Interval arithmetic methods to verify the stabilizing solution of an algebraic Riccati equation

Tayyebe Haqiri[1] Federico Poloni[2]

[1]Shahid Bahonar University of Kerman, Iran
[2]U Pisa, Italy, Dept of Computer Science

20th ILAS conference
Leuven, July 2016

Overview

Goal: compute a set $\boldsymbol{X}$ which contains (for sure, not "up to small computational errors") the stabilizing solution X_s of

$$0 = F(X) = A^\top X + XA + Q - XGX.$$

Do not use more than $O(n^3)$ flops.

Plan

- Convince you that interval arithmetic is a good idea.
- Show you what people did to verify Riccati equations.
- Show you the improvements we introduced.
- Competitors, experiments, and other ideas.

Interval arithmetic [Rump, '10 review]

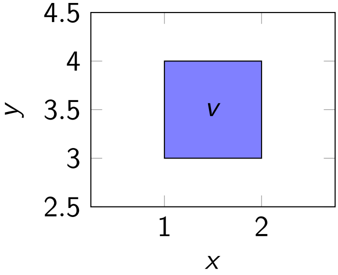
Basic idea if $a \in [1,2]$ and $b \in [3,4]$, then $a+b \in [4,6]$ and $ab \in [3,8]$. Store (min, max) (or $(center, radius)$) and operate on them. $\mathbb{IR}, \mathbb{IC}$.

With IEEE arithmetic + rounding in the correct direction, the inclusions work irrespective of machine errors.

Machine numbers can be embedded in $\mathbb{IR}$ as radius-0 intervals.

Wrapping effect

$$\begin{bmatrix} x \\ y \end{bmatrix} \mapsto \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \qquad v = \begin{bmatrix} [1,2] \\ [3,4] \end{bmatrix}$$



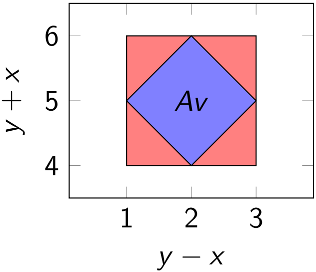


Image of v: blue square. Interval result: red+blue.
This happens even though $\kappa(A) = 1$!

Verify, don't solve

The first rule of interval arithmetic

You don't solve your problem with interval arithmetic.

Things like back-substitution would create huge intervals.
E.g., solving $AX + XB = C$ with Bartels-Stewart: hopeless.

Instead:

$$g(\boldsymbol{x}) \subset \boldsymbol{x} \text{ implies } x = g(x) \text{ for some } x \in \boldsymbol{x}$$

- compute (with usual methods) an approximate solution $\tilde{x}$.
- reformulate as $x = g(x)$, e.g., $x = x - Rf(x)$.
- choose an interval $\boldsymbol{x} \ni \tilde{x}$, e.g., $\tilde{x} - [0.9, 1.1]f(\tilde{x})$
- check (hopefully) $g(\boldsymbol{x}) \subset \boldsymbol{x}$.
- if not, enlarge $\boldsymbol{x}$, e.g., $\boldsymbol{x} \leftarrow [0.9, 1.1]g(\boldsymbol{x})$ and retry.

Details omitted; e.g.: need care with computing $\boldsymbol{x} - Rf(\boldsymbol{x})$.

The Krawczyk method

Ingredients:

- approximate solution $\tilde{x}$.
- slope $\boldsymbol{S_x}$: set such that there is $S \in \boldsymbol{S_x}$ satisfying

$$f(x) - f(\tilde{x}) = S(x - \tilde{x}) \quad \text{for all } x \in \boldsymbol{x}. \qquad (*)$$

  Often related to an interval evaluation of $f'(\boldsymbol{x})$.
- preconditioner R: approximate inverse of some matrix in $\boldsymbol{S_x}$.

Theorem [Krawczyk '69, Rump '83]

If, for some interval $\boldsymbol{\delta}$,

$$\operatorname{int}(\boldsymbol{\delta}) \supseteq -Rf(\tilde{x}) + (I - R\boldsymbol{S}_{\tilde{x}+\boldsymbol{\delta}})\boldsymbol{\delta},$$

then $\tilde{x} + \boldsymbol{\delta}$ contains a solution of $f(x) = 0$.
If (*) holds replacing $\tilde{x}$ with every $y \in \boldsymbol{x}$, then the solution is unique.

Verifying Riccati equations

$$F(X) = A^\top X + XA + Q - XGX$$

$O(n^3)$ algorithm: [Hashemi '12]

- $\tilde{X}$ from your favorite method.
- $\boldsymbol{S_x}$: $F'(\boldsymbol{X}) = (A - G\boldsymbol{X})^\top \otimes I + I \otimes (A - G\boldsymbol{X})^\top$ works.
- R: can't use Bartels-Stewart. Instead: explicit eigendecomposition $(A - GX) \approx VDV^{-1}$ and

$$R = (V^{-\top} \otimes V^{-\top})(D^\top \otimes I + I \otimes D^\top)^{-1}(V^\top \otimes V^\top)$$

Additional manipulations: $(I - R\boldsymbol{S_X}) = (V^{-\top} \otimes V^{-\top})(\cdots)(V^\top \otimes V^\top)$

Again, many details omitted; for instance, dealing properly with $W \approx V^{-1}$.

Improving Hashemi's method

Our goal: construct an enclosure $\boldsymbol{X}$ for the stabilizing solution X_s.

Plan:

- Compute an enclosure $\boldsymbol{X}$ starting from $\tilde{X} \approx X_s$.
- Verify that each matrix in $A - G\boldsymbol{X}$ is stabilizing.
- Uniqueness follows from classical Riccati theory. [Brockett, '70]

Letting go of uniqueness allows some improvements:

1. Tighter slope $\boldsymbol{S_X}$.
2. Defer the change of basis as in [Frommer Hashemi '09].
3. Verify a different equation using tricks from [Mehrmann P. '12].

Improvements

1. Tighter slope $\boldsymbol{S_X}$
   We can use $\boldsymbol{S_x} = (A - G\boldsymbol{X})^\top \otimes I + I \otimes (A - G\tilde{X})^\top$.

2. Defer the change of basis

   Find $\boldsymbol{Y}$ that encloses a solution of $\hat{F}(Y) = V^\top f(V^{-\top} Y V^{-1}) V$:
   Easier, because $\hat{F}'$ is diagonal.

   Then, compute $\boldsymbol{X} = V^{-\top} \boldsymbol{Y} V^{-1}$.
   Even if $Y \in \boldsymbol{Y}$ unique solution, other solutions may end up in $\boldsymbol{X}$ due to wrapping effects.

   [Frommer Hashemi '09] introduced this trick for `sqrtm`.

Improvements

3. Verify a different equation

$$CARE \iff \begin{bmatrix} A & -G \\ -Q & -A^\top \end{bmatrix} \begin{bmatrix} I \\ X \end{bmatrix} = \begin{bmatrix} I \\ X \end{bmatrix} (A - GX)$$

[Mehrmann P. '12]: one can find a basis for $\operatorname{im}\left[\begin{smallmatrix} I \\ X \end{smallmatrix}\right]$ with an identity in different position (i.e., $\operatorname{im}\left[\begin{smallmatrix} I \\ X \end{smallmatrix}\right] = \operatorname{im} \Pi \left[\begin{smallmatrix} I \\ Y \end{smallmatrix}\right]$, Π permutation matrix) so that $|Y|_{ij} \leq \sqrt{2}$.

As above, we can verify a Riccati equation for Y rather than one for X.

Smaller / more balanced entries $\implies$ easier verification.

Verify a different equation

Algorithm

- Compute approximate CARE solution $\tilde{X}$
- Compute Π so that $\operatorname{im}\begin{bmatrix} I \\ \tilde{X} \end{bmatrix} = \operatorname{im} \Pi \begin{bmatrix} I \\ \tilde{Y} \end{bmatrix}$, with $\tilde{Y}$ bounded.
- Form the CARE associated with $\Pi^{-1} \begin{bmatrix} A & -G \\ -Q & -A^\top \end{bmatrix} \Pi$ instead of $\begin{bmatrix} A & -G \\ -Q & -A^\top \end{bmatrix}$.
- Compute an inclusion $\boldsymbol{Y} \supseteq Y_s$ of its stable solution.
- $\boldsymbol{X} = \boldsymbol{U}_2 \boldsymbol{U}_1^{-1}$, where $\Pi \begin{bmatrix} I \\ \boldsymbol{Y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{U}_1 \\ \boldsymbol{U}_2 \end{bmatrix}$. Other solutions may enter $\boldsymbol{X}$.

Summing up

- Start from an approximate stabilizing solution $\tilde{X}$
- Use the above methods to construct $\boldsymbol{X} \ni \tilde{X}$ containing a solution
- If all the matrices in $\boldsymbol{X}$ are stabilizing, bingo!

Alternative approach (main competitor): [Miyajima '15].
Mix between the above methods and explicit normwise bounds. Idea:

- Newton-like iteration $X = g(X)$, $g(X) = X - (F'_{\tilde{X}})^{-1}(F(X))$.
- Formula for $F'_{\tilde{X}}$ using an eigendecomposition of $A - G\tilde{X}$, as earlier.
- Expand $g(\boldsymbol{X})$, where $\boldsymbol{X} = (\tilde{X} - \eta R, \tilde{X} + \eta R)$ (for a specific choice of R), as a function of η.
- Using inequalities, determine η such $\boldsymbol{X} \supseteq g(\boldsymbol{X})$ (if possible).
- Compute η using interval arithmetic and rounding.
- Uniqueness and stabilizing-ness verified *a posteriori*.

Diagonalizability

Verification methods tested on the benchmarks in CAREX [Benner et al '95]

OK on many of them, but we are still not satisfied:

CAREX Example 1 [Benner et al '95]

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad G = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad Q = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}, \quad X_s = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

This example can be used for a first verification of any solver for [CAREs] since the solution may be computed by hand.

$A - GX_s$ is not diagonalizable $\implies$ all methods fail on this 'warm-up' example.

Non-diagonalizable problems

New algorithm: not as effective as the others, but it works in $O(n^3)$ even if $A - GX_s$ is (almost) not diagonalizable.

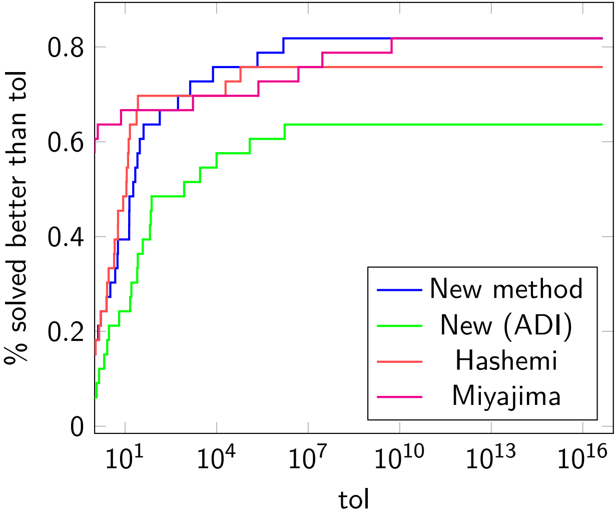
Idea

- Rewrite as a CARE in Δ, where $X = \tilde{X} + \Delta$: $\hat{A}^*\Delta + \Delta\hat{A} + \hat{Q} - \Delta G \Delta = 0$.
- Mimic ADI: fixed-point eqn $\Delta = (\hat{A} - sI)^{-\top}(\Delta G \Delta - \hat{Q} - \Delta(A + sI))$.
- Are there parameters that we can tune? Choice of s, and then change of basis:

$$\Delta_V = V^*\Delta V, \quad A_V = V^{-1}\tilde{A}V, \; Q_V = V^*\tilde{Q}V, \; G_V = V^{-1}GV^{-*}.$$

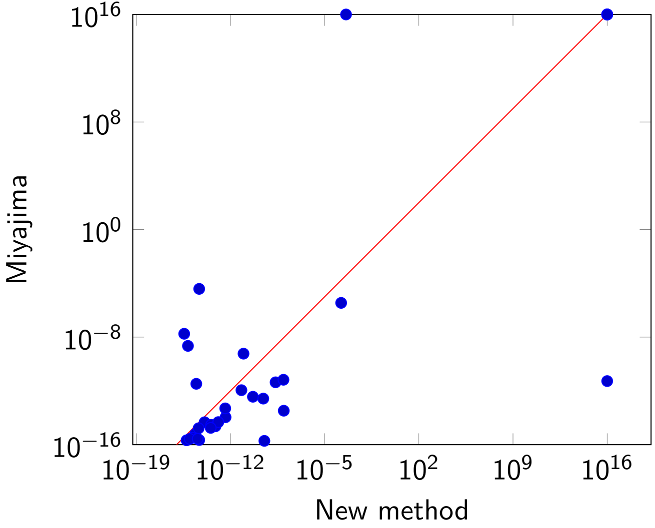
No need to diagonalize this time.
In practice, we choose $V =$ orthogonal Schur factor of $\hat{A}$, $s = -\lambda_{\max}(\hat{A})$

Performance profile on CAREX suite in [Chu et al '07]

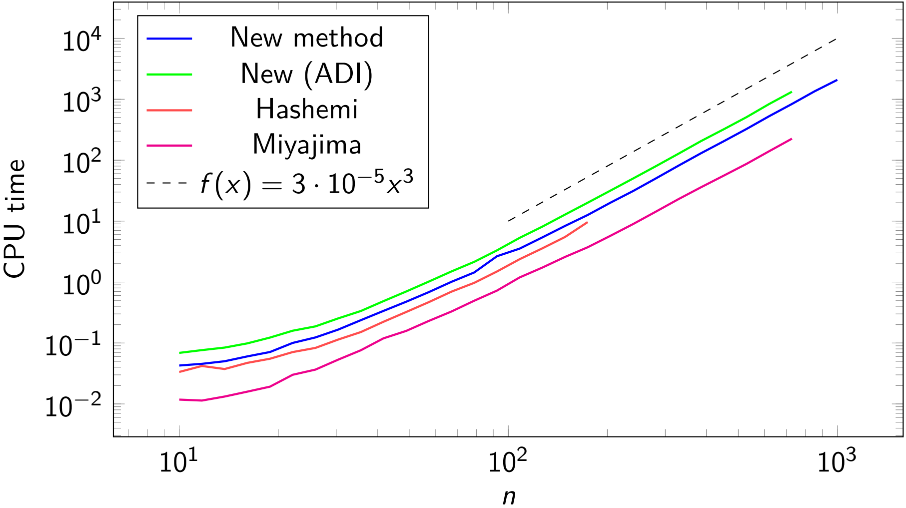


Top left = better.

(Norm-2) width of found interval



CPU time on CAREX 15



Lower = better. New = only method to reach $n = 1000$.

Conclusions

- Technical improvements and ideas from Riccati theory take Krawczyk-based method to state-of-the-art level.
- No method always better than the others, so it is useful to have more choice.
- In almost all cases, the first solution guess $\tilde{x} - [0.9, 1.1]f(\tilde{x})$ already works — so there is still room to optimize.
- Up next: transfer some of these improvements to Miyajima's method.

Conclusions

- Technical improvements and ideas from Riccati theory take Krawczyk-based method to state-of-the-art level.
- No method always better than the others, so it is useful to have more choice.
- In almost all cases, the first solution guess $\tilde{x} - [0.9, 1.1] f(\tilde{x})$ already works — so there is still room to optimize.
- Up next: transfer some of these improvements to Miyajima's method.

[Thanks for your attention!]