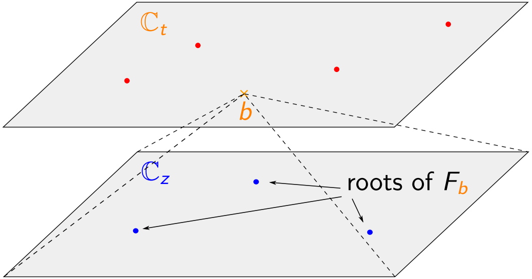
Certified Algebraic Path Tracking with Algpath

Alexandre Guillemot & Pierre Lairez
MATHEXP, Université Paris–Saclay, Inria, France

Siam AG 2025
July 7–11, 2025 | University of Wisconsin-Madison, Madison, USA

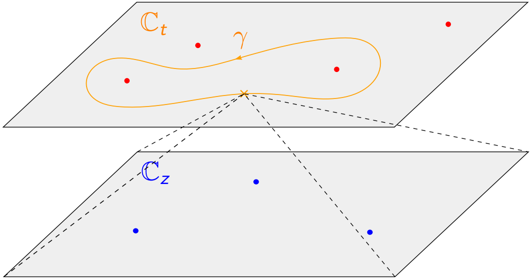
Motivation: braid computations



Setup

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

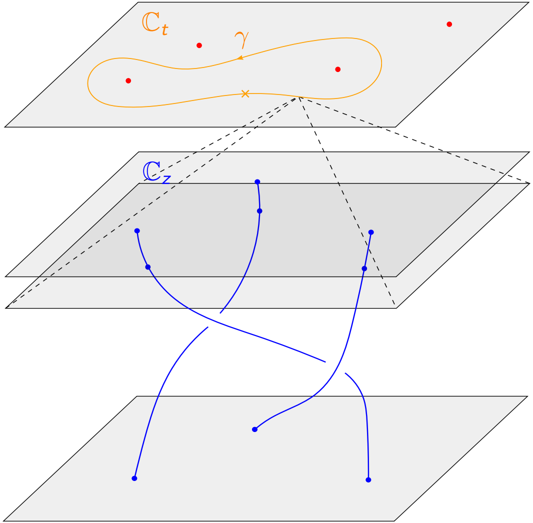
Motivation: braid computations



Setup

- Let $g \in \mathbb{C}[t, z]$,
- 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.

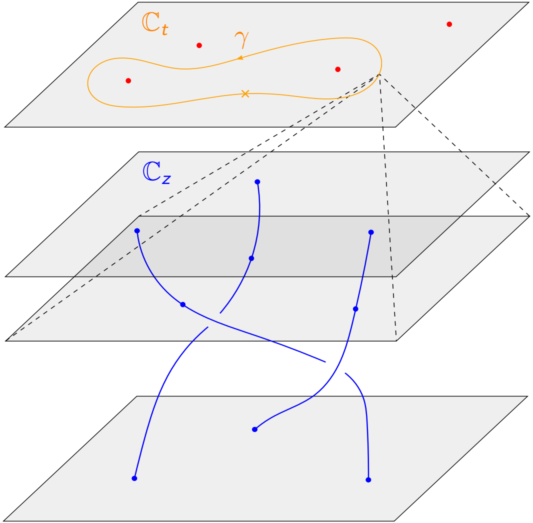
Motivation: braid computations



Setup

- Let $g \in \mathbb{C}[t, z]$,
- 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.

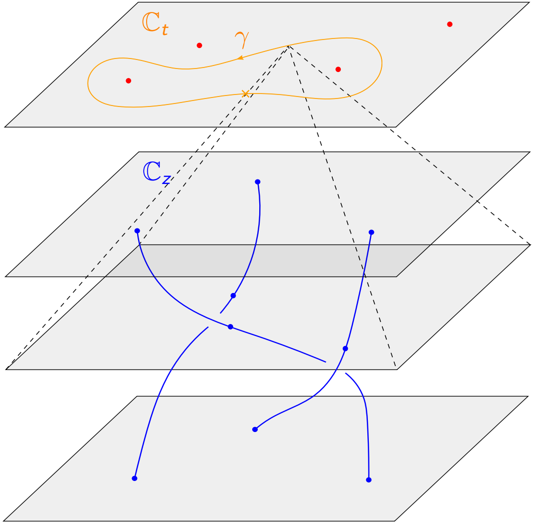
Motivation: braid computations



Setup

- Let $g \in \mathbb{C}[t, z]$,
- 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.

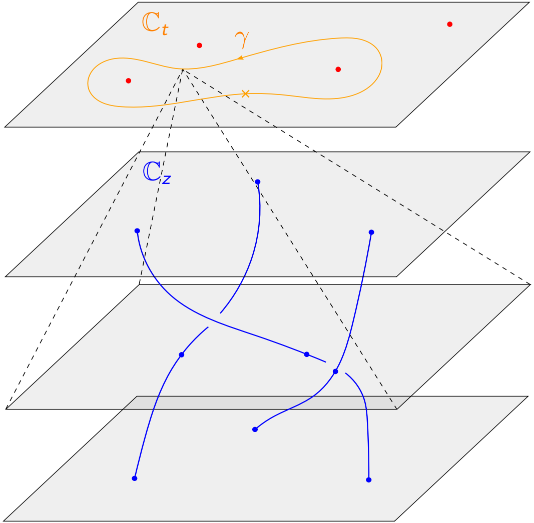
Motivation: braid computations



Setup

- Let $g \in \mathbb{C}[t, z]$,
- 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.

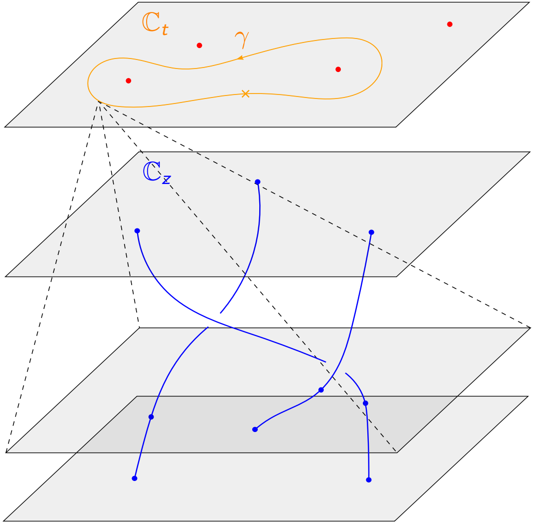
Motivation: braid computations



Setup

- Let $g \in \mathbb{C}[t, z]$,
- 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.

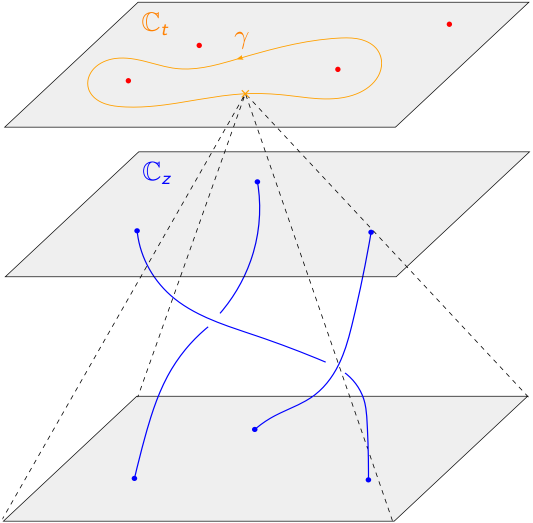
Motivation: braid computations



Setup

- Let $g \in \mathbb{C}[t, z]$,
- 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.

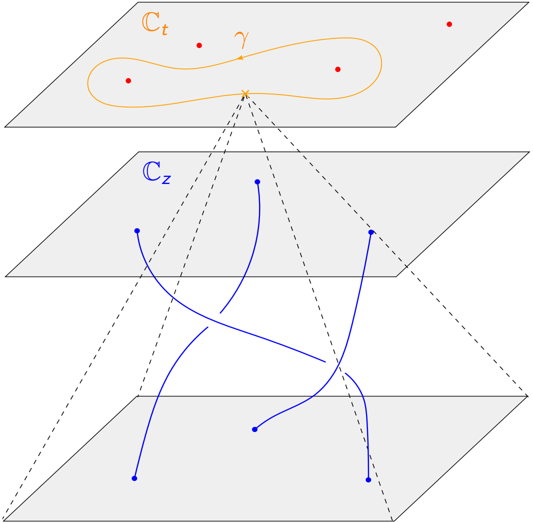
Motivation: braid computations



Setup

- Let $g \in \mathbb{C}[t, z]$,
- 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: braid computations



Setup

- Let $g \in \mathbb{C}[t, z]$,
- 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.

Algorithmic goal

Input: g, γ
Output: the associated braid
Tool: certified path tracking

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

Related 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

- SIROCCO by 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)

Algpath

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** following [Lambov, 2008],
- NEW! **adaptive precision** using **Arb**[1],
- NEW! **mixed precision** between double precision and Arb **without overhead**.

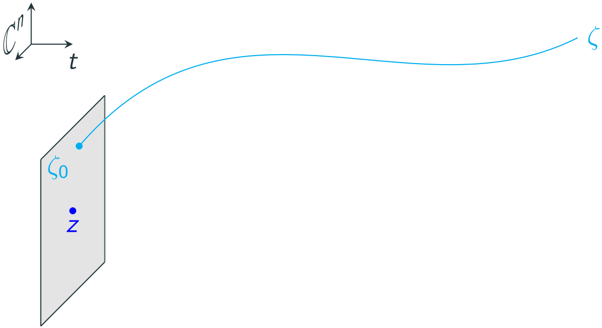
Applications

- Monodromy computations,
- **Braid computations**

[1] F. Johansson. "Arb: efficient arbitrary-precision midpoint-radius interval arithmetic"

Certified corrector-predictor loop

Recall: for all $t \in [0,1]$, $F_t(\zeta_t) = 0$

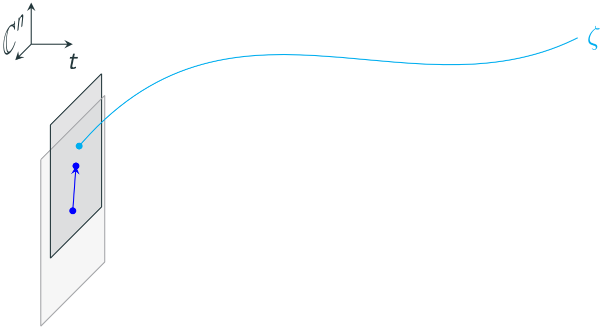


def track(F, z):

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

Certified corrector-predictor loop

Recall: for all $t \in [0, 1]$, $F_t(\zeta_t) = 0$

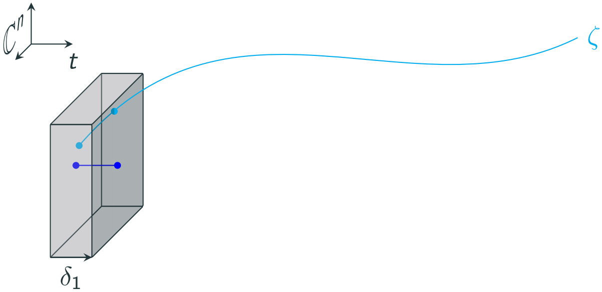


def `track`(F, z)**:**

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

Certified corrector-predictor loop

Recall: for all $t \in [0,1]$, $F_t(\zeta_t) = 0$

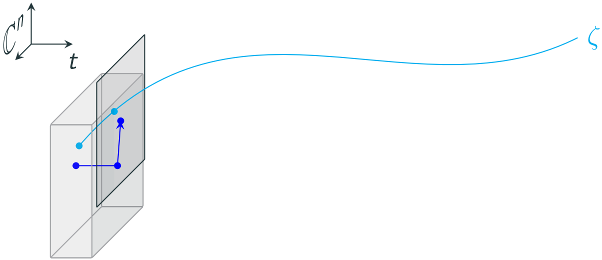


def `track`(F, z)**:**

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

Certified corrector-predictor loop

Recall: for all $t \in [0, 1]$, $F_t(\zeta_t) = 0$

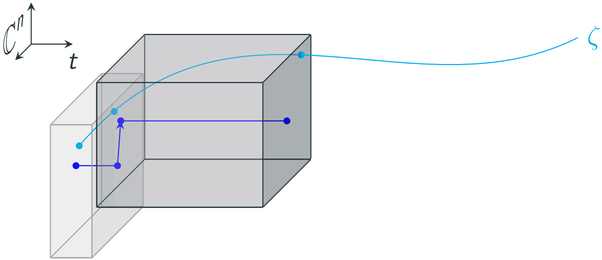


def `track`(F, z)**:**

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

Certified corrector-predictor loop

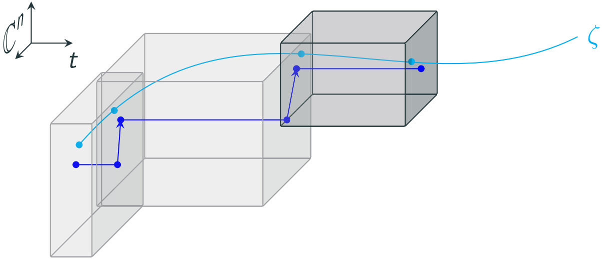
Recall: for all $t \in [0, 1]$, $F_t(\zeta_t) = 0$




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

Certified corrector-predictor loop

Recall: for all $t \in [0,1]$, $F_t(\zeta_t) = 0$



def `track`(F, z):

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

Interval arithmetic

Problem

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

Sufficient solution

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

$$x + y \in \mathrm{A} \boxplus \mathrm{B}, \; xy \in \mathrm{A} \boxtimes \mathrm{B}$$

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

! This is only a sufficient condition

Interval arithmetic

Rational enpoints interval arithmetic

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

Interval arithmetic

Rational enpoints interval arithmetic

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

$f = x^2 - x + 2$, $I = [0, 1]$

- If we decompose f as $(x \cdot x - x) + 2$, we get $[1, 3]$.
- If we decompose f as $x \cdot (x - 1) + 2$, we get $[1, 2]$.
- Actually, $f([0, 1]) = [1.75, 2]$.

Interval arithmetic

Rational enpoints interval arithmetic

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

$f = x^2 - x + 2$, $I = [0, 1]$

- If we decompose f as $(x \cdot x - x) + 2$, we get $[1, 3]$.
- If we decompose f as $x \cdot (x - 1) + 2$, we get $[1, 2]$.
- Actually, $f([0, 1]) = [1.75, 2]$.

! Coefficient swell

✔ Use double endpoints + correct roundings

✔ Arb: variant where intervals are of the form $[x \pm r]$ and x has arbitrary precision.

Krawczyk's method

Root isolation criterion [Krawczyk, 1969], [Moore, 1977], [Rump, 1983]

- $f : \mathbb{C}^n \to \mathbb{C}^n$ polynomial, $\rho \in (0, 1)$,
- $z \in \mathbb{C}^n$, $r \in \mathbb{R}_{>0}$, $A \in \mathbb{C}^{n \times n}$

such that for all $u, v \in B$ (where B is the ball of center 0 and radius r for $\| \cdot \|_\infty$),

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

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

Krawczyk's method

Root isolation criterion [Krawczyk, 1969], [Moore, 1977], [Rump, 1983]

- $f : \mathbb{C}^n \to \mathbb{C}^n$ polynomial, $\rho \in (0, 1)$,
- $z \in \mathbb{C}^n$, $r \in \mathbb{R}_{>0}$, $A \in \mathbb{C}^{n \times n}$

Let B be the ball of center 0 and radius r for $\| \cdot \|_\infty$. Assume that

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

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

Krawczyk's method

Root isolation criterion [Krawczyk, 1969], [Moore, 1977], [Rump, 1983]

- $f : \mathbb{C}^n \to \mathbb{C}^n$ polynomial, $\rho \in (0, 1)$,
- $z \in \mathbb{C}^n$, $r \in \mathbb{R}_{>0}$, $A \in \mathbb{C}^{n\times n}$

Let B be the ball of center 0 and radius r for $\|\cdot\|_\infty$. Assume that

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

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, r, A) which satisfies Moore's criterion.

Adaptive precision

Writting the algorithm in an idealized setup

- ➕ Easier termination proofs
- ➖ Cannot implement the theory, termination is not ensured in practice...

Adaptive precision

Writting the algorithm in an idealized setup

- ➕ Easier termination proofs
- ➖ Cannot implement the theory, termination is not ensured in practice...

The model we chose (also Arb's model)

- 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

Pros

- ➕ Algorithms written in this model can be implemented
- ⚠ Termination: careful precision management in theory
- ➕ **Precision decreases do not hinder correction**

Adaptive precision

Writting the algorithm in an idealized setup

- (+) Easier termination proofs
- (−) Cannot implement the theory, termination is not ensured in practice...

The model we chose (also Arb's model)

- 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

Pros

- (+) Algorithms written in this model can be implemented
- (!) Termination: careful precision management in theory
- (+) **Precision decreases do not hinder correction**

In practice we use Arb and decrease precision by 1 bit at each iteration of the main loop.

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):
    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):
    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, r, A) where $z \in \mathbb{C}^n$, $r \in \mathbb{R}$, $A \in \mathbb{C}^{n \times n}$. 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.

Benchmarks

			HomotopyContinuation.jl			algpath		
name	dim.	max deg	time (s)	fail.	max.	time (s)	prec.	max.
dense	1	1000	**6.8**		100	**12 min**	59	17 k
dense	1	2000	**26**	3	79	**1 h**	62	69 k
katsura	21	2	**4 h**		468	**60 h**	65	12 k
resultants	3	16	**5.6**		128	**92**	58	1857
resultants	2	40		200		**185**	69	1414
structured *	3	10	**3.0**		118	**1.5**	53	313
structured *	3	20	**3.0**	12	164	**4.2**	56	634
structured *	3	30	**2.9**	92	133	**24**	71	818

Figure 1: Total degree homotopy benchmarks. A * means that only 100 random roots were tracked.

[2] Breiding, P., Timme, S. HomotopyContinuation.jl: A Package for Homotopy Continuation in Julia.

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** following [Lambov, 2008],
- NEW! **adaptive precision** using **Arb**[2],
- NEW! **mixed precision** between double precision and Arb **without overhead**.

Todos

- Interface with Sage or Julia
- Avx512 ?

[2] F. Johansson. "Arb: efficient arbitrary-precision midpoint-radius interval arithmetic"

Test data

We tested systems of the form $g_t(z) = tf^{\odot}(z) + (1-t)f^{\triangleright}(z)$ ($f^{\triangleright}$ is the start system, $f^{\odot}$ is the target system).

Target systems

- Dense: $f_i^{\odot}$'s of given degree with random coefficients
- Structured: $f_i^{\odot}$'s of the form $\pm 1 + \sum_{i=1}^{\ell} \left(\sum_{j=1}^{n} a_{i,j} z_j \right)^d$, $a_{i,j} \in_R \{-1, 0, 1\}$
- Resultants: pick $h_1, h_2 \in \mathbb{C}[z_1, \cdots, z_n][y]$, compute their resultant $h \in \mathbb{C}[z_1, \cdots, z_n]$ and fill with random dense polynomials
- Katsura family (sparse - high dimension - low degree)

Start systems

- Total degree homotopies: $f_i^{\triangleright}$'s of the form $\gamma_i(z_i^{d_i} - 1)$, $\gamma_i \in_R \mathbb{C}$, $d_i = \deg f_i^{\odot}$

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.

Beltrán, C., & Leykin, A. (2012). Certified Numerical Homotopy Tracking. *Experimental Mathematics*, *21*(1), 69–83.

Beltrán, C., & Leykin, A. (2013). Robust Certified Numerical Homotopy Tracking. *Found Comput Math*, *13*(2), 253–295.

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.

Duff, T., & Lee, K. (2024). Certified homotopy tracking using the Krawczyk method. *Proc. ISSAC 2024*, 274–282.

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

Kranich, S. (2016). *An epsilon-delta bound for plane algebraic curves and its use for certified homotopy continuation of systems of plane algebraic curves*. arXiv: 1505.03432 [math].

Krawczyk, R. (1969). Newton-Algorithmen zur Bestimmung von Nullstellen mit Fehlerschranken. *Computing*, *4*(3), 187–201.

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.

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.

References ii

Moore, R. E. (1977). A Test for Existence of Solutions to Nonlinear Systems. *SIAM Journal on Numerical Analysis*, *14*(4), 611–615. Retrieved February 19, 2024, from https://www.jstor.org/stable/2156481

Rump, S. M. (1983). SOLVING ALGEBRAIC PROBLEMS WITH HIGH ACCURACY. In U. W. Kulisch & W. L. Miranker (Eds.), *A New Approach to Scientific Computation* (pp. 51–120). Academic Press.

van der Hoeven, J. (2015). *Reliable homotopy continuation* (Research Report). LIX, Ecole polytechnique. Retrieved February 19, 2024, from https://hal.science/hal-00589948

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

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