Computing Enclosures of Overdetermined Interval Linear Systems

Affiliation

Jaroslav Horáček

horacek@kam.mff.cuni.cz

[Charles University in Prague, Czech Republic]
[Department of Applied Mathematics]

supervisor: Mgr. Milan Hladík, Ph.D.

Contents

Basic notation

- $\boldsymbol{A}, \boldsymbol{b}$ indicate an interval matrix and vector respectively
- A, b indicate a point real matrix and vector respectively
- $\boldsymbol{A} = [\underline{A}, \overline{A}]$, where $\underline{A}$ is called *lower bound* and $\overline{A}$ is called *upper bound*
- Also $\boldsymbol{A} = \langle A_c, A_\Delta \rangle$, where A_c is *midpoint matrix* and A_Δ is *radius matrix*
- It holds $A_c = (\underline{A} + \overline{A})/2$
- It holds $A_\Delta = (\overline{A} - \underline{A})/2$

General interval linear system

Definition

$$\boldsymbol{A}x = \boldsymbol{b}, \quad \text{where}$$

- $\boldsymbol{A} \in \mathbb{IR}^{m \times n}$ (interval matrix)
- $\boldsymbol{b} \in \mathbb{IR}^{m \times 1}$ (interval vector)
- $\mathbb{IR}$ is the set of all real closed intervals

Solution of an interval linear system

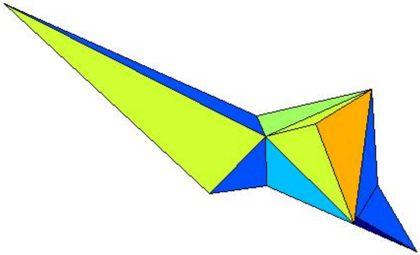
Definition

The solution set of $\mathbf{A}x = \mathbf{b}$ is

$$\Sigma = \{x \mid Ax = b \text{ for some } A \in \mathbf{A}, b \in \mathbf{b}\}.$$

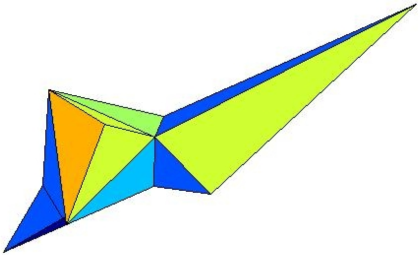
- Collection of all solutions of all point-real instances of an interval system.
- Not the least squares approach!
- If no instance has solution, we call the whole interval system - *unsolvable*

Solution of interval linear system



- It is a polyhedral set
- Not necessarily convex
- But convex in each orthant

Description of a solution



- Difficult to describe
- $\Rightarrow$ one possibility – a tight n-dimensional box *(interval hull)*
- NP-hard (even approximation)
- $\Rightarrow$ We are looking for a box as narrow as possible containing the hull *(interval enclosure)*

Overdetermined interval linear system

Definition

$$\mathbf{A}x = \mathbf{b}, \quad \text{where}$$

- $\mathbf{A} \in \mathbb{IR}^{m \times n}$, where $m > n$
- $\mathbf{b} \in \mathbb{IR}^{m \times 1}$
- $\mathbb{IR}$ is the set of all real closed intervals

"*More equations than variables.*"

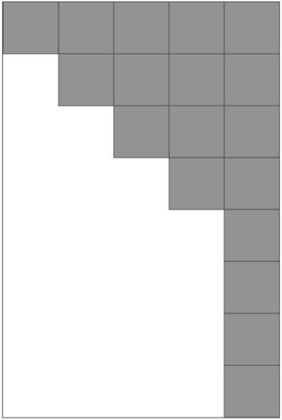
- More difficult than a square system.

Methods for solving OILS

- Gaussian elimination (GE)
- Iterative methods – Jacobi, Gauss-Seidel
- Supersquare methods
- Subsquare methods
- Rohn method
- Linear programming

Gaussian elimination (GE)

- Hansen 2006
- Adapted Gaussian elimination - we eliminate to the shape



Gaussian elimination (GE)

- We have $m - n + 1$ equations in the shape $x_n = [a_i, b_i]$
- We provide intersection
- Empty intersection - means no solution
- Infinite intersection means infinite solution or overestimation
- Then the backward substitution
- O.K. only for small systems $n \sim 4$
- For larger systems we get great overestimation

Grow by GE

x_1	0.1479	227.6698
x_2	15.2091	172.5929
x_3	11.1031	68.4653
x_4	9.7809	64.1056
x_5	-8.8168	27.2234
x_6	25.8164	25.8398
x_7	-19.0444	30.4596
x_8	-22.0799	11.0313
x_9	1.9649	12.1172
x_{10}	-19.1817	11.6841
x_{11}	-20.9670	1.9153
x_{12}	-4.6988	3.5407
x_{13}	3.1223	4.3894

(Variable|Midpoint|Radius) for a random system 15×13 with random radii $\leq 10^{-3}$

Gaussian elimination (GE) - preconditioning

- For square systems typical preconditioning is A_c the midpoint matrix
- We tested the preconditioning proposed by Hansen

$$B = \begin{bmatrix} A_1^c & 0 \\ A_2^c & I \end{bmatrix}.$$

- Preconditioning widens the solution $\Rightarrow$ unsolvable system becomes solvable
- When we want to test unsolvability of a system we can not use preconditioning (works only for $n \sim 10$)

Iterative methods

- For square systems – Jacobi, Gauss-Seidel, Krawczyk, etc. cannot be used for OILS
- Without preconditioning these method often have no sense when computing with intervals

Iterative methods - Solutions

- **(1)** After Hansen preconditioning we get almost this shape

$$\begin{pmatrix} \sim 1 & \sim 0 & \dots & \sim 0 \\ \sim 0 & \sim 1 & \dots & \sim 0 \\ \vdots & \vdots & \ddots & \vdots \\ \sim 0 & \sim 0 & \dots & \sim 1 \\ \sim 0 & \sim 0 & \dots & \sim 0 \\ \vdots & \vdots & \ddots & \vdots \\ \sim 0 & \sim 0 & \dots & \sim 0 \end{pmatrix}$$

- We can use only the upper $n \times n$ subsquare
- Solution enclosure is still rigorous but with loss of information
- For matrices with radii close to 10^{-2} preconditioned matrix often contains zeros on diagonal

Iterative methods - Solutions

- **(2)** do some simple tricks (later)
- **(3)** Rohn method (later)
- **(4)** Transform our system to the new square one

(4) transform the system to a new system

Supersquares method

- Classical $\boldsymbol{A}^T\boldsymbol{A}x = \boldsymbol{A}^T\boldsymbol{b}$ does not work!
- Even $(C\boldsymbol{A})^T(C\boldsymbol{A})x = (C\boldsymbol{A})^T\boldsymbol{b}$ does not work for some preconditioner C
- But what works is

$$\begin{pmatrix} I & \boldsymbol{A} \\ \boldsymbol{A}^T & 0 \end{pmatrix} \begin{pmatrix} y \\ x \end{pmatrix} = \begin{pmatrix} \boldsymbol{b} \\ 0 \end{pmatrix},$$

- Formula resembles the least squares, but it is actually not
- However the least squares solution is contained

Supersquares method

- Now we can apply our favourite iterative method
- Problem - for system 50×3 we have to solve a system 53×53
- This approach always returns solution!
- Implemented e.g. in Intlab - method verifylss()
- It is actually a parametric system

Subsquares methods

- Simple tricks to enable the use of the square iterative methods
- **(1)** solve some (random?) square subsystems of the overdetermined one and intersect the solutions
- If we check all of them we get something really close to the interval hull and for small systems it is cheaper than linear programming
- Also this way allows to detect unsolvability very quickly after few trials (random system 200×170 with $r \leq 10^{-2} \sim 5$ steps; with $r \leq 10^{-4} \sim 2$ steps)

Subsquares methods - times

matrix	time (sec)
5×3	0.1
9×5	0.6
11×6	2.15
13×7	7.67
15×13	0.62
15×7	28.16
23×19	42.43
40×39	0.30
40×38	4.47

Table: Subsquares - times when evaluating all subsystems

Subsquares methods

- **(2)** Use some distinct overlaping subsystems covering the whole OILS and apply some iterative method to them (Jacobi)
- Intersect all the partial solutions after each iteration - this works quite well
- The more systems we choose, the better
- The problem we currently work on is which and how many subsystems to choose

Rohn theorem

Let $\boldsymbol{A}x = \boldsymbol{b}$ be an IOLS with a solution S. Let R be arbitrary real $n \times m$ matrix and let x_0 and $d > 0$ are arbitrary n-dimensional real vector such that

$$Gd + g < d,$$

where

$$G = |I - RA_c| + |R|A_\Delta$$

and

$$g = |R(A_c x_0 - b_c)| + |R|(A_\Delta |x_0| + b_\Delta).$$

Then

$$S \subseteq [x_0 - d, x_0 + d].$$

Rohn method

- Iterative computing of d
- $x_0 \approx Rb_c.$, $R \approx (A_c^T A_c)^{-1} A_c^T$
- We does not have to use A_c we can use $A \in \mathbf{A} \Rightarrow$ iteration
- This method works well
- However sometimes we cannot find d when some radii are ≤ 0.1

Linear programming

Oettli-Prager theorem

Vector $x \in \mathbb{R}^n$ is a (weak) solution of an interval system if and only if

$$|A_c x - b_c| \leq A_\Delta |x| + b_\Delta .$$

- First absolute value can be rewritten
- Second one with the knowledge of current orthant we are at
- Then we have a system of point real inequalities $\Rightarrow$ Linear programming - we get interval hull
- Problem - we have to solve $2^n \times (2n)$ linear programming problems (a system $15 \times 9 \sim 28$ min)
- Solution - we can solve with some worse method and then compute LP only in the orthant returned

Linear programming - times (random systems)

matrix	time
5×3	6 sec
9×5	43 sec
13×7	5 min
15×9	28 min

Table: When searching all orthants

matrix	time
5×3	1 sec
15×10	3.5 sec
25×21	13 sec
35×23	19 sec
45×31	43 sec
55×35	1 min
73×55	9 min

Table: When searching orthants according to a sign vector – supersquares

Comparison of methods

Comparison of methods - widths of enclosures

- Tested widths according to verifylss (Intlab)
- $\frac{1}{n}\sum_{i=1}^{n} \frac{\mathrm{w}(x_i)}{\mathrm{w}(x_i^{intlab})}$

matrix \ *method*	*GSpre*	*Rohn*	*Intlab*	*Subsq*	*GE*
35×23	11.1655	1.0177	1.000	1.4045	11.1664
50×35	11.8722	1.0108	1.000	1.5445	11.8736
100×87	13.513	1.0014	1.000	1.4222	13.515
200×170	16.3620	0.999	1.000	1.8641	16.3639

Table: Width of enclosures for systems rad $< 10^{-4}$, $\epsilon = 10^{-5}$

Comparison of methods - times of computation

- Tested using Matlab functions – tic, toc

matrix \ *method*	*GSpre*	*GE*	*Rohn*	*Subsq*	*Intlab*
5×3	0.022	0.0239	0.00096	0.0217	0.0046
15×13	0.0731	0.1365	0.0018	0.0666	0.005
35×23	0.1257	0.5657	0.0026	0.1625	0.0065
50×35	0.1927	1.1618	0.0048	0.2513	0.0103
100×87	0.4961	4.9578	0.034	1.0431	0.0386
200×170	1.3616	19.7027	0.1678	6.3006	0.2428

Table: Times of computations for systems rad $< 10^{-4}$, $\epsilon = 10^{-5}$

- Matlab / Intlab / Versoft
- Toolbox LIME 1.0
- Documentation (html + sourcecode)
- Free for non-comercial use

Conclusions and future

- Solving of overdetermined interval linear systems is sometimes needed
- Information about unsolvability is sometimes needed
- Still much can be done in this area - derandomization, theoretical links, new effective methods

Thank you very much for your attention.