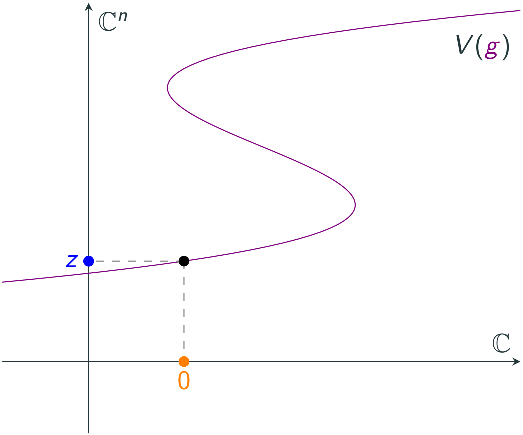
Validated numerics for algebraic path tracking

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

ISSAC 2024
July 19, 2024 | Raleigh, NC, USA

Setup

- Let $g : \mathbb{C} \times \mathbb{C}^n \to \mathbb{C}^n$ be a polynomial map.
- Notation: $g_t : \mathbb{C}^n \to \mathbb{C}^n$ defined by $g_t(z) = g(t, z)$.
- Let $z \in \mathbb{C}^n$ such that $g_0(z) = 0$.



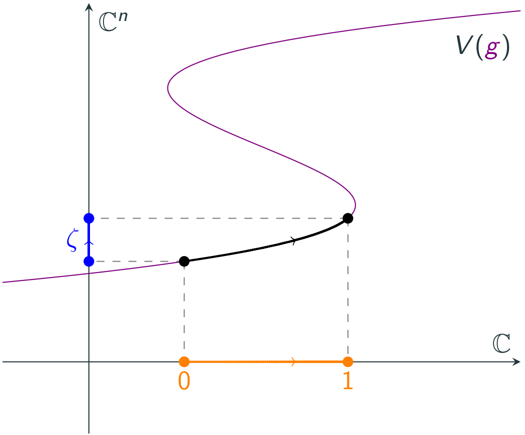
Introduction

Setup

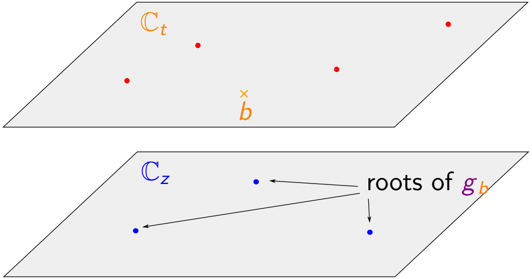
- Let $g : \mathbb{C} \times \mathbb{C}^n \to \mathbb{C}^n$ be a polynomial map.
- Notation: $g_t : \mathbb{C}^n \to \mathbb{C}^n$ defined by $g_t(z) = g(t, z)$.
- Let $z \in \mathbb{C}^n$ such that $g_0(z) = 0$.

$\rightsquigarrow$ Moving the parameter from 0 to 1 induces $\zeta : [0, 1] \to \mathbb{C}^n$ s.t. $\zeta(0) = z$ and $g_t(\zeta(t)) = 0$.

- Goal: "Track" ζ, with some topological guarantees.



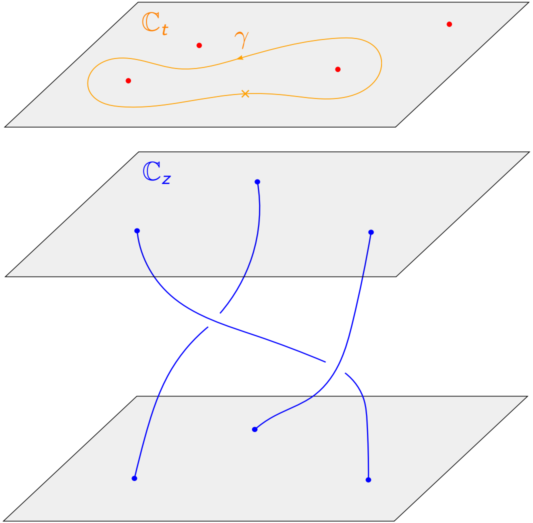
Motivation: braid computations



Setup

- Let $g \in \mathbb{C}[t, z]$,
- let $b \in \mathbb{C} \backslash \Sigma$ be a base point,

Motivation: braid computations



Setup

- Let $g \in \mathbb{C}[t, z]$,
- 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 g_t when t moves along γ defines a braid.

Algorithmic goal

Input: g, γ (p.w. linear)

Output: the associated braid

Tool: certified path tracking

Previous work

Noncertified path trackers

- PHCpack by Verschelde (1999)
- Bertini by Bates, Hauenstein, Sommese, 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 (2015)
- 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*

Contributions

- Specify an *algorithm* implementing the Krawczyk + Taylor approach.
- Prove *termination*.
- In which model?
  - Exact arithmetic is not realistic.
  - We can't prove anything with 64-bits floating point numbers.
  - We want an adaptive precision model (as implemented by MPFI or Arb).
  - $\rightsquigarrow$ We have to recognize when the working precision may not be enough.
- How good is the Krawczyk + Taylor approach?
  - $\rightsquigarrow$ Competitive Rust implementation

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**, denoted by $\square f$), 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

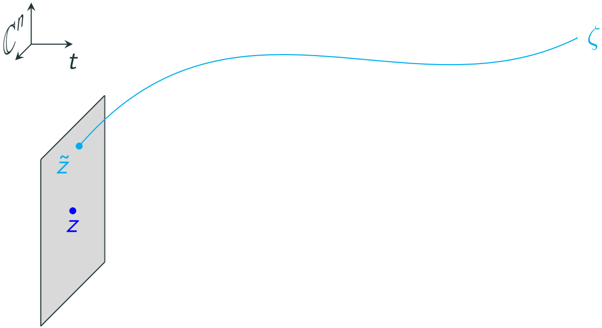
- Interval endpoints : $\{m2^e \in \mathbb{R} : m, e \in \mathbb{Z}\}$,
- $[a, b] \boxplus_u [c, d] \subseteq [a + c - Mu, b + d + Mu]$,
- $[a, b] \boxtimes_u [c, d] \subseteq [\min\{ac, ad, bc, bd\} - M^2 u, \max\{ac, ad, bc, bd\} + M^2 u]$,

for all $M \geq 1$, all $a, b, c, d \in [-M, M]$, and $u \in (0, 1)$ is the unit roundoff that should be specified at each operation.

Pros and cons

- ✔ good theoritical properties as $u \to 0$
- ✔ fast when we can maintain low precision
- implemented by MPFI and Arb
- compatible with IEEE-754 floating point arithmetic, when $u > 2^{-53}$
- when implemented with double precision only, a computation is guaranteed to terminate or fail with a precision error, it cannot hang

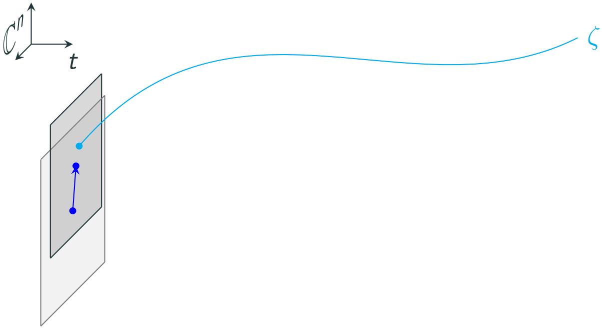
The folklore meta-algorithm



"Algorithm"

def $\mathit{track}(g, Z)$**:**
  $t \leftarrow 0$
  $L \leftarrow []$
  while $t < 1$**:**
    $Z \leftarrow \mathit{refine}(g_t, Z)$
    pred $\leftarrow$ a predictor
    $\delta \leftarrow \mathit{validate}(g, t, Z, \text{pred})$
    $t \leftarrow t + \delta$
    append (t, Z) to L
  return L

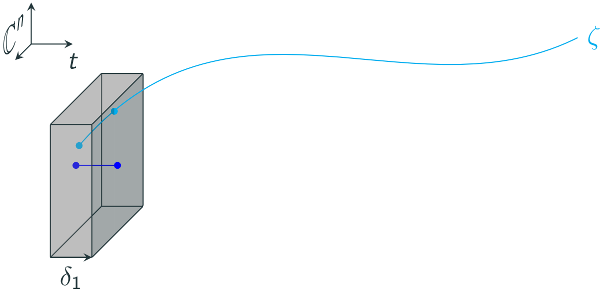
The folklore meta-algorithm



"Algorithm"

def *track*(g, Z)**:**
    $t \leftarrow 0$
    $L \leftarrow [\,]$
    while $t < 1$**:**
        $Z \leftarrow \mathit{refine}(g_t, Z)$
        pred $\leftarrow$ a predictor
        $\delta \leftarrow \mathit{validate}(g, t, Z, \mathrm{pred})$
        $t \leftarrow t + \delta$
        append (t, Z) to L
    return L

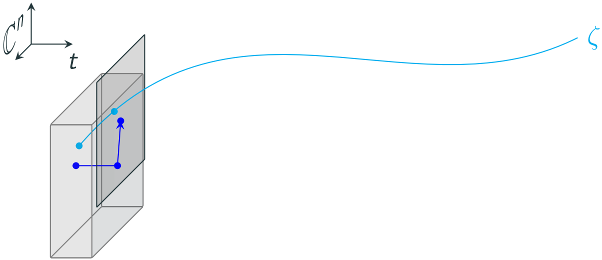
The folklore meta-algorithm



“Algorithm”

```
def track(g, Z):
    t ← 0
    L ← []
    while t < 1:
        Z ← refine(g_t, Z)
        pred ← a predictor
        δ ← validate(g, t, Z, pred)
        t ← t + δ
        append (t, Z) to L
    return L
```

The folklore meta-algorithm



“Algorithm”

def $\mathit{track}(g, Z)$**:**

  $t \leftarrow 0$

  $L \leftarrow []$

  while $t < 1$**:**

    $Z \leftarrow \mathit{refine}(g_t, Z)$

    pred $\leftarrow$ a predictor

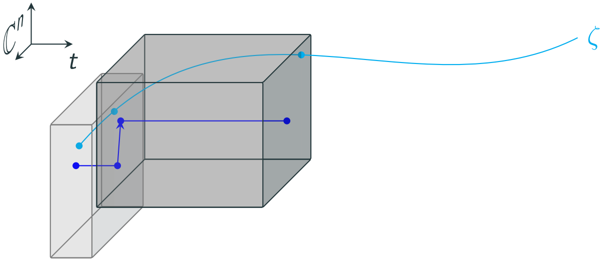
    $\delta \leftarrow \mathit{validate}(g, t, Z, \text{pred})$

    $t \leftarrow t + \delta$

    append (t, Z) to L

  return L

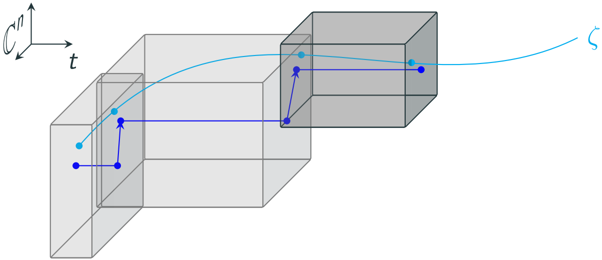
The folklore meta-algorithm



"Algorithm"

def *track*(g, Z)**:**
    $t \leftarrow 0$
    $L \leftarrow []$
    while $t < 1$**:**
        $Z \leftarrow \mathit{refine}(g_t, Z)$
        pred $\leftarrow$ a predictor
        $\delta \leftarrow \mathit{validate}(g, t, Z, \text{pred})$
        $t \leftarrow t + \delta$
        append (t, Z) to L
    return L

The folklore meta-algorithm



“Algorithm”

def $track(g, Z)$**:**
  $t \leftarrow 0$
  $L \leftarrow []$
  while $t < 1$**:**
    $Z \leftarrow refine(g_t, Z)$
    pred $\leftarrow$ a predictor
    $\delta \leftarrow validate(g, t, Z, \text{pred})$
    $t \leftarrow t + \delta$
    append (t, Z) 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 $u, v \in B$,

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

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,

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

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,

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

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

Refinement of Moore boxes

Algorithm

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

  $U \leftarrow A$; $B \leftarrow 2B$

  while not $-A \cdot \square f(z) + [I - A \cdot \square Jf(z+B)]\, B \subseteq \frac{1}{8} B$

    if $-U \cdot \square f(z) \subseteq \frac{1}{512} B$: # left term is small

      $B \leftarrow \frac{1}{2} B$

    else: # left term is big

      $z \leftarrow z - U f(z)$

    $A \leftarrow Jf(z)^{-1}$ # 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

Algorithm

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

    $U \leftarrow A$; $B \leftarrow 2B$; shrink_cnt $\leftarrow 0$

    while not $-A \cdot \square f(z) + [I - A \cdot \square Jf(z+B)]\, B \subseteq \frac{1}{8} B$

        if $-U \cdot \square f(z) \subseteq \frac{1}{512} B$: # left term is small

            $B \leftarrow \frac{1}{2} B$; shrink_cnt $\leftarrow$ shrink_cnt $+ 1$

            if shrink_cnt > 8:

                double working precision

        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

Algorithm

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

    $U \leftarrow A$; $B \leftarrow 2B$; shrink_cnt $\leftarrow 0$

    while not $-A \cdot \square f(z) + [I - A \cdot \square Jf(z+B)]\, B \subseteq \frac{1}{8} B$

        if $-U \cdot \square f(z) \subseteq \frac{1}{512} B$: # left term is small

            $B \leftarrow \frac{1}{2} B$; shrink_cnt $\leftarrow$ shrink_cnt $+ 1$

            if shrink_cnt > 8:

                double working precision

        else: # left term is big

            $\delta \leftarrow U \cdot \square f(z)$

            if $\mathit{width}(z - \delta) > \frac{1}{40} \|\delta\|_\square$:

                double working precision

            else:

                $z \leftarrow \mathit{mid}(z - \delta)$

        $A \leftarrow Jf(z)^{-1}$ # 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

Algorithm

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

    $U \leftarrow A$; $B \leftarrow 2B$; shrink_cnt $\leftarrow 0$

    while not $-A \cdot \Box f(z) + [I - A \cdot \Box Jf(z+B)]\, B \subseteq \frac{1}{8} B$

        if $-U \cdot \Box f(z) \subseteq \frac{1}{512} B$: # left term is small

            $B \leftarrow \frac{1}{2} B$; shrink_cnt $\leftarrow$ shrink_cnt $+ 1$

            if shrink_cnt > 8:

                double working precision

        else: # left term is big

            $\delta \leftarrow U \cdot \Box f(z)$

            if $width(z - \delta) > \frac{1}{40} \|\delta\|_{\Box}$:

                double working precision

            else:

                $z \leftarrow mid(z - \delta)$

        $A \leftarrow Jf(z)^{-1}$ # 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.

Proposition

refine terminates and is correct.

Step validation

Input

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

Output

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

Proposition

validate terminates and is correct.

Algorithm

def $\mathit{validate}(g, t, \delta_{\mathrm{hint}}, z, B, A)$**:**

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

 while $-A \cdot \Box g_T(z) + [I - A \cdot \Box J g_T(z + B)]\, B \not\subseteq \frac{7}{8} B$**:**

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

  if $\delta < u$**:**

   double working precision

 return δ

Step validation

Input

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

Output

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

Proposition

validate terminates and is correct.

Algorithm

def $\mathit{validate}(g, t, \delta_{\mathrm{hint}}, z, B, A)$**:**

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

  while $-A \cdot \Box g_T(z) + [I - A \cdot \Box J g_T(z + B)]\, B \not\subseteq \frac{7}{8} B$**:**

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

    if $\delta < u$**:**

      double working precision

  return δ

Remark

It is possible to modify this algorithm to validate along a predictor. It requires the use of Taylor models.

Implementation

- ✔ Rust implementation.
- ✔ Available at https://gitlab.inria.fr/numag/algpath.
- ✘ Double precision only, abort instead of raising precision.
- ✔ SIMD interval arithmetic, following Lambov (2008).
- ✔ Hermite's cubic predictor.
- ✔ Benchmarked against HomotopyContinuation.jl and NAG for Macaulay2.

We have benchmarks, but caveat

- They may not be relevant for your application.
- Timings are difficult to get consistent, especially with Julia.
- Large variability of the metrics.

So, is it fast?

Total time

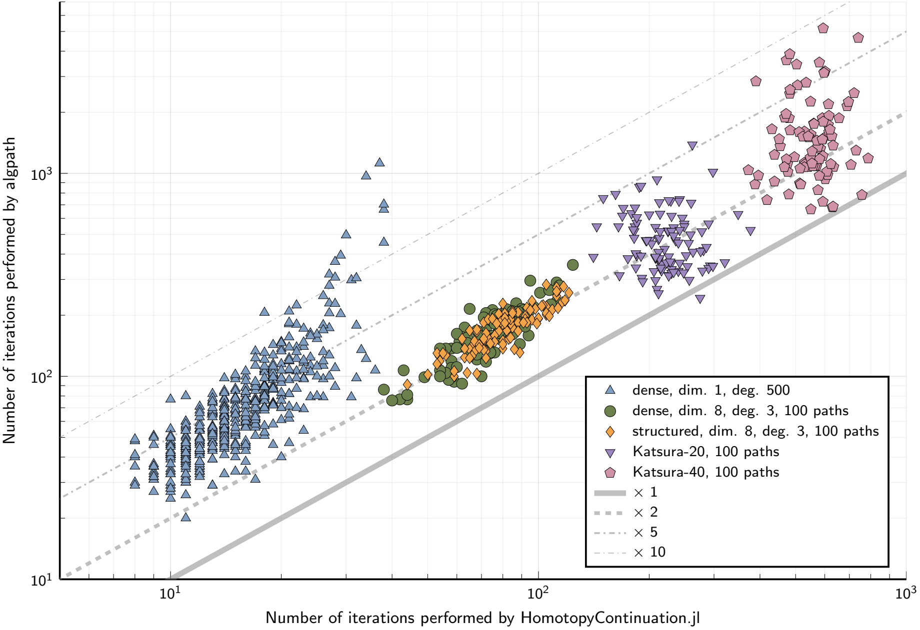
- 1-30$\times$ slower than HC.jl, lot of variability
- orders of magnitude faster than M2

Iterations/sec. (measure the efficiency *refine* and *validate*)

- ! 2$\times$ slower than M2 (I have been surprised by the efficiency of M2!)
- 5-10$\times$ slower than HC.jl

Total number of iterations (measure the theoretical merit)

- ! 1-5$\times$ more iterations than HC.jl + large deviations
- orders of magnitude less than M2
- ! 1-8$\times$ the theoretical minimum of the method (this has a precise meaning)
- Maybe we can gain a factor 2 by imitating Duff and Lee (2024).


Number of iterations performed by algpath
Number of iterations performed by HomotopyContinuation.jl
10^1
10^2
10^3
dense, dim. 1, deg. 500
dense, dim. 8, deg. 3, 100 paths
structured, dim. 8, deg. 3, 100 paths
Katsura-20, 100 paths
Katsura-40, 100 paths
× 1
× 2
× 5
× 10

				circuit size		HomotopyContinuation.jl					algpath					Macaulay2				
name	dim.	deg	# paths	f	df	fail.	med.	max.	ksteps/s	time	fail.	med.	max.	ksteps/s	time	fail.	med.	max.	ksteps/s	time
dense	1	30	30	248	314		10	25	41	2.0		23	372	25	$<$ 0.1		830 k	3478 k	30	18 min
dense	1	40	40	328	416		14	30	45	2.0		34	197	24	$<$ 0.1			$>$ 1 h		
dense	1	50	50	408	520		12	61	37	1.9		30	5567	13	0.7			$>$ 1 h		
dense	1	100	100	808	1054		13	51	23	1.9		38	5289	7.4	1.4			$>$ 1 h		
dense	1	500	500	4008	5466		14	59	3.8	3.9	2	60	1121	2.3	17	500				4.0
dense	1	1000	1000	8008	10952		15	100	1.7	12	35	74	976	1.1	82	1000				29
dense	2	10	100	1016	1280		22	74	33	2.6		53	307	9.2	0.7		33 k	301 k	28	158
dense	2	20	400	3616	4612		25	63	13	3.1		74	401	2.9	12			$>$ 1 h		
dense	2	30	900	7816	9952		24	127	5.8	6.4		85	690	1.4	72			$>$ 1 h		
dense	2	40	1600	13616	17284		25	95	3.4	14		100	998	0.81	268			$>$ 1 h		
dense	2	50	2500	21016	26624		27	84	2.3	33		117	1675	0.53	12 min			$>$ 1 h		
katsura	9	2	256	448	228		82	132	54	4.2		148	286	9.5	4.2		12 k	59 k	18	186
katsura	11	2	1024	606	308		100	179	41	6.7		177	359	6.3	30		21 k	88 k	13	30 min
katsura	16	2	32768	1090	548		153	303	22	235		304	1847	2.7	1 h			$>$ 50 h		
katsura	21	2	1048576	1696	844		209	469	13	4 h	483	427	8798	1.4	101 h			not benchmarked		
katsura *	26	2	100	2430	1202		305	466	6.9	8.8	1	800	2930	0.73	125			$>$ 1 h		
katsura *	31	2	100	3286	1614		382	538	4.9	12	1	852	5021	0.47	219			$>$ 1 h		
katsura *	41	2	100	5376	2618		554	787	2.7	24	9	1371	5182	0.19	13 min			$>$ 1 h		
dense *	4	3	100	1080	1318		39	67	41	2.4		66	127	8.3	0.9		3384	9936	35	10
dense *	6	3	100	4092	5384		54	96	9.0	3.3		112	224	2.3	5.1		11 k	24 k	18	62
dense *	8	3	100	11120	15242		73	124	2.1	6.3		157	354	0.86	19		21 k	74 k	9.5	243
structured *	4	3	100	244	418		40	78	92	4.0		75	199	24	0.4		4531	8925	41	11
structured *	6	3	100	426	778		66	101	59	3.9		130	254	13	1.1		18 k	61 k	23	85
structured *	8	3	100	670	1252		81	121	40	3.9		182	283	7.9	2.3		36 k	97 k	13	305
structured [N]	5	5	1	302	545		42	42	4.9	3.1		99	99	18	$<$ 0.1		252 k	252 k	12	22
structured [N]	10	10	1	1034	2024		53	53	0.18	3.1		123	123	4.9	$<$ 0.1			$>$ 1 h		
structured [N]	15	15	1	2366	5079			$>$ 8 GB				628	628	2.0	0.4			$>$ 8 GB		
structured [N]	20	20	1	3554	6721			$>$ 8 GB				1591	1591	1.2	1.5			$>$ 8 GB		
structured [N]	25	25	1	5466	10541			$>$ 8 GB				1734	1734	0.69	2.9			$>$ 8 GB		
structured [N]	30	30	1	7788	15239			$>$ 8 GB				1989	1989	0.43	5.2			$>$ 8 GB		

			HomotopyContinuation.jl			algpath			Macaulay2		
name	dim.	max deg	med.	ksteps/s	time (s)	med.	ksteps/s	time (s)	med.	ksteps/s	time (s)
dense	1	10	6	30	1.8	11	55	< 0.1	629	55	0.2
dense	1	50	12	37	1.9	30	13	0.7		> 1 h	
dense	2	10	22	33	2.6	53	9.2	0.7	33 k	28	158
dense	2	30	24	5.8	6.4	85	1.4	72		> 1 h	
dense	2	50	27	2.3	33	117	0.53	12 min		> 1 h	
katsura	9	2	82	54	4.2	148	9.5	4.2	12 k	18	186
katsura	11	2	100	41	6.7	177	6.3	30	21 k	13	30 min
katsura	16	2	153	22	235	304	2.7	1 h		> 50 h	
katsura	21	2	209	13	4 h	427	1.4	101 h		not benchmarked	
dense *	8	3	73	2.1	6.3	157	0.86	19	21 k	9.5	243
structured *	8	3	81	40	3.9	182	7.9	2.3	36 k	13	305
structured ^{N}	10	10	53	0.18	3.1	123	4.9	< 0.1		> 1 h	
structured ^{N}	20	20		> 8 GB		1591	1.2	1.5		> 8 GB	
structured ^{N}	30	30		> 8 GB		1989	0.43	5.2		> 8 GB	

References i

- Bates, D. J., Hauenstein, J. D., Sommese, A. J., & Wampler, C. W. (2013). *Numerically solving polynomial systems with Bertini* (Vol. 25). SIAM, Philadelphia, PA.
- Beltrán, C., & Leykin, A. (2012). Certified numerical homotopy tracking. *Exp. Math.*, *21*(1), 69–83. https://doi.org/10/ggck73
- Beltrán, C., & Leykin, A. (2013). Robust certified numerical homotopy tracking. *Found. Comput. Math.*, *13*(2), 253–295. https://doi.org/10/ggck74
- Breiding, P., & Timme, S. (2018). HomotopyContinuation.jl: A package for homotopy continuation in julia. *Int. Congr. Math. Softw.*, 458–465. https://doi.org/10/ggck7q
- Duff, T., & Lee, K. (2024). Certified homotopy tracking using the Krawczyk method.
- Kearfott, R. B., & Xing, Z. (1994). An interval step control for continuation methods. *SIAM J. Numer. Anal.*, *31*(3), 892–914.
- Kranich, S. (2015). An epsilon-delta bound for plane algebraic curves and its use for certified homotopy continuation of systems of plane algebraic curves.
- Krawczyk, R. (1969). Newton-Algorithmen zur Bestimmung von Nullstellen mit Fehlerschranken. *Computing*, *4*(3), 187–201. https://doi.org/10/css7z9

- 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. *Proc. ICMS 2016*, 191–197. https://doi.org/10/grqk32
- Moore, R. E. (1977). A test for existence of solutions to nonlinear systems. *SIAM J. Numer. Anal.*, *14*(4), 611–615. https://doi.org/10/c66n76
- 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. https://doi.org/10/kh8k
- van der Hoeven, J. (2015). Reliable homotopy continuation. https://hal.science/hal-00589948v4
- Verschelde, J. (1999). Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation. *ACM Trans. Math. Softw. TOMS*, *25*(2), 251–276. https://doi.org/10/fncfxj
- 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/ggck7k

Test data

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

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 $\pm 1 + \sum_{i=1}^{5} \left(\sum_{j=1}^{n} a_{i,j} z_j \right)^d$, $a_{i,j} \in_R \{-1, 0, 1\}$
- Katsura family (sparse - high dimension - low degree)

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 $\pm 1 + \sum_{i=1}^{5} \left(\sum_{j=1}^{n} a_{i,j} z_j \right)^d$, $a_{i,j} \in_R \{-1, 0, 1\}$
- Katsura family (sparse - high dimension - low degree)

Start systems

- Total degree homotopies: $f_i^{\triangleright}$'s of the form $\gamma_i(z_i^{d_i} - 1)$, $\gamma_i \in_R \mathbb{C}$, $d_i = \deg f_i^{\odot}$
- Newton homotopies: $f^{\triangleright}(z) = f^{\odot}(z) - f^{\odot}(z_0)$

Refine

Reminder

In a ρ-Moore box (z, r, A), the quasi Newton iteration $\varphi(w) = w - Af(w)$ is a ρ-contraction map, and the limit of iterated compositions of φ gives the associated zero $\tilde{z}$.

Refine

Reminder

In a ρ-Moore box (z, r, A), the quasi Newton iteration $\varphi(w) = w - Af(w)$ is a ρ-contraction map, and the limit of iterated compositions of φ gives the associated zero $\tilde{z}$.

Heuristic

$$-Af(z) + [\mathrm{I}_n - A \cdot Jf(z + B_r)]B_r \subseteq \frac{1}{8} B_r$$

Refine

Reminder

In a ρ-Moore box (z, r, A), the quasi Newton iteration $\varphi(w) = w - Af(w)$ is a ρ-contraction map, and the limit of iterated compositions of φ gives the associated zero $\tilde{z}$.

Heuristic

$$-Af(z) + [\mathrm{I}_n - A \cdot Jf(z + B_r)]B_r \subseteq \frac{1}{8} B_r$$

- We set A to always be $Jf(z)^{-1}$.

Refine

Reminder

In a ρ-Moore box (z, r, A), the quasi Newton iteration $\varphi(w) = w - Af(w)$ is a ρ-contraction map, and the limit of iterated compositions of φ gives the associated zero $\tilde{z}$.

Heuristic

$$\underbrace{-Af(z)}_{\xrightarrow[\text{q.n. iters}]{} 0} + [\mathrm{I}_n - A \cdot Jf(z + B_r)]B_r \subseteq \frac{1}{8} B_r$$

- We set A to always be $Jf(z)^{-1}$.
- By performing quasi Newton iterations, we are able to make the term $-Af(z)$ go to zero.

Refine

Reminder

In a ρ-Moore box (z, r, A), the quasi Newton iteration $\varphi(w) = w - Af(w)$ is a ρ-contraction map, and the limit of iterated compositions of φ gives the associated zero $\tilde{z}$.

Heuristic

$$\underbrace{-Af(z)}_{\xrightarrow[\text{q.n. iters}]{}0} + \underbrace{[\mathrm{I}_n - A \cdot Jf(z + B_r)]}_{\xrightarrow[r\to 0]{}0} B_r \subseteq \frac{1}{8} B_r$$

- We set A to always be $Jf(z)^{-1}$.
- By performing quasi Newton iterations, we are able to make the term $-Af(z)$ go to zero.
- By reducing r, we are able to make the term $[\mathrm{I}_n - A \cdot Jf(z + B_r)]B_r$ fit into any εB_r.

Refine

Reminder

In a ρ-Moore box (z, r, A), the quasi Newton iteration $\varphi(w) = w - Af(w)$ is a ρ-contraction map, and the limit of iterated compositions of φ gives the associated zero $\tilde{z}$.

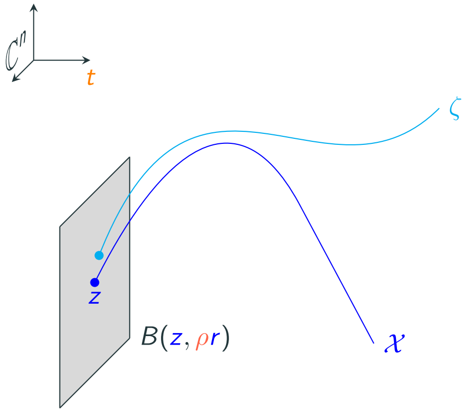
Heuristic

$$\underbrace{-Af(z)}_{\xrightarrow[\text{q.n. iters}]{}0} + \underbrace{[\mathrm{I}_n - A \cdot Jf(z + B_r)]}_{\xrightarrow[r\to 0]{}0} B_r \subseteq \frac{1}{8} B_r$$

- We set A to always be $Jf(z)^{-1}$.
- By performing quasi Newton iterations, we are able to make the term $-Af(z)$ go to zero.
- By reducing r, we are able to make the term $[\mathrm{I}_n - A \cdot Jf(z + B_r)]B_r$ fit into any εB_r.

Idea: a balance between reductions of r and quasi Newton iterations.

Thickening with a predictor

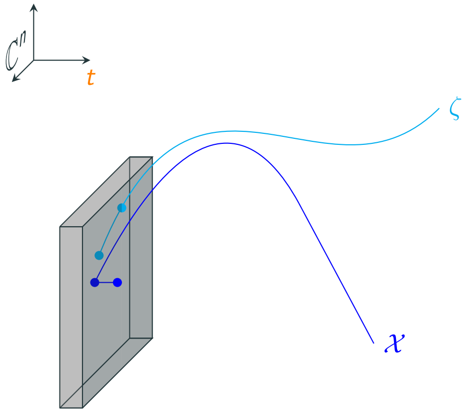


Predictor

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.

Thickening with a predictor



Predictor

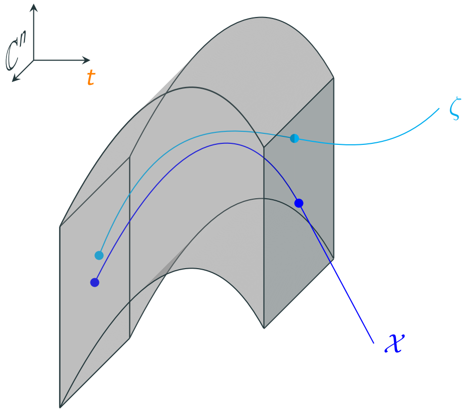
A map $\mathcal{X} : \mathbb{R} \to \mathbb{C}^n$ such that $\mathcal{X}(0) = z$.

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

Certifying the prediction

Pb: check that for all $s \in [0, \delta]$, (z, r, A) is a ρ-Moore box for g_{t+s}.
Soln: try $M(\square g_T, \square J g_T, z, r, A, \rho)$, where $T = [t, t + \delta]$.

Thickening with a predictor



Predictor

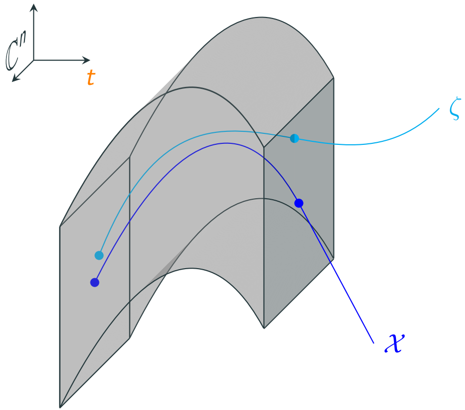
A map $\mathcal{X} : \mathbb{R} \to \mathbb{C}^n$ such that $\mathcal{X}(0) = z$.

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

Certifying the prediction

Pb: check that for all $s \in [0, \delta]$, $(\mathcal{X}(s), r, A)$ is a ρ-Moore box for g_{t+s}.
Soln: try $M(\square g_T, \square J g_T, \square \mathcal{X}([0,\delta]), r, A, \rho)$, where $T = [t, t+\delta]$.

Thickening with a predictor



Predictor

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

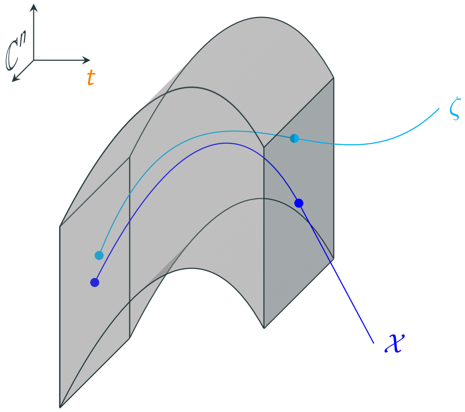
In practice, one should have $\mathcal{X}(s) \approx \zeta(t+s)$ around 0.

Certifying the prediction

Pb: check that for all $s \in [0, \delta]$, $(\mathcal{X}(s), r, A)$ is a ρ-Moore box for g_{t+s}.
Soln: try $M(\square g_T, \square J g_T, \square \mathcal{X}([0,\delta]), r, A, \rho)$, where $T = [t, t+\delta]$.

This is too strong !

Thickening with 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.

Certifying the prediction

Pb: check that for all $s \in [0, \delta]$, $(\mathcal{X}(s), r, A)$ is a ρ-Moore box for g_{t+s}.
Soln: try $M(\square g_T, \square J g_T, \square \mathcal{X}([0, \delta]), r, A, \rho)$, where $T = [t, t+\delta]$.

This is too strong !

Way around the dependency problem: Taylor models !

Taylor models with relative remainder

Definition

- An interval $S \in \square\mathbb{R}$ containing zero,
- a polynomial $P(\eta) = A_0 + A_1\eta + \cdots + A_{d+1}\eta^{d+1}$ where $A_i \in \square\mathbb{C}$.

d is the order of the Taylor model.

Taylor models with relative remainder

Definition

- An interval $S \in \square\mathbb{R}$ containing zero,
- a polynomial $P(\eta) = A_0 + A_1\eta + \cdots + A_{d+1}\eta^{d+1}$ where $A_i \in \square\mathbb{C}$.

d is the order of the Taylor model.

Definition

A Taylor model (S, P) encloses a function $f : \mathbb{R} \to \mathbb{C}$ if for all $s \in S$, there exists $a_i \in A_i$, for all $0 \leq i \leq d + 1$ s.t. $f(s) = a_0 + a_1 s + \cdots + a_{d+1} s^{d+1}$

Taylor models with relative remainder

Definition

- An interval $S \in \square\mathbb{R}$ containing zero,
- a polynomial $P(\eta) = A_0 + A_1\eta + \cdots + A_{d+1}\eta^{d+1}$ where $A_i \in \square\mathbb{C}$.

d is the order of the Taylor model.

Definition

A Taylor model (S, P) encloses a function $f : \mathbb{R} \to \mathbb{C}$ if for all $s \in S$, there exists $a_i \in A_i$, for all $0 \leq i \leq d+1$ s.t. $f(s) = a_0 + a_1 s + \cdots + a_{d+1} s^{d+1}$

Remark

If $J \subseteq S$, then $f(J) \subseteq P(J)$.

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 $(A_d \boxplus (A_{d+1} \boxtimes I))\eta^d$.

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 $(A_d \boxplus (A_{d+1} \boxtimes I)) \eta^d$.

Operations

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

Sum: Component-wise sum using $\boxplus$. Compatible with sums of enclosed functions.

Product: Usual product formula, 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

Recall what we want

(z, r, A) is a $\frac{1}{8}$-Moore box for g_t, $\mathcal{X} : \mathbb{R} \to \mathbb{C}^n$ polynomial s.t. $\mathcal{X}(0) = z$, $\delta > 0$. We want to check that for all $s \in [0, \delta]$,

$$-Ag_{t+s}(\mathcal{X}(s)) + [\mathrm{I}_n - A \cdot Jg_{t+s}(\mathcal{X}(s) + B_r)]B_r \subseteq \frac{7}{8}B_r.$$

Back to our problem

Recall what we want

(z, r, A) is a $\frac{1}{8}$-Moore box for g_t, $\mathcal{X} : \mathbb{R} \to \mathbb{C}^n$ polynomial s.t. $\mathcal{X}(0) = z$, $\delta > 0$. We want to check that for all $s \in [0, \delta]$,

$$-Ag_{t+s}(\mathcal{X}(s)) + [\mathrm{I}_n - A \cdot Jg_{t+s}(\mathcal{X}(s) + B_r)]B_r \subseteq \frac{7}{8}B_r.$$

Solution using Taylor models

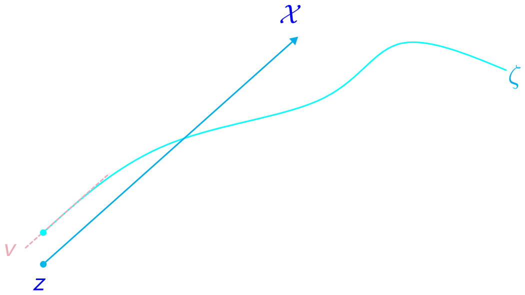
- Compute an order d taylor model $\mathcal{K}$ on $[0, \delta]$ of

$$-Ag_{t+\eta}(\mathcal{X}(\eta)) + [\mathrm{I}_n - A \cdot Jg_{t+\eta}(\mathcal{X}(\eta) + B_r)]B_r.$$

  This is just Taylor model arithmetic !

- Check that $\mathcal{K}([0, \delta]) \subseteq \frac{7}{8}B_r$ (interval arithmetic).

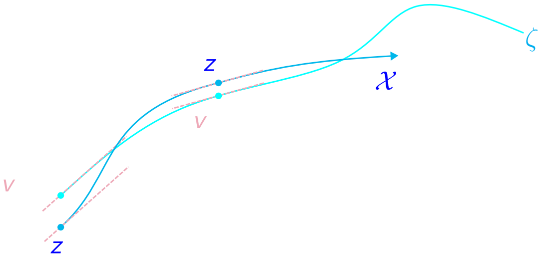
Choosing the right predictor



Tangent predictor

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

Choosing the right predictor



Tangent predictor

Idea: $-A \cdot \frac{\partial}{\partial t} g(t, z)$ 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.