Computing Guaranteed Error Bounds for Solutions of Matrix Equations

Andreas Frommer
Behnam Hashemi

Bergische Universität Wuppertal
Amirkabir University of Technology, Tehran

May 25, 2010

Contents

Basics of interval arithmetic

$\mathbf{a}, \mathbf{b}$ compact intervals $\subset \mathbb{R}$, $\circ \in \{+, -, *, /\}$

Theoretical IA:

$$\mathbf{a} \circ \mathbf{b} = \mathbf{c} = \{a \circ b : a \in \mathbf{a}, b \in \mathbf{b}\}$$

Note:

- bounds of $\mathbf{c}$ computable from bounds of $\mathbf{a}$ and $\mathbf{b}$
- bounds of $\mathbf{c}$ not necessarily representable on computer!

Machine IA: Ask for

- $\mathbf{a} \circ \mathbf{b} = \mathbf{c} \supseteq \{a \circ b : a \in \mathbf{a}, b \in \mathbf{b}\}$
- bounds of $\mathbf{c}$ representable, close to 'true' bounds

Expressions

Let $r(x_1, \ldots, x_n) = r(x)$ be a rational expression.

Interval arithmetic evaluation of r:

$$r(\mathbf{x}) = r(\mathbf{x}_1, \ldots, \mathbf{x}_n\}$$

Enclosure property of (machine) IA:

$$r(\mathbf{x}) \supseteq \{r(x) : x \in \mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)\}$$

Caveat: r_1, r_2 different expressions for same function. Then, in general, $r_1(\mathbf{x}) \neq r_2(\mathbf{x})$.

Intlab

- Machine IA can be implemented in IEEE 754 arithmetic using directed roundings
- Intlab: Does so, but allows for 'larger' intervals if this speeds up computation
- Intlab: Matlab toolbox by S. M. Rump
- **Intlab demo** ...

Proofs using (machine) IA

$r = (r_1, \ldots, r_n) : D \subseteq \mathbb{C}^n \to \mathbb{C}^n$, each r_i a rational expression.

Theorem:

a) $0 \notin r(\mathbf{x})$
$\Rightarrow$ the function represented by r has no zero in $\mathbf{x}$.

b) $r(\mathbf{x}) \subseteq \mathbf{x}$
$\Rightarrow$ the function represented by r has a fixed point $x^* \in \mathbf{x}$

Application to linear system $Ax = b$:

- $\check{x}$ approximate solution
- R approximation to A^{-1}
- find contracting fixed point operator for error z, i.e. $A(\check{x} + z) = b$

Krawczyk's method for z in $A(\check{x} + z) = b$

$$\begin{aligned} g(z) &= R(b - A\check{x}) + (I - RA)z \\ \mathbf{g}(\mathbf{z}) &= R(b - A\check{x}) + (I - RA)\mathbf{z} \text{ IA evaluation} \end{aligned}$$

Theorem: [Krawczyk 69, Rump 83] If

$$\mathbf{g}(\mathbf{z}) \subseteq \text{int}\,(\mathbf{z})$$

then A and R are non-singular and there exists $z^* \in \mathbf{z}$ such that $A(\check{x} + z^*) = b$.

Krawczyk's method for z in $A(\check{x} + z) = b$

$$\begin{aligned} g(z) &= R(b - A\check{x}) + (I - RA)z \\ \mathbf{g}(\mathbf{z}) &= R(b - A\check{x}) + (I - RA)\mathbf{z} \text{ IA evaluation} \end{aligned}$$

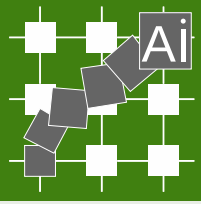
Theorem: [Krawczyk 69, Rump 83] If

$$\mathbf{g}(\mathbf{z}) \subseteq \text{int}\,(\mathbf{z})$$

then A and R are non-singular and there exists $z^* \in \mathbf{z}$ such that $A(\check{x} + z^*) = b$.

Remarks:

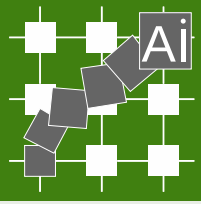
- *Slight extension*: Theorem still holds if $\text{range}(g, \mathbf{z}) \subseteq \text{int}\,\mathbf{z}$.
- $\mathbf{z}$ that works: centered around 0 with radius $\geq |R(b - A\check{x})|$
- Evaluate *everything* in machine IA



Sylvester equation

Sylvester (matrix) equation:

$$AX + XB = C, \qquad A \in \mathbb{C}^{m\times m}, B \in \mathbb{C}^{n\times n}, C \in \mathbb{C}^{m\times n}$$



Sylvester equation

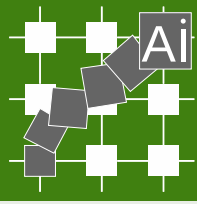
Sylvester (matrix) equation:

$$AX + XB = C, \qquad A \in \mathbb{C}^{m\times m}, B \in \mathbb{C}^{n\times n}, C \in \mathbb{C}^{m\times n}$$

Notation: For any matrix Y :

$$y := \mathrm{vec}(Y) = (y_{11}, \ldots, y_{k1}, y_{12}, \ldots, y_{k2}, \ldots, y_{1\ell}, \ldots, y_{k\ell})^T.$$

Simplifying: ,,lowercase $=$ vec(uppercase)"



Sylvester equation

Sylvester (matrix) equation:

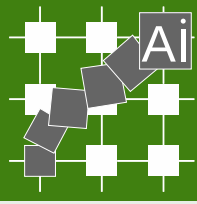
$$AX + XB = C, \qquad A \in \mathbb{C}^{m\times m}, B \in \mathbb{C}^{n\times n}, C \in \mathbb{C}^{m\times n}$$

Notation: For any matrix Y :

$$y := \mathrm{vec}(Y) = (y_{11}, \ldots, y_{k1}, y_{12}, \ldots, y_{k2}, \ldots, y_{1\ell}, \ldots, y_{k\ell})^T.$$

Simplifying: ,,lowercase = vec(uppercase)"

$$AX + XB = C \Longleftrightarrow \underbrace{(I_n \otimes A + B^T \otimes I_m)}_{\mathcal{P}} x = c.$$



Sylvester equation

Sylvester (matrix) equation:

$$AX + XB = C, \qquad A \in \mathbb{C}^{m\times m}, B \in \mathbb{C}^{n\times n}, C \in \mathbb{C}^{m\times n}$$

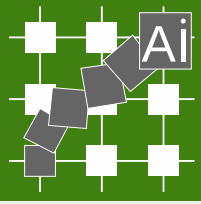
Notation: For any matrix Y :

$$y := \mathrm{vec}(Y) = (y_{11}, \ldots, y_{k1}, y_{12}, \ldots, y_{k2}, \ldots, y_{1\ell}, \ldots, y_{k\ell})^T.$$

Simplifying: ,,lowercase = vec(uppercase)"

$$AX + XB = C \Longleftrightarrow \underbrace{(I_n \otimes A + B^T \otimes I_m)}_{\mathcal{P}} x = c.$$

vec-of-three-factors formula: $\mathrm{vec}(RST) = (T^T \otimes R)s.$



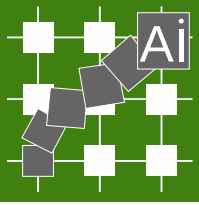
Computational effort for standard Krawczyk

Situation:

- $\check{X}$ approximate solution, $\check{x} = \mathrm{vec}(\check{X})$.
- $\mathbf{g}(\mathbf{z}) = \mathcal{R}(c - \mathcal{P}\check{x}) + (I - \mathcal{R}\mathcal{P})(\mathbf{z})$.

Krawczyk:

1. needs $\mathcal{R} \approx \mathcal{P}^{-1} = (I_n \otimes A + B^T \otimes I_m)^{-1} \in \mathbb{C}^{nm \times nm}$
2. computes $(I - \mathcal{R}\mathcal{P})$
3. R is full; cost for 2. is $\mathcal{O}((nm)^2(n+m))$ *"quintic"*



Challenge: Reduce cost to *cubic*.

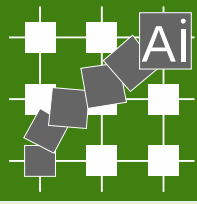
Idea: Allow for $\mathcal{R}$ *diagonal* via affine transformation.

For simplicity: A, B diagonizable

$$\begin{aligned} A &= V_A D_A W_A, \\ &\quad V_A, W_A, D_A \in \mathbb{C}^{m\times m}, D_A = \mathrm{diag}(\lambda_1, \ldots, \lambda_m), V_A \cdot W_A = I_m, \\ B &= V_B D_B W_B, \\ &\quad V_B, W_B, D_B \in \mathbb{C}^{n\times n}, D_B = \mathrm{diag}(\mu_1, \ldots, \mu_n), V_B \cdot W_B = I_n. \end{aligned}$$

Then

$$\mathcal{P} = (V_B^{-T} \otimes W_A^{-1})[I_n \otimes (W_A A W_A^{-1}) + (V_B^{-1} B V_B)^T \otimes I_m](V_B^T \otimes W_A).$$



Affine transformation

Define:

$$\begin{aligned} \mathcal{Q} &= I_n \otimes (W_A A W_A^{-1}) + (V_B^{-1} B V_B)^T \otimes I_m \\ y &= (V_B^T \otimes W_A) x \\ f &= (V_B^T \otimes W_A) c \end{aligned}$$

Affinely transformed Sylvester equation

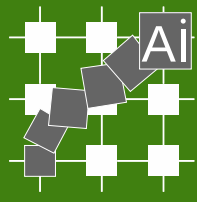
$$\mathcal{Q} y = f.$$

Important: We consider W_A, V_A, D_A as *computed* matrices, so

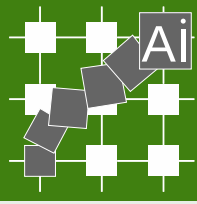
$$W_A A W_A^{-1} \approx D_A \quad \text{etc.}$$

Consequence: $\mathcal{Q} \approx I_n \otimes D_A + D_B^T \otimes I_m =: \Delta$, diagonal

Extremely **important:** Δ^{-1} can be used as $\mathcal{R}$



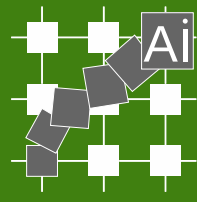
1: Put $D \in \mathbb{C}^{m \times n}$ s.t. column d_j of D is $\operatorname{diag}(D_A) + (D_B)_{jj}(1, \ldots, 1)^T$
2: Compute interval matrices $\mathbf{I}_{W_A}$ and $\mathbf{I}_{V_B}$ containing W_A^{-1} and V_B^{-1}, resp.
3: Compute $\mathbf{R} = W_A \cdot (A\tilde{X} + \tilde{X}B - C) \cdot V_B$ using interval arithmetic
4: Compute $\mathbf{S}_A = (W_A A)\mathbf{I}_{W_A}$ and $\mathbf{S}_B = \mathbf{I}_{V_B}(BV_B)$
5: Compute $\mathbf{U} = -\mathbf{R}./D$, put $k = 0$ {Prepare loop}
6: **repeat**
7: Put $\mathbf{Z} = \square(0, \mathbf{U} \cdot [1 - \varepsilon, 1 + \varepsilon])$, increment k {ε-inflation}
8: **for** $j = 1, \ldots, n$ **do**
9: Compute $\mathbf{m}_j = (\operatorname{Diag}(d_j) - \mathbf{S}_A - (\mathbf{S}_B)_{jj} I_m)\mathbf{z}_j$
10: **end for**
11: Compute $\mathbf{N} = -\mathbf{Z}\mathbf{E}_B$ {$\mathbf{E}_B$ is off-diagonal part of $\mathbf{S}_B$}
12: Compute $\mathbf{U} = (-\mathbf{R} + \mathbf{M} + \mathbf{N})./D$
13: **until** ($\mathbf{U} \subseteq \operatorname{int} \mathbf{Z}$ or $k = 15$)
14: **if** $\mathbf{U} \subseteq \operatorname{int} \mathbf{Z}$ **then** {successful termination}
15: output $\mathbf{V} = \mathbf{I}_{W_A}\mathbf{U}\mathbf{I}_{V_B}$
16: **end if**



Discussion of the algorithm

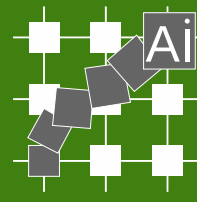
Remarks:

- Algorithm uses matrix-matrix operations whenever possible
  $\longrightarrow$ efficient in INTLAB
- Computational cost: $\mathcal{O}(mn^2 + nm^2)$ *cubic*
  $\longrightarrow$ comparable to cost for getting $\check{X}$
- Enclosing $W_A \cdot (A\tilde{X} + \tilde{X}B - C) \cdot V_B$ accurately is crucial
- Algorithm will not succeed if V_A or V_B are ill-conditioned
- In this case: block-diagonalization à la Bavely-Stewart
  (`bdiag` in Matlab's Optimization and Control Toolbox)



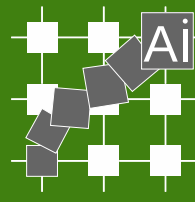
Well conditioned eqs. constructed as in Benner, Sima, Slowik

n	Versoft		Algorithm 1 double prec. res.			Algorithm 1 quad. prec. res.		
	time	mr mrr	time	k	mr mrr	time	k	mr mrr
50	663.1	$9.8 \cdot 10^{-14}$ $3.7 \cdot 10^{-9}$	0.4	1	$3.2 \cdot 10^{-12}$ $3.0 \cdot 10^{-8}$	0.7	1	$1.2 \cdot 10^{-26}$ $1.1 \cdot 10^{-16}$
75	-	- -	0.8	1	$1.1 \cdot 10^{-11}$ $2.2 \cdot 10^{-8}$	1.2	1	$3.6 \cdot 10^{-26}$ $1.1 \cdot 10^{-16}$
100	-	- -	1.4	1	$2.5 \cdot 10^{-11}$ $5.4 \cdot 10^{-7}$	2.2	1	$9.8 \cdot 10^{-26}$ $1.1 \cdot 10^{-16}$
300	-	- -	30.5	1	$1.4 \cdot 10^{-9}$ $3.0 \cdot 10^{-3}$	59.8	1	$7.0 \cdot 10^{-24}$ $1.1 \cdot 10^{-16}$
500	-	- -	149.9	1	$5.6 \cdot 10^{-8}$ $1.4 \cdot 10^{-1}$	298.8	1	$1.6 \cdot 10^{-21}$ $9.3 \cdot 10^{-16}$



Lyapunov eqs from CTDSX benchmark collection (Kressner, Mehrmann, Penzl)

Ex. no. / n	sep	Versoft time	Versoft mr / mrr	double prec. res. time	double prec. res. mr / mrr	quad. prec. res. time	quad. prec. res. mr / mrr
1.3 4	$1.9 \cdot 10^{-2}$	0.0	$1.5 \cdot 10^{-14}$ 1.0	0.1	$4.5 \cdot 10^{-14}$ 1.0	0.1	$6.9 \cdot 10^{-29}$ $1.6 \cdot 10^{-13}$
1.10 8	$2.2 \cdot 10^{-2}$	0.0	$1.4 \cdot 10^{-7}$ 1.0	0.2	$1.3 \cdot 10^{-7}$ 1.0	0.2	$3.0 \cdot 10^{-21}$ $1.3 \cdot 10^{-13}$
1.8 9	$1.8 \cdot 10^{-13}$	0.1	$2.0 \cdot 10^{-7}$ $1.6 \cdot 10^{-11}$	0.2	$8.3 \cdot 10^{-7}$ $3.0 \cdot 10^{-12}$	0.2	$1.9 \cdot 10^{-20}$ $1.1 \cdot 10^{-16}$
1.6 30	$6.1 \cdot 10^{-6}$	11.5	$2.6 \cdot 10^{-8}$ 1.0	0.4	$5.8 \cdot 10^{-7}$ 1.0	0.5	$4.3 \cdot 10^{-21}$ $2.2 \cdot 10^{-12}$
3.2 40	$1.2 \cdot 10^{-1}$	44.1	$1.1 \cdot 10^{-14}$ $8.1 \cdot 10^{-13}$	0.4	$6.0 \cdot 10^{-14}$ $7.1 \cdot 10^{-12}$	0.6	$4.7 \cdot 10^{-28}$ $1.1 \cdot 10^{-16}$
1.9 55	$7.6 \cdot 10^{-9}$	956.8	$9.0 \cdot 10^{-7}$ 1.0	1.1	$2.4 \cdot 10^{-6}$ 1.0	1.4	$5.5 \cdot 10^{-21}$ 1.0
3.4 211	-	-	- -	237.4	$2.6 \cdot 10^{-12}$ 1.0	325.0	$2.3 \cdot 10^{-24}$ $9.8 \cdot 10^{-10}$

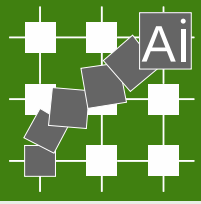


Ill-cond. Lyapunov eqs from CTLEX (Kressner, Mehrmann, Penzl)

Ex. no.	n			double prec. res.			quad. prec. res.		
		time	mr mrr	time	k	mr mrr	time	k	mr mrr
4.1	10	0.1	$3.8 \cdot 10^{-7}$ $6.1 \cdot 10^{-7}$	0.1	1	$5.1 \cdot 10^{-5}$ $1.3 \cdot 10^{-5}$	0.2	1	$2.3 \cdot 10^{-20}$ $1.1 \cdot 10^{-16}$
4.1	15	0.8	NaN NaN	0.2	1	NaN NaN	0.2	1	$4.7 \cdot 10^{-15}$ $9.0 \cdot 10^{-14}$
4.1	50	470.2	$1.4 \cdot 10^{-9}$ $6.0 \cdot 10^{-6}$	0.6	1	$2.0 \cdot 10^{-9}$ $3.2 \cdot 10^{-5}$	1.0	1	$2.4 \cdot 10^{-24}$ $1.1 \cdot 10^{-16}$
4.2	31	13.4	$4.4 \cdot 10^{-13}$ $2.8 \cdot 10^{-9}$	0.7	1	NaN NaN	1.0	1	NaN NaN
4.2	25	4.5	$1.5 \cdot 10^{-5}$ $1.0 \cdot 10^{-3}$	0.4	1	NaN NaN	0.4	1	NaN NaN
4.2	20	2.4	NaN NaN	0.3	1	NaN NaN	0.3	1	$8.7 \cdot 10^{-2}$ $8.7 \cdot 10^{-1}$

Conclusions I

- Verification method with cost $\mathcal{O}(mn(m+n))$
- Cost comparable to that for getting approximate solution
- Efficient implementation in INTLAB
- Block modifications for ill-conditioned problems possible ...
- ... but issues are: cost, implementation
- only dense matrices
- potential for proof of stability

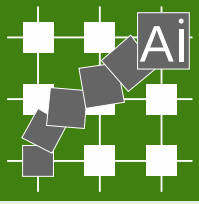


Definition

$$A \in \mathbb{C}^{n \times n}$$

Definition: An isolated solution X of $X^2 - A = 0$ is a *square root* of A.

Theorem [Gantmacher]:
Such a square root is one in the operator theoretic sense.



Krawczyk's method for nonlinear equations

Challenge: given a good *approximate* zero $\check{x}$ of $f : \mathbb{C}^n \to \mathbb{C}^n$

- prove existence of *"true"* zero x^* close to $\check{x}$
- provide correct (and narrow) enclosure for the error $\check{x} - x^*$

Krawczyk's method for nonlinear equations

Challenge: given a good *approximate* zero $\check{x}$ of $f : \mathbb{C}^n \to \mathbb{C}^n$

- prove existence of *"true"* zero x^* close to $\check{x}$
- provide correct (and narrow) enclosure for the error $\check{x} - x^*$

Generalization of approach for linear systems:

- turn into equivalent *fixed point* problem $x = g(x)$, where

$$g(x) = x - f'(\check{x})^{-1} f(x)$$

- from $\check{x}$ construct *box* $\mathbf{x} \subset \mathbb{C}^n$ (convex, compact)
- use interval arithmetic to computationally prove

$$\text{range}(g, \mathbf{x}) \subseteq \mathbf{x}$$

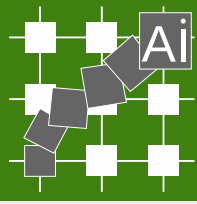
- by Brouwer's fixed point theorem: $x^* \in \mathbf{x}$
- *'contraction bonus'*: uniqueness of x^* in $\mathbf{x}$

Enclosing the range

Slopes: $f(x) = f(\check{x}) + A_f(x, \check{x})(x - \check{x}), \quad A_f \in \mathbb{C}^{n\times n}$ slope function for f

Consequence: Let $\mathbf{A} \supseteq \{A_f(x, \check{x}), x \in \mathbf{x}\}$. Then

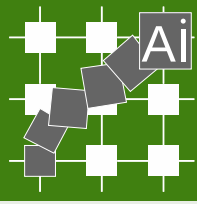
$$\text{range}(x - Rf(x), \mathbf{x}) \subseteq \underbrace{\{\check{x} - Rf(\check{x}) + (I - RA)(x - \check{x}) : A \in \mathbf{A}, x \in \mathbf{x}\}}_{\mathcal{K}_f(\check{x}, R, \mathbf{x}, \mathbf{A})}.$$



Existence of zeros

Theorem: If $\mathbf{A} \supseteq \{A_f(x,y), x, y \in \mathbf{x}\}$ and $\check{x} \in \mathbf{x}$ and $\mathcal{K}_f(\check{x}, R, \mathbf{x}, \mathbf{A}) \subset \operatorname{int} \mathbf{x}$, then

- R is non-singular
- f has a zero x^* in $\mathcal{K}_f$ which is unique in $\mathbf{x}$



Existence of zeros

Theorem: If $\mathbf{A} \supseteq \{A_f(x,y), x, y \in \mathbf{x}\}$ and $\check{x} \in \mathbf{x}$ and $\mathcal{K}_f(\check{x}, R, \mathbf{x}, \mathbf{A}) \subset \operatorname{int} \mathbf{x}$, then

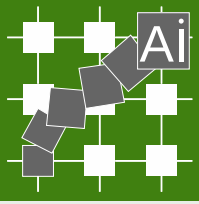
- R is non-singular
- f has a zero x^* in $\mathcal{K}_f$ which is unique in $\mathbf{x}$

Remarks:

- If f is real, the mean value theorem gives $A(x,y) \in \mathbf{f}'(\mathbf{x})$ for $x, y, \in \mathbf{x}$.
- Krawczyk '69, Rump '83: For f real replace $\mathcal{K}_f(\check{x}, R, \mathbf{x}, \mathbf{A})$ by

$$\check{x} - Rf(\check{x}) + (I - R\mathbf{f}'(\mathbf{x}))(\mathbf{x} - \check{x}) \supseteq \mathcal{K}_f(\check{x}, R, \mathbf{x}, \mathbf{f}'(\mathbf{x}))$$

- $\mathbf{x}$ that works: centered around $\check{x}$ with radius $\geq |Rf(\check{x})|$



f and f' for matrix square root

For matrix square root:

$$F(X) := X^2 - A = 0 \quad \Leftrightarrow \quad f(x) = 0,$$

where

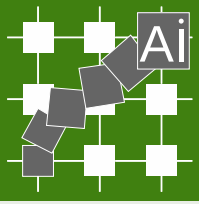
$$f : \mathbb{C}^{n^2} \to \mathbb{C}^{n^2}, \quad x \mapsto (I \otimes X)x - a,$$
$$f'(x) = (I \otimes X + X^T \otimes I) \in \mathbb{C}^{n^2 \times n^2}.$$

Back to our old problem ...

Standard Krawczyk evaluates $\check{x} - Rf(\check{x}) + (I - R\mathbf{f}'(\mathbf{x}))(\mathbf{x} - \check{x})$:

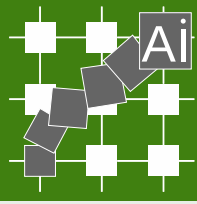
- $\mathbf{f}'(\mathbf{x}) = I \otimes \mathbf{X} + \mathbf{X}^T \otimes I$
- $R = [I \otimes \check{X}) + (\check{X}^T \otimes I)]^{-1}$
  cost is $\mathcal{O}(n^4)$ if you are clever
- $I - R\mathbf{f}'(\mathbf{x})$: cost is $\mathcal{O}(n^5)$!!

As in the linear case: Reduce cost to $\mathcal{O}(n^3)$.



The idea

- diagonalization of $\check{X}$ induces affine transformation $\hat{f}$ of f
- since $\hat{f}'(\hat{\check{x}})$ is (almost) diagonal, R can be chosen as diagonal
- Reasonable superset $\hat{\mathbf{k}}$ of $\mathcal{K}_{\hat{f}}(\hat{\check{x}}, R, \hat{\mathbf{x}})$ computable at cost $\mathcal{O}(n^3)$
- **Theorem shows:** If $\hat{\mathbf{k}} \subset \operatorname{int} \hat{\mathbf{x}}$, then $\hat{f}$ has an isolated zero $\hat{x}^* \in \hat{k}$.
  So there is an isolated square root X^* of A in $\mathbf{I}_W \hat{\mathbf{X}} \mathbf{I}_V$.
- Block diagonalization of $\check{X}$ if V, W are ill-conditioned.
- `bdiag` from the MATLAB optimal control toolbox.



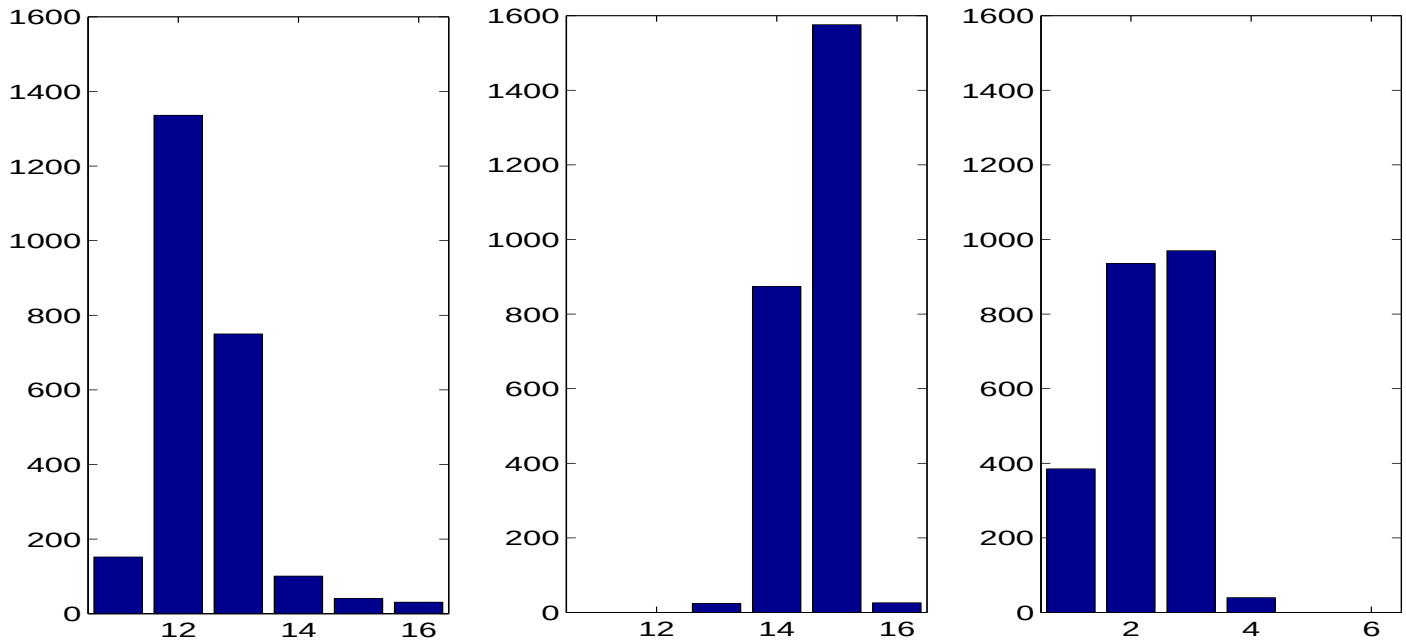
Numerical results

- matrices from MATLAB, Matrix Market and literature
- dimensions from 5 to 2 000
- (unfair) comparison with VERSOFT (Rohn)
- On 2.5 GHz Pentium IV Laptop with 1 GByte main memory
- Higham/Smith's `rootm.m` for $\check{X}$

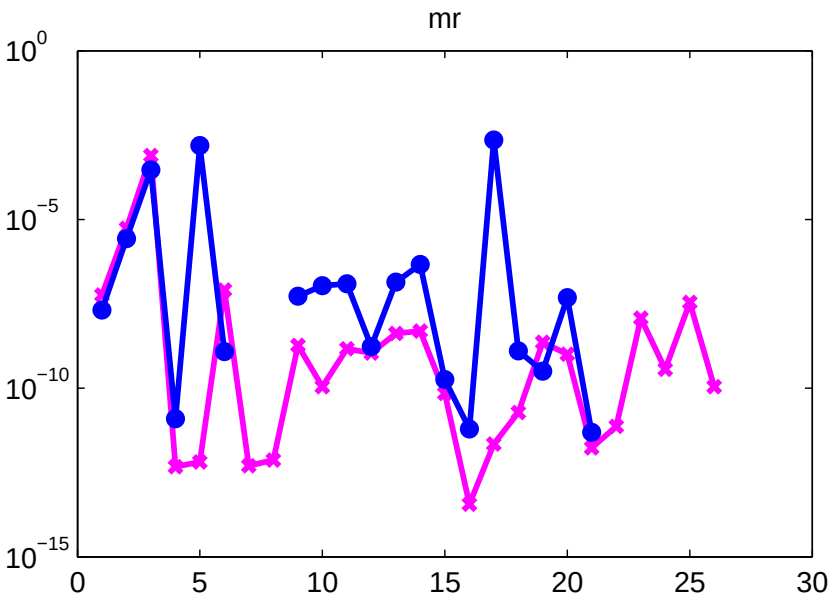
Correct digits

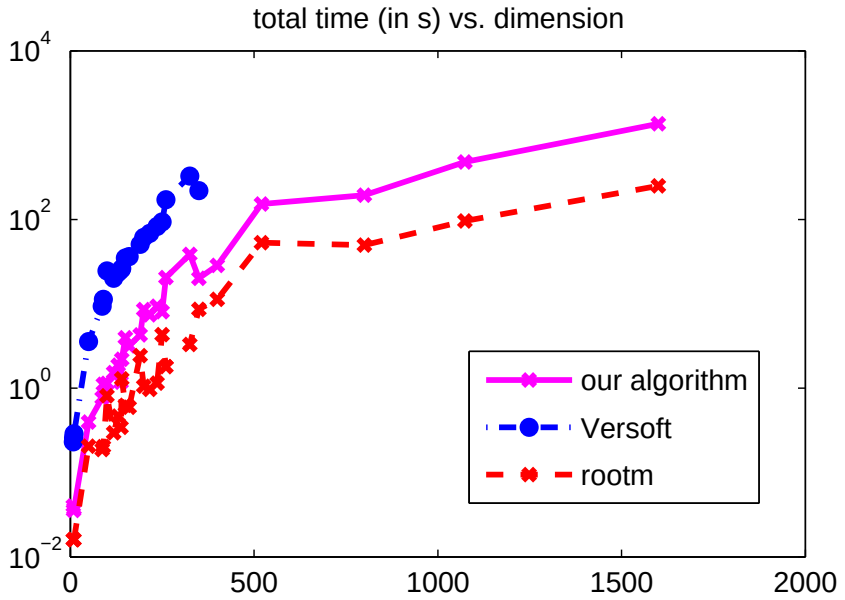
Matrix $A \in \mathbb{R}^{50\times 50}$, $a_{ij} = 0.1$ for $i \neq j$, $a_{ii} = i^2$, $i,j = 1,\ldots,50$
(from Higham's book)

in $\check{X}$ from interval matrix improvement



Radii and timings





Conclusions II

- slight generalization of theory for Krawczyk's method allows the use of factorization for $f'(\check{x})$
- cost is $\mathcal{O}(n^3)$
- fast in INTLAB
- needs (block) diagonalization of $\check{X}$
- works for multiple eigenvalues
- comparable accuracy as VERSOFT
- . . . but much faster
- poisson(50) from MATLAB: nonlinear system with 6.25 Mio of variables. Verification in 15 minutes, accuracy 10^{-11}.