A Novel Algorithm to Compute all Vertex Matrices of an Interval Matrix: A Computational Approach
Mohammed Tawfik Hussein
Pages 137 - 142

A Novel Algorithm to Compute All Vertex-Matrices of an Interval Matrix: A Computational Approach

Mohammed Tawfik Hussein
Electrical & Computer Engineering Department
Islamic University of Gaza, Gaza City, Palestine, mhussein@iugaza.edu

Abstract: *In this paper an algorithm was developed and implemented to obtain all possible vertex matrices for nxn interval matrix. Given a matrix A ∈ Rnxn with entries aij ∈[aijL , aijH] ⊂ R for i.j=1,…,n. The uncertain parameters generate an infinite family of nxn matrices. From the above matrix let's generate all of the vertex matrices, by allowing each uncertain entry of A to vary over the entire interval. The problem treated in this research is of considerable practical significance. Illustrative examples are given to validate the proposed method*

Keywords: *Interval, Uncertainty, Vertex Matrix, State Space, Bound, Binary.*

Received: *November, 1, 2004* | **Revised:** *December 10, 2004* | **Accepted:** *December 23, 2004*

1. Introduction

Interval analysis is a new and intensively developing branch of computational mathematics. It has been in existence for only three decades: the first monograph in the field by R. Moore was published in 1966 [8]. Originating from the need to control propagation of errors in computations on digital computers, interval analysis presently covers a variety of problems in applied mathematics which are difficult to solve using traditional (noninterval) approaches. The basic concept in interval analysis is that of an interval is a bounded segment of the real line.

1.1 Definitions and Problem Statement

The interval function has envelope bounds, such that Xlower (t) ≤ X (t) ≤ Xupper (t), where Xlower (t) and Xupper (t) are deterministic functions which delimit the range of variation of X (t). A matrix is said to be an interval matrix if at least one of its entries is allowed to vary in some interval. Assume that A is an interval matrix with entries aij ∈[aijL , aijH] ⊂ R., for some i, j =1..n. From the above matrix let's generate all of the vertex matrices, by allowing each uncertain entry of A to vary over the entire interval. From these vertex matrices, the corresponding vertex characteristic polynomials can be calculated as well.

1.2 Review Of Literature

In recent years, along with development of interval mathematics, researches on tolerance analysis have focused on interval mathematics methods, especially the interval functions inclusion methods [1],[7],[9-11], also the concept of uncertainty was discussed by Keel and Bhattacharyya [2-6].

1.3 Objective and Contributions

The objective of this research is to develop and implement an algorithm to obtain all possible vertex matrices for any size of a given interval matrix.

2. Methodology and Algorithm

In this research an algorithm was developed to obtain all possible vertex matrices for nxn interval matrix

2.1 Algorithm for the Computational Approach

The following algorithm is used for the computational approach.

1. Obtain State Space Equations that describe the Linear System as it shown below in Equation 4.1-4.2.

$$\dot{X} = Ax + Bu \qquad (4.1)$$

$$\begin{aligned}\dot{X}_1 &= a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n + b_1u\\ \dot{X}_2 &= a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n + b_2u\\ &\vdots\\ \dot{X}_n &= a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n + b_nu\end{aligned} \qquad (4.2)$$

The a's are interval, x_1 x_2 .... x_n are the state variables, u is the circuit input, and b is a constant.The parameters of the state space matrices are assumed to vary independently. Only bounds to parameters are known. All parameter may vary at the same time over a large range.

2. Obtain all possible matrices from the system's

$$A = \begin{bmatrix} [a_{11\min} \;\; a_{11\max}] & [a_{12\min} \;\; a_{12\max}] & \cdots & [a_{1n\min} \;\; a_{1n\max}] \\ [a_{21\min} \;\; a_{21\max}] & [a_{22\min} \;\; a_{22\max}] & \cdots & [a_{2n\min} \;\; a_{2n\max}] \\ \vdots & \vdots & \vdots & \vdots \\ [a_{n1\min} \;\; a_{n1\max}] & \cdots & \cdots & [a_{nn\min} \;\; a_{nn\max}] \end{bmatrix}$$

interval matrix as follow

3. Given an interval matrix A

a. Generate the set of vertex matrices.

i. Read

$$A_L = [a_l(i,j)]$$

$$A_U = [a_u(i,j)]$$

ii. Find the location of the perturbed entries, and store in matrix L, the lower bounds and in matrix U, the upper bounds.

iii. For each element in L take the one combination in U, except for the same index

iv. For each element in L take the two combinations in U, except for the same index

v. For each element in L take the nth combinations in U, except for the same index

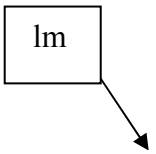
vi. Do steps (iii-v) for each element of U taking the nth combinations L and erase duplications

b. From the set of vertex matrices, obtain the set of all characteristic polynomials.

2.2 Description of Algorithm

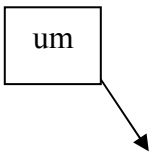
The input to the program consists of a square matrix in which certain elements are uncertain, meaning that they could have on of two potential values. In the implementation stage, this input matrix is actually represented by two separate matrices whose values are identical except in those elements that are to be uncertain. For instance, let assume that the element with row and column indices 2 and 3 respectively is uncertain in the input array, and the two values it can assume are 11 and 7. Then in the dual matrix representation, element (2,3) in one array will have the value 11 and element (2,3) in the second array will be 7. If all elements of the input array are to be uncertain, then the dual representation will consist of two arrays in which each element of the first array differs in value from its corresponding element in the other array. Consequently, there can be a maximum of dxd uncertainties in the input array.

i. The first step of the algorithm is to obtain this input from the user. This may be entered interactively by answering the prompts of the program, or through the use of input redirection and a file with the input data in it. This input consists of the size of the two square input matrices, and then the two matrices themselves. Let refer to the two matrices as "um" (upper matrix) and "lm" (lower matrix). See Tables 1 and 2 for a 3x3 interval matrix.



	0	1	2
0	0	0	1
1	0	2	0
2	3	0	1

Table 1- Lower Bounds of a 3 x 3 Interval Matrix



	0	1	2
0	0	0	4
1	0	5	0
2	6	0	8

Table 2 - Upper Bounds a 3 x 3 Interval Matrix

The uncertainties (their values as well as locations in the matrices) must now be determined. The algorithm examines each pair of corresponding elements in the two matrices and determines if the two values differ.

ii. If the values do differ, the row and column indices for that pair of values are stored in two separate arrays allocated for this purpose. Let refer to these two arrays as "row" and "col". The location of the first uncertainty found in the input arrays, therefore, can be represented as (row [0], col [0]). The arrays row and col are each dxd in length since there could be a maximum of that many uncertainties to track. See Table 3 for a 3x3 interval matrix.

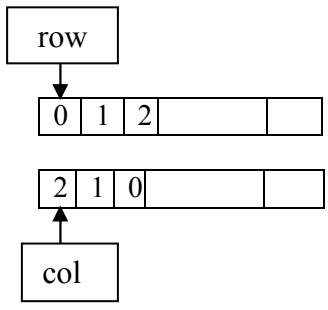
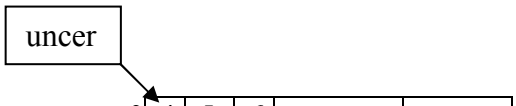


Table 3- Uncertainty Entries Indices of a 3 x 3 Interval Matrix

iii. In addition to recording the row and column indices of each uncertainty, the two possible values are themselves stored in a 2 by dxd matrix. Let refer to this matrix as "uncert". Again, this matrix has dxd columns since there could exist that many uncertainties in the input. Each uncertainty is represented by the two values of the uncertainty found in element (row [0], col [0],) of the input are stored in uncert [0] [0] and uncert [1] [0]; values of the uncertainty in element (row [1], col [1]) would be in uncert [0] [1] and uncert [1] [1], and so on. See Table 4 for a 3x3 interval matrix.



0	4	5	6			
1	1	2	3			

Table 4 - Uncertainty Entries of a 3 x 3 Matrix

iv. By the time each corresponding pair of elements of the two input matrices has been examined in turn, we will have recorded the location and values of all the uncertainties. We will also know the total count (or number) of uncertainties found, which we may call "cnt".

Now that we have explained the rationale of the data structures used thus far, let us examine how the program constructs all the combinations of unique matrices from the initial input matrices.

v. We first need to determine the number of unique matrices that can be generated from the input. Let us assume that the program discovers a total of "cnt" uncertainties by applying the steps discussed above. Since each one of the cnt uncertainties can assume two possible values, there will be 2cnt unique combinations of values, and therefore 2cnt unique matrices. Let refer to this number as "n".

In order to generate each unique set of uncertainties, we have to pick up one of the two possible values for each uncertainty in the set. Recall these uncertainties are stored in the uncertain matrix.

vii. The uncertain matrix contains a column for each uncertainty in the set and each column has two rows, corresponding to the two values the uncertainty can assume. So for each column of the uncert matrix that is examined, a choice has to be made as to which row index to select to pick one of the two values. The row index will be either 0 or 1. For instance, if we have 3 uncertainties in the set, then one combination of values may be represented by sequences of row indices "000"; another one would be "111". In fact, for three uncertainties there would be 8 unique combinations" 000, 001,011,100,101,111. Interestingly, these are the binary representations of the decimal number 0 through 7. See Table 5

0	0	0	→	4	5	6
0	0	1	→	4	5	3
0	1	0	→	4	2	6
0	1	1	→	4	2	3
1	0	0	→	1	5	6
1	0	1	→	1	5	3
1	1	0	→	1	2	6
1	1	1	→	1	2	3

Table 5 - Binary Representations of a 3 x 3 Matrix

viii. Utilizing this relationship, our implementation runs a for loop from index 0 to n, using the binary representation of the loop index to extract a unique set of values from the uncert matrix.

ix. Each iteration of the loop therefore places its own unique set of values into a two dimensional matrix using the row and col arrays to determine the locations at which these values are to be placed. It should be mentioned here that before starting this loop, the program allocates a three dimensional matrix to store the values of all the unique 2D matrices that would be generated by the loop.

Once the unique matrices have been generated, the characteristic polynomial for each of these matrices is also generated and stored in a matrix called "p".

4. Results and Discussions

3.1 Numerical Example

Consider the following 2x2 matrix

$$A=\begin{bmatrix} a_{11} & a_{12} \\ 1 & a_{22} \end{bmatrix}$$

$$A=\left[a_{11,\min},a_{11,\max}\right]\times\left[a_{12,\min},a_{12,\max}\right]\times\left[a_{22,\min},a_{22,\max}\right]\subset R^3.$$

$$V_A=\left\{(a_{11},a_{12},a_{22})\in A: a_{ij}=a_{ij,\min}\ or\ a_{ij}=a_{ij,\max}\ i,j=1,2,3\right\}$$

where

$$a_{11}\in\left[a_{11,\min}\quad a_{11,\max}\right], a_{12}\in\left[a_{12,\min}\quad a_{12,\max}\right], and\ a_{22}\in\left[a_{22,\min}\quad a_{22,\max}\right]$$

Suppose

$$-6\le a_{11}\le -3$$

$$-5\le a_{12}\le -4$$

$$-5\le a_{22}\le -3$$

The family of all matrices can be mapped into the set

$$A=\left[a_{11,\min},a_{11,\max}\right]\times\left[a_{12,\min},a_{12,\max}\right]\times\left[a_{22,\min},a_{22,\max}\right]\subset R^3.$$

This is a box with vertices

$$V_A=\left\{(a_{11},a_{12},a_{22})\in A: a_{ij}=a_{ij,\min}\ or\ a_{ij}=a_{ij,\max}\ i,j=1,2,3\right\}$$

as shown in Figure 1.

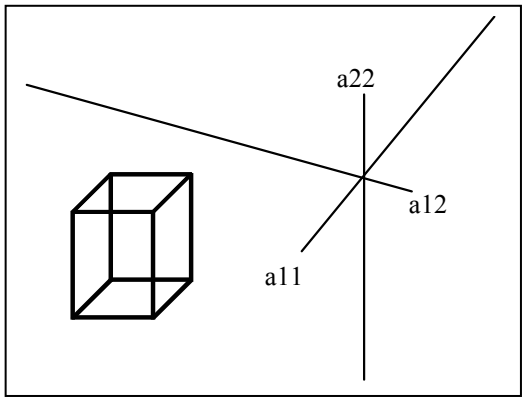


Figure 1. Parameter Space of Family of Matrices

The vertices of this box correspond to the matrices

$$A_1 = \begin{bmatrix} -6 & -5 \\ 1 & -5 \end{bmatrix}, A_2 = \begin{bmatrix} -6 & -5 \\ 1 & -3 \end{bmatrix}, A_3 = \begin{bmatrix} -6 & -4 \\ 1 & -5 \end{bmatrix},$$
$$A_4 = \begin{bmatrix} -6 & -4 \\ 1 & -3 \end{bmatrix} A_5 = \begin{bmatrix} -3 & -5 \\ 1 & -5 \end{bmatrix}, A_6 = \begin{bmatrix} -3 & -5 \\ 1 & -3 \end{bmatrix},$$
$$A_7 = \begin{bmatrix} -3 & -4 \\ 1 & -5 \end{bmatrix}, A_8 = \begin{bmatrix} -3 & -4 \\ 1 & -3 \end{bmatrix}$$

The characteristic polynomial for A is

$$P(s, a_{11}, a_{12}, a_{22}) = a_{11}a_{22} - a_{12} + (-a_{11} - a_{22})s + s^2 = p_0 + p_1 s + s^2$$

Thus, for the each of the vertices the characteristic polynomials are

$$P_1(s) = 35 + 11s + s^2$$
$$P_2(s) = 23 + 9s + s^2$$
$$P_3(s) = 34 + 11s + s^2$$
$$P_4(s) = 22 + 9s + s^2$$
$$P_4(s) = 22 + 9s + s^2$$
$$P_5(s) = 20 + 8s + s^2$$
$$P_6(s) = 14 + 6s + s^2$$
$$P_7(s) = 19 + 8s + s^2$$
$$P_8(s) = 13 + 6s + s^2$$

3.2 Computational Example 2: a 4x4 Interval Matrix

The following example illustrates the determination of all possible vertex matrices of a 4x4 interval matrix. The A matrix with four uncertainty entries is shown below, where Al matrix is the lower entries of A matrix, while the Au is the upper entries of A matrix. The results of the simulation are shown as follow:

```
A= [2       (-2,1)      0          4
   (3,5)    5           2          1
    7       4          (-3,3)      -2
   -1       0           5         (1,3)]
```

where

```
Al= [ 2  -2  0  4
      3   5  2  1
      7   4 -3 -2
     -1   0  5  1 ]

Au= [ 2   1  0  4
      5   5  2  1
      7   4  3 -2
     -1   0  5  3 ]
```

3.2.1 Computational Results

```
Avertex =

    2   -2    0    4
    3    5    2    1
    7    4   -3   -2
   -1    0    5    1

CHPOL =

   1.0000   -5.0000    5.0000 -135.0000  548.0000

Avertex =

    2   -2    0    4
    3    5    2    1
    7    4   -3   -2
   -1    0    5    3

CHPOL =

   1.0000   -7.0000   13.0000 -109.0000  364.0000

Avertex =

    2   -2    0    4
    3    5    2    1
    7    4    3   -2
   -1    0    5    1

CHPOL =
   1.0000  -11.0000   53.0000 -297.0000  776.0000

Avertex =

    2   -2    0    4
    3    5    2    1
    7    4    3   -2
   -1    0    5    3

CHPOL =

   1.0000  -13.0000   73.0000 -355.0000  784.0000
```

```
Avertex =
    2    1    0    4
    3    5    2    1
    7    4   -3   -2
   -1    0    5    1

CHPOL =

    1.0000   -5.0000   -4.0000 -192.0000  419.0000

Avertex =

    2    1    0    4
    3    5    2    1
    7    4   -3   -2
   -1    0    5    3

CHPOL =

    1.0000   -7.0000    4.0000 -148.0000  373.0000
Avertex =

    2    1    0    4
    3    5    2    1
    7    4    3   -2
   -1    0    5    1
CHPOL =

    1.0000  -11.0000   44.0000 -300.0000  575.0000

Avertex =

    2    1    0    4
    3    5    2    1
    7    4    3   -2
   -1    0    5    3
CHPOL =

    1.0000  -13.0000   64.0000 -340.0000  613.0000

Avertex =

    2   -2    0    4
    5    5    2    1
    7    4   -3   -2
   -1    0    5    1

CHPOL =

    1.0000   -5.0000    9.0000 -127.0000  416.0000

Avertex =

    2   -2    0    4
    5    5    2    1
    7    4   -3   -2
   -1    0    5    3

CHPOL =

    1.0000   -7.0000   17.0000 -109.0000  208.0000

Avertex =

    2   -2    0    4
    5    5    2    1
    7    4    3   -2
   -1    0    5    1

CHPOL =

    1.0000  -11.0000   57.0000 -313.0000  668.0000

Avertex =

    2   -2    0    4
    5    5    2    1
    7    4    3   -2
   -1    0    5    3

CHPOL =

    1.0000  -13.0000   77.0000 -379.0000  700.0000

Avertex =

    2    1    0    4
    5    5    2    1
    7    4   -3   -2
   -1    0    5    1

CHPOL =

    1.0000   -5.0000   -6.0000 -196.0000  245.0000

Avertex =

    2    1    0    4
    5    5    2    1
    7    4   -3   -2
   -1    0    5    3

CHPOL =

    1.0000   -7.0000    2.0000 -148.0000  211.0000

Avertex =

    2    1    0    4
    5    5    2    1
    7    4    3   -2
   -1    0    5    1
```

```
CHPOL =

   1.0000  -11.0000   42.0000 -292.0000  389.0000

Avertex =

    2    1    0    4
    5    5    2    1
    7    4    3   -2
   -1    0    5    3

CHPOL =

   1.0000  -13.0000   62.0000 -328.0000  415.0000
```

5. Concluding Remarks

Indeed, an A matrix with uncertainty (interval) entries is of paramount importance to control system engineers, especially on finding eigenvalues to determine system stability.. In summary, the problem treated in this paper is of considerable practical significance.

This research was conducted on different type of example in order to validate the proposed method. The most pertinent conclusions of this research are as follows:

1. This research makes it possible, by simple algorithms, to all vertex matrices and corresponding characteristics polynomials.
2. The program and the computational effort associated with this problem can accommodate any size of a matrix.
3. All parameters may vary at the same time over a large range.

References

[1] G. Alefeld, *Introduction to Interval Computations*. New York: Academic, 1993.

[2] S.P. Bhattacharyya, H. Chapellat, and L.H. Keel, *Robust Control The Parametric Approach*, Prentice Hall, 1995.

[3] L.H. Keel, and S. P. Bhattacharyya, "Parametric stability margin for multilinear interval control systems," *in Proc. of the American Control Conferences*. San Francisco, CA, pp. 262 –266, 1993.

[4] L. H. Keel and S. P. Bhattacharyya, "Robust Stability of Interval Matrices: a computational approach," *Int. J. Control*, 1995, Vol.62, No. 6, pp. 1491-1506.

[5] L. H. Keel and S. P. Bhattacharyya, "Robust Stability of Interval Matrices: a computational approach," *Int. J. Control*, 1995, Vol.62, No. 6, pp. 1491-1506.

[6] L.H. Keel, and S. P. Bhattacharyya,"Parametric Stability Margin for Multilinear Interval Control Systems," *Proc. Of the American Control Conferences*. San Franssico, California pp 262 –266,June 1993.

[7] L.V. Kolev, "Interval mathematics algorithms for tolerance analysis," *IEEE Transactions on Circuits and Systems*, vol. 35, pp. 967-975, 1988.

[8] R. E. Moore, *Interval Analysis*, Englewood Cliffs, NJ: Prentice Hall, Inc., 1966.

[9] A. Neumaier, *Interval Methods for Systems Equations*. Cambridge, U.K.: Cambridge University Press, 1990.

[10] E. P. Oppenhermer and A. N. Mickel, "Application of interval analysis techniques to linear systems," *IEEE Transactions on Circuits and Systems,* vol. 35, pp. 1129 – 1138, 1230-1242, 1243-1256, 1988.

[11] S. Skelobe, "True worst-case analysis of linear electrical circuits by interval arithmetic," *IEEE Transactions on Circuits and Systems*, vol. 26, pp. 874-879, 1979.

Dr. Mohammed T. Hussein Joined the department of Electrical and Computer engineering at Islamic university in August 2003. Dr. Hussein was named Director of e-Learning Center in November 1, 2003. Prior to this appointment he served as a department Head of Engineering Technology in the College of Engineering at Prairie View A&M University, Prairie View, Texas. Dr. Hussein earned a Ph.D. degree in electrical engineering from Texas A&M University, College Station, Texas, USA. Dr. Hussein is a Registered Professional Engineer in the State of Texas. Dr. Hussein worked for Motorola Inc., in Tempe, Az., and Oak Ridge National Laboratory in state of Tennessee. His research interests include Robust Control Systems, Computer algorithms and applications, and e-Learning. Dr. Hussein holds scientific and professional memberships in IEEE(SM), Eta Kappa Nu, Tau Beta Pi. He is the recipient of numerous national, state, university, college, and departmental awards including Who's Who among America best teachers and Teaching Award in the College of Engineering. Dr. Hussein was nominated and selected on 2003 as an evaluator for Accreditation board for Engineering and Technology (ABET), USA.