C S 9 6

INTERVAL ARITHMETIC DETERMINANT EVALUATION AND ITS USE IN TESTING FOR A CHEBYSHEV SYSTEM

BY

LYLE B. SMITH

TECHNICAL REPORT NO. CS 96
APRIL 26, 1968

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY

INTERVAL ARITHMETIC DETERMINANT EVALUATION AND ITS USE IN TESTING FOR A CHEBYSHEV SYSTEM*

BY

Lyle B. Smith

*This research was supported in part by ONR.

ABSTRACT

Two recent papers by Hansen and by Hansen and R. R. Smith have shown how interval arithmetic (I.A.) can be used effectively to bound errors in matrix computations. This paper compares a method proposed by Hansen and R. R. Smith to straightforward use of I.A. in determinant evaluation. Computational results show what accuracy and running times can be expected when using I.A. for determinant evaluation. An application using I.A. determinants in a program to test a set of functions to see if they form a Chebyshev system is then presented.

Interval Arithmetic Determinant Evaluation And Its Use In Testing For A Chebyshev Svstem

1. Introduction

Recently Hansen [1] and Hansen and R. R. Smith [2] have shown how interval arithmetic (I.A.) can be used effectively to bound errors in matrix computations. In [2] a method for evaluating the determinant of a real square interval matrix, AI, was proposed. An interval matrix, A^I, has elements which are closed intervals

$$A^I_{ij} = [u_{ij}, v_{ij}] .$$

Here we present the results of an implementation of that method and compare it to a straightforward use of interval arithmetic in determinant evaluation. First we give an algorithm for determinant evaluation which simply uses I.A. for all computations. Then we detail the method given in [2] and give results which compare the effectiveness of the two methods. Finally, interval determinant calculations are used to mathematically test a set of functions for the property of being a Chebyshev system of functions.

2. Straightforward Use Of Interval Arithmetic

For the purpose of comparision with Hansen's method for determinant evaluation we chose a previously published algorithm [3] and inserted calls on I.A. routines where appropriate. This algorithm evaluates a

determinant by triangularization with searching for pivot in row and with row equilibration. The I.A. routines are similar to those given in [4], and are described in detail in [6]. The calls are implemented as shown in Table 1, where in each-case the resultant interval is [CL,CR] and the left and right operand intervals are [AL,AR] and

I.A. Routines

operation	name of routine	Parameters
addition	IADD	(CL,CR,AL,AR,BL,BR,L)
subtraction	ISUB	(CL,CR,AL,AR,BL,BR,L)
multiplication	IMPY	(CL,CR,AL,AR,BL,BR,L)
division	IDIV	(CL,CR,AL,AR,BL,BR,L)

Table 1

[BL,BR] respectively. The label L is for an error return when overflow is likely or if the denominator interval contains zero when IDIV is called. Thus a call of ISUB(CL,CR,AL,AR,BL,BR,L) will give [CL,CR] ← [AL,AR] - [BL,BR] with the subtraction done in I.A. and a transfer of control to the label L if overflow is likely.

The algorithm given in [3] as modified to use I.A. is now given:

Algorithm 269 with Interval Arithmetic

```
PROCEDURE IDETERMINANT(ALEFT,ARIGHT,N,DLEFT,DRIGHT,LBL);
    VALUE N;
    ARRAY ALEFT[0,0],ARIGHT[0,0];
    INTEGER N;
    REAL DLEFT,DRIGHT; LABEL LBL;
    BEGIN COMMENT COMPUTES THE INTERVAL VALUE OF AN
        INTERVAL DETERMINANT. SEE ALGORITHM 269 IN
        COMMUNICATIONS OF THE A.C.M. NOV.1965 ;
        LABEL RETURN;
        REAL PRODUCTL,PRODUCTR,TEMPL,TEMPR,TEMLL,TEMRR;
        INTEGER I,J,R,S;
        ARRAY MULTL[0:N],MULTR[0:N];
        REAL PROCEDURE MAX(X,Y);
            VALUE X,Y;
            REAL X,Y;
            MAX := IF X ≤ Y THEN Y ELSE X;
        PROCEDURE EQUILIBRATE(AL,AR,N,MULTL,MULTR,LBL,
            DLEFT,DRIGHT,RETURN);
            VALUE N; INTEGER N; LABEL LBL;
            AKRAY AL[0,0],AR[0,0],MULTL[0],MULTR[0];
            REAL DLEFT,DRIGHT; LABEL RETURN;
            BEGIN
                INTEGER I,J; REAL MXL,MXR;
                FOR I := 1 STEP 1 UNTIL N DO
                BEGIN
                    MXL := 0.0; MXR := 0.0 ;
                    FOR J := 1 STEP 1 UNTIL N DO
                    BEGIN
                        IFABS(AL[I,J])> MXL THEN MXL :=ABS(AL[I,J]);
                        IFABS(AR[I,J])> MXL THEN MXL :=ABS(AR[I,J]);
                    END; MXR := MXL;
                    IF MXL = 0.0 THEN
                    BEGIN
                        DLEFT := 0.0; DRIGHT :=0.0;
                        GO TO RETURN;
                    END;
                    MULTL[I]:= MXL ;
                    MULTR[I]:= MXR ;
                    IF MXL ≠ 1.0 THEN
                    FUR J := 1 STEP 1 UNTIL N DO
                    RDIV(AL[I,J],AR[I,J],AL[I,J],AR[I,J],MXL,MXR,LBL);
                END;
            END EQUILIBRATE ;
        EQUILIBRATE(ALEFT,ARIGHT,N,MULTL,MULTR,LBL,DLEFT,DRIGHT,RETURN);
        PRODUCTL := 1.0;
        PRODUCTR := 1.0;
        FOR K := 1 STEP 1 UNTIL N-1 DO
        BEGIN
            S := R ;
            TEMPL := MAX(ABS(ALEFT[R,R]),ABS(ARIGHT[R,R])) ;
            FUK J := R+1 STEP 1 UNTIL N DO
            IF TEMPL < MAX(ABS(ALEFT[R,J]),ABS(ARIGHT[R,J])) THEN
            BEGIN
```

```
            TEMPL :=MAX(ABS(ALEFT[R,J]),ABS(ARIGHT[R,J])) ;
            S := J ;
         END ;
         IF TEMPL = 0.0 THEN
         BEGIN
            DLEFT := 0.0;
            DRIGHT := 0.0;
            GO TO RETURN ;
         END;
         It S ≠ R THEN
         BEGIN
            RMPY(PRODUCTL,PRODUCTR,-1.0,-1.0,PRODUCTL,PRODUCTR,LBL);
            FOR I := R STEP 1 UNTIL N DO
            BEGIN
               TEMPL := ALEFT[I,R];
               TEMPR := ARIGHT[I,R];
               ALEFT[I,R] := ALEFT[I,S];
               ARIGHT[I,R] := ARIGHT[I,S];
               ALEFT[I,S]:= TEMPL;
               ARIGHT[I,S] := TEMPR ;
            END;
         END;
         RMPY(PRODUCTL,PRODUCTR,PRODUCTL,PRODUCTR,ALEFT[R,R],
              ARIGHT[R,R],LBL);
         FUR I := R+1 STEM 1 UNTIL N DO
         BEGIN
            RDIV(TEMPL,TEMPR,ALEFT[I,R],ARIGHT[I,R],
                             ALEFT[R,R],ARIGHT[R,R],LBL);
            FUR J:= R+1 STEP 1 UNTIL N DO
            BEGIN
               RMPY(TEMLL,TEMRR,ALEFT[R,J],ARIGHT[R,J],TEMPL,TEMPR,LBL);
               RSUB(ALEFT[I,J],ARIGHT[I,J],ALEFT[I,J],ARIGHT[I,J],
                    TEMLL,TEMRR,LBL);
            END;
         END;
      END ;
      RMPY(TEMPL,TEMPR,PRODUCTL,PRODUCTR,ALEFT[N,N],ARIGHT[N,N],LBL);
      FOR R:= 1 STEP 1 UNTIL N Do
      RMPY(TEMPL,TEMPR,TEMPL,TEMPR,MULTL[R],MULTR[R],LBL);
      DLEFT := TEMPL;
      DRIGHT := TEMPR ;
RETURN:
   END IDETERMINANT ;
```

3. Hansen's Method

In [2] a method for determinant evaluation of an interval matrix, A^I, is proposed. It can be presented as 4 steps.

(i) Determine a lower triangular matrix, L, with unit diagonal elements such that $LA_c = U$, where A_c is the center of A^I and U is upper triangular. L will contain roundoff errors but the determinant of L is exactly one.

(ii) Using I.A., multiply L times A^I obtaining B^I which will in general have very small intervals below the main diagonal.

(iii) Perform Gaussian elimination on B^I using I.A. This will result in an upper triangular matrix T^I. It is noted in [2] that an exact zero for a zeroed element can not be computed during the elimination but it is correct to insert such zeros

(iv) Compute the determinant as the I.A. product of the diagonal elements of T^I. That is,

$$d^I = \prod_{i=1}^{N} t_{ii}^I .$$

The implementation of this technique was accomplished by using an algorithm given by Ralston [5, p. 411] for determination of the appropriate lower triangular matrix L with unit diagonal elements. The algorithm in [5] takes a matrix A and reduces it to two matrices L and U such that A = LU where L is lower triangular with unit diagonal elements and U is upper triangular. It is then simple to determine L^{-1} which is the lower matrix required by Hansen's method.

The decomposition was implemented both with column pivot selection and without any pivot selection. It is interesting to compare the results of Hansen's method with and without pivot selection* This comparison is presented in the next section. -

A Burroughs Extended Algol implementation of Hansen's method is now given:

```
PROCEDURE HANSENSMETHOD(AL,AR,DL,DR,N,IAERR) ;
     VALUE N; INTEGER N;
     REAL DL,DR;
     LABEL IAERR;
     REAL ARRAYAL,AR[0,0];
     BEGIN COMMENT
        THIS PROCEDURE COMPUTES THE-INTERVAL DETERMINANT
        OF AN INTERVAL MATRIX USING THE METHOD DESCRIBED
        IN SIAM JOURNAL UN NUMERICAL ANALYSIS,VOL 4(1967),NO.1,
        By ELDON HANSEN AND R.R.SMITH.

        THE INPUT IS AL,AR, AND N, WHERE
           AL  IS THE LEFT END POINTS OF AN INTERVAL MATRIX,
           AH  IS THE RIGHT END POINTS, AND
           N   IS THE ORDER OF THE MATRIX,

        THE OUTPUT IS DL AND DR ,THE LEFT AND RIGHT END POINTS
           OF THE DETERMINANT,

        IF A DIVISION BY ZERO,FOR EXAMPLE, OCCURS DURING THE
           INTERVAL ARITHMET CALCULATIONS, CONTROL IS TRANSFERRED
           TO THE LAHEL IAERR ;

       COMMENT PUT DECOMP,INVLOWER, AND INTDET HERE;
PROCEDURE DECOMP(A,N,P);
   VALUE N ;
   INTEGER N;
   ARRAY A[0,0];
   INTEGER ARRAY P[0];
   BEGIN COMMENT SEE RALSTON (1ST COURSE IN N.A.)P.414;
      INTEGER R,K,I,J,DMAX;
      ARRAY  D[0:N] ;
      LABEL ZRO,DONE;
REAL TMP;
      FOR R := 1 STEP 1 UNTIL N DO
      BEGIN
         FOR K := 1 STEP 1 UNTIL N DO
         D[K] := A[K,R] ;
         FOR J := 1 STEP 1 UNTIL R-1 DO
         BEGIN
            A[J,R] := D[P[J]] ;
            D[P[J]] := D[J] ;.
            FOR I := J+1 STEP 1 UNTIL N DO
            D[I] := D[I] -A[I,J]x A[J,R] ;
         END;                                    DMAX := R;
         FOR I := R STEP 1 UNTIL N DO
         IF ABS(D[I]) > ABS(D[DMAX]) THEN DMAX := I ;
         A[R,R] := D[DMAX] ;
         P[R] := DMAX;
         D[DMAX] := D[R] ;
         FOR I := R+1 STEP 1 UNTIL N DO
         BEGIN
            IF A[R,R] = 0-THEN GO TO ZRO;
```

```
            A[I,R] := D[I] / A[R,R] ;
          END;
        END;
        GO TO DONE;
ZRO:    WRITE(<"A[R,R]=0.0",I10>,R);
DONE:

FOR I := 1 STEP 1 UNTIL N-1 DO
FOR J := I+1 STEP 1 UNTIL N DO
IF   P[J] ≠ J THEN
BEGIN
   TMP := A[J,I];
   A[J,I] := A[P[J],I] ;
   A[P[J],I] := TMP ;
END;

   END DECUMP;

PROCEDURE INVLOWER(LIN,LOUT,N);
   VALUE N; INTEGER N;
   ARRAY LIN,LOUT[0,0];
   BEGIN
      REAL SUM; INTEGER I,J,P;
      FOR I:=1 STEP 1 UNTIL N DO
      BEGIN
         LOUT[I,I] := 1.0;
         FOR J := 1 STEP 1 UNTIL I-1 DO
         BEGIN SUM := 0.0;
            FOR P:=J STEP 1 UNTIL I-1 DO
            SUM := SUM + LIN[I,P]×LOUT[P,J];
            LOUT[I,J] := -SUM;
         END;
 END;
      END INVLOWER;

PROCEDURE INTDET(L,AL,AR,DL,DR,N);
   VALUE N; INTEGER N;
   ARRAY L,AL,AR[0,0];
   REAL DL,DR;
   BEGIN COMMENT THIS PROCEDURE TAKES A LOWER TRIANGLE
      SUCH THAT L×A= U(UPPER),WHERE A IS INTERVAL SO
      USING I.A.,U IS NOT REALLY UPPER AND I.A. GAUSSIAN
      ELIMINATION IS THEN DONE TO GET THE INTERVAL DETERMINANT;
      INTEGER I,J,K;
      REAL SUML,SUMR,MULTL,MULTR,TL,TR;
      ARRAY UL[0:N,0:N],UR[0:N,0:N];     LABEL LBL,DONE;
      FOR I := 1 STEP 1 UNTIL N DO
      FOR J := 1 STEP 1 UNTIL N DO
      BEGIN
        SUML := AL[I,J] ; SUMR := AR[I,J] ;
        FOR K := 1STEP 1 UNTIL I-1 DO        BEGIN
         RMPY(TL,TR,L[I,K],L[I,K],AL[K,J],AR[K,J],LBL);
         RADD(SUML,SUMR,SUML,SUMR,TL,TR,LBL);END;
        UL[I,J] := SUML;
        UR[I,J] := SUMR;
```

```
        END;
        COMMENT NOW DO GAUSSIAN ELIMINATION ONUL,UR
                OR USE IDET ON IT;
        FUR I := 1 STEP 1 UNTIL h DO
        FOR J:= I+1 STEP 1 UNTIL N DO
        BEGIN
           RDIV(MULTL,MULTR,UL[J,I],UR[J,I],UL[I,I],UR[I,I],LBL);
           FUR K := I+1 STEP 1 UNTIL N DO
           BEGIN
              RMPY(TL,TR,MULTL,MULTR,UL[I,K],UR[I,K],LBL);
              RSUB(UL[J,K],UR[J,K],UL[J,K],UR[J,K],TL,TR,LBL);
           END;
        END;
        COMMLNT NOW COMPUTE THE DETERMINANT;
        DL := UL[1,1];
        DR := UR[1,1];
        FOR I:= 2 STEP 1 UNTIL N DO
        RMPY(DL,DR,DL,DR,UL[I,I],UR[I,I],LBL);
        GO TO DONE ;
LBL:    WRITE(<"I.A. TROUBLE IN INTDET">);
DONE:
    END INTDET;

        REAL ARRAYA,LI[0:N,0:N];
        INTEGER ARRAY P[0:N];
        INTEGER I,J ;
        REAL TEMPL,TEMPR,SGN ;

        FOR I := 1 STEP 1 UNTIL N DO
        FOR J := 1 STEP 1 UNTIL N DO
           A[I,J]:=(AL[I,J] + AR[I,J]) /2.0 ;
        COMMENT  A HAS THE MIDPOINTS. NOW DECOMP WILL FIND L AND U
           SUCH  THAT  A=L×U AND PUT L IN A ;

        DECOMP(A,N,P);
        COMMENT  NOW INVLOWER WILL INVERT THE LOWER TRIANGLE IN A
           AND PUT THE INVERSE IN LI;

        INVLOWER(A,LI,N);

        COMMENT  IF INTERCHANGES OCCURRED DURING DECOMP WE NOW
           INTERCHANGE  AL AND AR ACCORDINGLY;
        FOR I := 1 STEP 1 UNTIL N DO
        BEGIN
           IF P[I]≠ I THEN
           FOR J := 1 STEP 1 UNTIL N DO
           BEGIN
              TEMPL := AL[I,J] ;
              TEMPR := AR[I,J] ;
              AL[I,J] := AL[P[I],J];
              AL[P[I],J]:= TEMPL ;
              AR[I,J] := AR[P[I],J];
              AR[P[I],J]:= TEMPR ;
           END
        END ;
```

```
      COMMENT NOW INTDET COMPUTES BI = LI×A(INTERVAL)AND
         PERFORMS GAUSSIAN ELIMINATION ON BITOFIND THE
         DETERMINANT AS THE PRODUCT OF THE DIAGONAL ELEMENTS)

      INTDET(LI,AL,AR,DL,DR,N);
      COMMENT NOW ADJUST THE SIGN ACCORDING TO THE INTERCHANGES;
      SGN := 1.0 ;
      FOR I := 1 STEP  1 UNTIL NDO
         IF P[I] ≠ I  THEN SGN := -SGN;
      IF SGN < 0    THEN
         RMPY(DL,DR,DL,DR,-1.0,-1.0,IAERR);
   END HANSENSMETHOD ;
```

4. Comparison of results

The test matrices were generated using a mixed congruential method of generating uniform pseudo-random numbers in the interval (0,1) as implemented on a Burroughs B5500 computer. A matrix A was filled with random numbers, then a small positive number, ζ, was added to and subtracted from each element of A to obtain the right and left end points respectively of the interval elements of A^I.

Various values of ζ were tried for a range of values of N, the order of the test matrices. Selected results are shown in Table 2. Note that the Burroughs B5500 can hold approximately 11 decimal digits of accuracy so the input interval widths are quite significant compared to machine accuracy.

The value of pivot selection is strikingly illustrated by columns four and five of Table 2. For the case $N=5, \zeta=10^{-6}$ no pivoting was necessary and the results are identical as expected. However, at the other extreme, the case $N=9, \zeta=10^{-6}$ without pivoting shows an interval width of over 600 times the interval computed with piuoting.

Table 2 also shows that Hansen's method can retain correct significant digits for matrices of order at least 20 whereas the straightforward use of I.A. begins to lose all correct significant digits for matrices whose order approaches 20. Consider the case $N=17, \zeta=10^{-8}$. Hansen's method gives a result which can be expressed as $d^I = -.00334 \pm .0000075$, leaving two correct significant digits. However for the same case the straightforward approach gives $d^I = -.00334 + .0004$ which has no correct significant digits. The case $N=20, \zeta=10^{-8}$

provides a similar comparison. Hansen's method yields d^I = -0.00776 + .0000065 (2 correct significant digits) and the alternate method gives d^I = -0.00776 $\pm$ .00023 (barely 1 correct significant digit).

Comparison of Interval Determinant Widths

(2ζ is the width of the elements in the original matrix)

N	ζ	Value of Determinant	Width Hansen's w/pivot	Width Hansen's w/o pivot	Width Algorithm 269 w/I.A.	Multiplicative factor for all 3 widths
3	10^{-8}	0.01311..	2.18	2.24	3.28	$x10^{-8}$
3	10^{-6}	-0.25862..	5.0	13.1	5.66	$x10^{-6}$
4	10^{-8}	0.12517..	1.4	12.0	2.1	$x10^{-7}$
4	10^{-6}	-0.18143..	0.95	9.1	2.0	$x10^{-5}$
5	10^{-8}	-0.01023..	0.7	3.4	0.9	$x10^{-7}$
5	10^{-6}	0.02672..	0.69	0.69	0.78	$x10^{-5}$
6	10^{-8}	0.02365..	1.44	2.51	3.6	$x10^{-7}$
6	10^{-6}	-0.09218..	0.26	1.03	0.52	$x10^{-4}$
7	10^{-8}	0.02758..	1.80	8.38	6.8	$x10^{-7}$
7	10^{-6}	-0.00818..	0.81	1.5	108	$x10^{-5}$
8	10^{-8}	0.03106..	5.2	83.2	13.7	$x10^{-7}$
8	10^{-6}	0.00680..	0.13	2.3	0.28	$x10^{-4}$
9	10^{-8}	0.07876..	1.09	282.	1.79	$x10^{-6}$
9	10^{-6}	0.02018..	0.29	178.6	1.14	$x10^{-4}$
10	10^{-8}	0.01337..	1.05	13.1	0.66	$x10^{-6}$
10	10^{-6}	0.00023..	1.75	23.25	2.19	$x10^{-5}$
15	10^{-8}	0.00087..	0.47	---	4.43	$x10^{-6}$
17	10^{-8}	-0.00334..	0.15	---	8.15	$x10^{-4}$
20	10^{-8}	-0.00776..	0.13	---	4.59	$x10^{-4}$

Table 2

5. Average Relative Widths using Hansen's Method

In an attempt to generate some useful guidelines as to what accuaracy to expect when using Hansen's method for determinant evaluation we have done tests using random matrices with elements in the interval $(-1,1)$. Thus, if a matrix is scaled so that all elements are less than one in modulus the results given in Table 3 will provide an estimate of the size of interval determinant value that can be expected when using Hansen's method. A relative width of 1.0×10^{-m} means that at least m-l significant digits are correct in the interval determinant value.

Consider, for example, a problem involving the determinant of a matrix of order 8 with interval elements of maximum width 10^{-6}. If the matrix elements are scaled to lie in the interval $[-1,1]$ and Hansen's method is used to evaluate the determinant, a crude interpolation in column 4 of Table 3 will provide an estimate of the accuracy that can be achieved. For this example a relative width of about 10×10^{-4} can be expected which means at least three correct significant digits in the interval determinant value.

Average Relative Widths of I.A. Determinants

using Hansen's Method (averages calculated for 3 matrices)

{ w = width of interval determinant
d = true value of determinant
ζ = half the width of origjnal elements of matrices }

N	$\frac{w}{d}$, $\zeta=10^{-10}$	$\frac{w}{d}$, $\zeta=10^{-8}$	$\frac{w}{d}$, $\zeta=10^{-6}$	$\frac{w}{d}$, $\zeta=10^{-4}$
3	0.65×10^{-8}	0.21×10^{-6}	1.58×10^{-4}	0.49×10^{-2}
5	2.02×10^{-8}	1.17×10^{-6}	1.74×10^{-4}	1.64×10^{-2}
7	4.84×10^{-8}	4.84×10^{-6}	3.98×10^{-4}	13.19×10^{-2}
9	7.34×10^{-8}	9.51×10^{-6}	24.57×10^{-4}	19.40×10^{-2}
11	35.46×10^{-8}	41.97×10^{-6}	52.69×10^{-4}	38.23×10^{-2}

Table 3

6. Comparison of Running Times

Table 4 gives average times for determinant evaluation on the Burroughs B5500. All I.A. calculations are performed with the I.A. subroutines coded in Burroughs Extended Algol and are obviously quite slow compared to noninterval arithmetic.

Hansen's method is significantly slower than the I.A. version of algorithm 269. Thus in any application a study of Tables 2, 3, and 4, together with a knowledge of the input element widths and the accuracy desired, should indicate which routine to use from the standpoint of efficient machine time utilization.

Average Times to Evaluate Determinants
using various Methods (Times in seconds)

N	85C	269(I.A.)	Hansen's	Number of matrices used in averaging
3	.0183	.7100	.9017	10
4	00200	1.4450	2.1233	10
5	.0250	2.5933	4.2917	10
6	.0483	4.0817	7.4350	10
7	.0778	6.2222	11.8056	3
8	.1000	9.3667	18.9500	1
9	.1167	120 9167	26.3500	1
10	.1667	17.2000	36.6833	1
15	.4166	56.8500	125.4333	1
17	.5833	84.7167	185.8333	1
20	.8500	128.1333	309.3333	1

85C is Stanford Library program number 85C which uses Gaussian elimination with row equilibration and row interchanges.

269I.A. is C.A.C.M. algorithm 269 as modified to use interval arithmetic.

Hansen's is the method of Hansen described in this report.

Table 4

7. An Application - Testing for a Chebyshev System

7.1 Introduction

Given that a set of functions form a Chebyshev system, certain algorithms in approximation theory can be proved to converge. In particular, the second algorithm of Remez [7] can be applied to find the best approximation in the Chebyshev (minimax) sense to a continuous function by a linear combination of the functions forming the Chebyshev system. However, in some cases it is not known a priori whether a given set of functions form a Chebyshev system or not. In these cases the program-to be described can be used to indicate the presence or absence of the desired property. In case the set does not form a Chebyshev system the use of interval determinants can possibly prove this fact.

7.2 Definition of a Chebyshev system of functions

Given a set of linearly independent continuous functions, $\varphi_1(x), \ldots, \varphi_n(x)$ defined on a closed interval $[a,b]$, form a function

$$F(x) = \sum_{i=1}^{n} \lambda_i \varphi_i(x) \,. \tag{7.2.1}$$

If any such function, which is not identically zero on $[a,b]$, has not more than n - 1 zeros in $[a,b]$ with double zeros counted twice, then the set $\{\varphi_i(x)\}_1^n$ forms a Chebyshev system. For more an Chebyshev systems see [9] for example.

An equivalent definition is the following: given the set $\{\varphi_i(x)\}_1^n$ if for any set of n arbitrary distinct points $\{x_i\}_1^n$, $x_i \in [a,b]$, the determinant whose i, j element is $\varphi_i(x_j)$ is non-zero, then the set $\{\varphi_i(x)\}_1^n$ forms a Chebyshev system. That is, the determinant, D, given by

$$(7.2.2) \quad D = \begin{vmatrix} \varphi_1(x_1) & \varphi_1(x_2) & \cdots & \varphi_1(x_n) \\ \varphi_2(x_1) & \varphi_2(x_2) & \cdots & \varphi_2(x_n) \\ \vdots & \vdots & & \vdots \\ \varphi_n(x_1) & \varphi_n(x_2) & \cdots & \varphi_n(x_n) \end{vmatrix}$$

must be non-zero for any set of n distinct points in [a,b] .

The second definition is used in this program to test for a Chebyshev system. The determinant is tested to see if it has a zero for any possible set of distinct points $\{x_i\}_1^n$. It can be shown that D, given by (7.2.2) is a continuous function of $\{x_i\}_1^n$. Thus, if we order the $\{x_i\}$ by requiring that $x_1 < x_2 < \ldots < x_n$, we know that if there exists two sets, $\{x_i^+\}$ and $\{x_i^-\}$, such that $D(\{x_i^+\}) > 0$ and $D(\{x_i^-\}) < 0$, then D must be equal to zero for some other set $\{x_i^0\}$. This property is used in conjuction with interval determinant calculations to prove that a system is not a Chebyshev system.

Two examples of Chebyshev systems for any closed interval of the real line are the set $\{\varphi_i(x)\}_1^n$ with $\varphi_i(x) = x^{i-1}$, and the set $\{\psi_i(x)\}_1^n$ with $\psi_i(x) = T_{i-1}(x)$, where $T_i(x)$ represents the Chebyshev polynomial of the first kind of order i . That is,

(7.2.3) $$T_i(x) = \cos(i\theta), \cos\theta = x .$$

A third example is the set $\{1, \cos(x), \sin(x), \cos(2x), \sin(2x), \ldots, \cos(nx), \sin(nx)\}$ on the interval $[0, 2\pi]$.

An example of a set of functions that is not a Chebyshev system is $\{\varphi_i(\theta) = \cos(2i\theta), i=1,2, \ldots, n\}$ on the interval $0 \leq \theta \leq \pi/2$. This can be seen easily since as θ varies from 0 to $\pi/2$ the argument of the cosine varies, from 0 to $i\pi$, thus passing through i odd integer multiples of $\pi/2$ where the cosine takes the value zero. 'Hence, we have that $\varphi_i(\theta)$ possesses i distinct zeros in the interval $[0,\pi/2]$ which contradicts the requirement of no more than i - 1 zeros set forth in the definition.

7.3 The problem

Assume that we are given a set of functions $\{\varphi_i(x)\}_1^n$ defined on an interval $[a,b]$. We ask whether the functions form a Chebyshev system on that interval or not.

7.4 Description of the method

The method utilizes the determinant definition of a Chebyshev system to test a set $\{\varphi_i(x)\}_1^n$ for that property in an interval $[a,b]$. The steps involved can be outlined as follows:

(i) Choose an arbitrary initial set of n distinct points $\{x_i\}_1^n$ in $[a,b]$.

(ii) Knowing a priori the errors in calculating the functions $\varphi_i(x)$, $i=1, \ldots, n$, create an interval matrix which contains the matrix shown in (7.2.2). Calculate the interval determinant of this matrix. The interval obtained will be greater than zero, less than zero or contain zero. If it contains zero, minimize or maximize D until the interval determinant does not contain zero. (Always require that $x_1 < x_2 < \ldots < x_n$).

(iii) Depending on whether the interval determinant is positive or negative, then minimize or maximize $D(\{x_i\})$. When (if) a change in sign of $D(\{x_i\})$ occurs, use the interval determinant calculation again to see if the interval has changed sign. If so, it is proved that the $\{\varphi_i\}$ does not form a Chebyshev system. If not, try to minimize (maximize) D further and use the interval determinant calculation again. If no further minimization or maximization is possible and the interval determinant contains zero, this may indicate a zero determinant and thus not a Chebyshev system but it does not prove anything. However,, if both positive and negative interval values of the determinant can be found, it is proved that the $\{\varphi_i\}$ do not form a Chebyshev system.

In practice, there are an infinite number of choices for a new set $\{x_j\}$ in the interval, therefore a direct computation is impossible, To surmount this problem we use an approximate minimization technique on the determinant, D, as a function of n parameters, the set $\{x_j\}_1^n$. Thus, if an interval change in sign is found, we can be sure

that the set of functions does not form a Chebyshev system, however, if an interval containing zero or an interval of the same sign as the starting point is found, we can not be positive that we do not or do have a Chebyshev system. This dichotomy of certainty is due to the fact that any known numerical procedure for minimization can failto locate the absolute minimum of a function, thereby locating a non-zero minimum whereas in fact a zero minimum exists.

In spite of the uncertainty involved in this method, if a reasonably faithful minimization procedure is employed, a non-zero minimum or a zero minimum with very close points is a very good indication of Chebyshev system. A zero minimum (an interval containing zero), with well separated points is convincing evidence that a Chebyshev system is not at hand. A change in sign (a positive interval and a negative interval) is proof that a Chebyshev system is not at hand.

The program to test for Chebyshev systems incorporates three basic algorithms Algorithm 178 [8] as coded for a Burroughs 135500 was modified slightly and used to perform the minimization. A routine similar to Algorithm 269 [3] was used to evaluate intermediate determinants for the minimization and Hansen's method was used for the interval determinant calculations. The interval [a,b] and the functions $\{\varphi_i(x)\}_{i=1}^{n}$ must be specified for each particular problem.

7.5 Examples

The program as implemented on a Burroughs B5500 computer has been used to test several sets of functions. Some of these are given below with the points chosen and the corresponding interval determinant values.

The functions such as exp and cos which occur in the examples were computed by the B5500 system routines to approximately 11 accurate decimal digits. To ensure that the interval determinants that were evaluated contained the mathematically correct values, we added (subtracted) 10^{-8} to each computed element of D (7.2.2) to obtain the right (left) endpoint of the interval determinant.

example 1. Not a Chebyshev system

$$\{\varphi_i(x)\}_{i=1}^{2} = \{x, \exp(x)\} \text{ on } [0,3] . \quad ([10, p. 55])$$

	x_1	x_2	interval determinant
start	1.0	2.0	[1.9524922, 1.9524926]
finish	0.0	3.0	[-3.0000003, -2.9999997]

example 2. Not a Chebyshev system

$$\varphi_i(x) = \cos(2 \otimes i \otimes x),\ i = 1,2,3,4 \text{ on } [0,\pi/2] .$$

	x_1	x_2	x_3	x_4	interval determinant
start	0.3	0.6	0.9	1.4	[0.0343986, 0.0343990]
finish	0.1	0.8	0.9	1.5	[-2.4963293, -2.4963286]

example 3. A Chebyshev system

$$\varphi_i(x) = x^{i-1},\ i = 1,2,3 \text{ on } [-1,1] .$$

	x_1	x_2	x_3	interval determinant
start	-0.5	0.0	0.5	[0.24999993, 0.25000007]
finish	-0.040601	-0.0404	-0.0405	$[-8.571\times10^{-12}, +4.510\times10^{-12}]$

In this case the functions do form a Chebyshev system and we know that there should be no set of distinct points for which the determinant, D, is zero. The program is written to prevent points from becoming closer than 10^{-4}. As shown by-these results, the only way the minimization routine could obtain smaller values for D was to use points as close together as possible. This is typical of the results obtained when the given set of functions does form a Chebyshev system.

8. Conclusions

The computational results show that Hansen's method with pivot selection provides a smaller interval fcr the determinant than straightforward use of I.A. for all but one of the test matrices. This is as would be expected since the elements in the lower triangle of B^I are very small intervals and thus interval widths are kept small during the elimination.

As is well known, and as shown dramatically by Table 2, the correct selection of pivots during decomposition (or elimination) can produce a striking difference in the resultant accuracy of a determinant evaluation. Table 2 also shows that for matrices of order as high as 20, and with original element widths less than or equal to 2.0×10^{-8}, Hansen's method will compute interval determinants which retain useful significance. However, the interval determinants computed by the straightforward use of I.A. begin to lose significance for matrices whose order approaches 20 .

The timing results given in Table 4 indicate that some trade-off between accuracy and running time might be appropriate in particular applications. That is, some a priori information about input widths and order of the matrices, together with the information given in the Tables,, might indicate that the straightforward use of I.A. would give sufficient accuracy and save a considerable amount of machine time.

The application of interval determinant calculation in a program to test for Chebyshev systems shows that this particular use of interval arithmetic can be used in mathematical proofs.

[1] Hansen, E.; 'Interval arithmetic in matrix computations, part I," SIAM Journal on Numerical Analysis, 2 (1965), pp. 308-320.

[2] Hansen, E. and Smith, R. R.; --"Interval arithmetic in matrix computations, part II," SIAM Journal on Numerical Analysis, 4 (1967), pp. 1-9.

[3] Pfann, J. and Straka, J.; Algorithm 269, "Determinant Evaluation," Comm. ACM, vol. 8, No. 11 (1965), p. 668.

[4] Gibb, A.; Algorithm 61, "Procedures for range arithmetic," Comm. ACM, vol. 4, No. 7 (1961), p. 319.

[5] Ralston, A.; A First Course in Numerical Analysis. San Francisco: McGraw-Hill Book Co., 1965.

[6] Lord, A.; "B5500 Algol procedures for range arithmetic," CS239 report, Stanford Computer Science Department, (Aug. 1964).

[7] Remez, E. Y.: "General computational methods of Chebyshev Approximation". In The Problems with Linear Real Parameters. AEC-tr-4491, Books 1 and 2, English translation by US AEC, (1962).

[8] Kaupe, A. F.; Algorithm 178, "Direct Search," Comm. ACM, vol. 6, No. 6 (1963), p. 313.

[9] Karlin, S. J. and Studden, W. J.; Tchebycheff systems: with applications in analysis and statistics, New York: Interscience Publishers a division of John Wiley & Sons, 1966.

[10] Rice, J. R.; The Approximation of Functions Vol. 1, Palo Alto: Addison-Wesley Publishing Co., 1964.