Certified Algebraic Path Tracking with Algpath

Alexandre Guillemot
Join work with Pierre Lairez
MATHEXP, Université Paris–Saclay, Inria, France

Ouragan seminar
December 2, 2025

Certified path tracking

$$F$$

Parametrized polynomial system

Certified homotopy continuation

Input: F

Certified path tracking

Point in $\mathbb{C}^n$

$$F_0(\zeta_0) = 0$$

Parametrized polynomial system

Certified homotopy continuation

Input: F, ζ_0

Certified path tracking

Unique continuous extension

$$F_t(\zeta_t) = 0, \quad \forall t \in [0, 1]$$

Parametrized polynomial system

Certified homotopy continuation

Input: F, ζ_0

Certified path tracking

Unique continuous extension

$$F_t(\zeta_t) = 0, \quad \forall t \in [0,1]$$

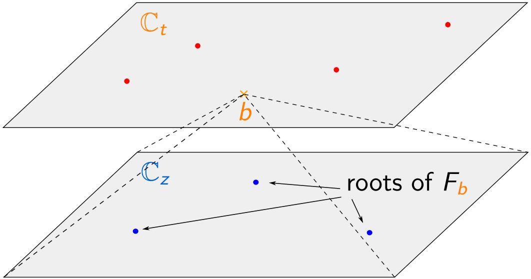
Parametrized polynomial system

Certified homotopy continuation

Input: F, ζ_0

Output: A "certified approximation" of ζ

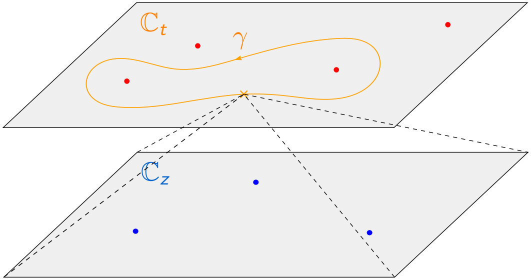
Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ ($n = \deg_z(g)$),
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \backslash \Sigma$ be a base point,

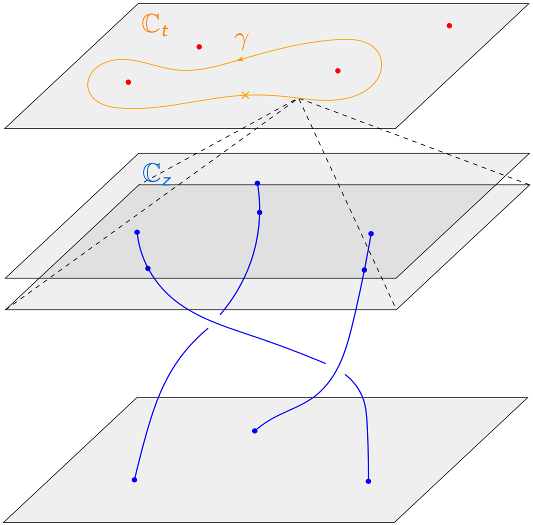
Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ $(n = \deg_z(g))$,
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \backslash \Sigma$ be a base point,
- let $\gamma : [0, 1] \to \mathbb{C} \backslash \Sigma$ be a loop starting at b.

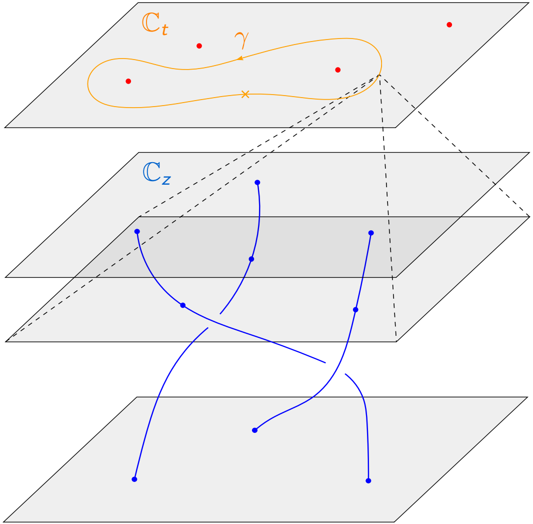
Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ $(n = \deg_z(g))$,
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \backslash \Sigma$ be a base point,
- let $\gamma : [0, 1] \to \mathbb{C} \backslash \Sigma$ be a loop starting at b.
- The displacement of all roots of F_t when t moves along γ defines a braid.

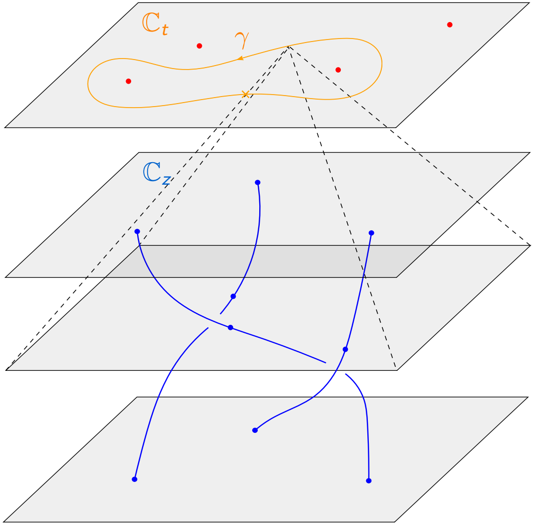
Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ ($n = \deg_z(g)$),
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \backslash \Sigma$ be a base point,
- let $\gamma : [0, 1] \to \mathbb{C} \backslash \Sigma$ be a loop starting at b.
- The displacement of all roots of F_t when t moves along γ defines a braid.

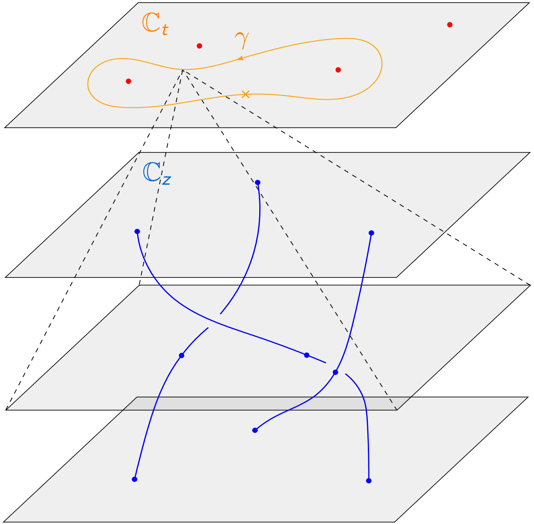
Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ $(n = \deg_z(g))$,
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \backslash \Sigma$ be a base point,
- let $\gamma : [0, 1] \to \mathbb{C} \backslash \Sigma$ be a loop starting at b.
- The displacement of all roots of F_t when t moves along γ defines a braid.

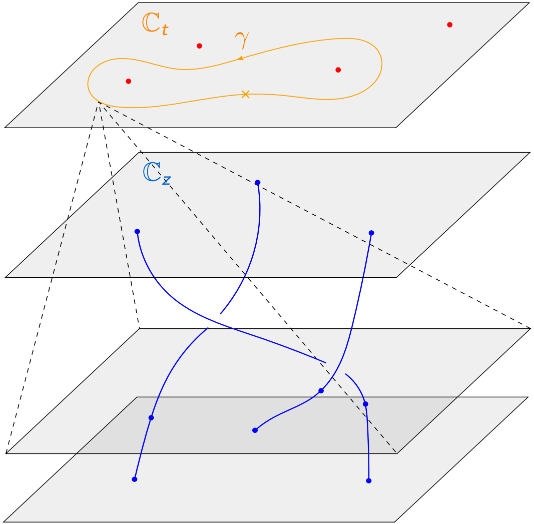
Motivation



Setup

- Let $g \in \mathbb{C}[t,z]$ $(n = \deg_z(g))$,
- define $F_t(z) = g(t,z)$.
- Let $b \in \mathbb{C} \setminus \Sigma$ be a base point,
- let $\gamma : [0,1] \to \mathbb{C} \setminus \Sigma$ be a loop starting at b.
- The displacement of all roots of F_t when t moves along γ defines a braid.

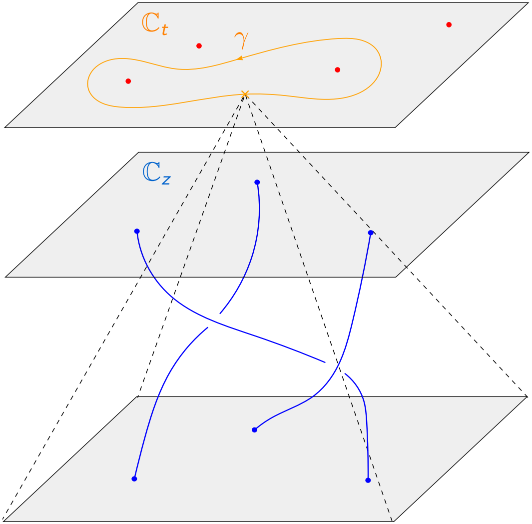
Motivation



Setup

- Let $g \in \mathbb{C}[t,z]$ ($n = \deg_z(g)$),
- define $F_t(z) = g(t,z)$.
- Let $b \in \mathbb{C} \backslash \Sigma$ be a base point,
- let $\gamma : [0,1] \to \mathbb{C} \backslash \Sigma$ be a loop starting at b.
- The displacement of all roots of F_t when t moves along γ defines a braid.

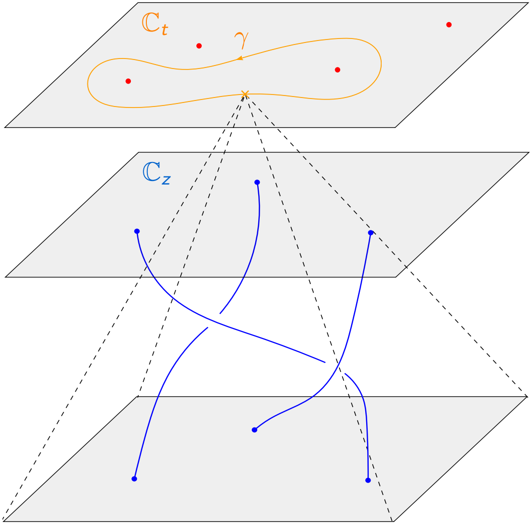
Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ ($n = \deg_z(g)$),
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \backslash \Sigma$ be a base point,
- let $\gamma : [0, 1] \to \mathbb{C} \backslash \Sigma$ be a loop starting at b.
- The displacement of all roots of F_t when t moves along γ defines a braid.

Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ $(n = \deg_z(g))$,
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \backslash \Sigma$ be a base point,
- let $\gamma : [0, 1] \to \mathbb{C} \backslash \Sigma$ be a loop starting at b.
- The displacement of all roots of F_t when t moves along γ defines a braid.

Algorithmic goal

Input: g, γ

Output: the associated braid

Previous work

Noncertified path trackers

- PHCpack by Verschelde (1999)
- Bertini by Bates, Sommese, Hauenstein, and Wampler (2013)
- HomotopyContinuation.jl by Breiding and Timme (2018)

Certified path trackers using Smale's alpha-theory

- NAG for M2 by Beltrán and Leykin (2012, 2013)

Certified path trackers in one variable

- Marco-Buzunariz and Rodríguez (2016)
- Kranich (2016)
- Xu, Burr, and Yap (2018)

Certified path trackers using interval arithmetic

- Kearfott and Xing (1994)
- van der Hoeven (2015) *Krawczyk operator + Taylor models*
- Duff and Lee (2024) *similar to us but independent work*

Interval arithmetic

Problem

Given $f \in \mathbb{R}[x]$, I and J intervals, check $f(\mathsf{I}) \subseteq \mathsf{J}$.

Sufficient solution

- Define interval binary operations $\boxplus$ and $\boxtimes$ that take two intervals, give an interval and is such that for all $x \in \mathsf{I}$, $y \in \mathsf{J}$,

$$x + y \in \mathsf{I} \boxplus \mathsf{J}, \; xy \in \mathsf{I} \boxtimes \mathsf{J}.$$

- Write f as a composition of binary operations and replace each operation by its interval counterpart (interval extension), then plug I and check if the result is contained in J.

! This is only a sufficient condition.

Computational model

- Interval endpoints: $\mathbb{Q}$,
- $[a, b] \boxplus [c, d] = [a + c, b + d]$,
- $[a, b] \boxtimes [c, d] = [\min\{ac, ad, bc, bd\}, \max\{ac, ad, bc, bd\}]$.

Pros and cons

✔ Good theoritical properties.

✘ Coefficient swell.

Computational model

- Interval endpoints: {IEEE-754 64-bits floating-point numbers},
- $[a, b] \boxplus [c, d] = [\underline{a + c}, \overline{b + d}]$,
- $[a, b] \boxtimes [c, d] = [\min\{\underline{ac}, \underline{ad}, \underline{bc}, \underline{bd}\}, \max\{\overline{ac}, \overline{ad}, \overline{bc}, \overline{bd}\}]$.

Pros and cons

✔ Fast.

✘ Bad theoritical properties.

✘ Not enough representable numbers.

Computational model

- Interval endpoints: $\{m2^e \in \mathbb{R} : m, e \in \mathbb{Z}\}$,
- precision $P \in \mathbb{Z}_>$ (number of bits) that can be changed at will,
- interval operations: endpoints rounding done at precision P.

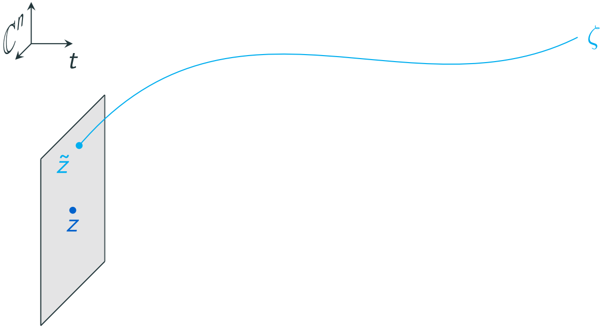
Pros and cons

✔ Good theoritical properties as $P \to \infty$.

✔ Fast when we can maintain low precision.

⚙ Need to carefully manage P in the algorithms to ensure termination.

The folklore meta-algorithm

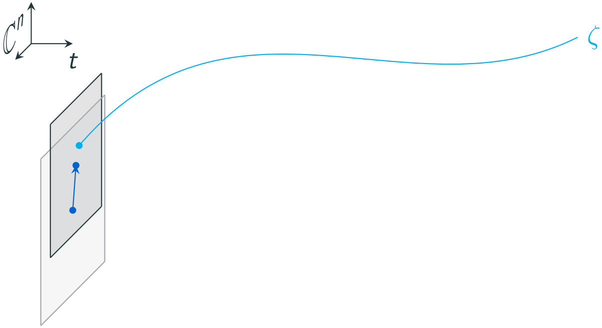


$F : \mathbb{C} \times \mathbb{C}^n \to \mathbb{C}^n$, $F_t(z) = F(t, z)$,
m isolates a zero of F_0.

def *track*(F, m)**:**

$t \leftarrow 0; \quad L \leftarrow []$
while $t < 1$**:**
  $m \leftarrow \mathit{refine}(F_t, m)$
  $\delta \leftarrow \mathit{extend}(F, t, m)$
  $t \leftarrow t + \delta$
  append (t, m) to L
return L

The folklore meta-algorithm

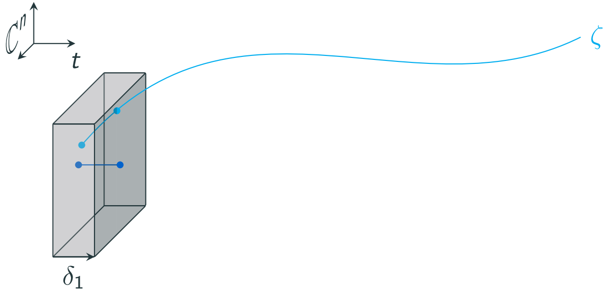


$F : \mathbb{C} \times \mathbb{C}^n \to \mathbb{C}^n$, $F_t(z) = F(t, z)$,
m isolates a zero of F_0.

def *track*(F, m)**:**

$t \leftarrow 0$; $\quad L \leftarrow []$
while $t < 1$**:**
    $m \leftarrow \mathit{refine}(F_t, m)$
    $\delta \leftarrow \mathit{extend}(F, t, m)$
    $t \leftarrow t + \delta$
    append (t, m) to L
return L

The folklore meta-algorithm

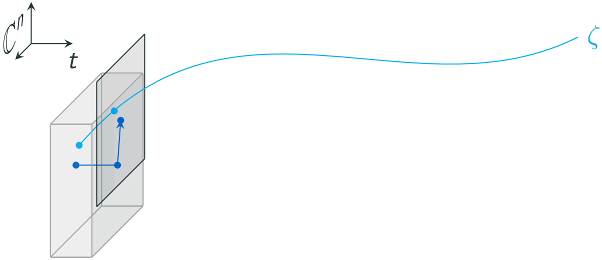


$F : \mathbb{C} \times \mathbb{C}^n \to \mathbb{C}^n$, $F_t(z) = F(t, z)$,
m isolates a zero of F_0.

def *track*(F, m)**:**

```
t ← 0;     L ← []
while t < 1:
    m ← refine(F_t, m)
    δ ← extend(F, t, m)
    t ← t + δ
    append (t, m) to L
return L
```

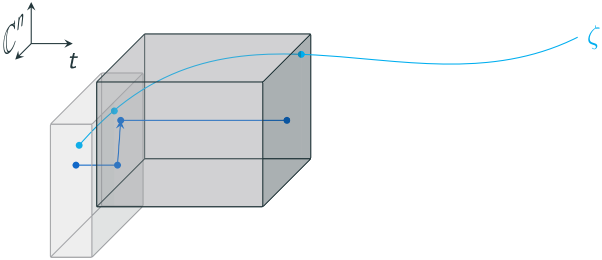
The folklore meta-algorithm



$F : \mathbb{C} \times \mathbb{C}^n \to \mathbb{C}^n$, $F_t(z) = F(t, z)$,
m isolates a zero of F_0.

```
def track(F, m):
    t ← 0;    L ← []
    while t < 1:
        m ← refine(F_t, m)
        δ ← extend(F, t, m)
        t ← t + δ
        append (t, m) to L
    return L
```

The folklore meta-algorithm



$F : \mathbb{C} \times \mathbb{C}^n \to \mathbb{C}^n$, $F_t(z) = F(t, z)$,

m isolates a zero of F_0.

def *track*(F, m)**:**

$t \leftarrow 0$; $L \leftarrow []$

while $t < 1$**:**

  $m \leftarrow \mathit{refine}(F_t, m)$

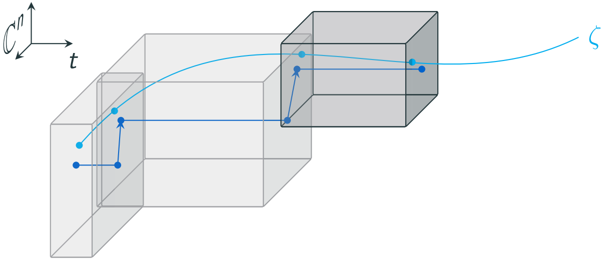
  $\delta \leftarrow \mathit{extend}(F, t, m)$

  $t \leftarrow t + \delta$

  append (t, m) to L

return L

The folklore meta-algorithm



$F : \mathbb{C} \times \mathbb{C}^n \to \mathbb{C}^n$, $F_t(z) = F(t, z)$,
m isolates a zero of F_0.

def *track*(F, m)**:**

$t \leftarrow 0$; $\quad L \leftarrow []$
while $t < 1$**:**
  $m \leftarrow \mathit{refine}(F_t, m)$
  $\delta \leftarrow \mathit{extend}(F, t, m)$
  $t \leftarrow t + \delta$
  append (t, m) to L
return L

Moore boxes, the datastructure for isolating boxes

Root isolation criterion (Krawczyk˙1969, Moore˙1977, Rump˙1983)

- $f : \mathbb{C}^n \to \mathbb{C}^n$ polynomial, $\rho \in (0, 1)$,
- $z \in \mathbb{C}^n$, $A \in \mathbb{C}^{n\times n}$, $B \subseteq \mathbb{C}^n$ a ball of center 0,

such that for all $u, v \in B$,

$$-A \cdot f(z) + [\mathrm{I}_n - A \cdot df(z + u)]v \in \rho B.$$

Then f has a unique zero in $z + \rho B$.

Moore boxes, the datastructure for isolating boxes

Root isolation criterion (Krawczyk˙1969, Moore˙1977, Rump˙1983)

- $f : \mathbb{C}^n \to \mathbb{C}^n$ polynomial, $\rho \in (0, 1)$,
- $z \in \mathbb{C}^n$, $A \in \mathbb{C}^{n\times n}$, $B \subseteq \mathbb{C}^n$ a ball of center 0,

$$-A \cdot f(z) + [\mathrm{I}_n - A \cdot df(z + B)]B \subseteq \rho B.$$

Then f has a unique zero in $z + \rho B$.

Moore boxes, the datastructure for isolating boxes

Root isolation criterion (Krawczyk˙1969, Moore˙1977, Rump˙1983)

- $f : \mathbb{C}^n \to \mathbb{C}^n$ polynomial, $\rho \in (0, 1)$,
- $z \in \mathbb{C}^n$, $A \in \mathbb{C}^{n\times n}$, $B \subseteq \mathbb{C}^n$ a ball of center 0,

$$-A \cdot f(z) + [\mathrm{I}_n - A \cdot df(z + B)]B \subseteq \rho B.$$

Then f has a unique zero in $z + \rho B$.

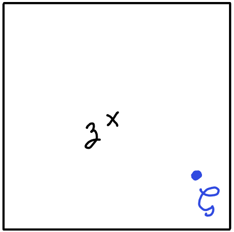
Proof sketch

We show that $\varphi : z + \rho B \to \mathbb{C}^n$ defined by $\varphi(w) = w - Af(w)$ is a ρ-contraction map with values in $z + \rho B$.

Definition

A ρ-Moore box for f is a triple (z, B, A) which satisfies Moore's criterion.

Refinement of Moore boxes

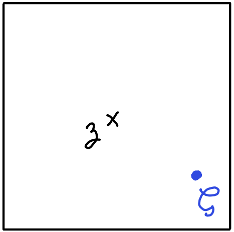


Strategy

(z, B, A) a $\frac{7}{8}$-Moore box for f, ζ its associated zero.

$$-A \cdot f(z) + [\mathrm{I}_n - A \cdot df(z + B)]B \subseteq \frac{7}{8}B.$$

Refinement of Moore boxes



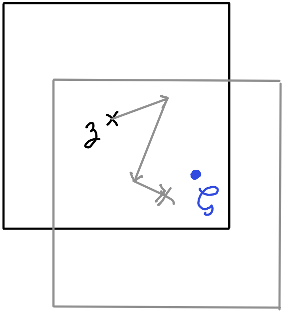
Strategy

(z, B, A) a $\frac{7}{8}$-Moore box for f, ζ its associated zero.

We want to transform it into a $\frac{1}{8}$-Moore box.

$$-A \cdot f(z) + [\mathrm{I}_n - A \cdot df(z + B)]B \overset{?}{\subseteq} \frac{1}{8}B.$$

Refinement of Moore boxes



Strategy

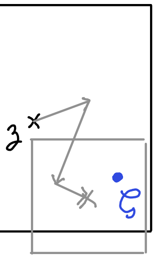
(z, B, A) a $\frac{7}{8}$-Moore box for f, ζ its associated zero.

We want to transform it into a $\frac{1}{8}$-Moore box.

$$-A \cdot f(z) + [\mathrm{I}_n - A \cdot df(z+B)]B \overset{?}{\subseteq} \frac{1}{8}B.$$

Reduce left term with quasi-Newton $z \mapsto z - Af(z)$.

Refinement of Moore boxes



Strategy

(z, B, A) a $\frac{7}{8}$-Moore box for f, ζ its associated zero.

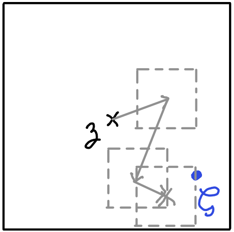
We want to transform it into a $\frac{1}{8}$-Moore box.

$$-A \cdot f(z) + [\mathrm{I}_n - A \cdot df(z+B)]B \overset{?}{\subseteq} \frac{1}{8}B.$$

Reduce left term with quasi-Newton $z \mapsto z - Af(z)$.

Reduce right term with ball radius reductions.

Refinement of Moore boxes



Strategy

(z, B, A) a $\frac{7}{8}$-Moore box for f, ζ its associated zero.

We want to transform it into a $\frac{1}{8}$-Moore box.

$$-A \cdot f(z) + [\mathrm{I}_n - A \cdot df(z + B)]B \overset{?}{\subseteq} \frac{1}{8}B.$$

Reduce left term with quasi-Newton $z \mapsto z - Af(z)$.
Reduce right term with ball radius reductions.
How do we handle numerical errors?

Refinement of Moore boxes

def *refine*(f, z, B, A)**:**

$U \leftarrow A$; $B \leftarrow 2B$
while not $-A \cdot f(z) + [I - A \cdot df(z + B)]\, B \subseteq \frac{1}{8} B$
  if $-Uf(z) \subseteq \frac{1}{512} B$**:** # left term is small
    $B \leftarrow \frac{1}{2} B$
  else: # left term is big
    $z \leftarrow z - Uf(z)$
  $A \leftarrow Jf(z)^{-1}$ # unchecked arithmetic
return z, B, A

Input

- $f : \mathbb{C}^n \to \mathbb{C}^n$ polynomial,
- z, B, A a $\frac{7}{8}$-Moore box for f.

Output

A $\frac{1}{8}$-Moore box for f with same associated zero as z, B, A.

Refinement of Moore boxes

def *refine*(f, z, B, A):

$U \leftarrow A$; $B \leftarrow 2B$; shrink_cnt $\leftarrow 0$
while not $-A \cdot f(z) + [I - A \cdot df(z + B)]\, B \subseteq \frac{1}{8} B$
  if $-Uf(z) \subseteq \frac{1}{512} B$: # left term is small
    $B \leftarrow \frac{1}{2} B$; shrink_cnt $\leftarrow$ shrink_cnt $+ 1$
    if shrink_cnt > 8:
      increase working precision
  else: # left term is big
    $z \leftarrow z - Uf(z)$
  $A \leftarrow df(z)^{-1}$ # unchecked arithmetic
return z, B, A

Input

- $f : \mathbb{C}^n \to \mathbb{C}^n$ polynomial,
- z, B, A a $\frac{7}{8}$-Moore box for f.

Output

A $\frac{1}{8}$-Moore box for f with same associated zero as z, B, A.

Refinement of Moore boxes

def *refine*(f, z, B, A):

```
U ← A;  B ← 2B;  shrink_cnt ← 0
while not −A·f(z) + [I − A·df(z + B)] B ⊆ (1/8) B
    if −U f(z) ⊆ (1/512) B:  # left term is small
        B ← (1/2) B;  shrink_cnt ← shrink_cnt + 1
        if shrink_cnt > 8:
            increase working precision
    else:  # left term is big
        δ ← U f(z)
        if width(z − δ) > (1/40) mag(δ):
            increase working precision
        else:
            z ← mid(z − δ);
    A ← df(z)^(−1)  # unchecked arithmetic
return z, B, A
```

Input

- $f : \mathbb{C}^n \to \mathbb{C}^n$ polynomial,
- z, B, A a $\frac{7}{8}$-Moore box for f.

Output

A $\frac{1}{8}$-Moore box for f with same associated zero as z, B, A.

Refinement of Moore boxes

def *refine*(f, z, B, A):

$U \leftarrow A$; $B \leftarrow 2B$; shrink_cnt $\leftarrow 0$
while not $-A \cdot f(z) + [I - A \cdot df(z + B)]\, B \subseteq \frac{1}{8} B$
  if $-Uf(z) \subseteq \frac{1}{512} B$: # left term is small
    $B \leftarrow \frac{1}{2} B$; shrink_cnt $\leftarrow$ shrink_cnt $+ 1$
    if shrink_cnt > 8:
      increase working precision
  else: # left term is big
    $\delta \leftarrow Uf(z)$
    if $\mathit{width}(z - \delta) > \frac{1}{40}\, \mathit{mag}(\delta)$:
      increase working precision
    else:
      $z \leftarrow \mathit{mid}(z - \delta)$;
  $A \leftarrow df(z)^{-1}$ # unchecked arithmetic
return z, B, A

Input

- $f : \mathbb{C}^n \to \mathbb{C}^n$ polynomial,
- z, B, A a $\frac{7}{8}$-Moore box for f.

Output

A $\frac{1}{8}$-Moore box for f with same associated zero as z, B, A.

Proposition

refine terminates and is correct.

Step validation

Input

- $F : \mathbb{C} \times \mathbb{C}^n \to \mathbb{C}^n$,
- $t \in [0, 1]$,
- (z, B, A) a $\frac{1}{8}$-Moore box for F_t, returned by refine.

recall $F_s(z) = F(s, z)$

Output

$\delta > 0$ s.t. for all $s \in [t, t + \delta]$, (z, B, A) is a $\frac{7}{8}$-Moore box for F_s.

def *extend*$(F, t, \delta_{\mathrm{hint}}, z, B, A)$

$\delta \leftarrow \delta_{\mathrm{hint}}$**;** $T \leftarrow [t, t + \delta]$

while not $-AF_T(z) + [I - AdF_T(z + B)]\, B \subseteq \frac{7}{8} B$**:**

  $\delta \leftarrow \frac{\delta}{2}$**;** $T \leftarrow [t, t + \delta]$

  if $\delta < 2^{-P}$**:** # P working precision in bits

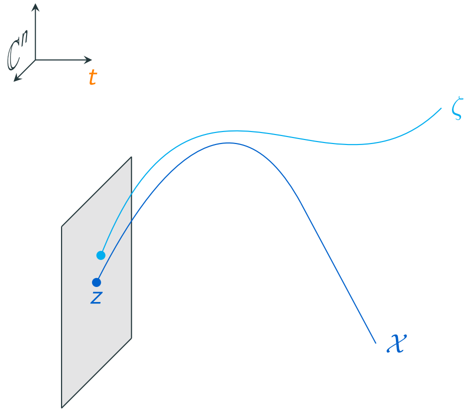
    increase working precision

return δ

Proposition

extend terminates and is correct.

Extending along a predictor

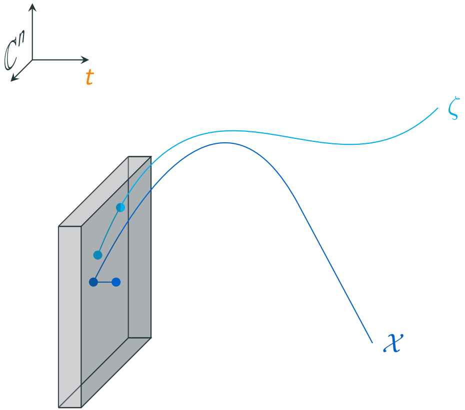


Predictor

A map $\mathcal{X} : \mathbb{R} \to \mathbb{C}^n$ such that $\mathcal{X}(0) = z$.

In practice, one should have $\mathcal{X}(s) \approx \zeta(t+s)$ around 0.

Extending along a predictor



Predictor

A map $\mathcal{X} : \mathbb{R} \to \mathbb{C}^n$ such that $\mathcal{X}(0) = z$.

In practice, one should have $\mathcal{X}(s) \approx \zeta(t+s)$ around 0.

How to extend?

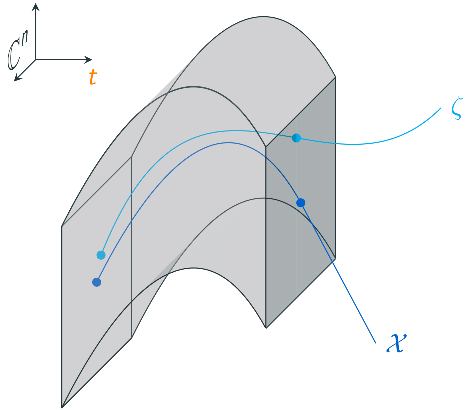
Pb: check that for all $s \in [0, \delta]$, (z, B, A) is a ρ-Moore box for F_{t+s}.

Soln: try

$$-AF_{t+S}(z) + [1 - AdF_{t+S}(z+B)]B \subseteq \rho B$$

where $S = [0, \delta]$.

Extending along a predictor



Predictor

A map $\mathcal{X} : \mathbb{R} \to \mathbb{C}^n$ such that $\mathcal{X}(0) = z$.

In practice, one should have $\mathcal{X}(s) \approx \zeta(t+s)$ around 0.

How to extend along $\mathcal{X}$?

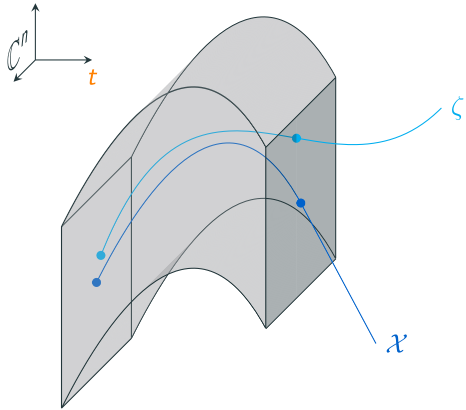
Pb: check that for all $s \in [0, \delta]$, $(\mathcal{X}(s), B, A)$ is a ρ-Moore box for F_{t+s}.

Soln? try

$$-AF_{t+S}(\mathcal{X}(S)) + [1 - AdF_{t+S}(\mathcal{X}(S) + B)]B \subseteq \rho B$$

where $S = [0, \delta]$.

Extending along a predictor



Predictor

A map $\mathcal{X} : \mathbb{R} \to \mathbb{C}^n$ such that $\mathcal{X}(0) = z$.

In practice, one should have $\mathcal{X}(s) \approx \zeta(t+s)$ around 0.

How to extend along $\mathcal{X}$?

Pb: check that for all $s \in [0, \delta]$, $(\mathcal{X}(s), B, A)$ is a ρ-Moore box for F_{t+s}.

Soln? try

$$-AF_{t+S}(\mathcal{X}(S)) + [1 - AdF_{t+S}(\mathcal{X}(S) + B)]B \subseteq \rho B$$

where $S = [0, \delta]$.

This is too strong! Way around the dependency problem: **Taylor models**

Taylor models with relative remainder

Definition

- A real interval S containing zero,
- a polynomial $P(\eta) = A_0 + A_1\eta + \cdots + A_{d+1}\eta^{d+1}$ where A_i is a (complex) interval.

d is the order of the Taylor model.

Taylor models with relative remainder

Definition

- A real interval S containing zero,
- a polynomial $P(\eta) = A_0 + A_1\eta + \cdots + A_{d+1}\eta^{d+1}$ where A_i is a (complex) interval.

d is the order of the Taylor model.

Definition

A Taylor model (S, P) of degree d encloses a function $\mathcal{X} : \mathbb{R} \to \mathbb{C}$ if

$$\forall s \in S, \forall i \in 0, \ldots, d+1, \exists a_i \in A_i \text{ s.t. } \mathcal{X}(s) = a_0 + a_1 s + \cdots + a_{d+1} s^{d+1}.$$

Taylor models with relative remainder

Definition

- A real interval S containing zero,
- a polynomial $P(\eta) = A_0 + A_1\eta + \cdots + A_{d+1}\eta^{d+1}$ where A_i is a (complex) interval.

d is the order of the Taylor model.

Definition

A Taylor model (S, P) of degree d encloses a function $\mathcal{X} : \mathbb{R} \to \mathbb{C}$ if

$$\forall s \in S, \forall i \in 0, \ldots, d+1, \exists a_i \in A_i \text{ s.t. } \mathcal{X}(s) = a_0 + a_1 s + \cdots + a_{d+1} s^{d+1}.$$

Remark

If $J \subseteq S$, we have $\mathcal{X}(J) \subseteq P(J)$ (where $P(J)$ is computed using interval arithmetic).

Reduction

Let (S, P) be a Taylor model of order d.

Goal : reduce it order to $d - 1$, s.t. if (S, P) encloses a function, so does its reduction.

Solution : replace $A_d \eta^d + A_{d+1} \eta^{d+1}$ by $[A_d \boxplus (A_{d+1} \boxtimes S)]\eta^d$.

Arithmetic

Reduction

Let (S, P) be a Taylor model of order d.

Goal : reduce it order to $d-1$, s.t. if (S, P) encloses a function, so does its reduction.

Solution : replace $A_d \eta^d + A_{d+1}\eta^{d+1}$ by $[A_d \boxplus (A_{d+1} \boxtimes S)]\eta^d$.

Let (S, P) and (S, Q) be Taylor models of order d.

Sum

Component-wise sum of P and Q using $\boxplus$. Compatible with sums of enclosed functions.

Product

Usual product formula with P and Q, gives a Taylor model of order $2d+1$, then reduce it to make it of order d. Compatible with products of enclosed functions.

Back to our problem

- (z, B, A) is a $\frac{1}{8}$-Moore box for F_t,
- $\mathcal{X} : \mathbb{R} \to \mathbb{C}^n$ polynomial s.t. $\mathcal{X}(0) = z$.

We want to check that for all $s \in [0, \delta]$, $(\mathcal{X}(s), B, A)$ is a $\frac{7}{8}$-Moore box for F_{t+s}. i.e.

$$-A \cdot F_{t+s}(\mathcal{X}(s)) + [1 - A \cdot dF_{t+s}(\mathcal{X}(s) + B)]B \subseteq \frac{7}{8}B.$$

Back to our problem

- (z, B, A) is a $\frac{1}{8}$-Moore box for F_t,
- $\mathcal{X} : \mathbb{R} \to \mathbb{C}^n$ polynomial s.t. $\mathcal{X}(0) = z$.

We want to check that for all $s \in [0, \delta]$, $(\mathcal{X}(s), B, A)$ is a $\frac{7}{8}$-Moore box for F_{t+s}. i.e.

$$-A \cdot F_{t+s}(\mathcal{X}(s)) + [1 - A \cdot dF_{t+s}(\mathcal{X}(s) + B)]B \subseteq \frac{7}{8}B.$$

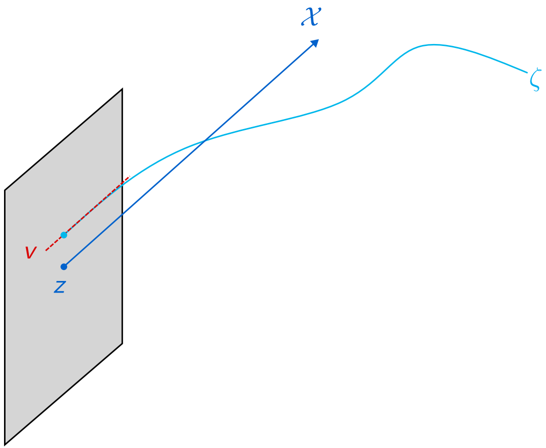
Solution using Taylor models

- Compute an order d taylor model $\mathcal{K}$ on $[0, \delta]$ enclosing

$$s \mapsto -A \cdot F_{t+s}(\mathcal{X}(s)) + [1 - A \cdot dF_{t+s}(\mathcal{X}(s) + B)]B.$$

  This is just Taylor model arithmetic !
- Check that $\mathcal{K}([0, \delta]) \subseteq \frac{7}{8}B$ (interval arithmetic).

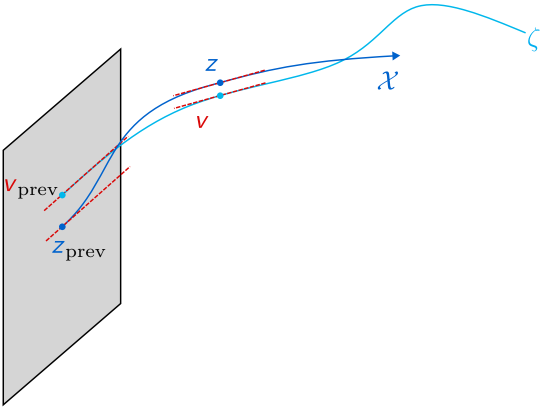
Choosing the right predictor



Tangent predictor

Idea: $-A \cdot \frac{\partial F}{\partial t}(t, z)$ is a good approximation of $\zeta'(t)$. Use it to do a order 1 correction.

Choosing the right predictor



Tangent predictor

Idea: $-A \cdot \frac{\partial F}{\partial t}(t, z)$ is a good approximation of $\zeta'(t)$. Use it to do a order 1 correction.

Hermite predictor

Idea: use previous point z_{prev} and previous tangent vector v_{prev}, z and v to do a Hermite cubic spline approximation.

Algpath

Features

- **Rust** implementation,
- available at **https://gitlab.inria.fr/numag/algpath,**
- **SIMD double precision interval arithmetic** following Lambov (2008),
- **adaptive precision** using **Arb**[1],
- **mixed precision** between double precision and Arb **without overhead**.

[1] Johansson, 2017.

Adaptive precision, in practice

Precision decreases

The algorithm presented only increases precision.

High precision is computationally costly.

Can we introduce precision decreases?

def *track*(F, m)**:**

```
t ← 0;    L ← []
while t < 1:
    decrease P by 1?
    m ← refine(F_t, m)
    δ ← extend(F, t, m)
    t ← t + δ
    append (t, m) to L
return L
```

Adaptive precision, in practice

Precision decreases

The algorithm presented only increases precision.

⚠ High precision is computationally costly.

Can we introduce precision decreases?

Computational model, again

- Precision is managed globally,
- a change of precision induces **no changes on data, only operations are changed**,
- precision of data is indirectly changed by performing operations on it.

def $track(F, m)$**:**

$t \leftarrow 0$; $L \leftarrow []$

while $t < 1$:

  decrease P by 1!

  $m \leftarrow refine(F_t, m)$

  $\delta \leftarrow extend(F, t, m)$

  $t \leftarrow t + \delta$

  append (t, m) to L

return L

Mixed precision

Double precision SIMD interval arithmetic is faster than Arb, but it lacks the ability to manage precision. . .

Goal

Use double precision when possible, else use Arb. We want to have no overhead over double precision only.

- Data can either be double precision or Arb balls. Operations manage arithmetic switch depending on precision
- ! Overhead
- ! Challenging implementation

```
enum MixedRI {
  Fast(F64RI),
  Accurate(Arb),
}
```

Spacing arithmetic switches

One iteration of the main loop

```
def one_step(F, m): # m isolating box
    try:
        convert m to double precision
        perform a corrector-predictor round at double precision
    except:
        convert m to Arb
        perform a corrector-predictor round using Arb
```

Spacing arithmetic switches

One iteration of the main loop

```
def one_step(F, m): # m isolating box
    try:
        convert m to double precision
        perform a corrector-predictor round at double precision
    except:
        convert m to Arb
        perform a corrector-predictor round using Arb
```

Can we always convert m to Arb?

Can we always convert m to double precision when the working precision is 53?

Exact conversions

Exact conversions fail both ways !

Consider double precision interval $[-2^{-50}, 2]$. The exact ball associated is $[(1-2^{-51}) \pm (1+2^{-51})]$. $1+2^{-51}$ cannot be represented by a `mag_t`!

Remark

- Recall: a moore box is a triple (z, B, A) where $z \in \mathbb{C}^n$, $A \in \mathbb{C}^{n\times n}$ and B is given by its radius $r > 0$. In practice, represented by singleton intervals.
- Conversions of singleton intervals behave as expected!

			algpath	algpath (fixed precision)
name	dim.	max deg	time (s)	time (s)
dense	1	100	0.4	0.4
katsura	16	2	42 min	41 min
dense	2	50	588	588

Implementation details

We would like to avoid writting the algorithm for each arithmetic.

Challenges

- Rust is statically typed,
- our functions depend on the type of intervals (double precision, Arb balls) but also on higher level types (e.g. complex intervals, interval matrices),
- Rust's generics are interface based.

Still we tried

- ⊕ Very little code duplication.
- ⊕ Easy to integrate additional arithmetics.
- ⊖ Lots of complicated interfaces trying to avoid "where clause" swell.
- ⊖ High level generic functions require heavy setup for only a few lines of code.

Conclusion

Features

- Rust implementation available at **https://gitlab.inria.fr/numag/algpath,**
- **certified** corrector-predictor loop,
- relies on **interval arithmetic** and **Krawczyk's method**,
- **SIMD double precision interval arithmetic**,
- **adaptive precision** using **Arb**,
- **mixed precision** between double precision and Arb **without overhead**.

Todos

- Interface with Sage or Julia
- Avx512?

References i

Bates, D. J., Sommese, A. J., Hauenstein, J. D., & Wampler, C. W. (2013). *Numerically Solving Polynomial Systems with Bertini*. Society for Industrial; Applied Mathematics. https://doi.org/10.1137/1.9781611972702

Beltrán, C., & Leykin, A. (2012). Certified Numerical Homotopy Tracking. *Experimental Mathematics*, *21*(1), 69–83. https://doi.org/10.1080/10586458.2011.606184

Beltrán, C., & Leykin, A. (2013). Robust Certified Numerical Homotopy Tracking. *Found Comput Math*, *13*(2), 253–295. https://doi.org/10.1007/s10208-013-9143-2

Breiding, P., & Timme, S. (2018). HomotopyContinuation.jl: A Package for Homotopy Continuation in Julia. In J. H. Davenport, M. Kauers, G. Labahn, & J. Urban (Eds.), *Mathematical Software – ICMS 2018* (pp. 458–465). Springer International Publishing. https://doi.org/10.1007/978-3-319-96418-8_54

Duff, T., & Lee, K. (2024). Certified homotopy tracking using the Krawczyk method. *Proc. ISSAC 2024*, 274–282. https://doi.org/10.1145/3666000.3669699

Johansson, F. (2017). Arb: Efficient Arbitrary-Precision Midpoint-Radius Interval Arithmetic. *IEEE Transactions on Computers*, *66*(8), 1281–1292. https://doi.org/10.1109/TC.2017.2690633

Kearfott, R. B., & Xing, Z. (1994). An Interval Step Control for Continuation Methods. *SIAM J. Numer. Anal.*, *31*(3), 892–914. https://doi.org/10.1137/0731048

References ii

Kranich, S. (2016). An epsilon-delta bound for plane algebraic curves and its use for certified homotopy continuation of systems of plane algebraic curves. https://doi.org/10.48550/arXiv.1505.03432

Lambov, B. (2008). Interval Arithmetic Using SSE-2. In P. Hertling, C. M. Hoffmann, W. Luther, & N. Revol (Eds.), *Reliab. Implement. Real Number Algorithms* (pp. 102–113). Springer. https://doi.org/10.1007/978-3-540-85521-7_6

Marco-Buzunariz, M. Á., & Rodríguez, M. (2016). SIROCCO: A Library for Certified Polynomial Root Continuation. In G.-M. Greuel, T. Koch, P. Paule, & A. Sommese (Eds.), *Mathematical Software – ICMS 2016* (pp. 191–197). Springer International Publishing. https://doi.org/10.1007/978-3-319-42432-3_24

van der Hoeven, J. (2015). *Reliable homotopy continuation* (Research Report). LIX, Ecole polytechnique.

Verschelde, J. (1999). Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation. *ACM Trans. Math. Softw.*, *25*(2), 251–276. https://doi.org/10.1145/317275.317286

Xu, J., Burr, M., & Yap, C. (2018). An Approach for Certifying Homotopy Continuation Paths: Univariate Case. *Proc. ISSAC 2018*, 399–406. https://doi.org/10.1145/3208976.3209010