

Résolution de systèmes linéaires par intervalles : raffinement itératif

Master S2IFA
Algorithmique numérique et fiabilité des calculs en arithmétique flottante

20 mars 2013

Agenda

Verified linear system solving

Nguyen's algorithm

Recap

Interval linear system

Input:

$\mathbf{A} \in \mathbb{IR}^{n \times n}$ a matrix with interval coefficients

$\mathbf{b} \in \mathbb{IR}^n$ a vector with interval coefficients.

Output:

$$\Sigma_{\exists, \exists}(\mathbf{A}, \mathbf{b}) = \{x \in \mathbb{R}^n : \exists A \in \mathbf{A}, \exists b \in \mathbf{b}, Ax = b\}.$$

Interval linear system

Input:

$\mathbf{A} \in \mathbb{IR}^{n \times n}$ a matrix with interval coefficients

$\mathbf{b} \in \mathbb{IR}^n$ a vector with interval coefficients.

Output:

$$\Sigma_{\exists, \exists}(\mathbf{A}, \mathbf{b}) = \{x \in \mathbb{R}^n : \exists A \in \mathbf{A}, \exists b \in \mathbf{b}, Ax = b\}.$$

Oettli-Präger theorem

Let $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{b} \in \mathbb{R}^n$.

$$x \in \Sigma_{\exists, \exists}(\mathbf{A}, \mathbf{b}) \Leftrightarrow |\mathbf{A}_m x - \mathbf{b}_m| \leq \mathbf{A}_r |x| + \mathbf{b}_r.$$

Proof:

Let us denote by $\mathbf{a} = \mathbf{A}x$.

$$\mathbf{A}x \cap \mathbf{b} \neq \emptyset \Leftrightarrow |\mathbf{a}_m - \mathbf{b}_m| \leq \mathbf{a}_r + \mathbf{b}_r.$$

Consequence:

$\Sigma_{\exists, \exists}(\mathbf{A}, \mathbf{b})$ is defined by the following inequalities, on each orthant defined by $D = \text{diag}(d_i)$ (where $d_i = \text{sign}(x_i)$):

$$\begin{aligned} \mathbf{A}_m x - \mathbf{A}_r D x &\leq \mathbf{b}_m + \mathbf{b}_r \\ \mathbf{A}_m x - \mathbf{A}_r D x &\geq -\mathbf{b}_m - \mathbf{b}_r \\ D x &\geq 0 \end{aligned}$$

Solution of an interval linear system: what does it look like?

$$\mathbf{A} = \begin{pmatrix} [2, 4] & [-1, 1] \\ & [2, 4] \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} [-3, 3] \\ 0 \end{pmatrix}$$

and the solution set is defined by

$$\Sigma_{\exists, \exists}(\mathbf{A}, \mathbf{b}) = \{x \in \mathbb{R}^2 : 2|x_2| \leq |x_1|, 2|x_1| \leq 3 + |x_2|\}.$$

Complexity of solving an interval linear system

Complexity: almost every problem is NP-hard

Gaganov 1982, Rohn 1994 ff, Kreinovich...

- ▶ evaluate a function on a box (cartesian product of intervals)
- ▶ evaluate a function on a box up to ε
- ▶ solve a linear system
- ▶ solve a linear system up to $1/4n^4$ ($n = \text{dim. of the system}$)
- ▶ determine if the solution of a linear system is bounded
- ▶ compute the matrix norm $\|\mathbf{A}\|_{\infty,1}$
- ▶ determine if an interval matrix (= a matrix with interval coefficients) is regular, i.e. if every possible punctual matrix in it is regular
- ▶ ...

An idea (but not a proof!) is given by Oettli-Präger theorem: one has to explore every orthant. In \mathbb{R}^n , there are 2^n orthants.

When does a solution exist?

M-matrix: $\mathbf{A} \in \mathbb{R}^{n \times n}$ is an M-matrix iff $\mathbf{A}_{i,j} \leq 0$ for all $i \neq j$ and there exists $u \in \mathbb{R}^n$ such that $\mathbf{A}u > 0$.

Theorem about M-matrices:

- ▶ every M-matrix in $\mathbb{R}^{n \times n}$ is regular and its inverse has all its coefficients nonnegative;
- ▶ every M-matrix $\mathbf{A} = [\underline{\mathbf{A}}, \bar{\mathbf{A}}]$ is regular and the set of the inverses of $A \in \mathbf{A}$ is $[\bar{\mathbf{A}}^{-1}, \underline{\mathbf{A}}^{-1}]$;
- ▶ if $\mathbf{A} \in \mathbb{R}^{n \times n}$ is an M-matrix and if $\mathbf{B} \subset \mathbf{A}$ then \mathbf{B} is an M-matrix; item in particular, \mathbf{A} is an M-matrix iff $\underline{\mathbf{A}}$ and $\bar{\mathbf{A}}$ are M-matrices in $\mathbb{R}^{n \times n}$.

When does a solution exist?

Comparison matrix:

for $\mathbf{A} \in \mathbb{R}^{n \times n}$, its comparison matrix $\langle \mathbf{A} \rangle \in \mathbb{R}^{n \times n}$ is defined by:

- ▶ $\langle \mathbf{A} \rangle_{i,i} = \min(|\underline{A}_{i,i}|, |\bar{A}_{i,i}|)$ if $0 \notin \mathbf{A}_{i,i}$ and 0 otherwise,
- ▶ $\langle \mathbf{A} \rangle_{i,j} = -|\mathbf{A}_{i,j}|$ for $i \neq j$.

H-matrix: $\mathbf{A} \in \mathbb{R}^{n \times n}$ is an H-matrix iff its comparison matrix $\langle \mathbf{A} \rangle$ is an M-matrix.

Theorem about H-matrices:

- ▶ every H-matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is regular and satisfies
 $|\mathbf{A}^{-1}| \leq \langle \mathbf{A} \rangle^{-1}$
- ▶ ...

Hansen-Blik-Rohn-Ning-Kearfott formula

Theorem

Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be an H-matrix and $\mathbf{b} \in \mathbb{R}^n$.

Let us denote

$$\begin{aligned} z &= \langle \mathbf{A} \rangle^{-1} \cdot |\mathbf{b}| \\ d &= \text{diag}(\langle \mathbf{A} \rangle^{-1}) : d_i = (\langle \mathbf{A} \rangle^{-1})_{i,i} \\ \alpha_i &= \langle \mathbf{A}_{i,i} \rangle - 1/d_i \\ \beta_i &= \frac{z_i}{d_i} - |\mathbf{b}_i|. \end{aligned}$$

Then it holds that $\Sigma_{\exists\exists}(\mathbf{A}, \mathbf{b}) \subset \mathbf{x}$ with

$$\mathbf{x}_i = \frac{\mathbf{b}_i + [-\beta_i, \beta_i]}{\mathbf{A}_{i,i} + [-\alpha_i, \alpha_i]}.$$

Both are equal if \mathbf{A} is diagonal.

Hansen-Bliek-Rohn-Ning-Kearfott formula ... in MatLab

The system considered here is the residual system, where the matrix is premultiplied by an approximate inverse: $\mathbf{A} = \mathbf{R} \cdot \mathbf{A}_{init}$ and $\mathbf{b} = \mathbf{b}_{init} - \mathbf{A}_{init}\mathbf{x}$.

```
cA = compmat (A)
B = inv(cA)
v = abs(B * ones(n,1))
setRound(-1)
u = cA * v
if ~ (all(u) > 0)
    x = NaN, return
else
    Res = cA * B - eye(n)
    setRound(1)
    w = ...
    d = v .* w' - diag(B)
    B = B + v * w'
    z = B * abs(b)
    beta = z ./ d - abs(b)
    alpha = diag(cA) + (-1) ./ diag(B)
    x=(b+midrad(0,beta))./(diag(A)+midrad(0,alpha))
```

Algorithm: linear systems solving (Hansen-Sengupta)

Problem: solve $\mathbf{Ax} = \mathbf{b}$, i.e. determine

$$\text{Hull}(\Sigma_{\exists\exists}(\mathbf{A}, \mathbf{b})) = \text{Hull}(\{x : \exists A \in \mathbf{A}, \exists b \in \mathbf{b}, Ax = b\}).$$

Pre-processing:

multiply the system by an approximate $\text{mid}(\mathbf{A})^{-1}$.

New system = $\text{mid}(\mathbf{A})^{-1}\mathbf{Ax} = \mathbf{b}$.

Hope: contracting iteration.

Algorithm: apply Gauss-Seidel iteration

while convergence not reached loop

for $i = 1$ to n do

$$\mathbf{x}_i := \left(\mathbf{b}_i - \sum_{j \neq i} \mathbf{A}_{i,j} \mathbf{x}_j \right) / \mathbf{A}_{i,i}$$

Algorithms to solve interval linear systems

- ▶ Complexity: polynomial in time ... for an NP-hard problem: no guarantee on the accuracy of the solution, failure is possible.
- ▶ The algorithm presented during this lecture is usually employed to verify the solution of a linear system with floating-point coefficients: interval arithmetic is used as a verification tool.

Agenda

Verified linear system solving

Nguyen's algorithm

Recap

Problem: verified solution of a linear system

Goals: For a linear system $Ax = b$ with $A \in \mathbb{F}^{n \times n}$ non-singular and $b \in \mathbb{F}^n$, we want to

1. compute an approximation $\tilde{x} \in \mathbb{F}^n$ of the exact solution x^* ,
2. simultaneously bound the error upon \tilde{x} , or enclose it in an interval

$$e \ni x^* - \tilde{x}.$$

Remark: denote by e the error $x^* - \tilde{x}$.

Then e is the solution of the residual system $Ae = b - A\tilde{x}$.

Indeed, $Ae = A(x^* - \tilde{x}) = Ax^* - A\tilde{x} - b + b - A\tilde{x}$.

Method: contractant iteration

Classical iterative refinement

Wilkinson (1963), Higham (2000), Demmel et al. (2006) ..

Algorithm (Classical iterative refinement)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$

% MatLab-like syntax

while(not converged)

$$\tilde{r} = b - A \tilde{x}$$

$$\tilde{e} = A \setminus \tilde{r}$$

$$\tilde{x} = \tilde{x} + \tilde{e}$$

end

Output: \tilde{x}



Method: contractant iteration Interval iterative refinement

Neumaier (1990), Rump (1999)

Algorithm (certifylss)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$

% MatLab-like syntax

while(not converged)

$\tilde{r} = b - A \tilde{x}$

$\tilde{e} = A \setminus \tilde{r}$

$\tilde{x} = \tilde{x} + \tilde{e}$

end

Output: \tilde{x}



Method: contractant iteration Interval iterative refinement

Neumaier (1990), Rump (1999)

Algorithm (certifylss)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$

% MatLab-like syntax

while(not converged)

$\mathbf{r} = [b - A \tilde{x}]$

% $A(x^ - \tilde{x}) \in \mathbf{r}$*

$\tilde{e} = A \setminus \tilde{r}$

$\tilde{x} = \tilde{x} + \tilde{e}$

end

Output: \tilde{x}



Method: contractant iteration Interval iterative refinement

Neumaier (1990), Rump (1999)

Algorithm (certifylss)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$

while(not converged)

$\mathbf{r} = [b - A \tilde{x}]$

$\mathbf{e} = A \setminus \mathbf{r}$

$\tilde{x} = \tilde{x} + \tilde{\mathbf{e}}$

end

Output: \tilde{x}

% MatLab-like syntax

% $A(x^ - \tilde{x}) \in \mathbf{r}$*

% $x^ - \tilde{x} \in \mathbf{e}$*



Method: contractant iteration Interval iterative refinement

Neumaier (1990), Rump (1999)

Algorithm (certifylss)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$ % MatLab-like syntax

while(not converged)

$\mathbf{r} = [b - A \tilde{x}]$ % $A(x^* - \tilde{x}) \in \mathbf{r}$

$\mathbf{e} = A \setminus \mathbf{r}$ % $x^* - \tilde{x} \in \mathbf{e}$

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$, $\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

end

Output: \tilde{x}



Method: contractant iteration Interval iterative refinement

Neumaier (1990), Rump (1999)

Algorithm (certifylss)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$ % MatLab-like syntax

while(not converged)

$\mathbf{r} = [b - A \tilde{x}]$ % $A(x^* - \tilde{x}) \in \mathbf{r}$

$\mathbf{e} = A \setminus \mathbf{r}$ % $x^* - \tilde{x} \in \mathbf{e}$

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$, $\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

end

Output: $\mathbf{x} = \tilde{x} + \mathbf{e}$



Method: contractant iteration Interval iterative refinement

Neumaier (1990), Rump (1999)

Algorithm (certifylss)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$ *% MatLab-like syntax*

while(not converged)

$\mathbf{r} = [b - A \tilde{x}]$ *% $A(x^* - \tilde{x}) \in \mathbf{r}$*

$\mathbf{e} = A \setminus \mathbf{r}$ *% $x^* - \tilde{x} \in \mathbf{e}$*

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$, $\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

end

Output: $\mathbf{x} = \tilde{x} + \mathbf{e}$



Solving interval residual system?

Method: contractant iteration Interval iterative refinement

Neumaier (1990), Rump (1999)

Algorithm (certifylss)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$, $R = \text{inv}(A)$, $K = [RA]$

while(not converged)

$r = [b - A \tilde{x}]$ % $A(x^* - \tilde{x}) \in r$

$e = A \setminus r$ % $x^* - \tilde{x} \in e$

$\tilde{x} = \tilde{x} + \text{mid}(e)$, $e = e - \text{mid}(e)$

end

Output: $x = \tilde{x} + e$



K is close to Identity \Rightarrow there are algorithms to solve this system. ☰ 🔍 ↻

Method: contractant iteration Interval iterative refinement

Neumaier (1990), Rump (1999)

Algorithm (certifylss)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$, $R = \text{inv}(A)$, $\mathbf{K} = [RA]$

while(not converged)

$\mathbf{r} = [Rb - \mathbf{K} \tilde{x}]$ % $RA(x^* - \tilde{x}) \in \mathbf{r}$

$\mathbf{e} = A \setminus \mathbf{r}$ % $x^* - \tilde{x} \in \mathbf{e}$

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$, $\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

end

Output: $\mathbf{x} = \tilde{x} + \mathbf{e}$



\mathbf{K} is close to Identity \Rightarrow there are algorithms to solve this system

Method: contractant iteration Interval iterative refinement

Neumaier (1990), Rump (1999)

Algorithm (certifylss)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$, $R = \text{inv}(A)$, $\mathbf{K} = [RA]$

while(not converged)

$\mathbf{r} = [Rb - \mathbf{K} \tilde{x}]$ % $RA(x^* - \tilde{x}) \in \mathbf{r}$

$\mathbf{e} = \mathbf{K} \setminus \mathbf{r}$ % $x^* - \tilde{x} \in \mathbf{e}$

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$, $\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

end

Output: $\mathbf{x} = \tilde{x} + \mathbf{e}$



\mathbf{K} is close to Identity \Rightarrow there are algorithms to solve this system

Method: contractant iteration Interval iterative refinement

Neumaier (1990), Rump (1999)

Algorithm (certifylss)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$, $R = \text{inv}(A)$, $\mathbf{K} = [RA]$

while(not converged)

$\mathbf{r} = [Rb - \mathbf{K} \tilde{x}]$ % $RA(x^* - \tilde{x}) \in \mathbf{r}$

$\mathbf{e} = \mathbf{K} \setminus \mathbf{r}$ % $x^* - \tilde{x} \in \mathbf{e}$

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$, $\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

end

Output: $\mathbf{x} = \tilde{x} + \mathbf{e}$



\mathbf{K} is close to Identity \Rightarrow there are algorithms to solve this system

Method: contractant iteration Interval iterative refinement

Neumaier (1990), Rump (1999)

Algorithm (certifylss)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$, $R = \text{inv}(A)$, $\mathbf{K} = [RA]$

while(not converged)

$\mathbf{r} = [Rb - \mathbf{K} \tilde{x}]$ % $RA(x^* - \tilde{x}) \in \mathbf{r}$

$\mathbf{e} = \mathbf{K} \setminus \mathbf{r}$ % $x^* - \tilde{x} \in \mathbf{e}$

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$, $\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

end

Output: $\mathbf{x} = \tilde{x} + \mathbf{e}$



This algorithm can fail, if it fails to solve the interval linear system.

Differences with `verifylss`

Iterative refinement performed on the interval residual.

- ▶ initialization of \mathbf{e} : heuristic trying to determine \mathbf{e}_0 , based on **Proposition**: let $A \in \mathbb{F}^{n \times n}$ and $R \in \mathbb{F}^{n \times n}$ be a floating-point approximate inverse of A .

If $\langle [RA] \rangle u \geq v > 0$ for some $u > 0$ then

$$\begin{aligned} |A^{-1}\mathbf{r}| &\leq \|R\mathbf{r}\|_v u \\ A^{-1}\mathbf{r} &\subset \|R\mathbf{r}\|_v [-u, u]. \end{aligned}$$

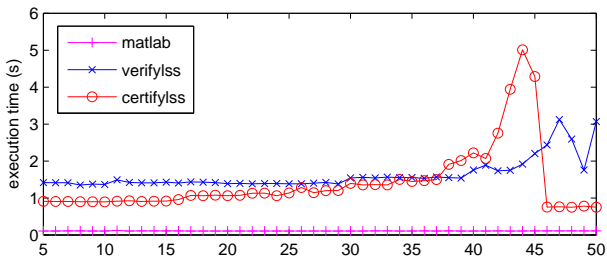
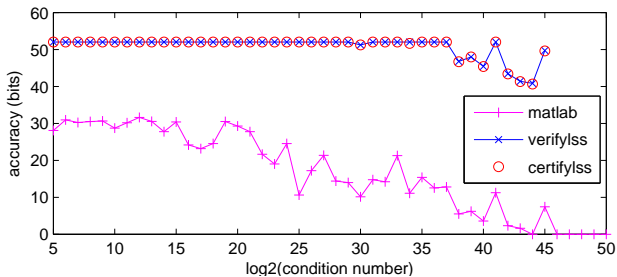
Idea: start from $u = e = (1, 1, \dots, 1)^t$ and modify u if v is not ≥ 0 .

Failure of the algo if failure of this step.

- ▶ solve $\mathbf{K}\mathbf{e} = \mathbf{r}$ using Gauss-Seidel iteration: known to converge quicker than Krawczyk.

Experimental Results

Dimension: 1000 $b = [1, \dots, 1]^T$



Method: contractant iteration

Relaxed interval iterative refinement

Algorithm (certifylss_relaxed)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$, $R = \text{inv}(A)$, $\mathbf{K} = [RA]$

while(not converged)

$\mathbf{r} = [Rb - \mathbf{K} \tilde{x}]$ % $RA(x^* - \tilde{x}) \in \mathbf{r}$

$\mathbf{e} = \mathbf{K} \setminus \mathbf{r}$ % $x^* - \tilde{x} \in \mathbf{e}$

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$, $\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

end

Output: $\mathbf{x} = \tilde{x} + \mathbf{e}$

Method: contractant iteration

Relaxed interval iterative refinement

Algorithm (certifylss_relaxed)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$, $R = \text{inv}(A)$, $\mathbf{K} = [RA]$,

$\hat{\mathbf{K}} = \text{inflated}(\mathbf{K})$ % $\hat{\mathbf{K}}$ is centered on a diagonal matrix

while(not converged)

$\mathbf{r} = [Rb - \mathbf{K} \tilde{x}]$ % $RA(x^* - \tilde{x}) \in \mathbf{r}$

$\mathbf{e} = \mathbf{K} \setminus \mathbf{r}$ % $x^* - \tilde{x} \in \mathbf{e}$

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$, $\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

end

Output: $\mathbf{x} = \tilde{x} + \mathbf{e}$

Method: contractant iteration

Relaxed interval iterative refinement

Algorithm (certifylss_relaxed)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$, $R = \text{inv}(A)$, $\mathbf{K} = [RA]$,

$\hat{\mathbf{K}} = \text{inflated}(\mathbf{K})$ % $\hat{\mathbf{K}}$ is centered on a diagonal matrix

while(not converged)

$\mathbf{r} = [Rb - \mathbf{K} \tilde{x}]$ % $RA(x^* - \tilde{x}) \in \mathbf{r}$

$\mathbf{e} = \hat{\mathbf{K}} \setminus \mathbf{r}$ % cost: 1 floating-point matrix-vector product

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$, $\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

end

Output: $\mathbf{x} = \tilde{x} + \mathbf{e}$

Method: contractant iteration

Relaxed interval iterative refinement

Algorithm (certifylss_relaxed)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$, $R = \text{inv}(A)$, $\mathbf{K} = [RA]$,

$\hat{\mathbf{K}} = \text{inflated}(\mathbf{K})$ % $\hat{\mathbf{K}}$ is centered on a diagonal matrix

while(not converged)

$\mathbf{r} = [Rb - \mathbf{K} \tilde{x}]$ % $RA(x^* - \tilde{x}) \in \mathbf{r}$

$\mathbf{e} = \hat{\mathbf{K}} \setminus \mathbf{r}$ % cost: 1 floating-point matrix-vector product

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$, $\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

end

Output: $\mathbf{x} = \tilde{x} + \mathbf{e}$

Relaxed method: details

The product of \mathbf{A} centered in zero by \mathbf{B} is $[-\bar{A} * \text{mag}\mathbf{B}, \bar{A} * \text{mag}\mathbf{B}]$,
i.e. 1 FP matrix product.

Let us decompose \mathbf{A} as $\mathbf{D} + \mathbf{L} + \mathbf{U}$. Then

$$\text{Jacobi:} \quad \mathbf{e}' = \mathbf{D}^{-1}(\mathbf{b} - (\mathbf{L} + \mathbf{U})\mathbf{e})$$

$$\text{Gauss-Seidel:} \quad \mathbf{e}' = \mathbf{D}^{-1}(\mathbf{b} - \mathbf{L}\mathbf{e}' - \mathbf{U}\mathbf{e})$$

If \mathbf{L} and \mathbf{U} are inflated so as to be centered in 0:

$$\mathbf{L}' = [-|\mathbf{L}|, |\mathbf{L}|] \quad \text{and} \quad \mathbf{U}' = [-|\mathbf{U}|, |\mathbf{U}|]$$

then Jacobi or Gauss-Seidel costs 1 FP matrix-vector product.

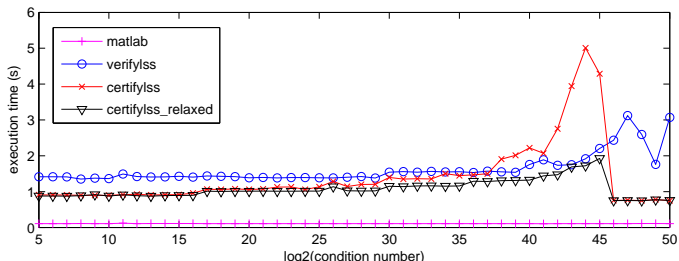
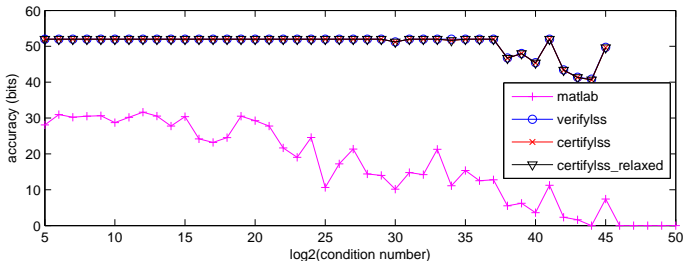
A BLAS2 routine can be used.

Convergence remains linear.

Accuracy of the solution: at most twice as wide as HBRNK.

Relaxed method: Results

Dimension: 1000 $b = [1, \dots, 1]^T$



Method: contractant iteration

Extra-precise relaxed interval iterative refinement

Algorithm (certifylssx)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$, $R = \text{inv}(A)$, $\mathbf{K} = [RA]$,

$\hat{\mathbf{K}} = \text{inflated}(\mathbf{K})$ % $\hat{\mathbf{K}}$ is centered on a diagonal matrix

while(not converged)

$\mathbf{r} = [Rb - \mathbf{K} \tilde{x}]$

$\mathbf{e} = \hat{\mathbf{K}} \setminus \mathbf{r}$

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$

$\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

end

Output: $\mathbf{x} = \tilde{x} + \mathbf{e}$

Method: contractant iteration

Extra-precise relaxed interval iterative refinement

Algorithm (certifylssx)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$, $R = \text{inv}(A)$, $\mathbf{K} = [RA]$,

$\hat{\mathbf{K}} = \text{inflated}(\mathbf{K})$ *% $\hat{\mathbf{K}}$ is centered on a diagonal matrix*

while(not converged)

$\mathbf{r} = [Rb - \mathbf{K} \tilde{x}]$ *% \mathbf{r} in twice the working precision*

$\mathbf{e} = \hat{\mathbf{K}} \setminus \mathbf{r}$

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$

$\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

end

Output: $\mathbf{x} = \tilde{x} + \mathbf{e}$

Method: contractant iteration

Extra-precise relaxed interval iterative refinement

Algorithm (certifylssx)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$, $R = \text{inv}(A)$, $\mathbf{K} = [RA]$,

$\hat{\mathbf{K}} = \text{inflated}(\mathbf{K})$ % $\hat{\mathbf{K}}$ is centered on a diagonal matrix

while(not converged)

$\mathbf{r} = [Rb - \mathbf{K} \tilde{x}]$ % \mathbf{r} in twice the working precision

$\mathbf{e} = \hat{\mathbf{K}} \setminus \mathbf{r}$

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$ % \tilde{x} in twice the working precision

$\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

end

Output: $\mathbf{x} = \tilde{x} + \mathbf{e}$

Method: contractant iteration

Extra-precise relaxed interval iterative refinement

Algorithm (certifylssx)

Input: $A \in \mathbb{F}^{n \times n}$, $b \in \mathbb{F}^n$

$\tilde{x} = A \setminus b$, $R = \text{inv}(A)$, $\mathbf{K} = [RA]$,

$\hat{\mathbf{K}} = \text{inflated}(\mathbf{K})$ *% $\hat{\mathbf{K}}$ is centered on a diagonal matrix*

while(not converged)

$\mathbf{r} = [Rb - \mathbf{K} \tilde{x}]$ *% \mathbf{r} in twice the working precision*

$\mathbf{e} = \hat{\mathbf{K}} \setminus \mathbf{r}$

$\tilde{x} = \tilde{x} + \text{mid}(\mathbf{e})$ *% \tilde{x} in twice the working precision*

$\mathbf{e} = \mathbf{e} - \text{mid}(\mathbf{e})$

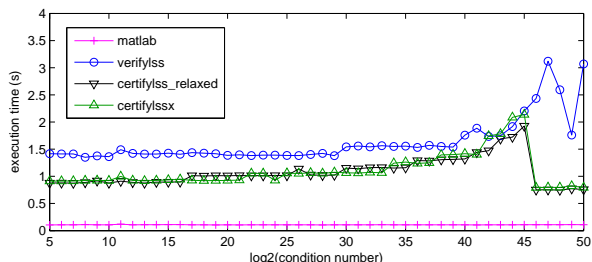
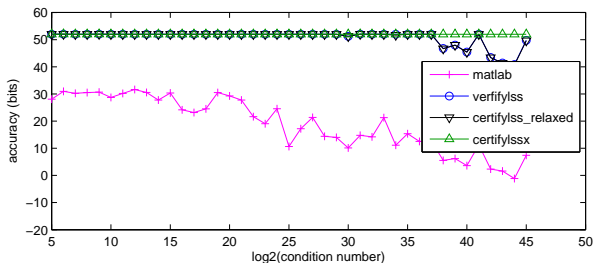
end

Output: $\mathbf{x} = \tilde{x} + \mathbf{e}$

Implementation: careful tuning of the precision of each variable
(no doubling for \mathbf{e} : useless and costly).

Extra-precise relaxed method: Results

Dimension: 1000 $b = [1, \dots, 1]^T$



Agenda

Verified linear system solving

Nguyen's algorithm

Recap

Recap of the chapter on interval arithmetic

Outline of the chapter

- ▶ historical and panoramic viewpoint:
including algorithms, complexity result on the difficulty to evaluate a polynomial exactly
- ▶ linear algebra:
building block = matrix multiplication
discussion of two representations: inf-sup versus mid-rad
goals = accuracy **and** efficiency
- ▶ linear algebra:
building block = linear system solving
complexity: NP-hard
algorithms (Rump, Nguyen) built by assembling basic blocks (ϵ -inflation, Krawczyk, Gauss-Seidel...)
- ▶ main use (here) for verification of floating-point computations

Philosophical conclusion

Morale

- ▶ keep your goals in mind (accuracy, efficiency)
- ▶ reuse optimized blocks (BLAS3)
- ▶ build algorithms by assembling building blocks
- ▶ interval arithmetic can be a tool for verification purposes

verifylss by S. Rump

- ▶ **1st step: approximate solving of the system** $A_c x = b_c$:


```

R = inv(Ac),    x = R * bc
while (not enough) loop    % iterative refinement
    res = bc - Ac * x      % residual computation
    e = R * res,    x = x + e
            
```
- ▶ **2nd step: verification = enclosure of the error**

1st attempt: ε -inflation and Krawczyk iteration

```

[z] = R * ([b]-[A]*x)
while (not enough) loop
    [y] = epsilon-inflate([z])
    [y'] = [z] + (I-R*[A])*[y]
    if [y'] in [y] then return x, [y']
            
```

2nd attempt: Hansen-Blik-Rohn-Ning-Kearfott formula

```

[b] = [b] - [A]*(x+[y]),    [A] = R*[A]
...
            
```

Comments on `verifylss`

- ▶ Complexity: polynomial in time ... for an NP-hard problem: no guarantee on the accuracy of the solution, failure is possible (either ε -inflation or Hansen-Bliek-Rohn-Ning-Kearfott if **A** is not an H-matrix).
- ▶ This algorithm is usually employed to verify the solution of a linear system with floating-point coefficients: interval arithmetic is used as a verification tool.

Comments on `verifylss`

- ▶ stopping criterion for the first loop (iterative refinement on the linear system formed by midpoints of **A** and **b**):
 - ▶ number of iterations < 15 , and
 - ▶ norm of new residual < 0.1 times norm of former residual.If this last criterion is not satisfied, the 1st step can also fail.
- ▶ stopping criterion for the second loop (determination of an enclosure of the error):
 - ▶ number of iterations < 7 , and
 - ▶ Brouwer theorem did not apply and thus it is not proven that $[y]$ contains the error.