Techniques for real solutions to nonlinear algebraic systems

Danielle Brake

10 September, 2018

ICERM Semester on Nonlinear Algebra
Bootcamp

University of Wisconsin
Eau Claire

The Power of
AND

Tired of those stupid navigation symbols in your beamer slides?

Put this just below the begin-document call:

```
\setbeamertemplate{navigation symbols}{}
```

- i still want to know how to embed emoji in my beamer slides

Differences between complex and real

It's been mentioned already a few times:

- real algebraic geometry is hard
- no statements of genericity
- real stuff is a set of measure zero
- complexity is brutal

Situations

- variety is pure 0-dimensional – “easy”
- a curve – ok, not so hard
- a surface – hmm
- anything higher – uh oh

Outline

- Zero-dim systems
  - solve the system
- Parameterized systems
  - parameter homotopy
- Positive dimensional sets
  - being content with a few points
  - decomposition of all the points
- Other tools
  - regeneration
  - deflation
  - endgames
  - certification
- Software

—

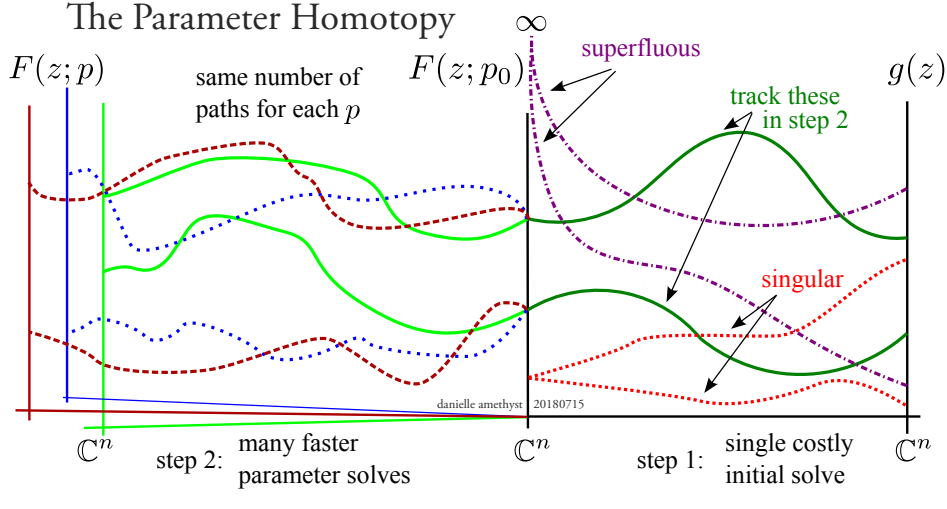
The zero-dimensional method

Just look at your solutions

Outline

- Zero-dim systems
  - solve the system
- Parameterized systems
  - parameter homotopy
- Positive dimensional sets
  - being content with a few points
  - decomposition of all the points
- Other tools
  - regeneration
  - deflation
  - endgames
  - certification
- Software

—



The savings of a parameter homotopy come from not tracking infinite and singular paths repeatedly.

Two parameter homotopies

1. *Cheater's Homotopy*, which moves the entire function. [1]

$$(1-t)\cdot f(x;p)+t\cdot f(x;p^*)$$

2. *Coefficient Parameter Homotopy*, which moves only the coefficients under study. [2]

$$(1-t)\cdot p_i+t\cdot p_i^*$$

for each coefficient p_i that is moving. Or,

$$f(x,(1-t)p+tp^*)$$

Practice the second.

[1] Li, Sauer, Yorke, 1989 [2] Morgan, Sommese. 1989

Example – chemical reaction network

$$
\begin{aligned}
x_0 + x_1 + x_2 + y_1 + y_2 + y_3 + y_4 &= k_1, \\
e + y_1 + y_2 &= k_2, \\
f + y_3 + y_4 &= k_3, \\
-a_1 x_0 e + b_1 y_1 + c_4 y_4 &= 0, \\
c_2 y_2 - a_3 x_2 f + b_3 y_3 &= 0, \\
a_1 x_0 e - (b_1 + c_1) y_1 &= 0, \\
a_2 x_1 e - (b_2 + c_2) y_2 &= 0, \\
a_3 x_2 f - (b_3 + c_3) y3 &= 0, \\
a_4 x_1 f - (b_4 + c_4) y_4 &= 0.
\end{aligned}
$$

The system has 9 variables, 9 equations, and 15 parameters. The parameters in the problem are a_i, b_i, c_i, k_j, for $i \in \{1, 2, 3, 4\}$, and $j \in \{1, 2, 3\}$.

Starting question

"The parameters are all assumed to be positive. For some of them, the system has only one positive solution (i.e., all coordinates are real and positive) and for the others, it has three. How can numerical methods help us determining values, regions, etc., for this to happen?"

Actual solution

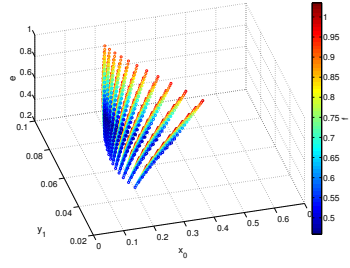
What I *really* wanted to do was to compute the discriminant locus in terms of the parameters.
$\Rightarrow$ ain't gonna happen

Initial answer

To start answering questions about this system:

- Perform grid-sample solves across the unit cube $[0, 1]^{15}$, to find the number of solutions

  Found only points with a single positive real equilibrium.

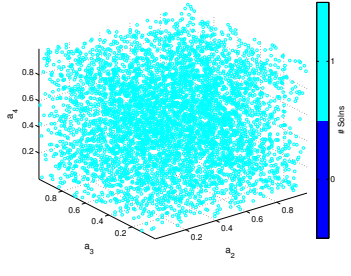


- Perform positive-dimensional numerical irreducible decomposition for random point in parameter space

  Only solutions are 'isolated' solutions – just points.

Monte Carlo sampling

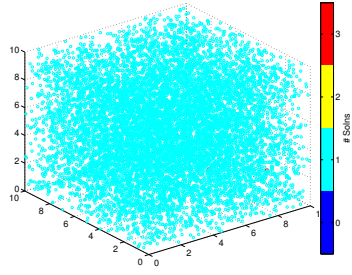
Grid-sampling the unit cube provided only single-equilibria points – maybe Monte Carlo sampling will help? Randomly sampling 10^5 times in all parameters yielded the same picture:



Monte carlo sampling of the unit cube seemed to lend itself to the conjecture that this portion of parameter space is devoid of points at which multiple positive equilibria exist. Perhaps the unit cube has only unique equilibria?

Beyond the Unit Cube

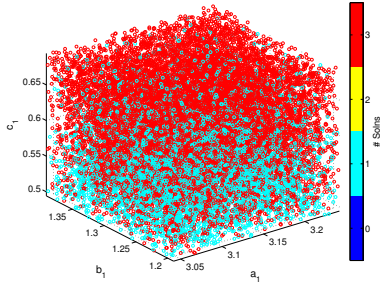
Let's move out a 10 units – the 10-cube, $[0, 10]^{15}$
Randomly sampling can be quite fruitful for system with large numbers of parameters, with estimates on numbers (such as volumes) converging as the square root.



My first pass at sampling, using only 10^4 points, produced *two* parameter points with non-unique equilibria. See the red points?

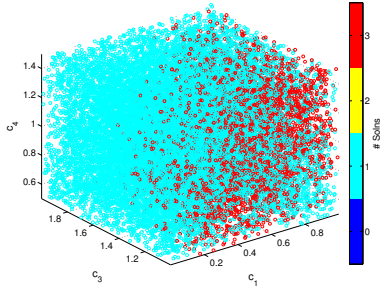
Zooming In

First zoom in around one of the special points.



Definite space containing multiple different numbers of positive real equilibria.

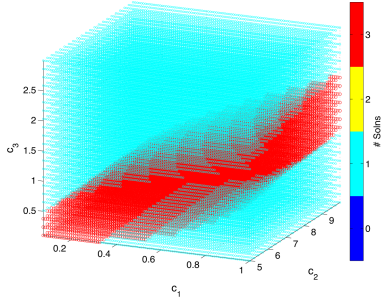
Second zoom, different projection.



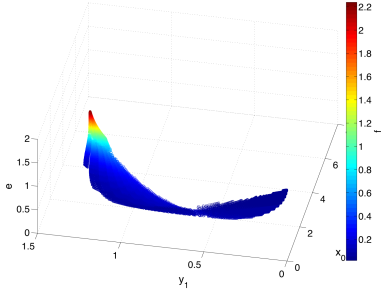
Structure begins to emerge.

Structure Emerges

Having seen a separation of the number of posreal equilibria depending on what looks like c_1, c_3, I chose to run a mesh around the point of interest.



Number of solutions - vs parameter point.



Solution set structure.

Outline

- Zero-dim systems
  - solve the system
- Parameterized systems
  - parameter homotopy
- Positive dimensional sets
  - being content with a few points
  - decomposition of all the points
- Other tools
  - regeneration
  - deflation
  - endgames
  - certification
- Software

—

A point on every connected component

If

- you just care about knowing whether there are any real solutions
- your variety or component is too complicated – dimension too high, etc

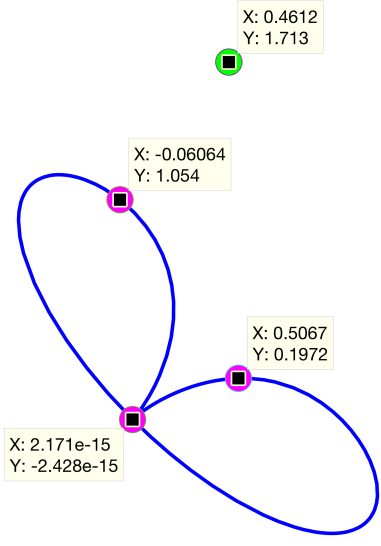
then you should probably settle for computing a point on every component.

A point on every connected component

Solve this homotopy:

$$H(x, \lambda, t) = \begin{bmatrix} f(x) - t\epsilon \\ \lambda_0(x - y) + \sum \lambda_j \nabla f_j(x)^\intercal \\ 1 - \sum \alpha_j \lambda_j \end{bmatrix}$$

- ϵ – a random complex perturbation
- λ_j – synthetic variables, representing nullvector of Jacobian
- α_j – a random complex patch on the λ_j
- y – a real point not on variety
- t – path variable, goes to 0


X: 0.4612
Y: 1.713
X: -0.06064
Y: 1.054
X: 0.5067
Y: 0.1972
X: 2.171e-15
Y: -2.428e-15

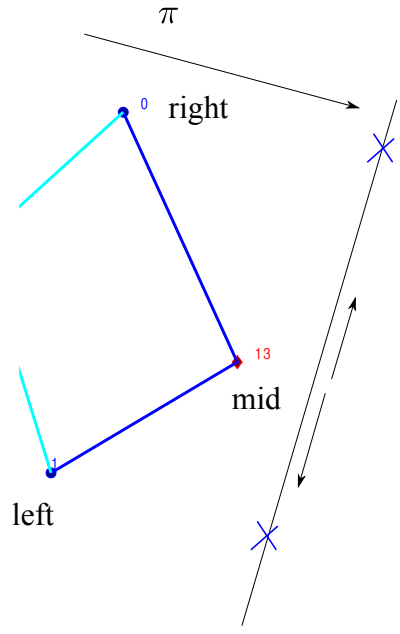
Cylindrical Algebraic Decomposition (CAD)

- Implemented in Mathematica and Maple
- Decomposes space into *cells* on which each polynomial takes constant sign
- Cylindrical in the projection
- Algebraic output
- Doubly exponential complexity
- Exact computations
- Effective for quantifier elimination

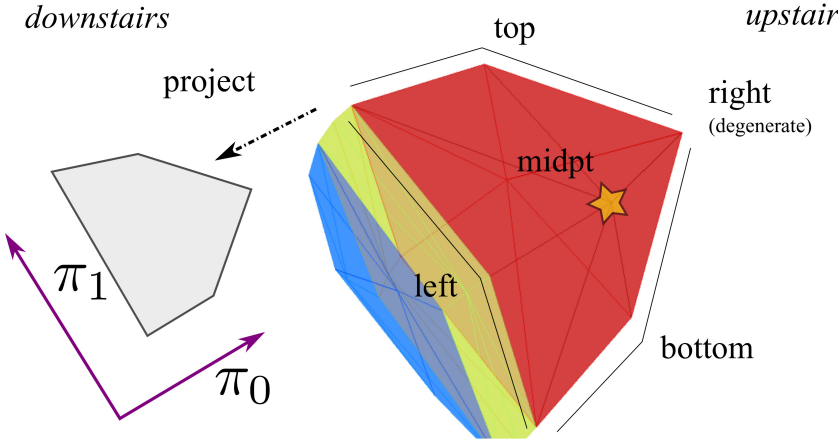
Numerical Real Cellular Decomposition

- Decomposes components into *cells* which are smooth except at their boundaries, and which intersect only at their boundaries.
- Each cell has a smooth point
- Boundaries are cells of dimension 1 lower
- Unknown complexity
- Numerical software with tolerances, etc – an art to operate

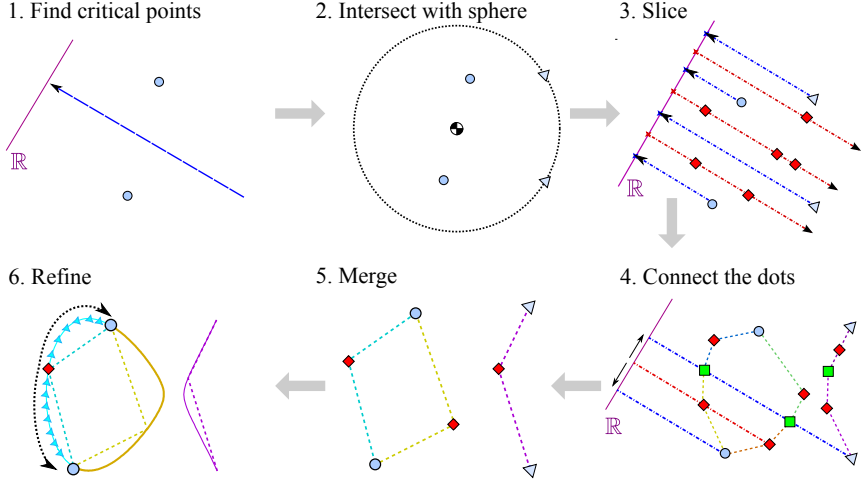
Edge



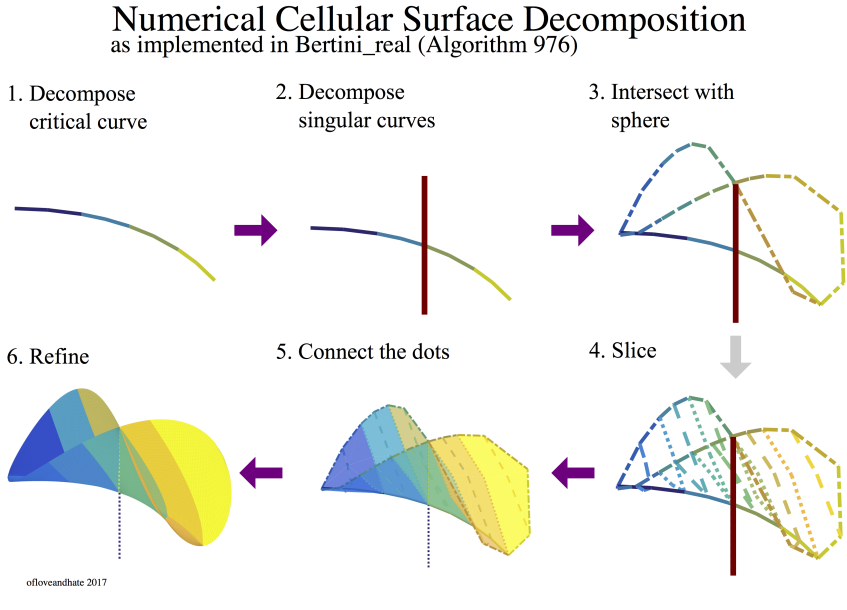
Face


downstairs
project
π_1
π_0
top
upstairs
right
(degenerate)
midpt
left
bottom

Curve decomposition



Surface decomposition



Outline

- Zero-dim systems
  - solve the system
- Parameterized systems
  - parameter homotopy
- Positive dimensional sets
  - being content with a few points
  - decomposition of all the points
- Other tools
  - regeneration
  - deflation
  - endgames
  - certification
- Software

—

Regeneration

Building up to new equations via linears

- this is totally happening on the board

Example – regeneration to find curve critical points (left nullspace method)

Regeneration uses products of linears to build up a start system for a homotopy. We're solving by homotoping as

$$H(x, v; t) = (1-t)\begin{bmatrix} f(x) \\ v^{\mathsf{T}} \cdot \begin{bmatrix} Jf(x) \\ J\pi_1 \end{bmatrix} \\ \text{patch}_v \end{bmatrix} + t\begin{bmatrix} f(x) \\ M_1(v)\prod_{i=1}^{\delta} L_{1,i}(x) \\ \vdots \\ M_N(v)\prod_{i=1}^{\delta} L_{N,i}(x) \\ \text{patch}_v \end{bmatrix} = 0$$

- δ is the maximum degree any polynomial in f.

Deflation

Deflation

- Making the untrackable, trackable.
- Regularizing witness and other points.
- Reduces one solution component at a time.
- Produces a new system for which Newton's method hypothetically converges quadratically.
- We are deflating the multiplicity of the singularity.
- Isosingular sets are closures of sets with the same deflation sequence.

Deflation – nullspace

given $f : \mathbb{C}^N \to \mathbb{C}^n$ with dnull(Jf) at a generic point x, choose a general linear system $\mathcal{L} : \mathbb{C}^N \to \mathbb{C}^d$, and set

$$g_e(x, \eta) = \begin{bmatrix} f(x) \\ Jf(x) \cdot \eta \\ \mathcal{L}(\eta) \end{bmatrix}$$

yields

$$g_e : \mathbb{C}^{2N} \to \mathbb{C}^{2n+d}$$

[Leykin, Verschelde, Zhao '06]

Deflation – determinental

This method of deflations adds no variables, but instead **lots** of functions

$$g_{\text{det}} = \begin{bmatrix} f(x) \\ \det J_{\sigma_1} f(x) \\ \vdots \\ \det J_{\sigma_m} f(x) \end{bmatrix}$$

where $\{\sigma_i\}$ indexes the set of $(n-d+1)$x$(n-d+1)$ submatrices of the Jabobian. d is still dimnull of Jf.

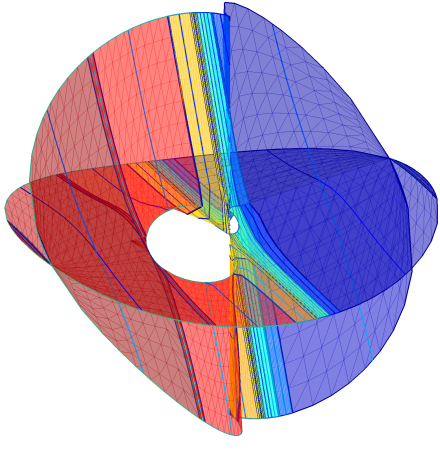
$$g_{\text{det}} : \mathbb{C}^N \to \mathbb{C}^{\text{scary}}$$

[Hauenstein, Wampler '13]

Application of deflation – finding singular curves

1. Compute witness set for critical curve.
2. Separate singular witness points from nonsingular.
   - Nonsingular $\rightarrow$ critical curve.
   - Singular $\rightarrow$ singular curve(s).
3. Separate singular witness points by deflation sequence.
4. Decompose each singular curve.

Example - Solitude



Example - Solitude

Solitude:

$$f(x, y, z) = x^2yz + xy^2 + y^3 + y^3z - x^2z^2$$

$$\mathcal{D}^1 = \begin{bmatrix} x^2yz + xy^2 + y^3 + y^3z - x^2z^2 \\ \\ y^2/4 - (xz^2)/2 + (xyz)/2 \\ (xy)/2 + (x^2z)/4 + (3y^2z)/4 + (3y^2)/4 \\ (x^2y)/4 - (x^2z)/2 + y^3/4 \end{bmatrix}$$

Example - Solitude

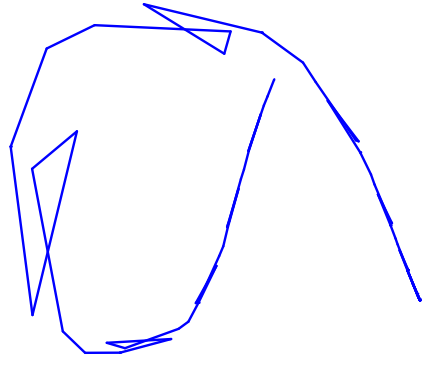
$$\mathcal{D}^2 = \begin{bmatrix} x^2yz + xy^2 + y^3 + y^3z - x^2z^2 \\ \\ y^2/4 - (xz^2)/2 + (xyz)/2 \\ (xy)/2 + (x^2z)/4 + (3y^2z)/4 + (3y^2)/4 \\ (x^2y)/4 - (x^2z)/2 + y^3/4 \\ \\ (y^2z^2)/8 - (x^2z^3)/24 + (y^2z^3)/8 - (y^3z^2)/8 - (y^3z)/8 + y^3/24 + (xy^2z)/24 + (x^2yz^2)/24 \\ (x^2z^3)/12 + (y^3z^2)/24 + (xy^3)/24 - (y^4z)/24 - (xy^2z)/12 - (x^2yz^2)/8 + (x^2y^2z)/24 \\ (x^2y^2)/24 + (xy^3)/8 - y^4/24 - (xy^2z)/4 - (x^2yz)/12 + (xy^3z)/12 - (xy^2z^2)/4 \\ -(x(x^2z^2 - 3y^2z^2 + 2xz^2 + 6yz^2 - 3y^2z + 6yz^3 + y^2 + xyz))/24 \\ (x^3z^2)/24 - (x^2y^2)/48 + y^4/48 + (x^2yz)/12 + (xy^3z)/12 - (xy^2z^2)/8 \\ (xy^3)/12 - (x^2y^2)/16 + (x^3z)/12 + (x^4z)/48 + (y^4z)/16 + y^4/16 + (x^2yz)/4 + (x^2yz^2)/4 \\ (x^3z^2)/24 - (x^2y^2)/16 - (xy^3)/8 + y^4/16 + (xy^2z)/4 + (x^2yz)/6 + (xy^2z^2)/8 \\ -(x(x^2y^2 + 2x^2z^2 + xy^2 - 2y^3z + y^4 - 2x^2yz))/24 \\ -(y(4x^2y^2 + 6x^2y + 4x^3 + x^4 + 3y^4))/48 \\ (y^2z^2)/12 - (x^2z^2)/36 - (xz^2)/36 - (yz^2)/12 + (y^2z)/12 - (yz^3)/12 - y^2/36 - (xyz)/36 \\ (x^2z^2)/24 - (y^2z^2)/24 - (xy^2)/36 + (y^3z)/24 - (x^2yz)/72 + (xyz)/18 \\ (x^2z)/18 - (x^2y)/72 - (xy^2)/12 + (x^3z)/72 + y^3/24 + (xyz^2)/6 - (xy^2z)/24 + (xyz)/6 \\ (x^2z^2)/24 - (y^2z^2)/24 - (xy^2)/36 + (y^3z)/24 - (x^2yz)/72 + (xyz)/18 \\ -(x^2(y^2 - 3yz + 3z^2))/36 \\ -(xy(2x - 6yz + x^2 + 3y^2))/72 \\ (x^2z)/18 - (x^2y)/72 - (xy^2)/12 + (x^3z)/72 + y^3/24 + (xyz^2)/6 - (xy^2z)/24 + (xyz)/6 \\ -(xy(2x - 6yz + x^2 + 3y^2))/72 \\ -(x^2y^2)/24 - (x^2y)/12 - x^3/36 - x^4/144 - y^4/16 - (x^2yz)/12 \end{bmatrix}$$

Endgames

Singular endpoints require special techniques to compute; using regular ODE methods will probably not work.
Choices:

1. Power series – approximate curve with a powerseries, extrapolate
2. Cauchy – walk circles around $t = 0$ in complex space, integrate

a numerical homotopy path



default bertini settings, a path projected onto its real coordinates

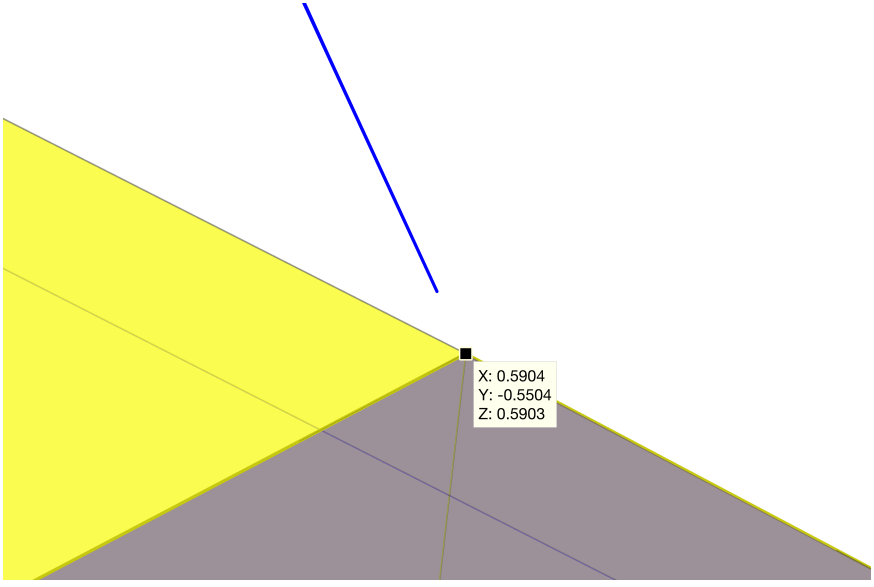
Power Series – procedure

- Fact: the path is a Puiseaux series in t.
- Method: Extrapolate to $t = t_0$ via a polynomial interpolant to the path.
- Tool: Rescale time by c, the cycle number of the path, to get a power series.

$$s = t^{1/c}$$

1. Guess the cycle number
   - 1.1 Approximate the first terms of the power series
   - 1.2 Extrapolate forward to t'
   - 1.3 Track to next time
   - 1.4 Choose c which minimizes error at t'
2. Extrapolate to t_0

Power Series – a picture



see how it doesn't even end at the endpoint? extrapolate!

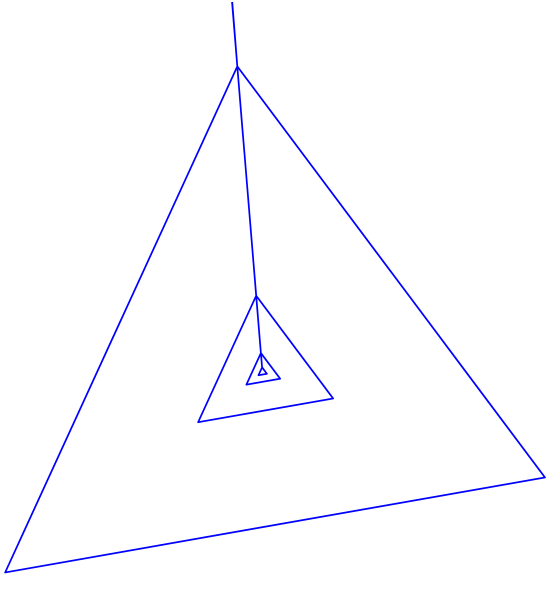
Cauchy – procedure

- Fact: the path is a Puiseaux series in t.
- Method: Extrapolate to $t = t_0$ via a Cauchy integral (just compute the mean!)
- Tool: Walk circles around t_0 to gather numerical data to do Trapezoid Rule quadrature.

$$f(s) = \frac{1}{2\pi i} \int_{\circ} \frac{f(s)}{s - t_0}\, dz$$

1. Guess the cycle number
   - 1.1 Approximate the first terms of the power series
   - 1.2 Extrapolate forward to t'
   - 1.3 Track to next time
   - 1.4 Choose c which minimizes error at t'
2. Extrapolate to t_0

Cauchy – a picture



Certification – general

Let

$$\beta(f,x) = ||x - \text{Newtonstep}(f,x)||$$
$$\gamma(f,x) = sup_{k\leq 2} \left|\left| \frac{Jf(x)^{-1} J^k f(x)}{k!} \right|\right|^{\frac{1}{k-1}}$$
$$\alpha(f,x) = \beta(f,x)\,\gamma(f,x)$$

if

$$\alpha(f,x) < \frac{13 - 3\sqrt{17}}{4} \approx 0.157671$$

then Newton's method will converge to some ξ with

$$f(\xi) = 0 \quad \text{exactly}$$

Certification – real

To check that $x = \bar{x}$ – that x is an approximate solution to a *real* ξ:

1. Compute β, γ, α
2. If $||x - \text{real}(x)|| > 2\beta$, not real
3. If $\alpha < 0.03$ && $||x - \text{real}(x)|| < \frac{1}{20\gamma}$, real
4. take a newton step, goto 1

Outline

Paramotopy

- Command-line software which enables rapid investigations into parametrized polynomial systems.
- Uses *Bertini 1* ($\mathbb{B}^1$) to perform solves, under the hood.
- Numeric-menu driven interface which simplifies and automates Bertini functions.
- Open source, free. `www.paramotopy.com`
- C++, must self-compile. Must also self-compile $\mathbb{B}^1$.

```
*******************************
 Welcome to Paramotopy.
     parametrized system analysis software by
     dani brake and matthew niemerg, with dan bates.
```

Bertini_real

- Command-line MPI-parallel program.
- Uses Bertini 1 as its path tracker.
- C++ code, with options for Matlab or Python for symbolic operations.
- Matlab and Python visualization suites.
- Must self-compile. Must also self-compile $\mathbb{B}^1$. Sorry.

```
BertiniReal(TM) v1.6.0

D.A. Brake with
D.J. Bates, W. Hao, J.D. Hauenstein,
A.J. Sommese, C.W. Wampler

(using GMP v6.1.2, MPFR v3.1.5)


  Library-linked Bertini(TM) v1.6
   (February 27, 2017)

D.J. Bates, J.D. Hauenstein,
A.J. Sommese, C.W. Wampler

(using GMP v6.1.2, MPFR v3.1.5)
```

Alpha certified

Alpha certified – by Jon Hauenstein and Frank Sottile

- Command-line program.
- See manual for input format.
- Certifies polynomial and poly-exponential systems.
- Soft or Hard certification.

```
alphaCertified v1.3.0 (October 16, 2013)
Jonathan D. Hauenstein and Frank Sottile
       GMP v6.1.2 & MPFR v4.0.1
```

Outline

- Zero-dim systems
  - solve the system
- Parameterized systems
  - parameter homotopy
- Positive dimensional sets
  - being content with a few points
  - decomposition of all the points
- Other tools
  - regeneration
  - deflation
  - endgames
  - certification
- Software

—

Thank you for your kind attention!