Institute of Computer Science
Academy of Sciences of the Czech Republic

INTLAB Primer

Jiří Rohn

Technical report No. V-1117

27.07.2011

Pod Vodárenskou věží 2, 182 07 Prague 8, phone: +420 266 051 111, fax: +420 286 585 789,
e-mail:rohn@cs.cas.cz

Institute of Computer Science
Academy of Sciences of the Czech Republic

INTLAB Primer

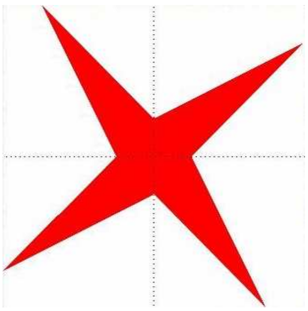
Jiří Rohn[1]

Technical report No. V-1117

27.07.2011

Abstract:

This report brings an extremely concise description of basic features of INTLAB, a MATLAB toolbox for verified computations. Only the very basics of INTLAB are presented; INTLAB itself contains much more.



Keywords:
INTLAB, MATLAB, verification software, primer.[2]

[1]This work was supported by the Institutional Research Plan AV0Z10300504.

[2]Above: logo of interval computations and related areas (depiction of the solution set of the system $[2,4]x_1 + [-2,1]x_2 = [-2,2]$, $[-1,2]x_1 + [2,4]x_2 = [-2,2]$ (Barth and Nuding [1])).

1 INTRODUCTION

This is an extremely concise description of basic features of INTLAB aimed at newcomers to this area. Only the very basics of INTLAB are presented; INTLAB itself contains much more.

2 INTLAB'S AUTHOR

INTLAB, a MATLAB toolbox for self-validating algorithms, is a one-man work. It has been created by Prof. Dr. Siegfried M. Rump, Institute for Reliable Computing, Technical University of Hamburg-Harburg, Germany, [2].

3 INTLAB'S DOWNLOAD

INTLAB can be downloaded (free for academic use; observe the license) from
`http://www.ti3.tu-harburg.de/rump/intlab/` .

4 INTLAB'S INSTALLATION

Follow the instructions in the file Readme.txt.

5 INTLAB'S DEMOS

Look at demointval.m, demointlab.m. Many important features contained therein are summarized below.

6 DEFAULT INITIALIZATIONS

There are two basic ways of displaying intervals:
intvalinit('displayinfsup') : display of intervals by endpoints,
intvalinit('displaymidrad') : display of intervals by midpoint and radius.
In what follows we use the first option. To invoke it, simply type

```
>> intvalinit('displayinfsup')
```

You will get the response

```
===> Default display of intervals by infimum/supremum (e.g. [ 3.14 , 3.15 ])
```

7 INTVAL TYPE

INTLAB uses newly defined MATLAB type "intval" for interval quantities.

8 BASIC INTERVAL ARITHMETIC PROPERTY

An operation always uses interval arithmetic if at least one of the operands is of type intval. The output is always rigorous (i.e., it encloses the real result within floating-point bounds).

9 INPUT

Given two real (of type double) matrices (vectors, numbers) A, B of the same size, the interval matrix (vector, number) C=[A,B] is inputed simply as C=infsup(A,B).
Example. For real[3] (double) matrices Ad, Ah given by

```
>> Ad=[2 -2; -1 2]
Ad =
     2    -2
    -1     2
>> Ah=[4 1; 2 4]
Ah =
     4     1
     2     4
>> A=infsup(Ad,Ah)
```

yields

```
intval A =
           [    2.0000,     4.0000] [   -2.0000,     1.0000]
           [   -1.0000,     2.0000] [    2.0000,     4.0000]
```

A is the interval matrix [Ad,Ah]. Alternatively, the same result can be achieved by using

```
>> A=[infsup(2,4)  infsup(-2,1);  infsup(-1,2)  infsup(2,4)]
intval A =
           [    2.0000,     4.0000] [   -2.0000,     1.0000]
           [   -1.0000,     2.0000] [    2.0000,     4.0000]
```

Similarly, A=midrad(C,D) gives the interval matrix [C-D,C+D] (i.e, the midpoint-radius representation). If you wish a real (double) matrix Ao to be handled as a thin interval matrix (so that the interval arithmetic could apply), use Ao=intval(Ao) (or, equivalently, Ao=infsup(Ao,Ao)):

```
>> Ao=[1 2;3 4]
Ao =
     1     2
     3     4
>> Ao=intval(Ao)
intval Ao =
            [    1.0000,     1.0000] [    2.0000,     2.0000]
            [    3.0000,     3.0000] [    4.0000,     4.0000]
```

[3]Throughout the text, “real” is meant as opposite of “interval”.

10 OUTPUT

Given an interval matrix A (of type intval), its bounds, midpoint and radius can be extracted as follows:
the lower bound is inf(A) or A.inf,
the upper bound is sup(A) or A.sup,
the midpoint is mid(A) or A.mid,
the radius is rad(A) or A.rad.

Example. With the above A, we have

```
intval A =
           [    2.0000,    4.0000] [   -2.0000,    1.0000]
           [   -1.0000,    2.0000] [    2.0000,    4.0000]
>> inf(A)
ans =
     2    -2
    -1     2
>> sup(A)
ans =
     4     1
     2     4
>> mid(A)
ans =
    3.0000   -0.5000
    0.5000    3.0000
>> rad(A)
ans =
    1.0000    1.5000
    1.5000    1.0000
```

11 DIFFERENCE BETWEEN A.inf AND inf(A)

A.inf and inf(A) are not entirely equivalent. A.inf allows indexing:

```
>> A.inf(1,1)
ans =
    2
```

whereas this does not work for inf(A). The same holds for A.sup and sup(A), etc.

12 ENCLOSURE OF THE INVERSE INTERVAL MATRIX

We shall demonstrate some typical INTLAB features on the function B=inv(A) which computes an enclosure of the interval inverse of A.

12.1 Interval input

```
>> A
intval A =
             [    2.0000,    4.0000] [   -2.0000,    1.0000]
             [   -1.0000,    2.0000] [    2.0000,    4.0000]
>> B=inv(A)
intval B =
             [   -2.9704,    3.6191] [   -3.2729,    3.3810]
             [   -3.3810,    3.2729] [   -2.9704,    3.6191]
```

This means that for each real (double) matrix Ao in A, its exact inverse is guaranteed to exist and to belong to B. The output uses outward rounding, so that the result is rigorous. The enclosure B is generally not optimal.

12.2 Real input

Inverse can also be used to obtain a rigorous output for a real (double) input:

```
Ao =
     1     2
     3     4
>> Ao=intval(Ao)
intval Ao =
              [    1.0000,    1.0000] [    2.0000,    2.0000]
              [    3.0000,    3.0000] [    4.0000,    4.0000]
>> B=inv(Ao)
intval B =
              [   -2.0001,   -1.9999] [    0.9999,    1.0001]
              [    1.4999,    1.5001] [   -0.5001,   -0.4999]
```

To see better the accuracy of the result, use

```
>> format long
>> B
intval B =
       [-2.00000000000001, -1.99999999999999] [ 0.99999999999999,  1.00000000000001]
       [ 1.49999999999999,  1.50000000000001] [-0.50000000000001, -0.49999999999999]
>> rad(B)
ans =
  1.0e-015 *
   0.44408920985006   0.33306690738755
   0.22204460492503   0.11102230246252
```

The very small radius matrix shows the high accuracy achieved.

12.3 Singularity

An attempt to invert a singular matrix

```
>> format short
Ao=[1 2;5 10]
Ao =
     1     2
     5    10
>> Ao=infsup(Ao,Ao)
intval Ao =
            [    1.0000,    1.0000] [     2.0000,     2.0000]
            [    5.0000,    5.0000] [    10.0000,    10.0000]
>> B=inv(Ao)
```

results in an interval matrix of NaN's:

```
intval B =
           [       NaN,        NaN] [       NaN,        NaN]
           [       NaN,        NaN] [       NaN,        NaN]
```

12.4 NaN output

Using NaN's for indication of an empty output is a typical feature of INTLAB. It enables feasibility of succeeding computations. E.g., in the above example

```
>> C=inv(Ao)*rand(2,2)
```

gives

```
intval C =
           [       NaN,        NaN] [       NaN,        NaN]
           [       NaN,        NaN] [       NaN,        NaN]
```

so that the computation does not break down despite the singularity of Ao.

13 SYSTEMS OF INTERVAL LINEAR EQUATIONS

A system of interval linear equations A*x=b (A, b of type intval, A square) can be solved (i.e., an enclosure of the solution set can be obtained) using X=verifylss(A,b).

13.1 Interval data

Example.

```
>> A
intval A =
           [    2.0000,    4.0000] [   -2.0000,    1.0000]
           [   -1.0000,    2.0000] [    2.0000,    4.0000]
>> bl=[-2 -2]'; bu=-bl;
>> b=infsup(bl,bu)
intval b =
           [   -2.0000,    2.0000] [   -2.0000,    2.0000]
```

```
>> X=verifylss(A,b)            % Barth-Nuding 1974 example, see [1]
intval X =
           [  -14.0001,   14.0001]
           [  -14.0001,   14.0001]
```

It is guaranteed that for each Ao in A and bo in b, Ao is nonsingular and the solution of Ao*x=bo is contained in X. The enclosure X is generally not optimal: the optimal enclosure XX would be

```
intval XX =
            [   -4.0000,    4.0000]
            [   -4.0000,    4.0000]
```

(see the logo on the abstract page). The overestimation is caused by too wide input intervals here.

13.2 Real data

The procedure can be used for solving systems with thin data (without converting them to type intval first, contrary to the procedure "inv" above).

Example.

```
>> A=[ 1 2 3;4 5 6;7 8 10]
A =
     1     2     3
     4     5     6
     7     8    10
>> b=A*ones(3,1)                % so that the exact solution is [1 1 1]'
b =
      6
     15
     25
>> format long
>> X=verifylss(A,b)
intval X =
           [   0.99999999999999,   1.00000000000001]
           [   0.99999999999999,   1.00000000000001]
           [   0.99999999999999,   1.00000000000001]
```

We can see that the degree of guaranteed accuracy is surprising.

14 DATA CONVERSION

Most numbers (like 0.1) cannot be exactly represented in binary finite precision arithmetic. To handle errors created by data conversion, INTLAB enables us to use

```
>> a=intval('0.1')           %  notice the apostrophes
intval a =
```

```
           [   0.09999999999999,    0.10000000000001]
>> rad(a)
ans =
      1.387778780781446e-017
```

("rigorous representation").

15 SOME ADDITIONAL INTERVAL FUNCTIONS

For x, y of type intval:

```
intersect(x,y) intersection of x, y
hull(x,y)      union of x, y
abss(x)        absolute value of x
mig(x)         mignitude of x
in(x,y)        x is included in y (logical array)
in0(x,y)       x is included in the interior of y (logical array)
```

16 WEB VERSION

Web version of this text can be found at
http://uivtx.cs.cas.cz/~rohn/matlab/primer/intlab_primer.html .

17 YOUR TURN NOW

Now it is your turn to explore INTLAB, this wonderful tool created by Siegfried Michael Rump. (Thanks also for his remarks on this text.)

Bibliography

[1] W. Barth and E. Nuding, *Optimale Lösung von Intervallgleichungssystemen*, Computing, 12 (1974), pp. 117–125. 1

[2] S. Rump, *INTLAB - INTerval LABoratory*, in Developments in Reliable Computing, T. Csendes, ed., Kluwer Academic Publishers, Dordrecht, 1999, pp. 77–104. http://www.ti3.tu-harburg.de/rump/. 2