n \). Recommendations as to how to choose \( X^{(0)} \) in the case of non-linear resistive circuits are given in [14], [15].

However, at their present stage of development, all interval methods known to date suffer from a serious drawback which severely limits their applicability, namely their numerical complexity grows too rapidly with the dimension \( n \) of the system and the size of the initial search region \( X^{(0)} \). Thus, for a system of 9 equations (representing an accurate Ebers-Moll model of a transistor) and a relatively small initial region \( X^{(0)} \) with \( X_i^{(0)} = [0, 10] \), a method suggested in [16] requires billions of functions evaluations in interval form to locate the unique zero of the GA problem considered and to prove uniqueness computationally.

Several attempts to improve the numerical efficiency of the interval methods have already been made. One is based on the use of the so-called modified interval extensions [17] of the functions
involved which have reduced overestimation. An alternative approach is associated with using interval slopes [18]-[21] rather than interval derivatives in evaluating the interval extensions. However, experimental evidence showed that the overall improvement of the interval methods efficiency obtained along these lines is still not satisfying, especially for more complex problems of larger \( n \) and initial search region [12], [22].

Recently, a more efficient interval method for global analysis has been developed [13] for the class of non-linear circuits and systems described by systems of separable form

\[
f_i(x) = \sum_{j=1}^{n} f_{ij}(x_j), \quad i = 1, \ldots, n
\]

In this paper, a new interval method for solving the GA problem considered is suggested. It is based on the following approach. First, by adding a certain number \( m \) of auxiliary variables, the original general form system (1) is transformed into a larger system of the special form

\[
f_i(x) \approx \sum_{j=1}^{n'} f_{ij}(x_j) + \sum_{k=1, l=1 \atop k \neq l}^{n', n'} \alpha_{kl}^{(i)} x_k x_l, \quad i = 1, \ldots, n', \quad n'' = n + m,
\]

defined in a \( n'' \)-dimensional box, i.e, when

\[
x \in X
\]

(some of the terms \( f_{ij}(x_j) \) or/and some of the products \( x_k x_l \) may be missing). Expressions (3a) differ from (2) in that now they include additionally the products \( x_k x_l \). If all the terms involving products are missing, we get a function in separable form. For this reason, the representation (3a) will be referred to as a system in semiseparable form. Next, the method [13] treating the separable form case (2) is generalized to cover the augmented size semiseparable form system (3). Finally, two important modifications in the computational scheme of the generalized method are introduced: (i) elimination of the auxiliary variables and (ii) improved solution of the resulting reduced size \( n \times n \) linear system.

The paper is organized as follows. Section II presents the transformation of the general form system (1) to the system of semiseparable form (3). In Section III, the previous method [13] is extended for the case of semiseparable form system (3). The improvements in the computational scheme of the extended method are presented in the next section. The overall efficiency of the resulting method is illustrated by several examples. Concluding remarks are given in Section V.
II. TRANSFORMATION TO SEMISEPARABLE FORM

In this section, it will be shown how a system of general form algebraic equations (1) can be transformed into a system if semiseparable form (3). More specifically, the system's components $f_i$ are assumed to be composed of four arithmetic operations ($+$, $-$, $\ast$, $/$), unary operations (sin, exp, log, sqrt, abs, etc.) and the power operation ($^\wedge$).

The transformation of (1) to (3) includes two stages: (i) transforming (1a) to (3a) and (ii) transforming (1b) to (3b).

2.1 Transforming the system of equations

The approach herein adopted is to transform the general form system (1a) to the semiseparable form (3a). The theoretical basis for such a transformation is a famous theorem due to a Kolmogorov's paper [23] as early as 1957. However, its proof is not constructive and only recently has an algorithm been proposed [24] to convert general form functions into separable functions automatically by computer. The present approach is a simplification of the algorithm from [24] which takes into consideration the specificity of the semiseparable form.

Let $f_L$ and $f_R$ be subexpressions of $f_i$ containing at least one variable. Consider the following four cases

\begin{align*}
  f &= f_L + f_R \\
  f &= f_L \cdot f_R \\
  f &= f_L / f_R \\
  f &= (f_L)^{f_R}
\end{align*}

If both $f_L$ and $f_R$ contain only one and the same variable then $f$ is obviously semiseparable in all the four cases.

If $f_L$ contains only one single variable and $f_R$ contains another single variable then only (4) is semiseparable. The remaining 3 cases can, however, be easily transformed into semiseparable form. Consider first the product (5). By introducing two auxiliary variables, (5) is transformed into semiseparable form as follows

\begin{align*}
  f &= f_L \cdot f_R \\
  f &= y_1 \cdot y_2 \\
  y_1 &= f_L \\
  y_2 &= f_R
\end{align*}
The third case can be easily reduced to the second by replacing $y_2 = f_R$ with $y_2 = 1/f_R$. It should be stressed that this is only possible if the original function $f_R \neq 0$ for all values of its argument. For the last case the transformation to semiseparable form is as follows

$$ f = (f_L f_R) \rightarrow \begin{cases} f' = \exp(y) \\ y = f_R \cdot \log(f_L) \end{cases} \quad (9) $$

where the product in (9) must be transformed using (8). It should be mentioned that (9) is only valid if $f_L > 0$ for all values of its argument.

If both $f_L$ and $f_R$ contain more than one variable we first introduce auxiliary variables and then apply the above approach. For instance, case (5) is transformed using (8). Now, by representing $f_L$ and $f_R$ in semiseparable form, $f$ can be put into semiseparable form.

In order to make $f_L$ and $f_R$ semiseparable, we perform the above transformation to $f_L$ and $f_R$, regarding them as $f$. This process can be implemented as a computer program (see [24] for the more complex case of transformation to separable form).

To illustrate the above approach we shall consider an example.

**Example 1.** Consider the system

$$ 1 - x_3 - x_4 + 2x_1(x_3 + x_4) = 0 $$
$$ 2x_2x_3 - x_4 = 0 $$
$$ x_3(x_1^2 + x_2^2 - 1) = 0 $$
$$ x_4(x_1^2 - x_2) = 0 $$

where $x = (x_1, \ldots, x_4)$ belongs to the initial box $X^{(0)}$ with components

$$ X_1^{(0)} = [-1, 1], \quad X_2^{(0)} = [0, 1], \quad X_3^{(0)} = [0, 1], \quad X_4^{(0)} = [0, 1] \quad (10b) $$

The problem is to transform (10) to an equivalent system of semiseparable form (3).

It is seen that only the first two equations of system (10a) are semiseparable. To transform the last two equations into semiseparable form, we introduce two auxiliary variables

$$ x_5 = x_1^2 + x_2^2 - 1 \quad (11a) $$
$$ x_6 = x_1^2 + x_2 \quad (11b) $$

On account of (10a) and (11), we get
System (12) is now in semiseparable form. However, while the original system (10a) has 4 equations of 4 variables, the equivalent system (12) has 6 equations of 6 variables. For this reason, (12) will be referred to as the augmented system.

So far, we have transformed (10a) to the semiseparable form system (12a). To complete the solution of Example 1, we need also to enlarge the initial 4-demensional box \( X^{(0)} \) to a corresponding 6-dimensional box \( X^{(0)} \).

2.2. Transforming the initial box

We first consider the transformation of \( X^{(0)} \) in Example 1. With this in mind, we have to compute bounds \( X_5 \) and \( X_6 \) on the auxiliary variables. Using (10b) and (11), it is easily seen that

\[
X_5^{(0)} = X_1^2 + X_2^2 - 1 = [-1, 1], \quad X_6^{(0)} = X_1^2 + X_2 = [-1, 1]
\]

Based of this example, we now proceed to considering the general case. Let the augmented system of separable form be denoted as

\[
f'(x) = 0
\]

where \( \tilde{x} = (x_1, \ldots, x_n) \) is the augmented vector of variables and \( n' = n + m \), with \( m \) being the number of auxiliary variables needed to transform (1a) into (3a). For the above example \( n = 4, \ m = 2, \ n' = 6 \) and \( \tilde{x} = (x_1, \ldots, x_6) \). It is necessary to determine the initial box \( X^{(0)} \) for \( \tilde{x} \) in order to complete (13a) with the condition

\[
\tilde{x} \in X^{(0)}
\]

To do this, we partition the augmented vector \( \tilde{x} \) into two parts

\[
x = (x_1, \ldots, x_n)
\]

\[
x_a = (x_{n+1}, \ldots, x_{n'})
\]
corresponding to the original and the auxiliary variables, respectively. Thus, the augmented vector can be put in the form
\[ \tilde{x} = (x, x_a) \]  
(15)

In the general case, the auxiliary variables and the original variables are related by the function
\[ f_a : R^n \rightarrow R^m, \] i.e.
\[ x_a = f_a(x) \]  
(16)

In Example 1, the function \( f_a \) is given by (11). We can compute bounds \( X_a \) on \( x_a \) for any box \( X \subseteq X^{(0)} \) by evaluating the interval extension \( F_a(X) \) of \( f_a(x) \) when \( x \in X \), i.e.
\[ X_a = F_a(X), \quad X \subseteq X^{(0)} \]  
(17)

Thus, we can compute the interval vector
\[ X_a^{(0)} = F_a(X^{(0)}) \]  
(18)

which determines the bounds on the auxiliary variables. Finally, the initial box is given in partitioned form by the interval vector \( \tilde{X} \)
\[ \tilde{X}^{(0)} = (X^{(0)}, X_a^{(0)}) \]  
(19)

Thus, it has been shown that the original problem (1) involving general form algebraic equations can be transformed into a corresponding augmented system (13). The components \( f_i \) in (13a) are in the semiseparable form (3a) and the initial box \( \tilde{X} \) related to (13a) can be computed using (18) and (19).

Once the transformation of (1) to (13) is completed, the solution of the original GA problem can be equated to that of finding all real solutions to (13). Indeed, let \( \tilde{x}^* = (x^*, x_a^*) \) be a solution to (13). Clearly, the vector \( x^* \) in \( \tilde{x}^* \) provides a solution to the original problem (1). Thus, our next objective is to devise an efficient method for solving systems of semiseparable form.

III. SOLVING SEMISEPARABLE FORM SYSTEMS

In this section, an interval method for solving the semiseparable form system (3) will be suggested. It is an extension of a previous method [13] developed for the special case of separable functions. More
specifically, the idea to enclose the separable terms in (3a) by an appropriate linear interval function [13] is generalized to encompass the terms representing products of two variables.

To simplify notation, the symbols \( x, X \in X^{(0)} \), \( X^{(0)} \) will stand for the \( n' \)-dimensional vectors \( x, X, X^{(0)} \), respectively, throughout this section.

### 3.1. Enclosures for semiseparable functions

To maintain completeness, first the linear interval approximation for the separable terms will be briefly presented.

**A. Enclosure for** \( f_{ij}(x_j) \)

Let \( X = (X_1, \ldots, X_{j'}, \ldots, X_{n'}) \subseteq X^{(0)} \). The linear interval enclosure (approximation) of \( f_{ij}(x_j) \) in \( X_j \) suggested earlier [13] is in the form

\[
L_{ij}(X_j) = B_{ij} + a_{ij}x_j, \quad x_j \in X_j
\]

where \( B_{ij} \) is an interval while \( a_{ij} \) is a real number. Both \( B_{ij} \) and \( a_{ij} \) (which, in fact, depend on \( X \)) are determined such that the following inclusion property holds

\[
f_{ij}(x_j) \in B_{ij} + a_{ij}x_j, \quad x_j \in X_j
\]

(It is on account of (21) that \( L_{ij}(X) \) is called a linear interval enclosure (approximation) of \( f_{ij}(x_j) \)). A procedure for determining \( a_{ij} \) and \( B_{ij} \) which will be called Procedure 1 has been suggested in [13].

Remark 1. In the original paper [13], no restrictions on the functions \( f_{ij}(x_j) \) are imposed except for the requirement that they be continuous. The procedure 1 therein suggested for determining the enclosure (20) is applicable for the case of continuously differentiable (CD) functions and piecewise linear (PWL) functions. It can, however, be easily shown (using simple geometrical considerations as in [13]) that the linear interval enclosure (20) can also be constructed even in the case of discontinuous functions having bounded discontinuities.

**B. Enclosure for** \( x_k x_l \)

To simplify notation, we shall consider the product
\[ xy, \ x \in X, \ y \in Y \]
(where \( X \) and \( Y \) are intervals) rather than the product \( x_kx_l \). If \( x_0 \) and \( y_0 \) are the centres of \( X \) and \( Y \), respectively, then
\[
xy = (x_0 + u)(y_0 + v) = x_0y_0 + y_0u + x_0v + uv = -x_0y_0 + y_0x + x_0y + uv \tag{22}
\]
When \( x \in X \) and \( y \in Y \), the centred variables \( u \in [-R_x, R_x] \) and \( v \in [-R_y, R_y] \) where \( R_x, R_y \) are the radii of \( X \) and \( Y \). Let \( R = R_x \cdot R_y \); it follows from (22) that
\[
xv \in -x_0y_0 + y_0x + x_0y + [-R, R], \ x \in X, \ y \in Y \tag{23}
\]
Thus, the product \( xv \) has been enclosed by a linear interval expression since
\[
xv = \alpha x + \beta y + B_{xy} \tag{24a}
\]
where \( \alpha = y_0, \ \beta = x_0 \) are real numbers while
\[
B_{xy} = -x_0y_0 + [-R, R] \tag{24b}
\]
is an interval.

Returning to the original term \( \alpha^{(i)}_{kl}x_kx_l \) of the \( i \)th function \( f_i(x) \) in (3a), it is easily seen that the corresponding linear interval enclosure denoted as \( L^{(i)}_{kl}(X_k,X_l) \) has the form
\[
L^{(i)}_{kl}(X_k,X_l) = \alpha_{ik}x_k + \alpha_{il}x_l + B_{kl}^{(i)} \tag{25a}
\]
with
\[
\alpha_{ik} = \alpha_{kl}^{(i)}x_k^{(i)}, \ \alpha_{il} = \alpha_{kl}^{(i)}x_l^{(i)} \tag{25b}
\]
\[
B_{kl}^{(i)} = \alpha_{kl}^{(i)}B_{kl}^{(i)} \tag{25c}
\]
where \( x_k^{(i)}, x_l^{(i)} \) are the centres of \( X_k \) and \( X_l \), respectively, while \( B_{kl}^{(i)} \) is an interval and is computed as in (24b) (with \( x_0, y_0 \) and \( R = R_xR_y \) replaced by \( x_k^{(0)}, x_l^{(0)} \) and \( R = R_kR_l \)). On account of (24a)
\[
\alpha_{kl}^{(i)}x_kx_l \in \alpha_{ik}x_k + \alpha_{il}x_l + B_{kl}^{(i)} \tag{26a}
\]
when
\[
x_k \in X_k, \ x_l \in X_l \tag{26b}
\]
The above procedure for determining the "slopes" \( \alpha_{ik}, \alpha_{il} \) and the interval \( B_{ik} \) will be referred to as Procedure 2.

3.2. The generalized method
Using the semiseparable representation (3a) and the inclusions (21) and (26), it is seen that the following inclusion is valid
\[ f_i(x) \in \sum_{j=1}^{n} a'_{ij}x_j + b_i^1, \quad i = 1, \ldots, n', \quad x \in X \]  
(27a)

where
\[ a'_{ij} = a_{ij} + \alpha_{ij} \]  
(27b)
\[ b_i^1 = \sum_j B_{ij} + \sum_{l<k} B_{lk}^{(l)} = [b^1_i, b^1_i] \]  
(27c)

Now a real \( n' \times n' \) matrix
\[ A = \{a'_{ij}\} \]  
(28)
is introduced and a \( n' \)-dimensional interval vector
\[ b^1 = (b_1^1, \ldots, b_{n'}^1) \]  
(29)
is formed. Thus, (27a) can be put in vector form
\[ f(x) \in Ax + b^1, \quad x \in X \]  
(30)

If \( y \) is a solution of (3), then \( f(y) = 0 \) and by (30)
\[ 0 \in Ay + b^1, \quad y \in X \]  
(31)

Now we can state the main result of the section.

**Theorem 1.** All the solutions \( y \) to
\[ f(x) = 0, \quad x \in X \]  
(32)
with \( f_i(x) \) given by (3a), which are contained in the \( n' \)-dimensional box \( X \) are also contained in the solution set \( S(X) \) of the system
\[ Ax + b = 0, \quad b \in b^1 \]  
(33)

where \( b \) is any real vector contained in \( b^1 \).

The proof of the present theorem is similar to that of Theorem 1 in [13] and will therefore be omitted.

**Remark 2.** It should be stressed that \( A \) and \( b^1 \) depend on the box \( X \), i.e. (33) should read
\[ A(X) + b = 0, \quad b \in b^1(X) \]  
(33')

To simplify notation, here and henceforth the shorter form \( A \) and \( b^1 \) is used.
Formally, Theorem 1 of this paper is an almost verbatim replica of Theorem 1 of the earlier paper [13] where the separable form equations case was considered. In reality, this paper's result is more general since it covers the case of equations in semiseparable form. The most important distinction is the inclusion (27) and, more specifically, the expressions (27b) and (27c) for the "slopes" $a_{ij}'$ and the intervals $b_i^1$, respectively.

Based on Theorem 1, the method of [13] is readily extended to cover the semiseparable case. Technically, the only difference is that now the square real matrix $A$ in (28) and interval vector $b^1$ in (29) are $(n+m)$-dimensional while their counterparts in [13] are $n$-dimensional. For this reason, only several facts, needed in the sequel, will be briefly presented here. Let

$$B = -b^1$$

(34)

The generalized method applicable to system (3) is an iterative method. It is based, essentially, on the following procedure.

Procedure 3. At each iteration $k$, a current $n'$-dimensional box $X^{(k)} \subseteq X^{(0)}$ is generated. Using Procedures 1 and 2, the corresponding $A^{(k)}$ and $B^{(k)}$ are formed (according to Remark 2 $A^{(k)}$ and $B^{(k)}$ stand for $A(X^{(k)})$ and $B(X^{(k)})$, respectively). Using $A^{(k)}$ and $B^{(k)}$, an interval vector $Y^{(k)}$ is then computed (by formulae given in [13]). The iterative procedure is defined as follows

$$X^{(k+1)} = Y^{(k)} \leq X^{(k)}, \quad k \geq 0$$

(35)

The algorithm of the generalized method which is based on Procedure 3 will be referred to as Algorithm 1.

Similarly to other interval methods the present method can be used as a computationally verifiable test for existence of solutions to (3a) in $X$.

**Theorem 2.** Let $f : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$ be a CD function in the domain $D$ and $X^{(0)} \subseteq D$. Introduce the interval operator

$$K(X^{(k)}) = A^{-1}(X^{(k)})B(X^{(k)}), \quad k \geq 0$$

(36)

where $A(X^{(k)})$ and $B(X^{(k)})$ defined as in Procedure 1 and 2, correspond to the current box $X^{(k)}$. Then, if at some $k$th iteration

$$K(X^{(k)}) \subseteq X^{(k)}$$

(37)
the above inclusion implies the existence of a solution to (3a) in $X^{(k)}$.

The proof of the theorem is given in the Appendix.

The numerical performance of Algorithm 1 has been tested on several systems of equations of the form (3a). It will be illustrated by the following two examples.

Example 2. The problem is to find all real solutions to the system (10a) contained in the box $X^{(0)} \in \mathbb{R}^4$ defined by (10b).

According to the approach adopted in Section II of this paper, we have to find all real solutions to the augmented system (12a) which are contained in the enlarged initial box $X^{(0)}$ determined by (10b) and (12b). The latter problem will be solved using Algorithm 1. To do this, we need to generate at each iteration $k$ the corresponding subbox $X^{(k)}$. At this stage, we apply Procedures 1 and 2 for the current box $X^{(k)}$ to obtain the following linear interval system

$$
\begin{align*}
    a_{11}x_1 + a_{13}x_3 + a_{14}x_4 &= B_1 \\
    a_{22}x_2 + a_{23}x_3 - x_4 &= B_2 \\
    a_{33}x_3 + a_{35}x_5 &= B_3 \\
    a_{44}x_4 + a_{46}x_6 &= B_4 \\
    a_{51}x_1 + a_{52}x_2 - x_5 &= B_5 \\
    a_{61}x_1 - x_2 - x_6 &= B_6
\end{align*}
$$

(38)

where for notational simplicity the dependence of the real numbers $a_{ij}$ and the interval vectors $B_i$ on $X^{(k)}$ is not shown explicitly. Now $Y^{(k)}$ is obtained as the interval solution to (38). According to Procedure 3 $Y^{(k)}$ is to be determined at each $k$th iteration. Thus, Algorithm 1 reduces, essentially, to repeatedly setting up and solving the linear interval system (38) (for each subbox $X^{(k)}$) until the desired accuracy of solution $\varepsilon$ [13] is met.

The augmented system (12), (10b) has been solved with $\varepsilon = 10^{-6}$ and the following two solutions were found

$$
\begin{align*}
    x^{(1)} &= (-1.000000, \ 0.000000, \ 0.333333, \ 0.000000, \ 0.000000, \ 0.999999) \\
    x^{(2)} &= (-0.786151, \ 0.618034, \ 0.173857, \ 0.214899, \ 0.000000, \ 0.000000)
\end{align*}
$$

(39)

where each component recorded to six decimal places is the midpoint of a corresponding side of the solution box enclosing a solution. (In actual computation the left end-point of all components $X_i^{(0)}$ of the starting box $X^{(0)}$ was lowered with $\partial = 10^{-7}$ to ensure the location of the first component of $x^{(1)}$.}
Algorithm 1 requires $N_i = 236$ iterations to locate the solutions (39). The solutions of the original nonlinear system (10) are given by the corresponding first 4 components of $x^{(1)}$ and $x^{(2)}$.

In the example considered, the original system (10a) is related to solving a global minimization problem [20] for which

$$x_5 \leq 0, \quad x_6 \leq 0$$  \hspace{1cm} (40)

On account of (40) the intervals (12b) are reduced to

$$X_5^{(0)} = [-1, 0], \quad X_6^{(0)} = [-1, 0]$$  \hspace{1cm} (41)

The augmented system (12a) has also been solved for the smaller box $X^{(0)}$, determined by (10b) and (41). A unique solution was thus found which is given by $x^{(2)}$. Now the number of iterations $N_i$ needed to locate $x^{(2)}$ within an accuracy $\varepsilon = 10^{-5}$ is $N_i = 24$. The latter example has also been solved using Krawczyk's method [6], [7], [11], [20]. Although the more efficient componentwise version of the method was programmed now 993 iterations were needed to locate the solution within the same accuracy. The corresponding execution times for a Pentium 166 MHz computer are 0.018 sec. and 0.684 sec., respectively. The example illustrates two imperically observed facts: (i) the global analysis problem considered can be solved more efficiently by the present method and (ii) both methods require approximately the same amount of calculation per iteration.

The following example is related to electric circuit synthesis ([11], p.276).

Example 3. The problem is to realize the following voltage transfer function

$$V(s) = \frac{0.186 s^2 + 2.474}{0.327 s^3 + 2.770 s^2 + 4.945 + 4.949}$$  \hspace{1cm} (42)

by means of the circuit shown in Fig.1. It is desired to determine the component values $G_1$, $C_1$, $C_2$, $C_3$, $G_L$ and $\Gamma_1=1/L_1$. It can be checked that these values are solutions of the following design equations system

$$
\begin{align*}
x_1 x_6 &= 2.474 \\
x_1 x_4 &= 0.186 \\
(x_1 + x_5) x_6 &= 4.949 \\
(x_2 + x_3) x_6 + x_1 x_5 &= 4.945 \\
x_1 (x_2 + x_3) + x_5 (x_2 + x_4) &= 2.770 \\
x_2 (x_3 + x_4) + x_3 x_4 &= 0.327
\end{align*}
$$

(43)
where $x_1 = G_1$, $x_2 = C_1$, $x_4 = C_2$, $x_3 = C_3$, $x_5 = G_L$, and $x_6 = \Gamma_1$. The initial box $X^{(0)}$ is given by

$$
X_1^{(0)} = [0.01, 2.0], \quad X_2^{(0)} = [0.01, 0.55],
X_3^{(0)} = [0.01, 2.0], \quad X_4^{(0)} = [0.01, 0.50],
X_5^{(0)} = [0.01, 2.5], \quad X_6^{(0)} = [0.01, 2.00],
$$

The desired solution accuracy $\epsilon$ was chosen to be $10^{-3}$. Using Algorithm 1, three solutions $x^{(1)} = (1.515, 0.0837, 1.538, 0.1220, 1.516, 1.632)$, $x^{(2)} = (1.783, 0.1780, 1.092, 0.1040, 1.784, 1.387)$, $x^{(3)} = (1.907, 0.3490, 0.656, 0.0974, 1.908, 1.296)$ of (43), contained in (44), have been found within the desired accuracy. The number of iterations $N_i$ required is 9169. It is worthwhile noting that Krawczyk's method requires 37928 iterations to find all the tree solutions (45) within the same accuracy.

**IV. IMPROVEMENTS**

As shown in Section 3.2 by Example 2, Algorithm 1 of the generalized method reduces, essentially, to setting up and solving the following $n' \times n'$ linear interval system

$$
A^{(k)} y = B^{(k)}
$$

at each iteration. In Example 2, $A^{(k)}$ and $B^{(k)}$ are given in (38).

In this section, two modifications will be introduced into the computational scheme of Algorithm 1. The first is associated with the elimination of all $m$ auxiliary variables from the linear system (46). Thus, (46) is transformed to a system

$$
A^{(k)} y = B^{(k)}
$$

of reduced $n \times n$ size. The second modification consists of applying the constraint propagation approach (e.g. [28]) to the reduced system (47). These modifications result in a considerable improvement of the numerical efficiency of Algorithm 1.

4.1. Eliminating the auxiliary variables

The computational efforts needed to solve (46) can be substantially reduced (for large $n$ and $m$) if the auxiliary variables are eliminated from system (46). This possibility will be shown by way of Example 2. Indeed, from (38)
Substituting (48) into the third and fourth equation of (38), we get

\[ a_{11}x_1 + a_{13}x_3 + a_{14}x_4 = B_1 \]
\[ a_{22}x_2 + a_{23}x_3 - x_4 = B_2 \]
\[ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = B'_3 \]
\[ a_{41}x_1 + a_{42}x_2 + a_{44}x_4 = B'_4 \]

where

\[ a_{31} = a_{35}, \quad a_{32} = a_{5} a_{52}, \quad a_{41} = a_{46} a_{61}, \quad a_{42} = -a_{46}, \]
\[ B'_3 = B_3 + a_{35} B_5, \quad B'_4 = B_4 + a_{46} B_6 \]

It is seen that the new system (49) has only 4 equations of 4 variables while the augmented linear system (38) has 6 equations of 6 variables.

In the general case, by elimination of the auxiliary variables the linear interval system (46) of augmented dimension \( n + m \) is reduced to system (47) of dimension \( n \). Solving (47) is a much easier problem than solving (46), especially for larger \( n \) and \( m \).

It should be emphasised that the elimination of the auxiliary variables is carried out at each iteration, i.e. for each current box \( \tilde{X} \). Therefore, we have to compute, at each iteration, bounds on the auxiliary variables. This can be done using (17), i.e. by evaluating some interval extension \( F_{a_i}(X) \) (or still better, the range \( f_{a_j}(X) \)) of \( f_{a_j}(x) \) when \( x \in X \) where \( f_{a_j}(x) \) is the \( i \)th component of the vector function \( f_{a}(x) \). In general, the interval extension \( X_{a_j} = F_{a_j}(X) \) of \( f_{a_j}(x) \) in \( X \) can be computed using some of the methods available in interval analysis [5]-[8], [18]-[20]. However, Procedure 1 allows an alternative, simpler approach to evaluating \( X_{a_j} \). Indeed, \( f_{a_j}(x) \) are semiseparable functions. Therefore

\[ f_{a_j}(x) = \sum_{j=1}^{n} f_{a_{i_j}}(x_j) + \sum_{k=l=1}^{n} a^{(a_j)}_{i_k l} x_k x_l \]  

But

\[ f_{a_{i_j}}(x) \in \alpha^{a_j}_{j} x_j + B^a_{i_j} \]

Hence
\[ f_{a,j} (x) \in l^a_i (x) + B^a_i \]  

where

\[ l^a_i (x) = \sum_{j=1}^{n} a_{ij} x_j + \sum_{k=1}^{n} \sum_{k \neq l}^{n} \alpha^{(a,j)}_{kl} x_k x_l \]  

\[ B^a_i = \sum_{j} B_{ij} \]  

Thus, the interval extension \( X_{a,j} \) can be computed in the form

\[ X_{a,j} = L^a_i (X) + B^a_i \]  

where \( L^a_i (X) \) is the interval extension (or, if possible, the range) of \( l^a_i (x) \) in \( X \).

The algorithm of the present method which implements the elimination of the auxiliary variables will be referred to as Algorithm 2.

Example 4. Consider the network shown in Fig. 2. We require to realise its voltage transfer function as a third-order maximally flat (Butterworth) function. If we now specify \( R = 0.5 \Omega \) and put \( x_1 = C_1, x_2 = L_2, x_3 = C_3 \), the design equations are

\[ \begin{align*}
    x_1 + 2x_2 + x_3 - 6 &= 0 \\
    2x_1x_2 + x_2x_3 - 6 &= 0 \\
    x_1x_2x_3 &= 3
\end{align*} \]  

(56)

It is known (e.g.,\, [27]) that system (56) has two solutions

\[ x^{(1)} = (1.0, \ 1.5, \ 2.0), \quad x^{(2)} = (3.261, \ 0.779, \ 1.181) \]  

(57)

The last equation of (75) is not in semiseparable form. It can, however, be easily transformed into such form by introducing an auxiliary variable

\[ x_4 = x_2x_3 \]  

(58)

The corresponding linear interval system (46) is

\[ \begin{align*}
    x_1 + 2x_2 + x_3 &= 6 \\
    a_{21}x_1 + a_{22}x_2 + a_{23}x_3 &= B_2 \\
    a_{31}x_1 + a_{34}x_4 &= B_3 \\
    a_{42}x_2 + a_{43}x_3 + a_{44}x_4 &= B_4
\end{align*} \]  

(59)

The initial \( X^{(0)} \) was chosen to have equal components

\[ X_i^{(0)} = [0, 4], \quad i = 1, \ldots, 4 \]  

(60)
The example considered was solved using Algorithm 1 (by solving the augmented system (59)) and Algorithm 2 (by eliminating \(x_4\) in (59) and solving a system of 3 equations). Both algorithms have located infallibly the two solutions (57) with \(\epsilon = 10^{-4}\). Data illustrating the improved efficiency of Algorithm 2 as compared to Algorithm 1 are given in Table 1.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>(N_i)</th>
<th>(t) (sec)</th>
<th>(N_{cl})</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>258</td>
<td>0.514</td>
<td>6</td>
</tr>
<tr>
<td>A2</td>
<td>88</td>
<td>0.141</td>
<td>3</td>
</tr>
</tbody>
</table>

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) [13].

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

4.2. Use of constraint propagation

The second modification is related to the application of the constraint propagation approach [28] to solving the reduced system (47). To simplify notation, it is written in the form

\[
Ay = B
\]

which actually stands for

\[
Ay = b, \quad b \in B
\] (62a)

Taking into account the fact that \(y\) must remain in the current box \(X\), (62a) is to be completed with the condition

\[
y \in X
\] (62b)
The problem is to find an interval solution to (62), that is an interval vector \( Y \) which contains the solution set of (62)

\[
S(A, B, X) = \{ y: Ay = b, \quad b \in B, \quad y \in X \}
\]  

The optimal interval solution \( Y^* \) will be the smallest interval solution still containing \( S(A, B, X) \).

It is readily seen that each component \( Y^*_i = [y^*_i, y^*_i] \), \( i = 1, 2, \ldots, n \), can be determined by solving two linear programming problems:

\[
y_i = \min_C y \quad s.t. \quad Cy - b = 0, \quad y \in X, \quad b \in B
\]  

and

\[
y_i = \max_C y \quad s.t. \quad Cy - b = 0, \quad y \in X, \quad b \in B
\]  

to find the endpoints \( y^*_i \) and \( y^*_i \), respectively. Thus, computing \( Y^* \) would require the solution of \( 2n \) linear programming problems. Such an approach to tackling the GA problem considered seems to be rather costly since \( Y^* \) is to be computed at each iteration \( k \). Therefore, a simpler approach will be adopted here which is based on computing a tight interval solution \( Y \) in a cheaper manner. This is made possible by resorting to the constraint propagation approach as a preliminary stage in solving (62). Several algorithms implementing the latter approach will be presented now.

**Algorithm A3.** This is an algorithm that is based on the following procedure involving two stages.

Stage A. For \( i = 1 \) to \( n \) do

\[
Y_i = \frac{1}{a_{ii}} \left[ B_i - \sum_{j \neq i} a_{ij} X_j \right]
\]  

\[
X_i := Y_i \quad 1 \quad X_i
\]

It is seen that this stage implements the known interval Gauss-Saidel scheme [2] - [4] (in fact, (66) is a simpler version since unlike other interval methods now all the coefficients \( a_{ij} \) are real numbers rather than intervals).

Stage B. Now one iteration of Procedure 3 is applied to the box \( X \) obtained on exit from Stage A.

**Algorithm A4.** 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

\[
Y_j = \frac{1}{a_{ij}} \left[ B_i - \sum_{k \neq j}^{n} a_{ik} X_k \right] \quad (67a)
\]

\[
X_j := Y_j \quad X_j \quad (67b)
\]

(In actual computation, (67a) is implemented in a more efficient manner by first computing

\[
S_1 = B_i - \sum_{k \neq i}^{n} a_{ik} X_k = [\underline{S}_1, \overline{S}_1] \quad (68)
\]

Then

\[
\overline{S}_2 = \underline{S}_1 + a_{i1} \underline{X}_1 - a_{i2} \overline{X}_2 \quad (69a)
\]

and

\[
\underline{S}_2 = \overline{S}_1 + a_{i1} \overline{X}_1 - a_{i2} \underline{X}_2 \quad (69b)
\]

The next sums \( S_j, \ j \geq 2 \) are computed in a similar way.)

Stage B. The same as in algorithm A3.

In the algorithms presented so far, all elements of the real matrix \( A \) 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 \( A \) and \( B \) is implemented in the next algorithm.

Algorithm 5. In this algorithm, stage A is modified in the following manner. Initially for \( i = 1 \), we compute the first row of \( A \) and the first element of \( B \) using the current box \( X \). We then apply (67) to (hopefully) reduce \( X \) to a new box \( X' \). Now \( X' \) is renamed \( X \) and the second row of \( A \) and the second element of \( B \) are determined. Now (67) is applied with \( i = 2 \). This process continues until \( i = n \).

Stage B. The same as in algorithm A4.

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

Example 5. The system to be solved is
The initial box $X^{(0)}$ has the following components

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

(70b)

The GA problem considered has 9 solutions:

$$x^{(1)} = (-1.7475, \ 0.8738) \quad 1, \ 0, \ 0, \ 0)$$
$$x^{(2)} = (-1.075, \ 0.5353) \quad 1, \ 0, \ 0, \ 0)$$
$$x^{(3)} = (-0.2398, \ -0.05648) \quad 0,5716, \ 0.4284, \ 0, \ 0)$$
$$x^{(4)} = (0.06604, \ -0.1929) \quad 0,8341, \ 0.1659, \ 0, \ 0)$$
$$x^{(5)} = (0.2398, \ 0.05648) \quad 0,5716, \ 0.4284, \ 0, \ 0)$$
$$x^{(6)} = (0, \ 0) \quad 1, \ 0, \ 0, \ 0)$$
$$x^{(7)} = (-0.06604, \ 0.1929) \quad 0,8341, \ 0.1659, \ 0, \ 0)$$
$$x^{(8)} = (1.075, \ -0.5353) \quad 1, \ 0, \ 0, \ 0)$$
$$x^{(9)} = (1.7475, \ -0.8738) \quad 1, \ 0, \ 0, \ 0)$$

They have been located by algorithms A2 to A5 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.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
<th>A5</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N_i$</td>
<td>3233</td>
<td>2780</td>
<td>1354</td>
<td>473</td>
</tr>
<tr>
<td>$t$ (sec)</td>
<td>6.86</td>
<td>5.68</td>
<td>2.97</td>
<td>1.18</td>
</tr>
</tbody>
</table>

Table 2

It is seen that algorithm A5 has relatively the best performance. For the same accuracy Krawczyk's method required $N_i = 33089$ iterations to solve the GA problem considered.
4.3. Comparison with other methods

At the present stage of their development, the numerical efficiency of previous interval methods is relatively low for even moderate $n$ and starting regions $X^{(0)}$. There seem to be four major reasons for this.

1. All known interval methods for solving the GA problem are one or another form of the interval Newton method [5]-[8], [16], [18], [20]. Thus, some interval extension $J^I(X)$ of the Jacobian $J(x)$ of $f(x)$ in $X$ (which is relatively expensive to evaluate) is needed at each iteration of the method. At the initial iterations $X$ is large and $J^I(X)$ is generally rather a crude overestimation of the range $J(X)$ of $J(x)$ in $X$.

2. The second reason is that all methods of this group are associated with solving a linear interval system

$$A^I(X)(y - x) = b(x)$$

(71)

with respect to $y$ for each iteration. Here $b(x)$ is a real vector (ignoring for simplicity of presentation the interval arithmetic implementation of the method considered), $x$ is usually the centre of $X$ while $A^I(X)$ is an interval matrix which is either the interval extension $J^I(X)$ itself or is in one way or another related to it. For instance, in methods using preconditioning [25]

$$A^I(X) = CJ^I(X), \quad b(x) = -C f(x)$$

where $C$ is some real matrix. Typically [8], [20]

$$C = [J(x)]^{-1}$$

Since the exact (optimal) interval solution $Y$ of (71) is extremely hard to determine, in practice an approximate interval solution $Y^I \supset Y$ is found which is, once again, rather a crude overestimation of $Y$. Indeed, most often a componentwise Gauss-Seidel procedure is used to compute $Y^I$ and it can be easily seen that the overestimation of $Y$ by $Y^I$ becomes more and more pronounced as $n$ increases. Now let $A(X)$ denote the interval matrix associated with the range $J(X)$, i. e.

$$A(X) = CJ(X)$$
Obviously, this is the narrowest possible interval matrix for the current box $X$ which can replace $A^I(X)$ in (71). Furthermore, let $\tilde{Y}$ denote the optimal interval solution of the "best" linear interval system

$$A(X)(y - x) = b(x)$$

(72)

On account of inclusion monotonicity

$$\tilde{Y} \subseteq Y \subseteq Y^I$$

(73)

and the inclusion is usually proper and rather pronounced (since $Y^I$ is large). Hence, initially, most often

$$X \subseteq Y^I$$

(74)

and therefore the new box

$$X' = Y^I \setminus X$$

(75)

generated by the method for the next iteration is the same as the box $X$ at the current iteration. In this case $X$ must be split into two subboxes $X_L$ and $X_R$. One of them is stacked into a queue $L$ for further processing while the other is renamed $X$ and the iterative process continues on $X$.

3. The above two factors substantiate the third cause for low efficiency: there are too many splittings at the early stages of the computation process. This, in turn, gives rise to a long queue of candidate boxes $X^{(v)}$ stored in $L$ and awaiting processing. In fact, splitting stops and reducing the size of $X$ by (75) starts only at that moment when the width of $X$ becomes so small that the first two factors associated with overestimation become insignificant.

4. Overestimation accounts for the ineffective functioning of the exclusion rule

$$Y^I \setminus X = \emptyset$$

(76)

which deletes the current box $X$ from further processing. Similar to reduction, deletion occurs only at the later iterations when the width of $X$ becomes small enough and the effects of overestimation become negligible.

It is to be noted that less effective reducing and deleting rules lead to a stronger clustering effect [13], [26].
The new interval method suggested in this paper and implemented as Algorithm 5 exploits in an efficient manner the semiseparability property (3a) and reduces, essentially, to setting up and solving the linear interval system (47) at each iteration of the computational process. Now, unlike (71) $A$ is a real matrix while $B$ is an interval vector. In contrast to (71) the optimal interval solution $Y$ to (47) is readily obtained

$$Y = A^{-1}B$$  \hspace{1cm} (77)

This is the main feature of the new interval method which distinguishes it favourably from previously known interval methods. Furthermore, application of the constraint propagation approach as suggested in Algorithm 5 additionally reduces the width of $Y$ at each iteration. This results, as observed empirically, in a faster rate of convergence of the new method.

IV. CONCLUSION

In this paper, the problem of finding (within preset accuracy) the set of all real solutions to a system of non-linear equations (1) contained in a given box $X^{(0)}$ (the GA problem) has been considered. The problem formulation is rather general since the non-linear functions involved can be only continuous or even discontinuous with finite discontinuities.

A new method for solving the general GA problem has been suggested. It comprises the following stages. First, the original system (1) of size $n$ is transformed into a new system (3) of size $n+m$ by introducing $m$ auxiliary variables. The equations of the latter system are in the semiseparable form (3a) which is a sum of the functions $f_j(x_j)$ of a single variable and of products $\alpha_{kl}^{(0)} x_k x_l$ . Each term $f_j(x_j)$ or $\alpha_{kl}^{(0)} x_k x_l$ is then enclosed tightly by the linear interval functions (20) or (25), respectively, at each iteration of the computational process. Using these enclosures, an augmented linear interval system (46) of size $n+m$ is obtained. Next, the auxiliary variables are eliminated from (46) and a reduced linear interval system (47) of size $n$, having a real matrix $A$ and an interval vector right-hand side is set up and solved at each iteration of the method. This advantageously distinguishes the present method from the other known interval methods where a much more complex linear interval system having an $n \times n$ interval matrix and a real right-hand side vector is to be solved at each iteration. Finally, (47) is
solved in a most efficient manner using the constraint propagation approach as implemented in Algorithm 5.

Experimental data show that as regards computer time and memory volume requirements the present method outperforms the other known interval methods for solving the global analysis problem considered.

There seems to exist a possibility for further improvement of the numerical efficiency of the new method by incorporating into its scheme ideas and techniques from affine arithmetic [29] in order to automate the transformation of the original non-linear system (1) into the linear interval system (47).

APPENDIX

Proof of Theorem 2.

Let $y$ be a solution to (32). Then (32) can be transformed into the fixed point format

$$x = C(x)b(x) = P(x)$$

where $C: X^{(k)} \rightarrow \mathbb{R}^n$ and $b: X^{(k)} \rightarrow \mathbb{R}^n$ are to be determined. We choose

$$C(x) = C^{(k)} = A^{-1}(X^{(k)}),$$

$$b(x) = b \in B(X^{(k)})$$

Obviously, for any $x \in X^{(k)}$

$$P(x) \in K(X^{(k)})$$

Thus, if (37) holds, then $P$ maps $X^{(k)}$ into itself. Therefore, by the Shrauder fixed point theorem $P$ has a fixed point in $X^{(k)}$ and hence (32) has a solution in $X^{(k)}$.

REFERENCES


ACKNOWLEDGEMENT

The author is grateful to the reviewers for their valuable recommendations and comments.

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