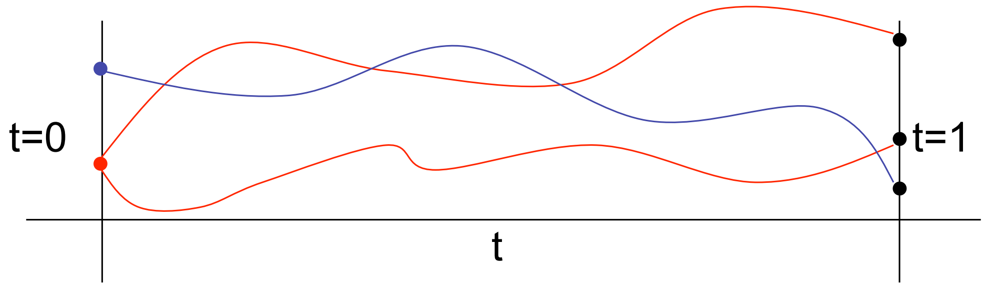


Numerical Algebraic Geometry Boot Camp

Dan Bates
Colorado State University

University of Notre Dame Workshop on…Numerical Algebraic Geometry May 23, 2016

Goals for this boot camp

- Introduce many of the basic
  - structures,
  - methods, and
  - assumptions

  of numerical algebraic geometry (and not the nitty-gritty theory underlying it)

- Show some Bertini Classic I/O

- Give us a **common language** for the next 2.5 days

Game plan

1. Polynomial systems and their solution sets
2. Finding isolated solutions (homotopy continuation)
3. Advanced topics for isolated solutions
4. Finding positive-dimensional solution sets (briefly)

Game plan

1. Polynomial systems and their solution sets

    A. Examples

    B. Intuition from linear algebra

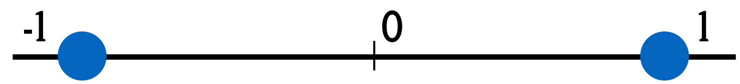
    C. Some words

    D. Bertini's theorem

Polynomial systems and their solution sets

Example 1: $x^2 - 1 = 0$

Two solutions: $x = \pm 1$

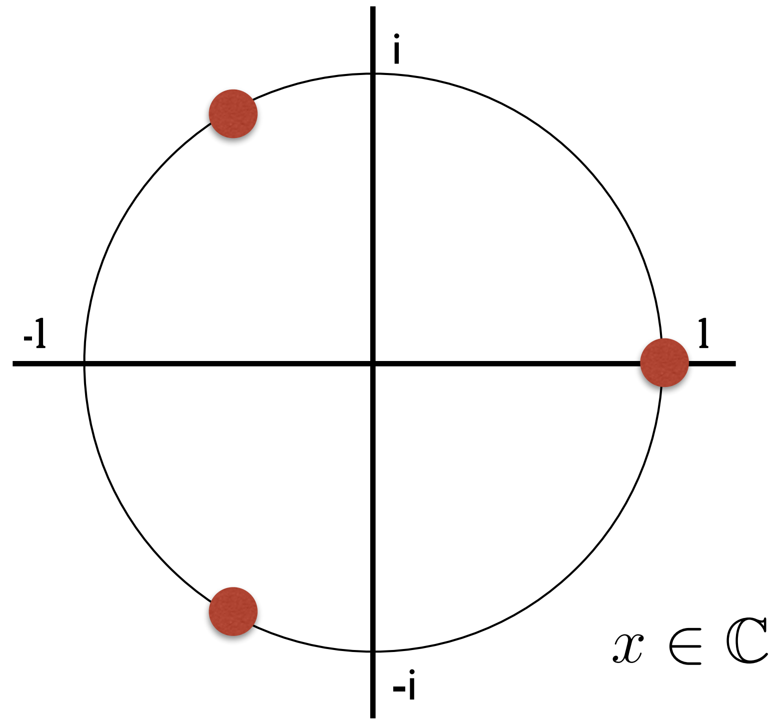


$x \in \mathbb{R}$

Polynomial systems and their solution sets

Example 2: $x^3 - 1 = 0$

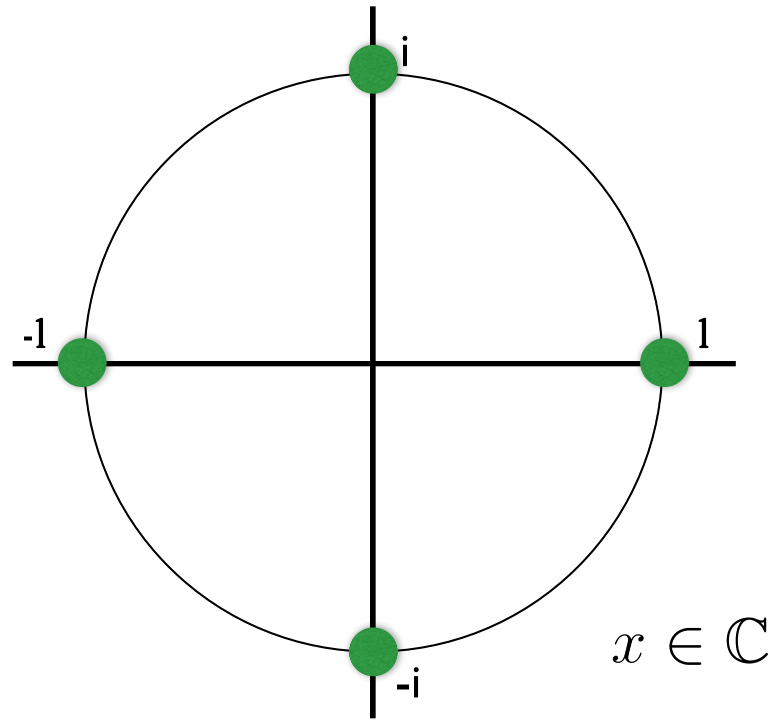
Three solutions (1 real, 2 non-real)



Polynomial systems and their solution sets

Example 3: $x^4 - 1 = 0$

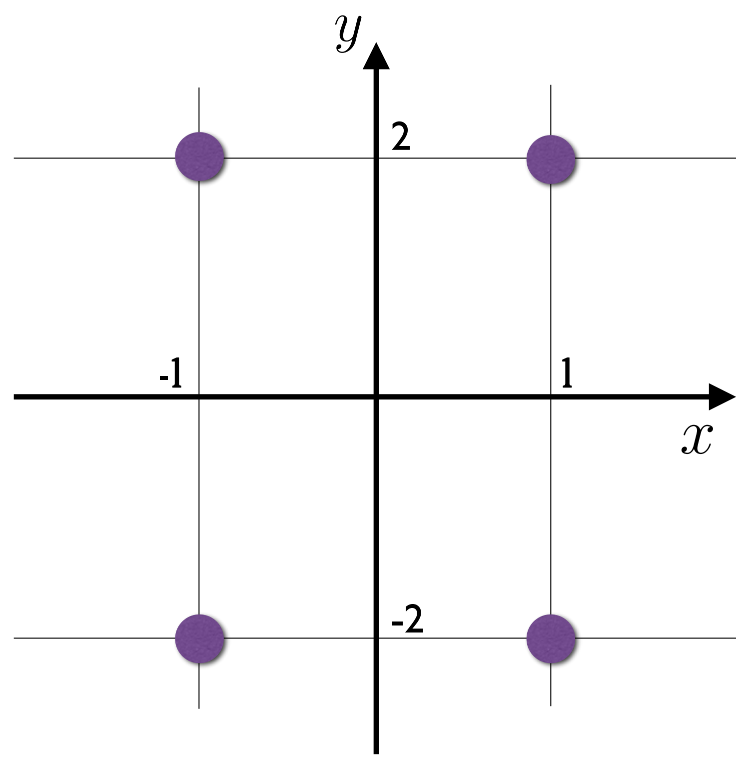
Four solutions (2 real, 2 non-real)



Polynomial systems and their solution sets

Example 4: $f(x,y) = \begin{bmatrix} x^2 - 1 \\ y^2 - 4 \end{bmatrix}$

Four solutions: $(\pm 1, \pm 2)$



Polynomial systems and their solution sets

Example 5: $g(x,y) = \begin{bmatrix} x^2+1 \\ y^2+4 \end{bmatrix}$

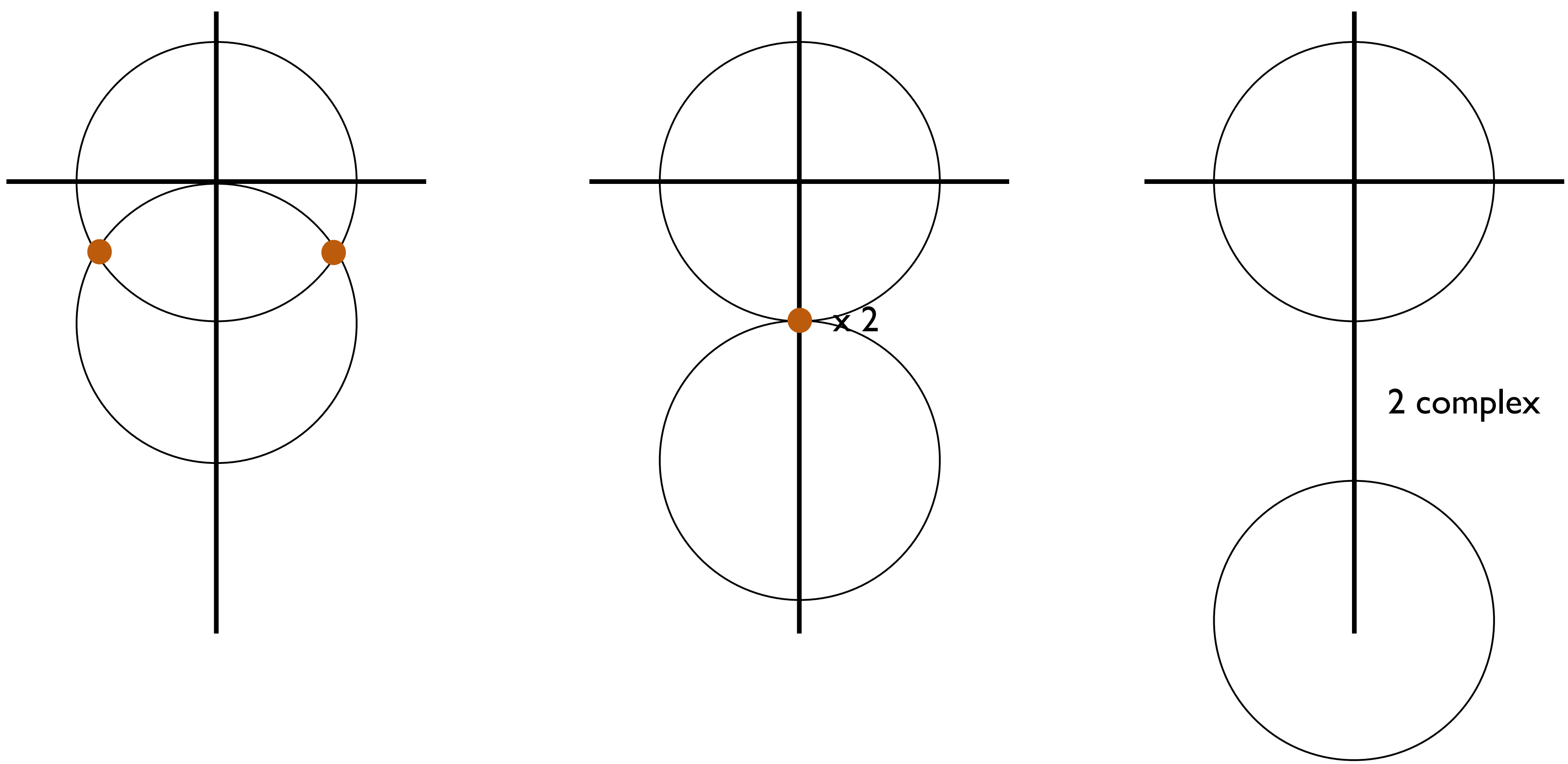
Four solutions: $(\pm i, \pm 2i) \in \mathbb{C}^2$

Visualization harder…

Polynomial systems and their solution sets

Example 6: $$h(x,y) = \begin{bmatrix} x^2 + y^2 - 1 \\ x^2 + (y+c)^2 - 1 \end{bmatrix}$$

$c = 1$ $\qquad$ $c = 2$ $\qquad$ $c = 3$



Polynomial systems and their solution sets

Example 7: $\begin{bmatrix} x(x-1) \\ x(y-1) \end{bmatrix}$

Solutions: $x = 0$ (a line) and $(1,1)$ (a point).

(Two sets, two dimensions!)

Polynomial systems and their solution sets

Example 8:

$$\begin{bmatrix} (y-x^2)(x^2+y^2+z^2-1)(x-2) \\ (z-x^3)(x^2+y^2+z^2-1)(y-2) \\ (z-x^3)(y-x^2)(x^2+y^2+z^2-1)(z-2) \end{bmatrix}$$

Solutions:

Dimension 2: One surface
Dimension 1: Three lines and one cubic curve
Dimension 0: One point

("Illustrative example" from Sommese-Verschelde-Wampler, *SIAM J. Numer. An.* '01)

Polynomial systems and their solution sets

Linear systems vs. polynomial systems

	Linear	Polynomial
# solutions	0 or 1	0 or more
# dimensions	1	1 or more
solution components	point, line, plane, etc.	points, curves, surfaces, etc.
vanishing set for each equation	hyperplane	hypersurface

Polynomial systems and their solution sets

Some words

Solution set: The set of all solutions of a polynomial system f. Also called an algebraic set and sometimes a variety. Sometimes denoted $\mathcal{V}(f)$.

Irreducible component: One "piece" of an algebraic set, e.g., a curve or a point.

Irreducible decomposition: $Z = \mathcal{V}(f) = \bigcup_{i=0}^{D} Z_i = \bigcup_{i=0}^{D} \bigcup_{j \in \Lambda_i} Z_{i,j}$, where

D is the dimension of Z,
i cycles through possible dimensions of irreducible components,
j is an index within dimension i, and the
$Z_{i,j}$ are the irreducible components.

Polynomial systems and their solution sets

Example 8:
$$\begin{bmatrix} (y-x^2)(x^2+y^2+z^2-1)(x-2) \\ (z-x^3)(x^2+y^2+z^2-1)(y-2) \\ (z-x^3)(y-x^2)(x^2+y^2+z^2-1)(z-2) \end{bmatrix}$$

Solutions:

Dimension 2: One surface
Dimension 1: Three lines and one cubic curve
Dimension 0: One point

$$Z = Z_{0,0} \cup Z_{1,0} \cup Z_{1,1} \cup Z_{1,2} \cup Z_{1,3} \cup Z_{2,0}$$

Polynomial systems and their solution sets

Some more words

Regular/nonsingular solution: A solution of a polynomial system f at which the Jacobian matrix (below) has full rank. Other solutions are called singular solutions.

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_N}{\partial x_1} & \cdots & \frac{\partial f_N}{\partial x_N} \end{bmatrix}$$

Example 6: $h(x,y) = \begin{bmatrix} x^2 + y^2 - 1 \\ x^2 + (y+c)^2 - 1 \end{bmatrix} \rightarrow \quad J = \begin{bmatrix} 2x & 2y \\ 2x & 2y + 2c \end{bmatrix}$

$c = 1$ (2 solutions): $J(0.866, -0.5) = \begin{bmatrix} 1.732 & -1 \\ 1.732 & 1 \end{bmatrix}$ (full rank)

$c = 2$ (1 double solution): $J(0, -1) = \begin{bmatrix} 0 & -2 \\ 0 & 2 \end{bmatrix}$ (rank-deficient)

Polynomial systems and their solution sets

Bertini's Theorem

Polynomial systems can have different numbers of function (n) and variables (N).

If N > n (more variables), there will be no isolated solutions (handled later).

If N < n (more equations), we can replace f with N random linear combinations of the polynomials of f. Call the new system $\tilde{f}$. Each solution of f is a solution of $\tilde{f}$, but $\tilde{f}$ might have more solutions (Bertini junk).

Example 9:

$$f(x,y) = \begin{bmatrix} x^2-1 \\ y^2-1 \\ xy-1 \end{bmatrix}$$

(not very random)

$$\begin{bmatrix} f_1+f_2+f_3 \\ f_1+2f_2+f_3 \end{bmatrix}$$

Solutions: $(1,1),(-1,-1)$

$$(1,1),(-1,-1),(2,-1),(-2,1)$$

Game plan

1. Polynomial systems and their solution sets
2. **Finding isolated solutions (homotopy continuation)**
3. Advanced topics for isolated solutions
4. Finding positive-dimensional solution sets (briefly)

Game plan

2. **Finding isolated solutions (homotopy continuation)**

   A. Homotopy continuation in a nutshell

   B. Start systems

   C. Bells & whistles

   D. Endgames

   E. Bertini Classic (1.x)

Finding isolated solutions

Given $f_1, \ldots, f_n \in \mathbb{C}[z_1, \ldots, z_N]$, we want to find all $\widehat{z} \in \mathbb{C}^N$ s.t. $f_i(\widehat{z}) = 0 \ \forall i$.

For each isolated solution, $\widehat{z}$, we aim to compute a numerical approximation $\tilde{z}$ such that $||\tilde{z} - \widehat{z}|| < \mathbf{FinalTol}$.

How do we accomplish this? Homotopy continuation.

Finding isolated solutions

Given polynomial system $f : \mathbb{C}^N \to \mathbb{C}^N$ (the target system) homotopy continuation is a 3-step process:

1. Choose and solve a polynomial system $g : \mathbb{C}^N \to \mathbb{C}^N$ (the start system) based on characteristics of $f(z)$ but relatively easy to solve.

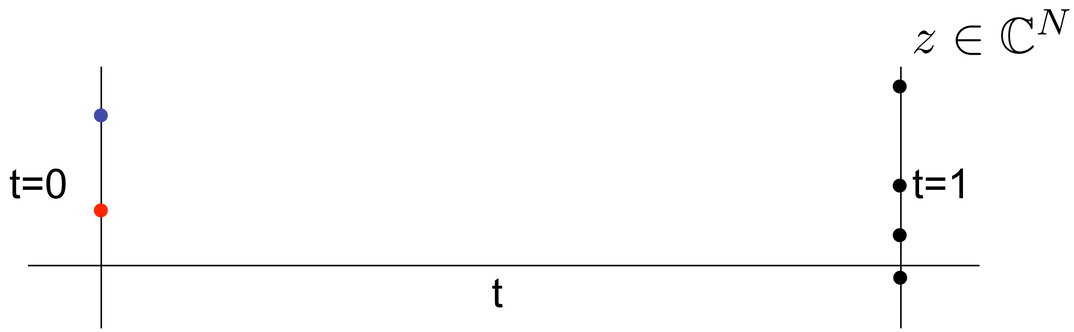
2. Form the homotopy $H : \mathbb{C}^N \times \mathbb{C} \to \mathbb{C}^N$ given by

$$H(z,t) = f(z) \cdot (1-t) + g(z) \cdot t$$

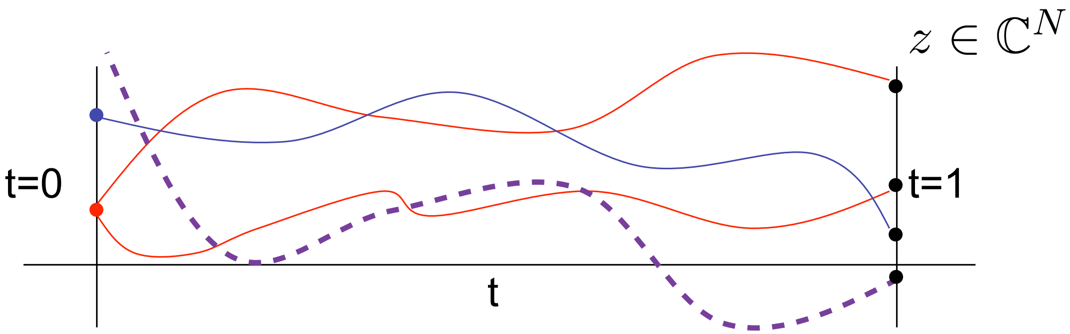
so that $H(z,1) = g(z)$ and $H(z,0) = f(z)$.

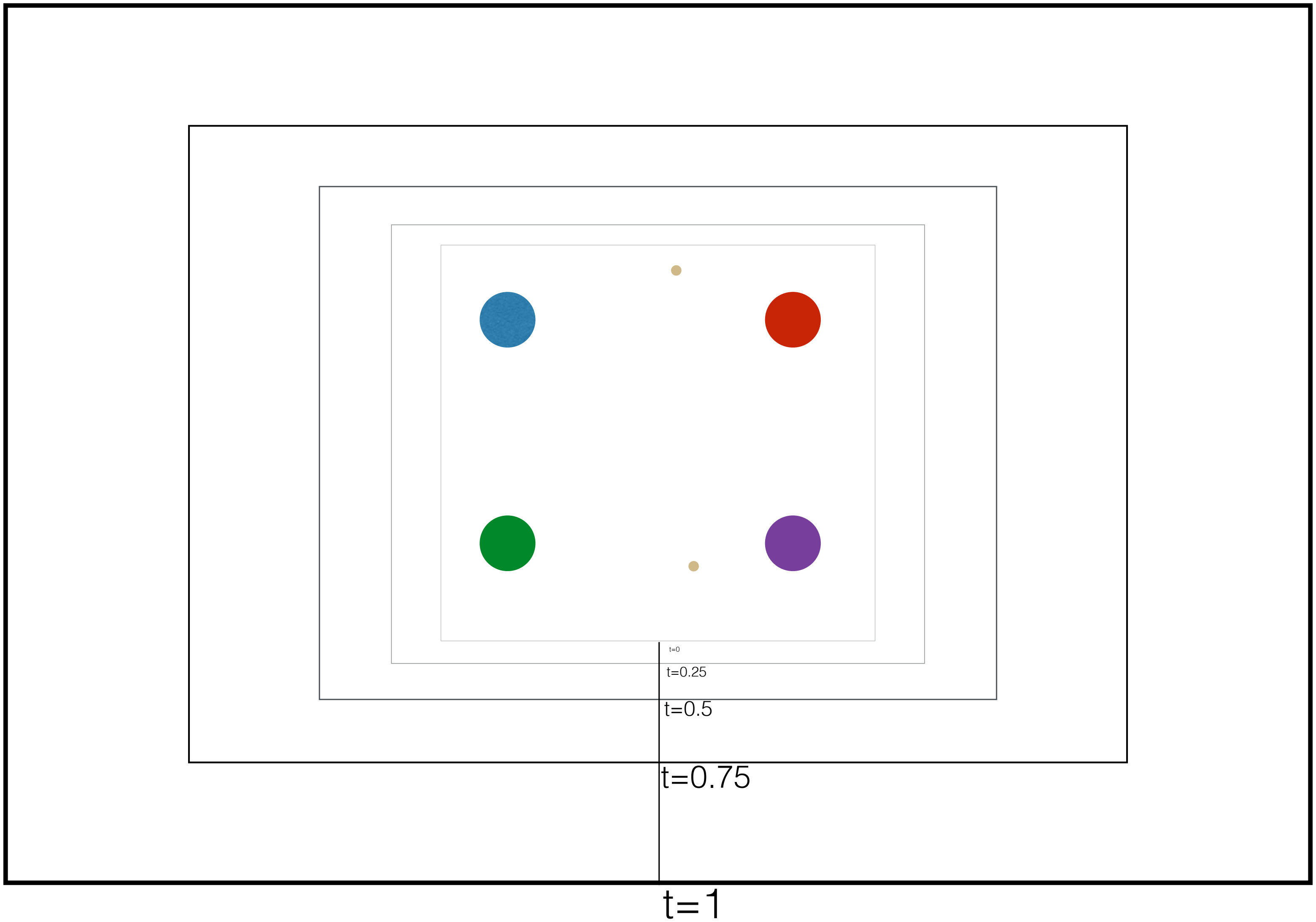
3. Use numerical predictor-corrector methods to follow the solutions as t marches from 1 to 0, one solution at a time. (Notice the parallelizability!)

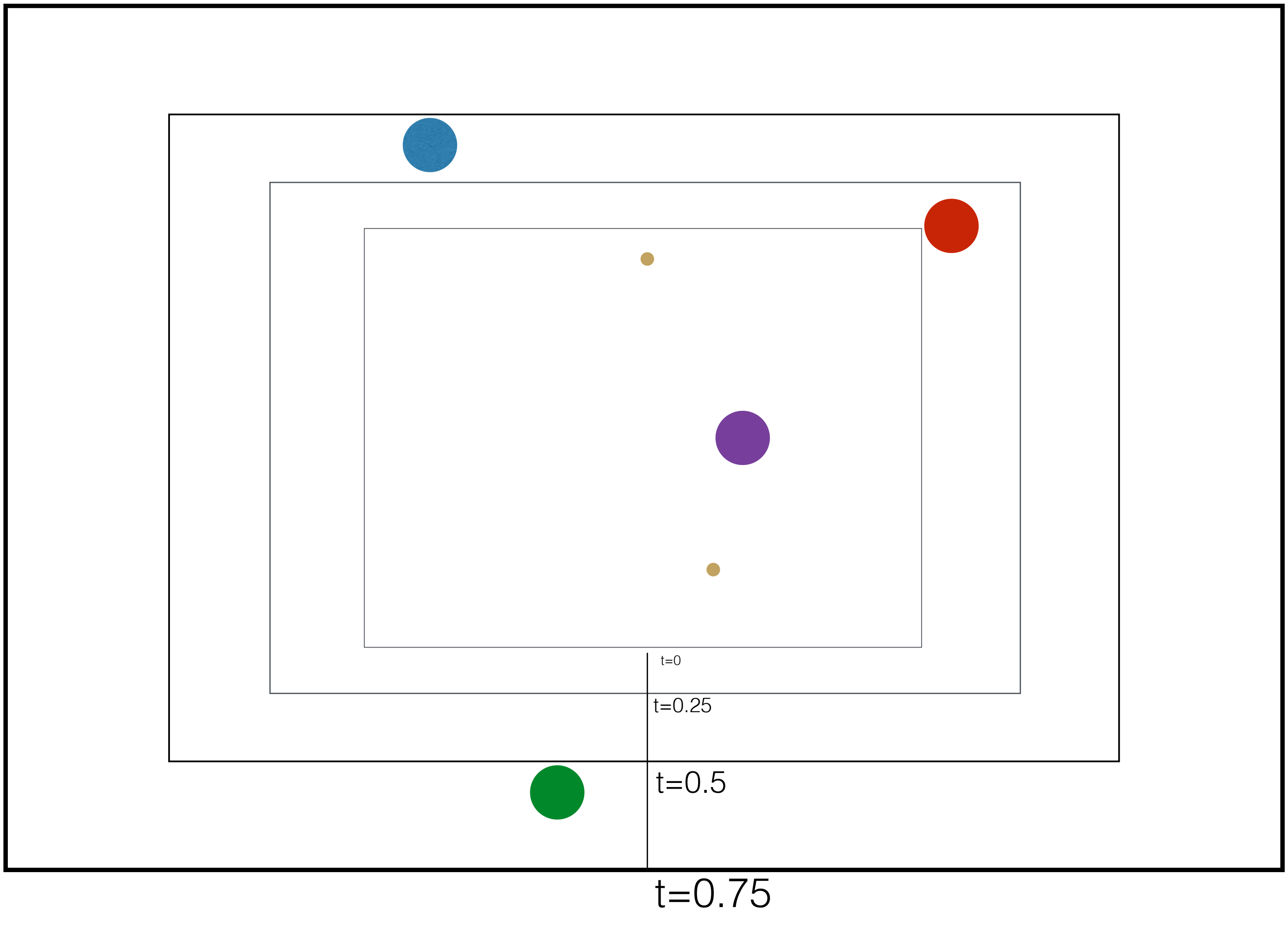
Finding isolated solutions



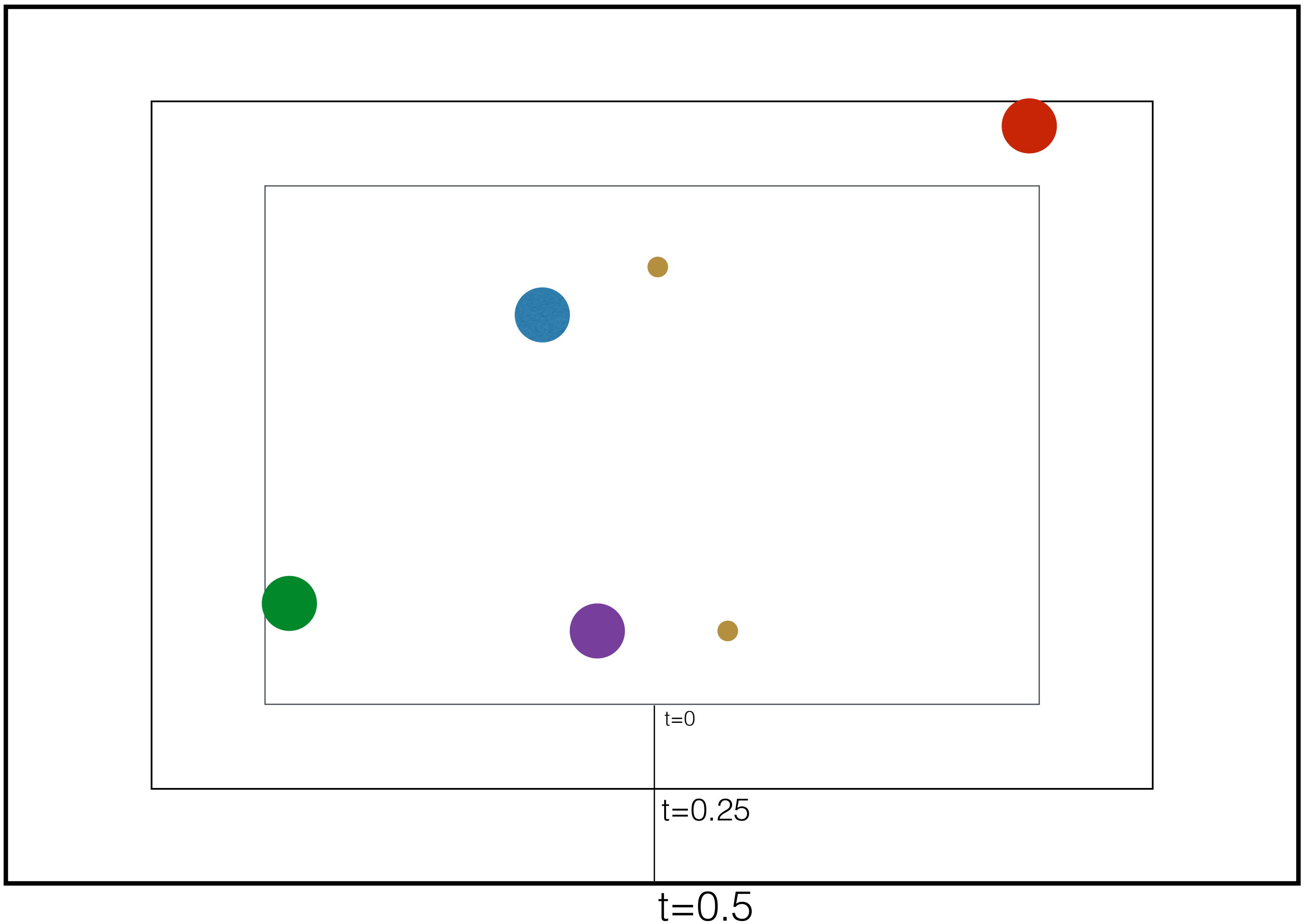
Finding isolated solutions

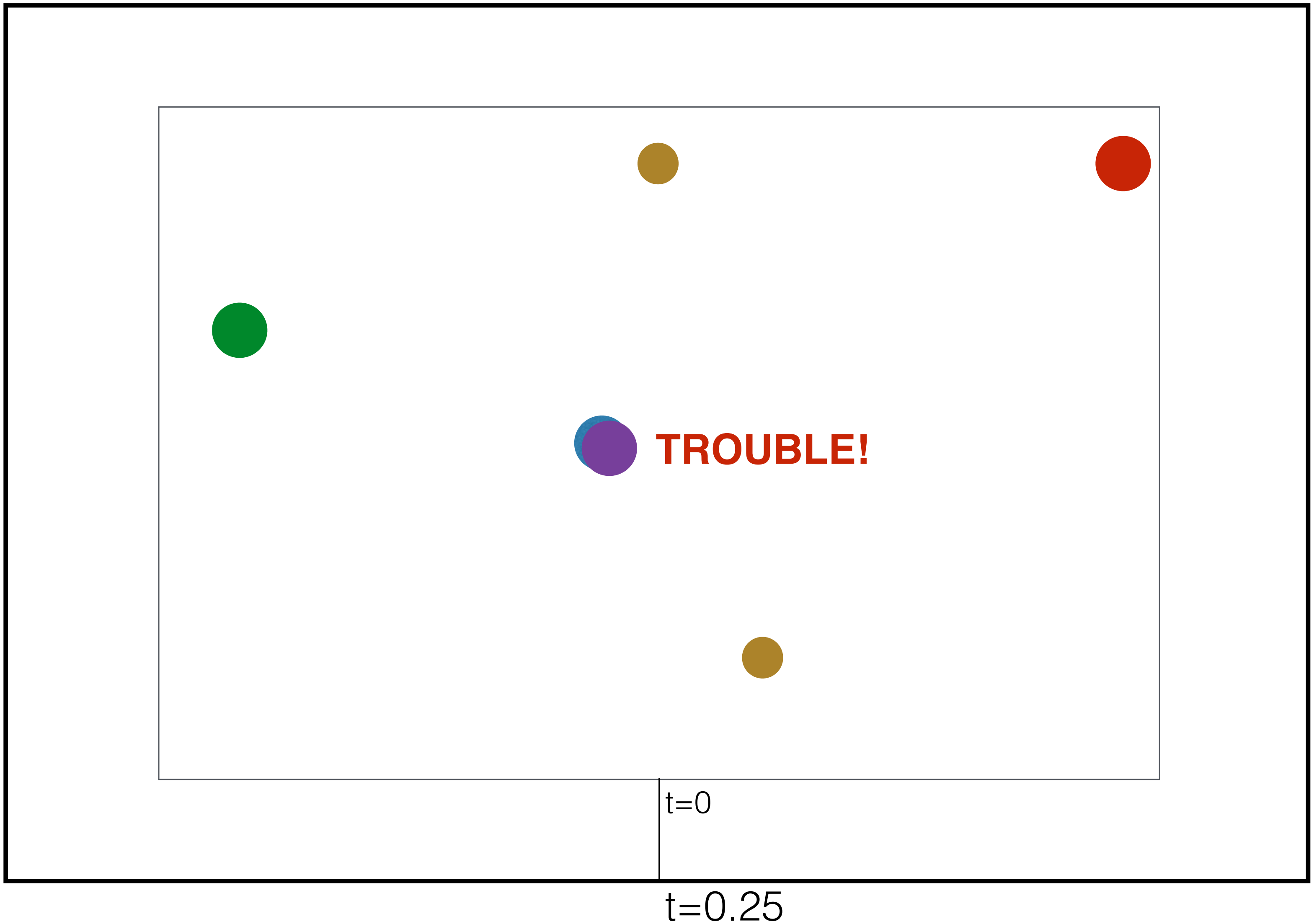


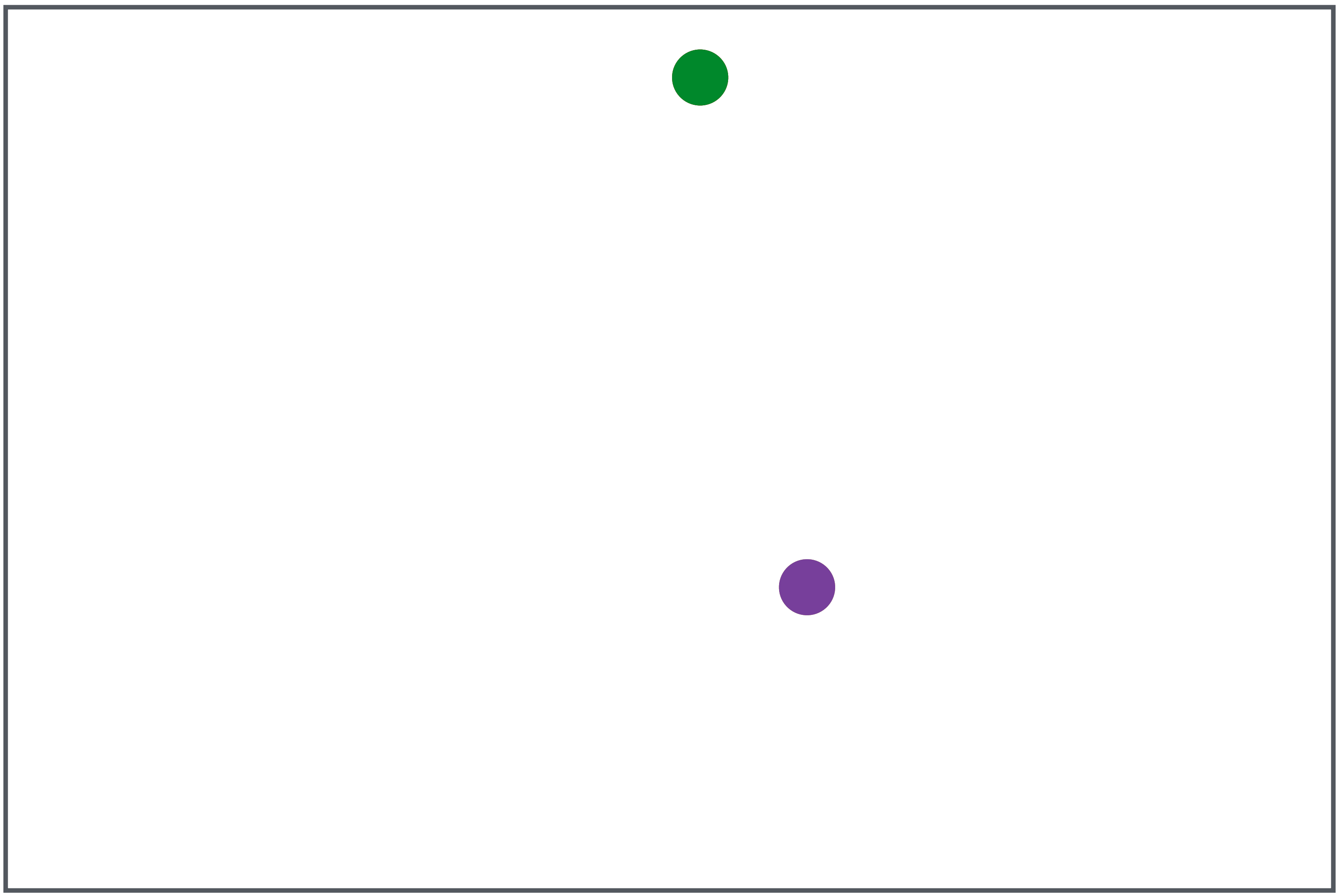

t=0
t=0.25
t=0.5
t=0.75
t=1



t=0
t=0.25
t=0.5
t=0.75

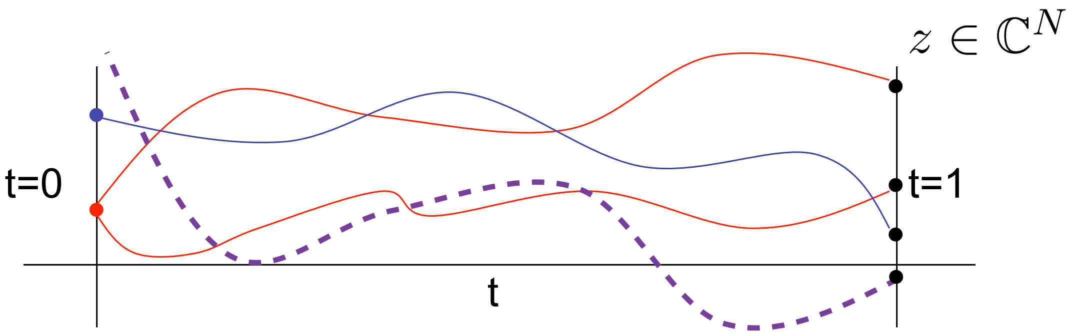

t=0
t=0.25
t=0.5

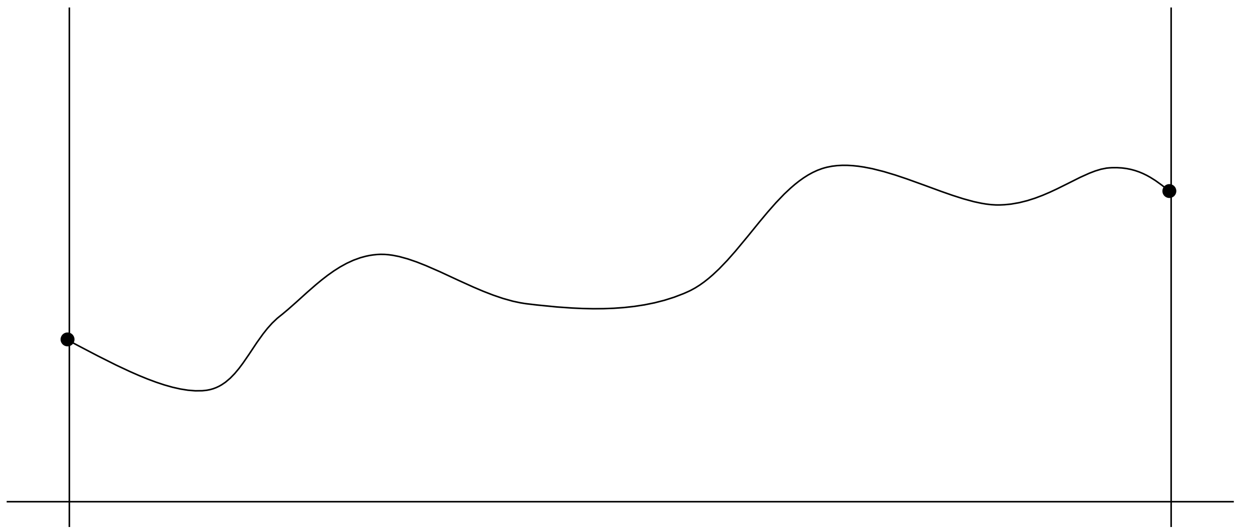

TROUBLE!
t=0
t=0.25



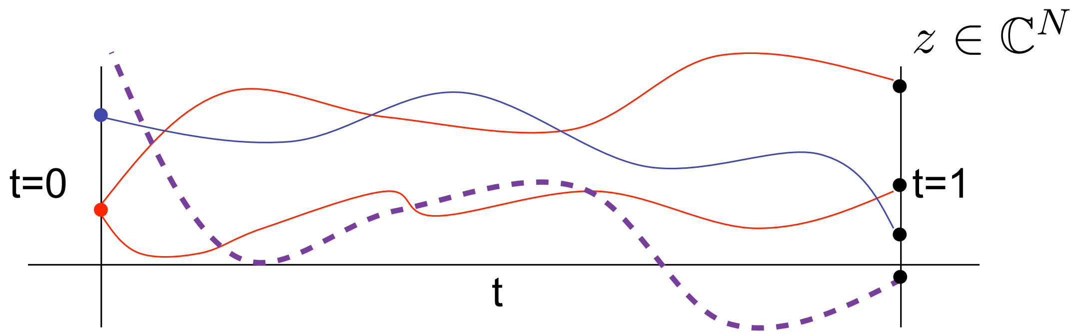
t=0

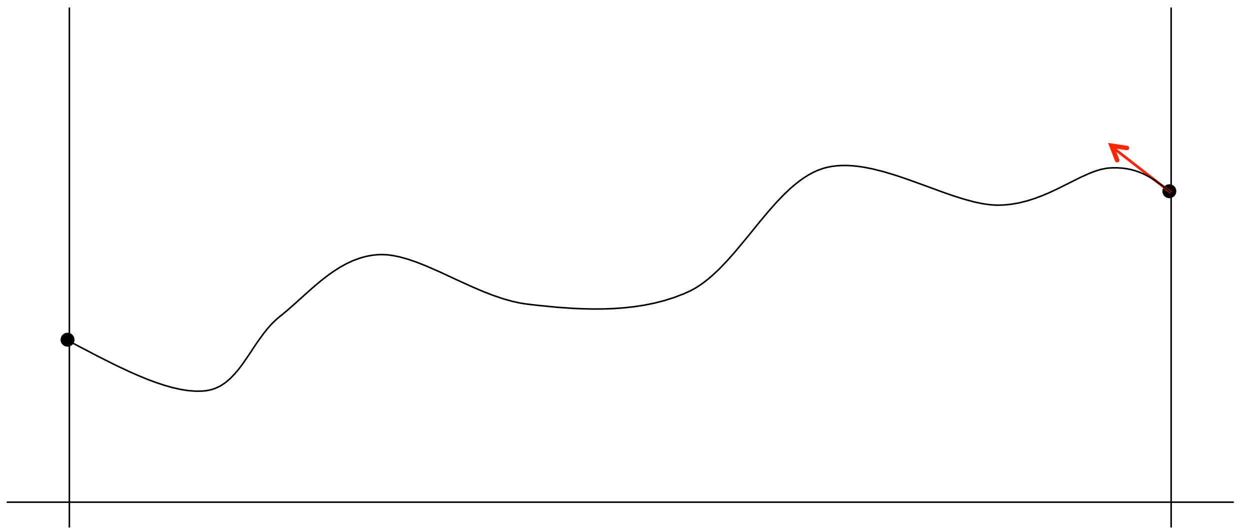
Finding isolated solutions



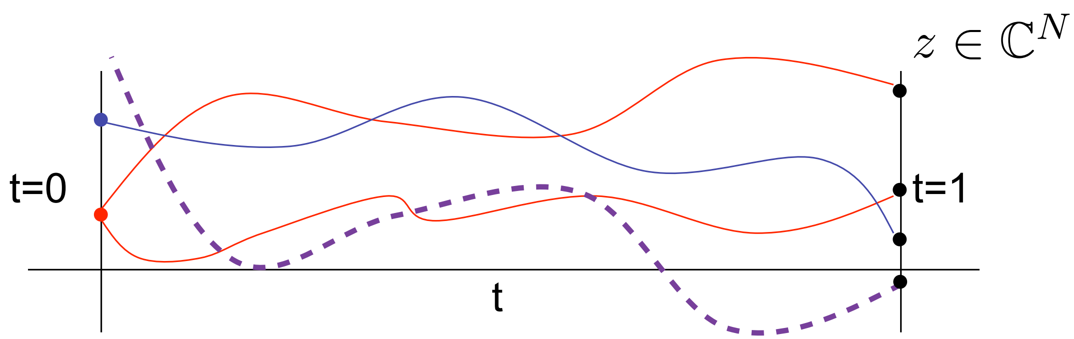


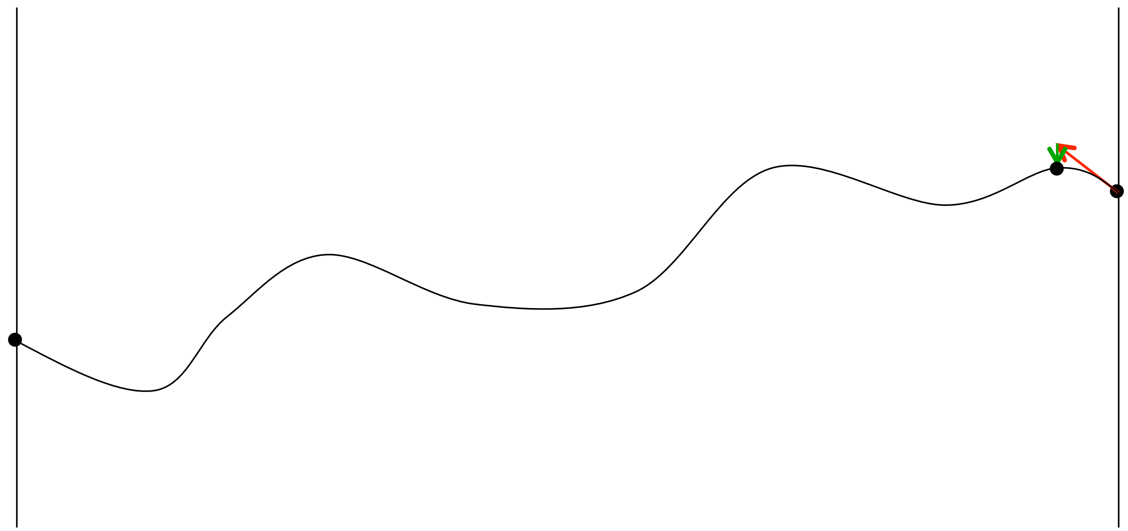
Finding isolated solutions



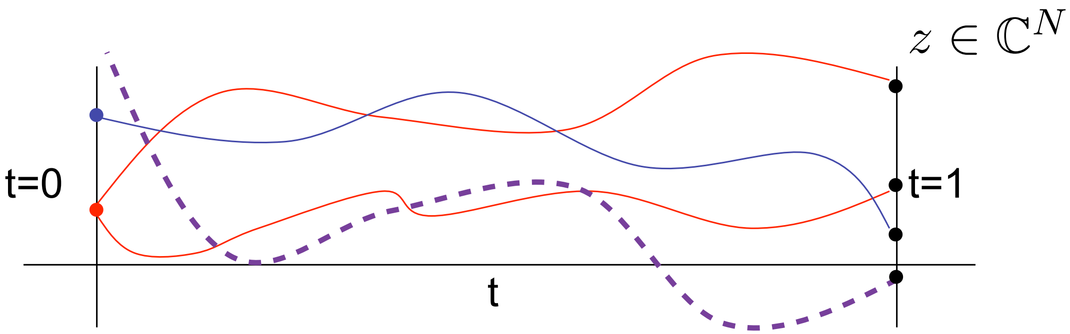


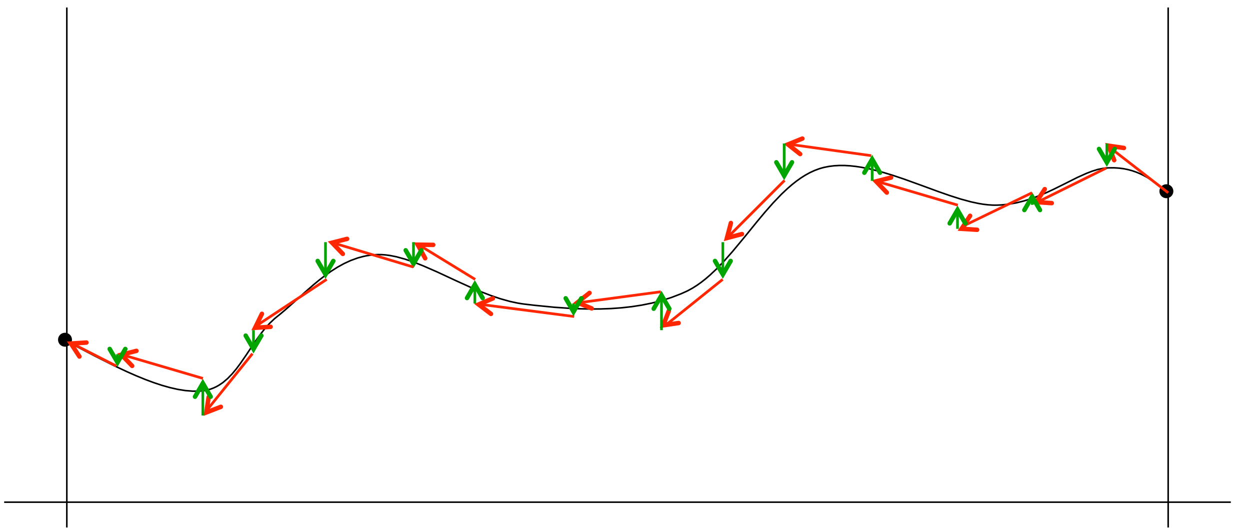
Finding isolated solutions





Finding isolated solutions





Finding isolated solutions

Root counts and start systems

Bézout: For a system with N polynomials and variables, the number of finite, isolated solutions $\leq \prod_{i=1}^{N} deg(f_i)$. (Over projective space, one can make more exact statements….)

One choice of start system is the total degree or Bézout start system:

$$g = \begin{bmatrix} z_1^{d_1} - 1 \\ \vdots \\ z_N^{d_N} - 1 \end{bmatrix}.$$

(Recall the first 3 examples!)

This has exactly $\prod_{i=1}^{N} deg(f_i)$ isolated, nonsingular, finite solutions.

This could be overkill — there might be many fewer solutions of f!

Different root counts lead to different start systems.

Finding isolated solutions

Root counts and start systems

Bézout (multihomogeneous version): The number of finite, isolated solutions is also bounded above by some combinatorial formula built from the multidegrees of the polynomials, when the variables are broken into multiple groups.

Depending on choice of variable groups, you might get fewer *or* more startpoints (typically more).

Example: $\begin{bmatrix} xy - 1 \\ x^2 - 1 \end{bmatrix}$

The total degree is 4.

Finding isolated solutions

Root counts and start systems

Example: $\begin{bmatrix} xy-1 \\ x^2-1 \end{bmatrix}$

	(x,y)
$xy-1$	2
x^2-1	2

The total degree is 4.

	(x)	(y)
$xy-1$	1	1
x^2-1	2	0

The 2-homogeneous (or 2-hom) degree is 2, so we can build a start system with 2 nonsingular, isolated solutions and follow only 2 paths to find all solutions!

Finding isolated solutions

Root counts and start systems

It is tempting to just try all possible variable groupings…DON'T!

The number of such groupings for n variables grows as the Bell number:

N	1	2	3	4	5	6	7	8
Bell(N)	1	2	5	15	52	203	877	4140

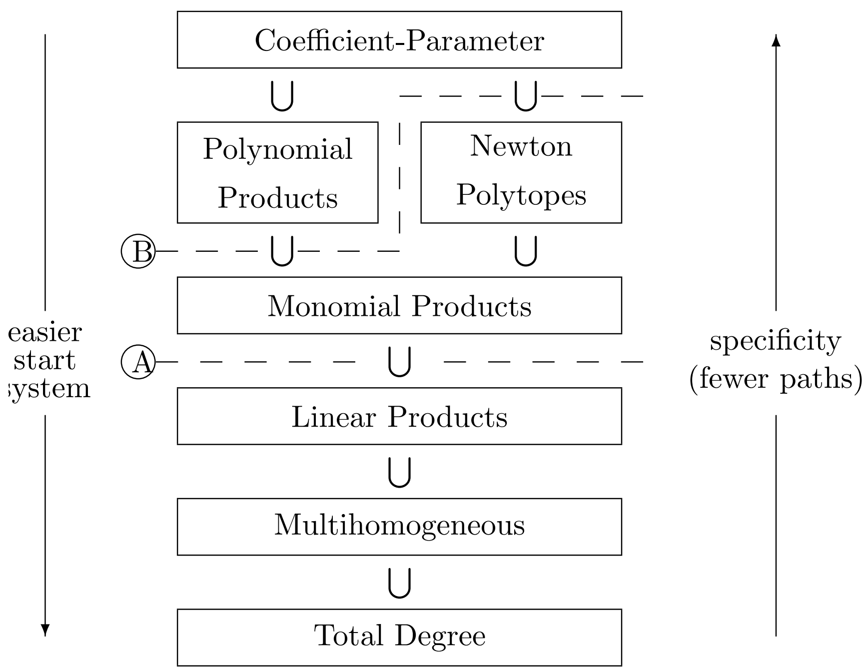
N	9	10	11	12
Bell(N)	21,147	115,975	678,570	4,213,597

N	25
Bell(N)	4,638,590,332,229,999,353

(See Wampler, *Numer. Math.*, '93 for more on this.)

Finding isolated solutions

Root counts and start systems

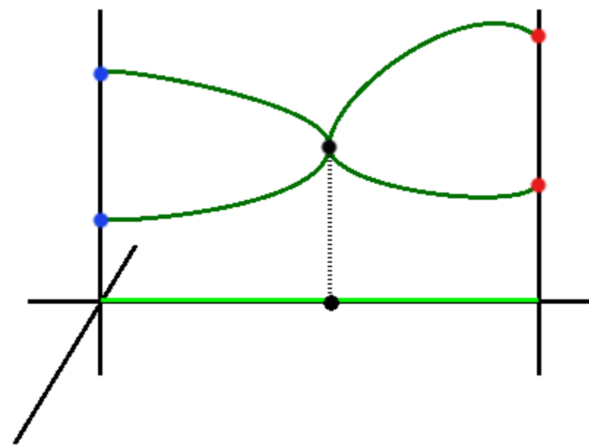


(from Sommese-Wampler '05 book)

Warning: I am entirely ignoring polyhedral homotopies, which can be very efficient in terms of the number of paths to be tracked but is sometimes expensive in terms of precomputation.

Finding isolated solutions

The gamma trick

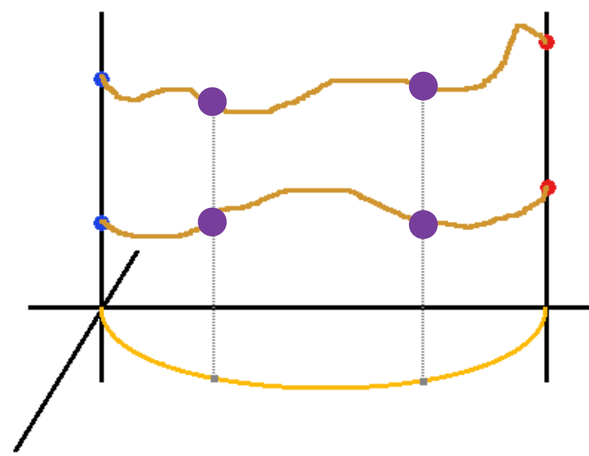


"2. Form the homotopy $H : \mathbb{C}^N \times \mathbb{C} \to \mathbb{C}^N$ given by

$$H(z,t) = f(z) \cdot (1-t) + g(z) \cdot t$$

so that $H(z,1) = g(z)$ and $H(z,0) = f(z)$."

In fact, we use the homotopy:



$$H(z,t) = f(z) \cdot (1-t) + \gamma g(z) \cdot t$$

where $\gamma \in \mathbb{C}$ is chosen at random.

Finding isolated solutions

Detecting divergence

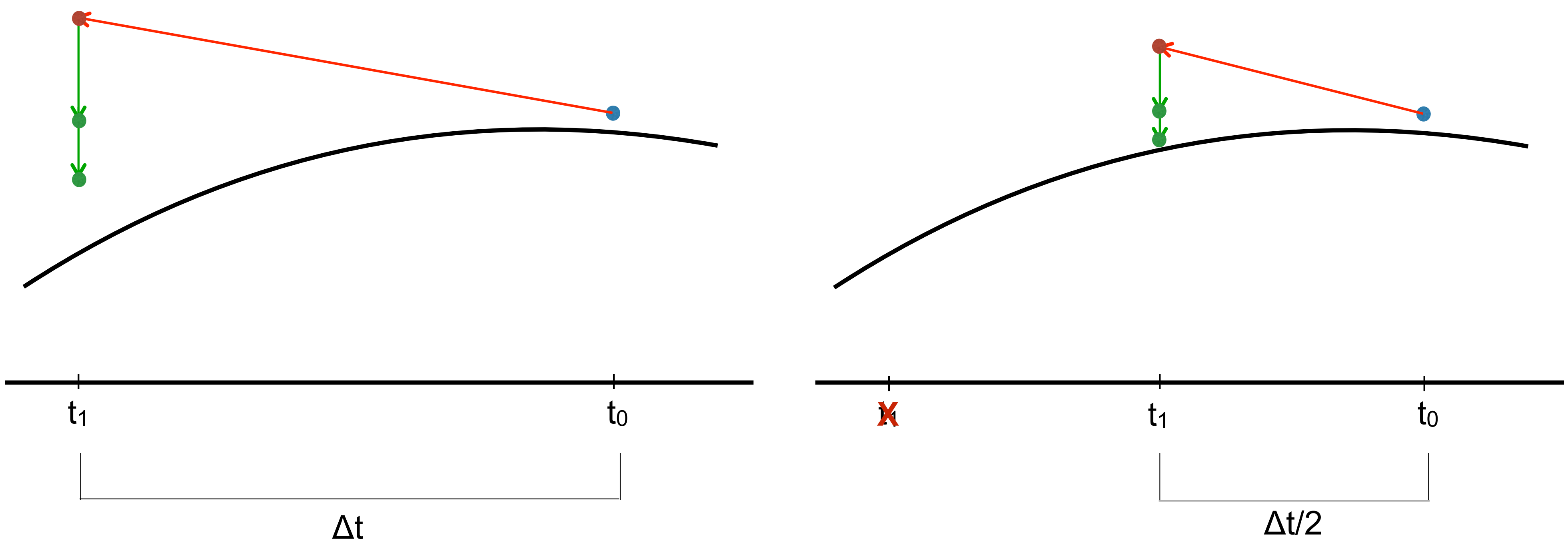
Input can be non-homogeneous (mixed degrees in each polynomial) or homogeneous (all terms in any one polynomial have the same degree).

Either way, Bertini will homogenize (if necessary) and work over a random patch of $\mathbb{P}^N$. Thus, paths of infinite length become paths of finite length.

Even so, paths diverging to infinity often have highly singular endpoints, so it is preferable to avoid them. In Bertini, we kill any path that exceeds the threshold $\mathrm{SecurityMaxNorm}$ after t reaches 0.1. (Good for speed!)

Finding isolated solutions

Adaptive steplength



Finding isolated solutions

Adaptive precision

For matrix $A \in \mathbb{C}^{N \times N}$, the singular value decomposition (SVD) of A is a decomposition $A = U\Sigma V^*$ with various properties.

For our purposes, the key is that Σ is diagonal with nonnegative real entries called the singular values of A.

Using this, we can define the condition number of A as:

$$\kappa(A) = \frac{s_{max}}{s_{min}}$$

Wilkinson: When solving linear system $Ax = b$,

$$\mathbf{ACC} \approx \mathbf{PREC} - log_{10}(\kappa(A))$$

Finding isolated solutions

Adaptive precision

$$\mathbf{ACC} \approx \mathbf{PREC} - log_{10}(\kappa(A))$$

So, when the condition number gets high, we can increase precision to salvage accuracy. (There are many details….)

This isn't free!!

The key point is that zones of ill-conditioning can cause numerical trouble but AMP reduces the size of these zones significantly. Of course, there is a limit on PREC and failure-causing pathologies can be constructed.

Bertini will either get through these zones or report path failures (rather than report incorrect results).

Finding isolated solutions

Endgames

We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Finding isolated solutions

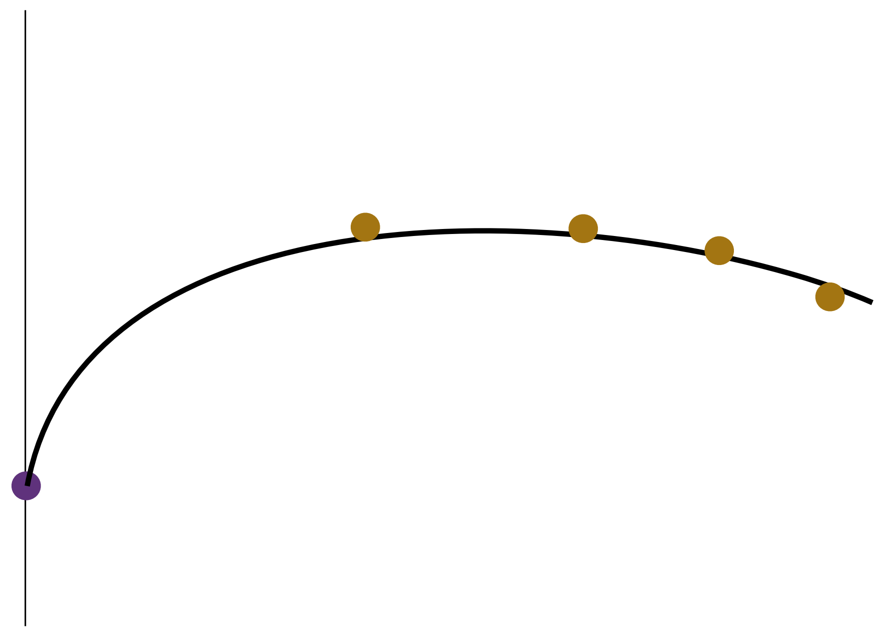
Endgames

We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. **Power series endgame**
2. Cauchy endgame



See Morgan-Sommese-Wampler, *Num. Math.* '92 for details.

Finding isolated solutions

Endgames

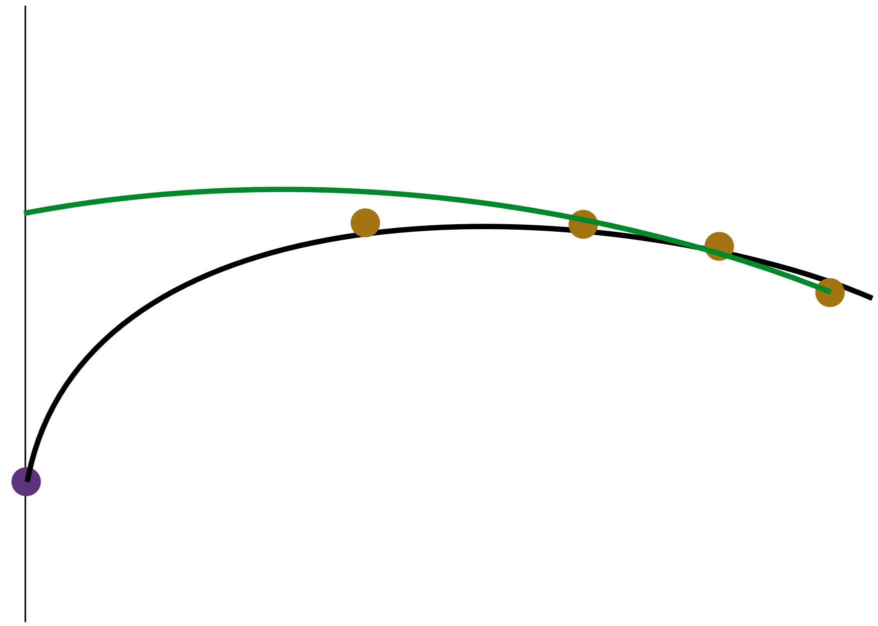
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

Endgames

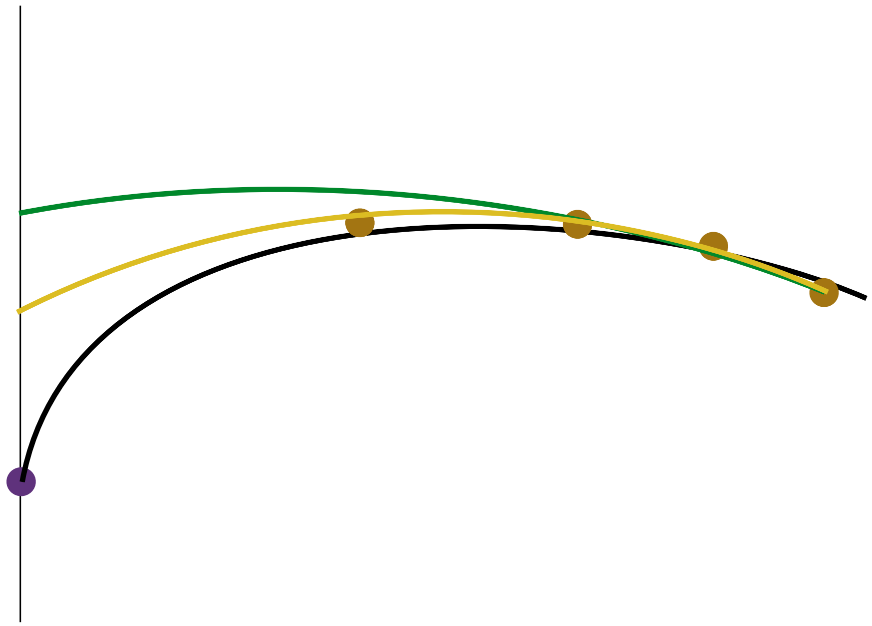
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame

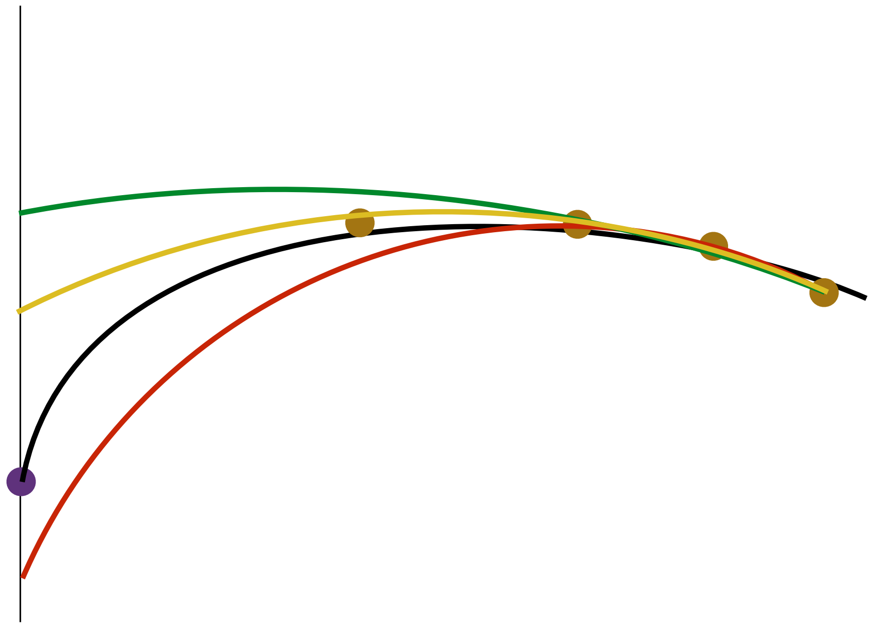


Finding isolated solutions

Endgames

We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.



Two main options:

1. Power series endgame

2. Cauchy endgame

Finding isolated solutions

Endgames

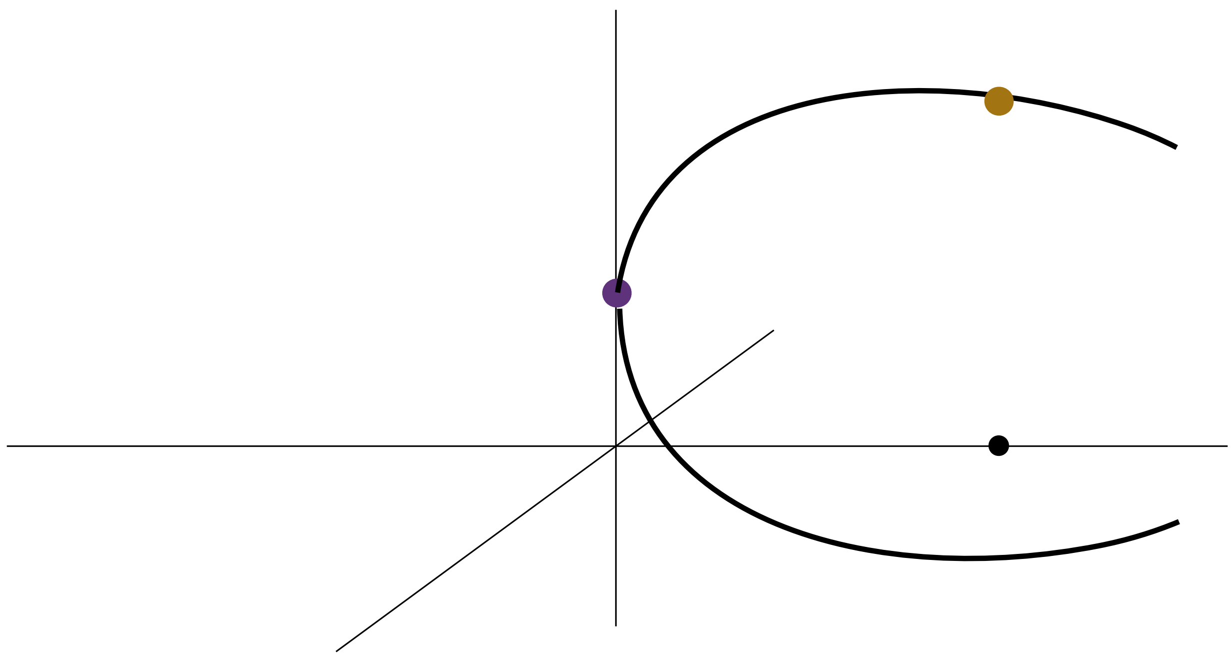
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. **Cauchy endgame**



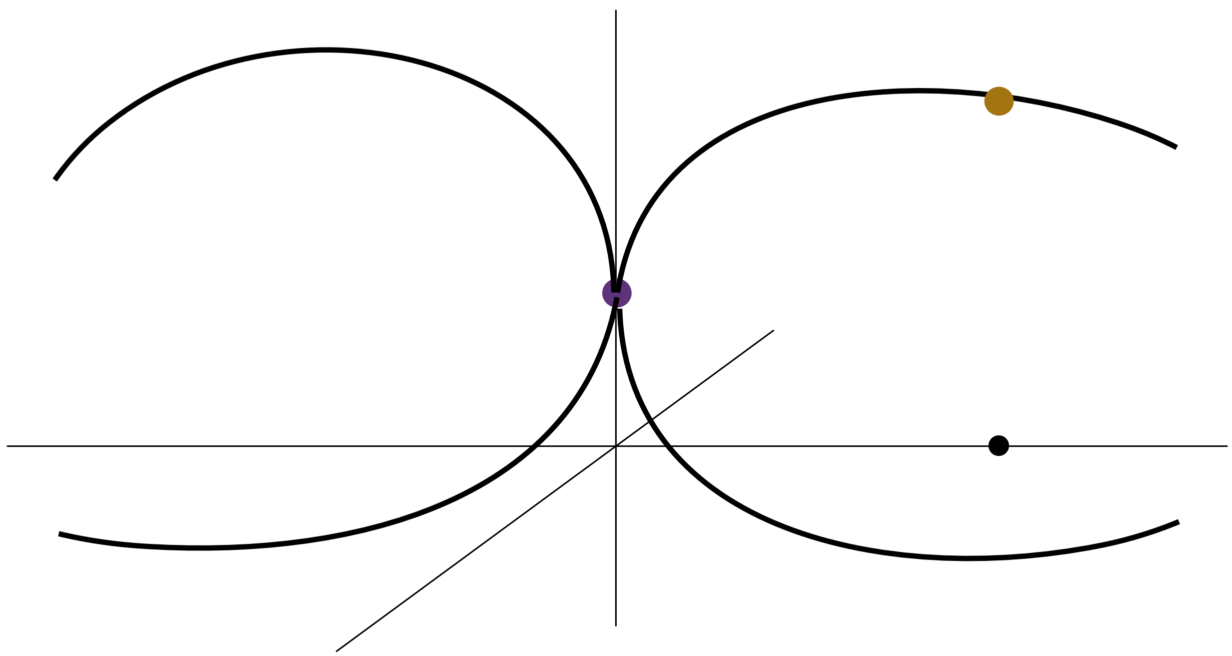
See Morgan-Sommese-Wampler, *Adv. in Appl. Math.* '92 for details.

Finding isolated solutions

Endgames

We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.



Two main options:

1. Power series endgame
2. Cauchy endgame

Finding isolated solutions

Endgames

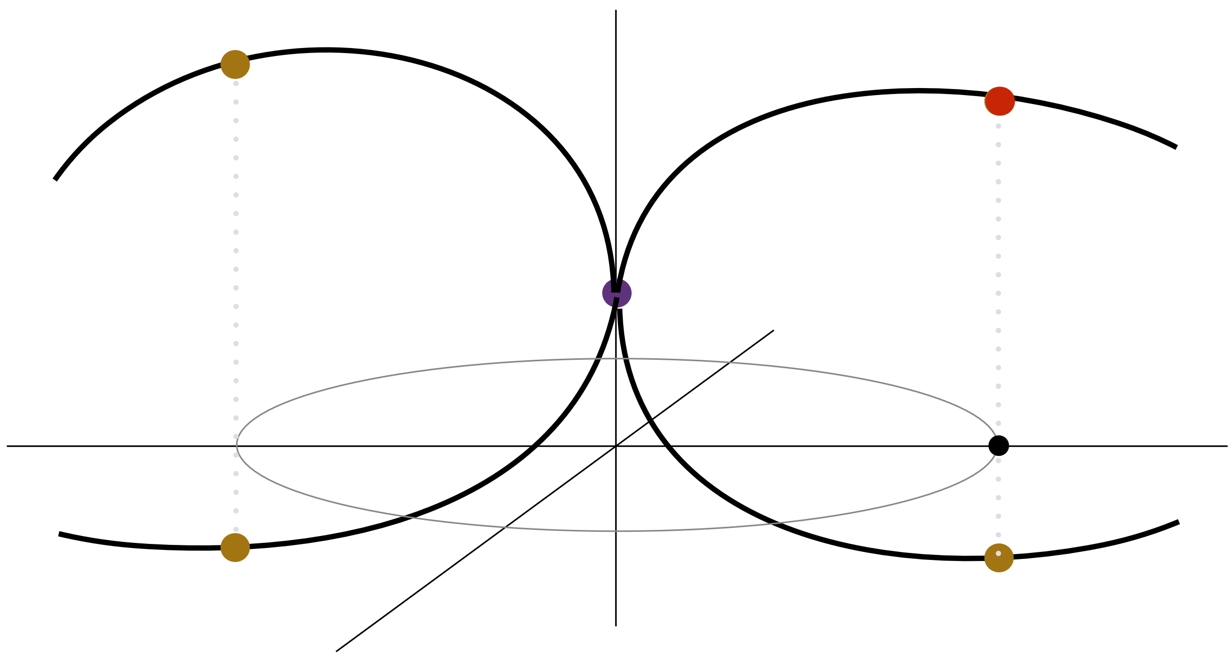
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

Endgames

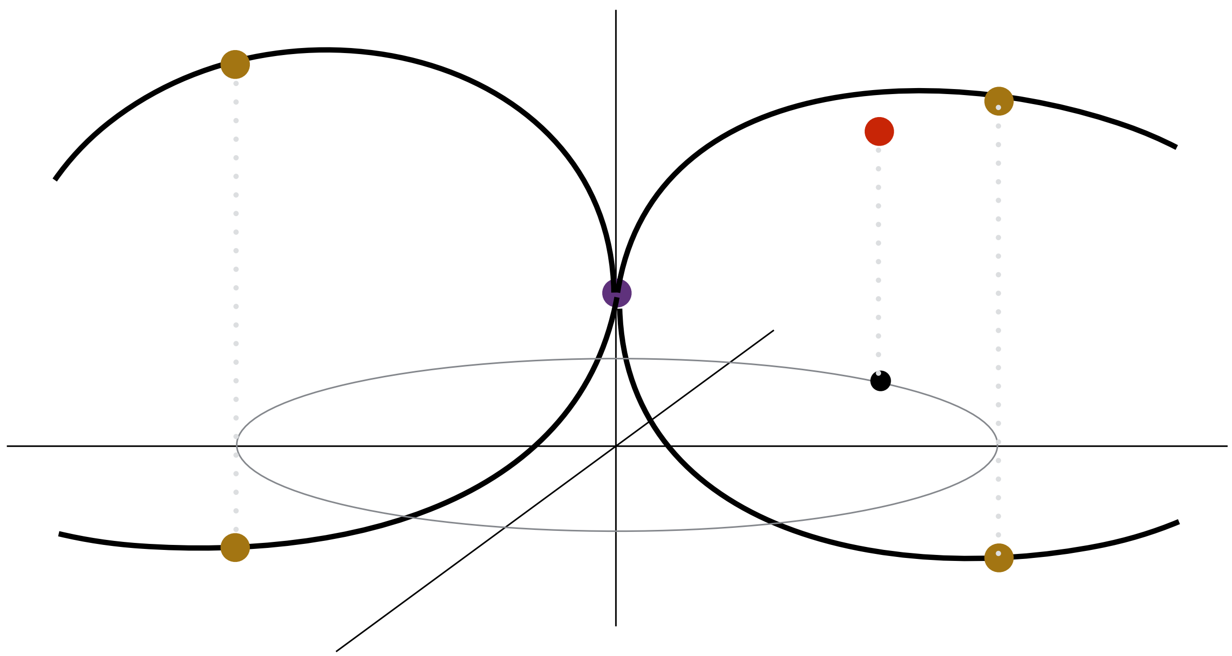
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

Endgames

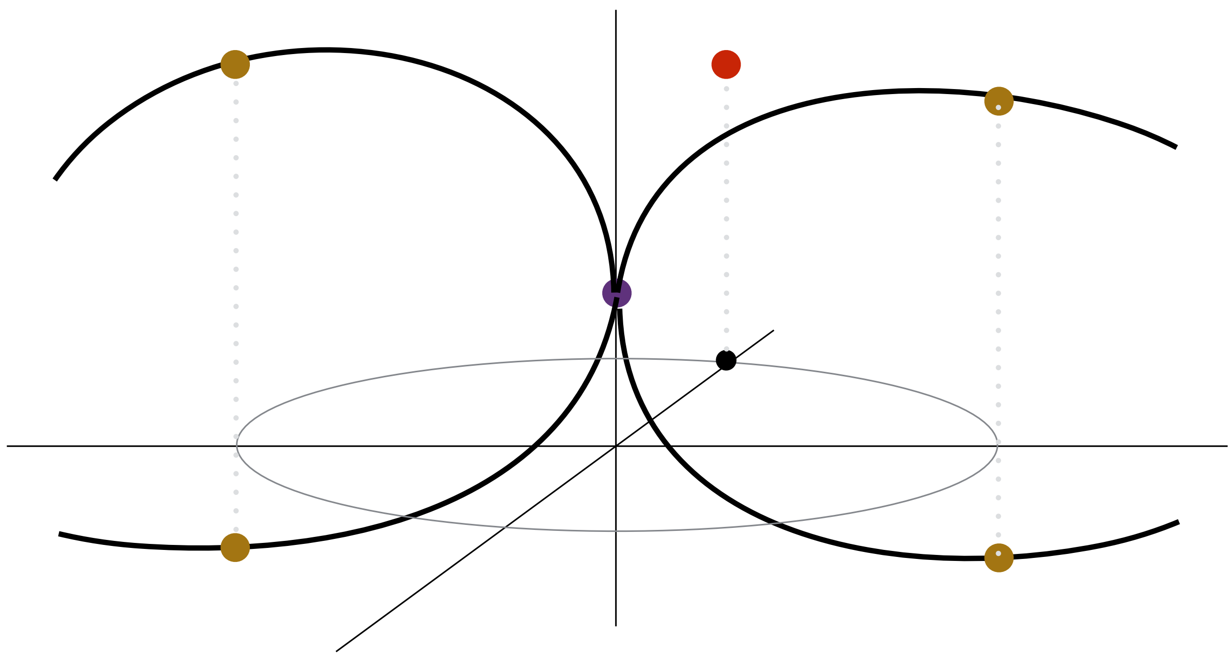
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

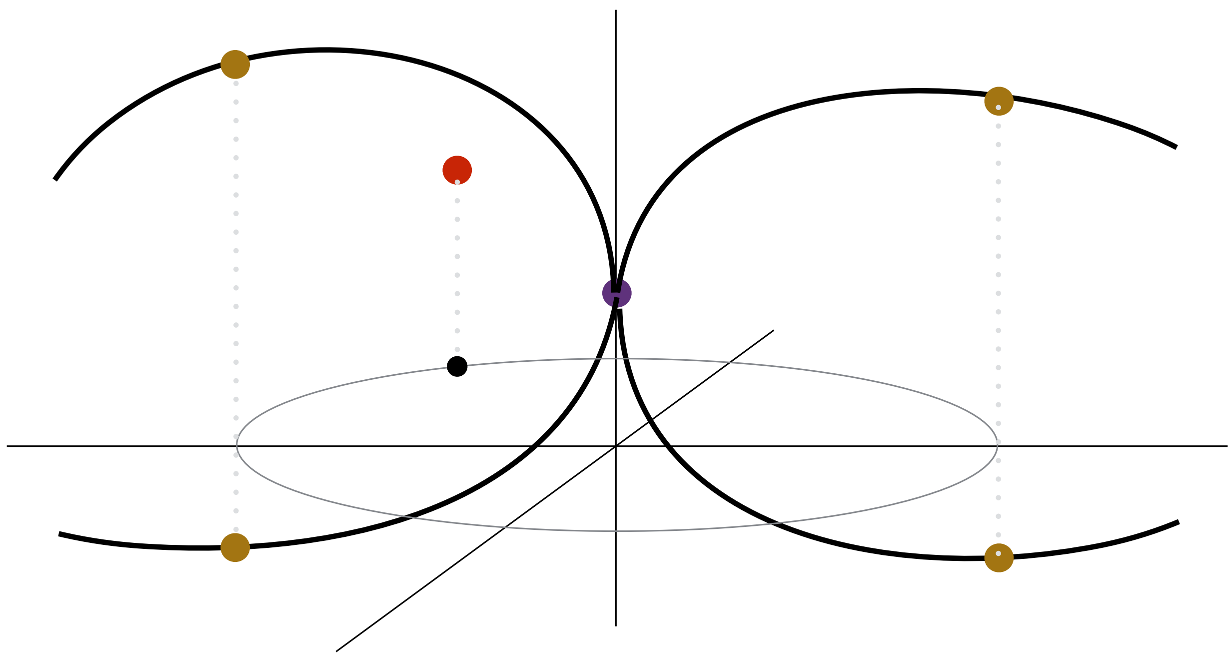
Endgames

We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame
2. Cauchy endgame



Finding isolated solutions

Endgames

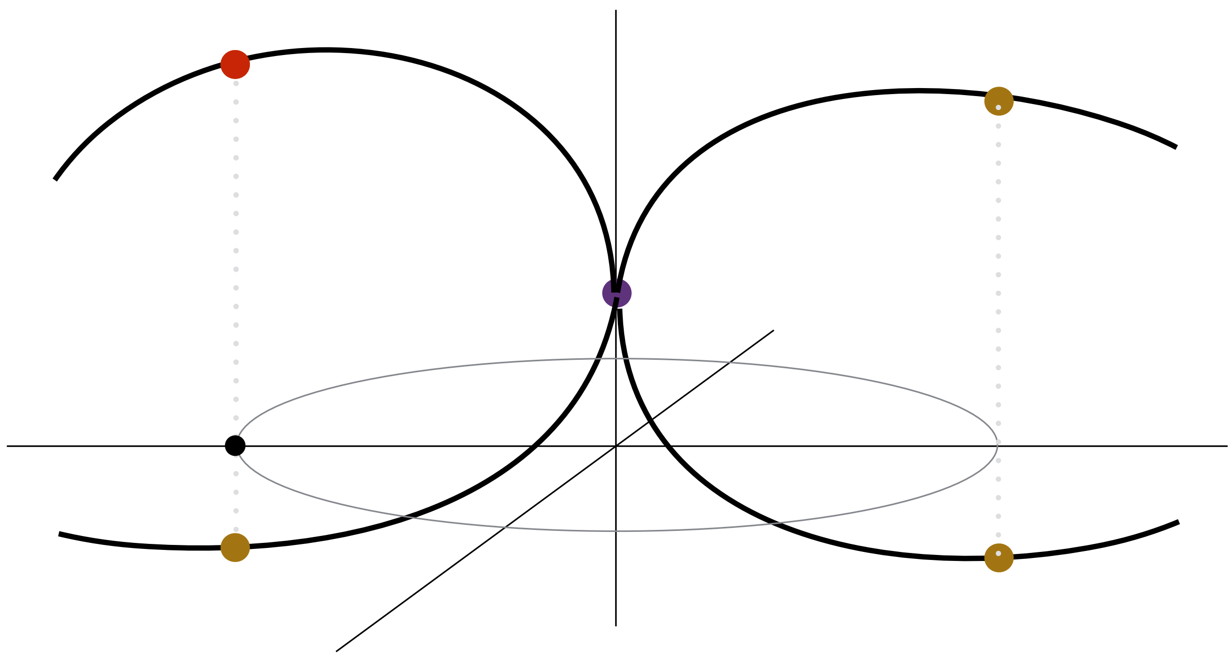
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

Endgames

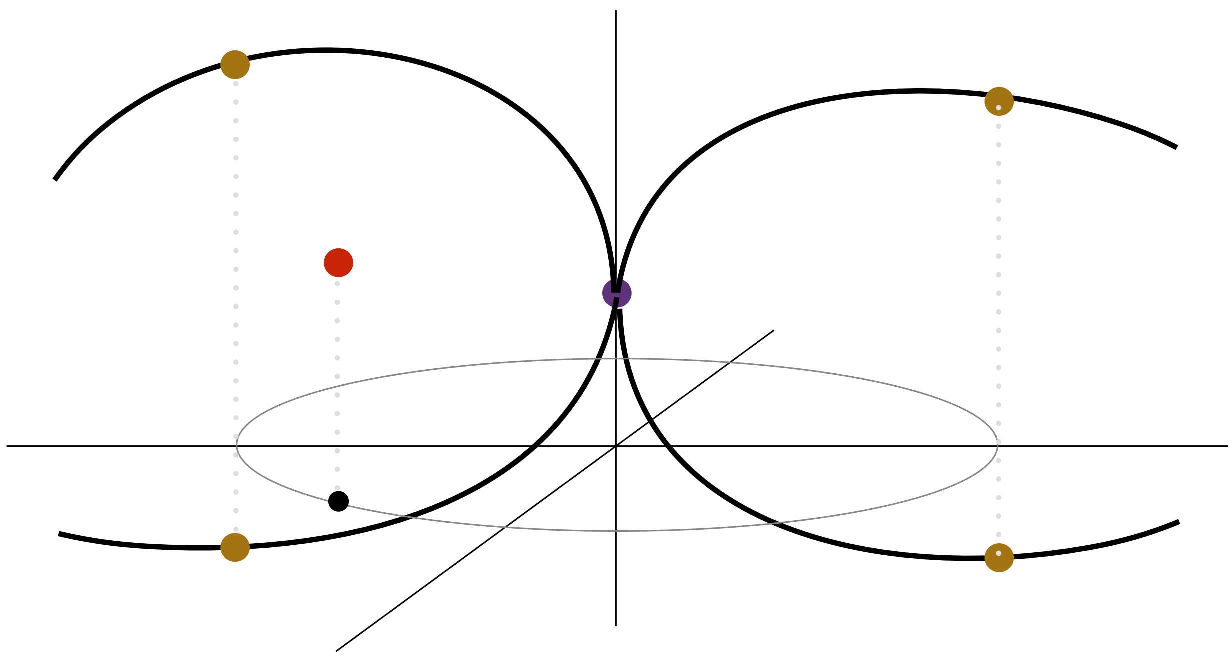
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

Endgames

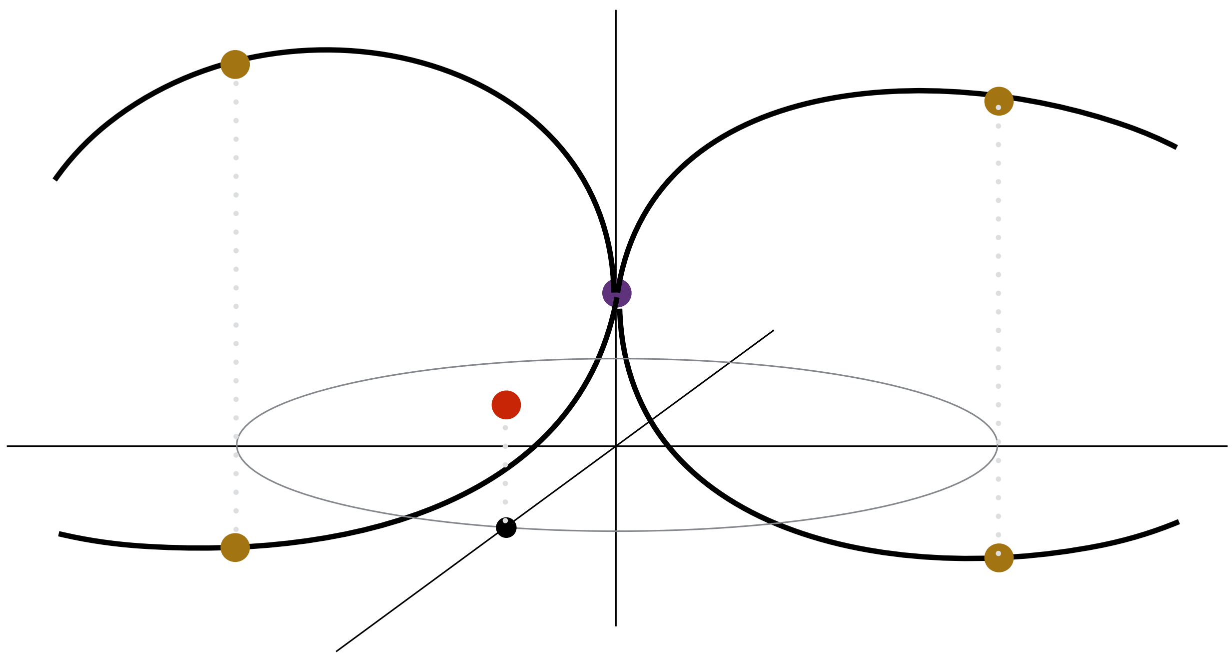
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

Endgames

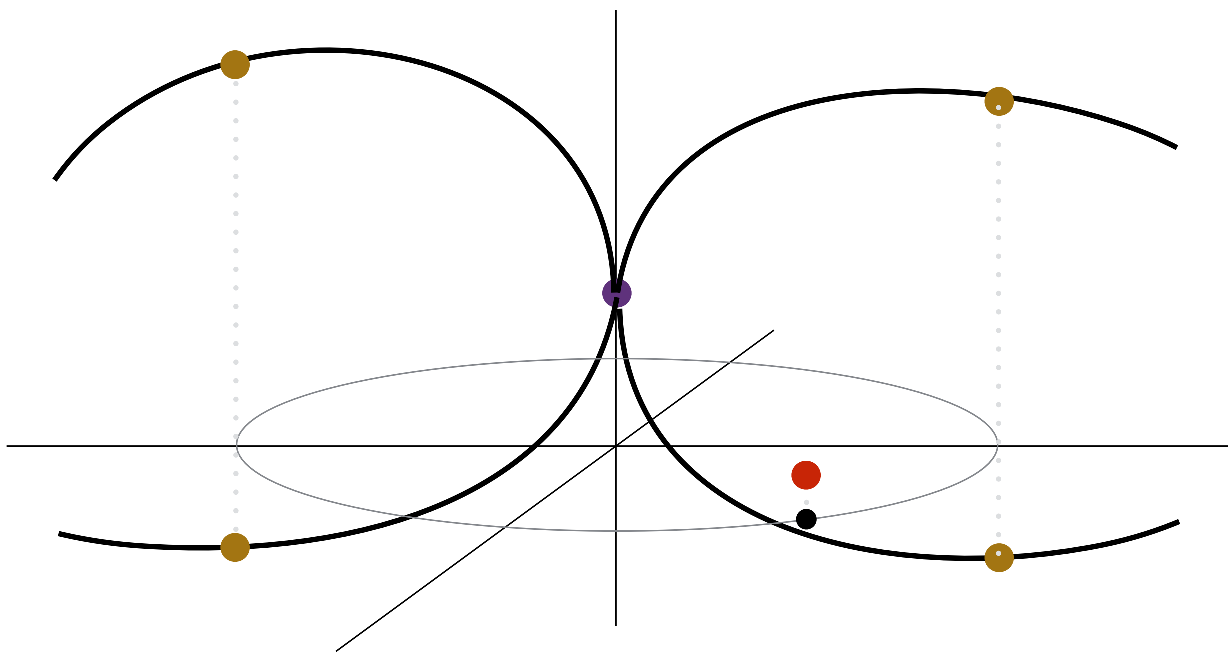
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

Endgames

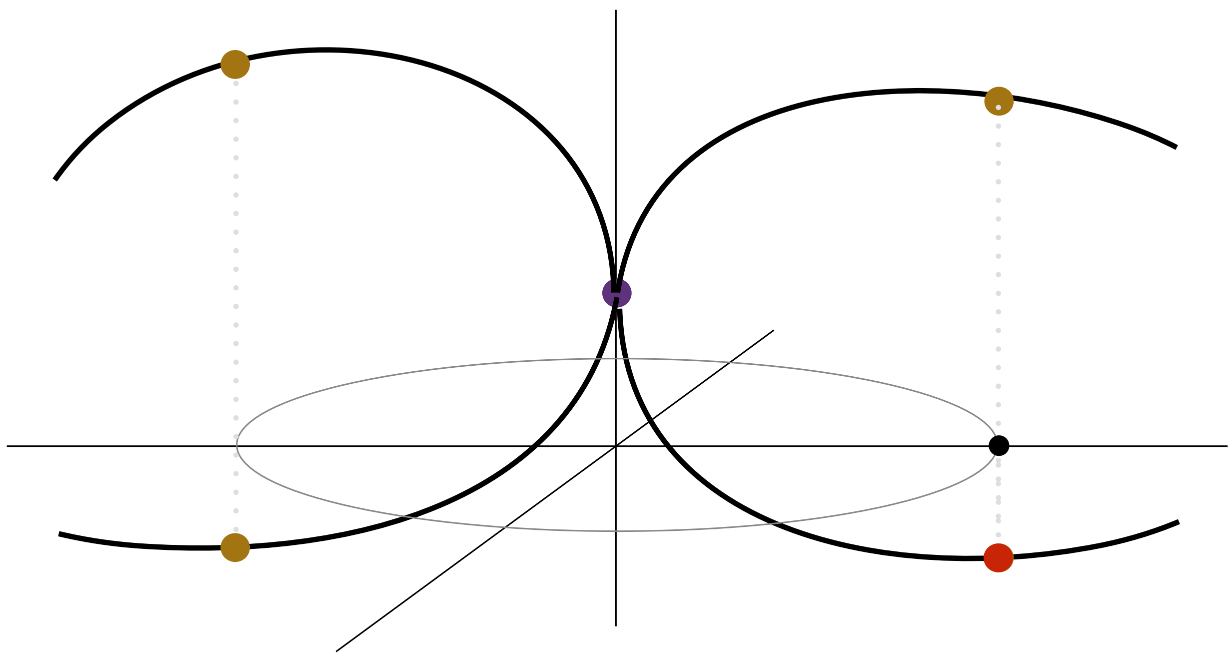
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

Endgames

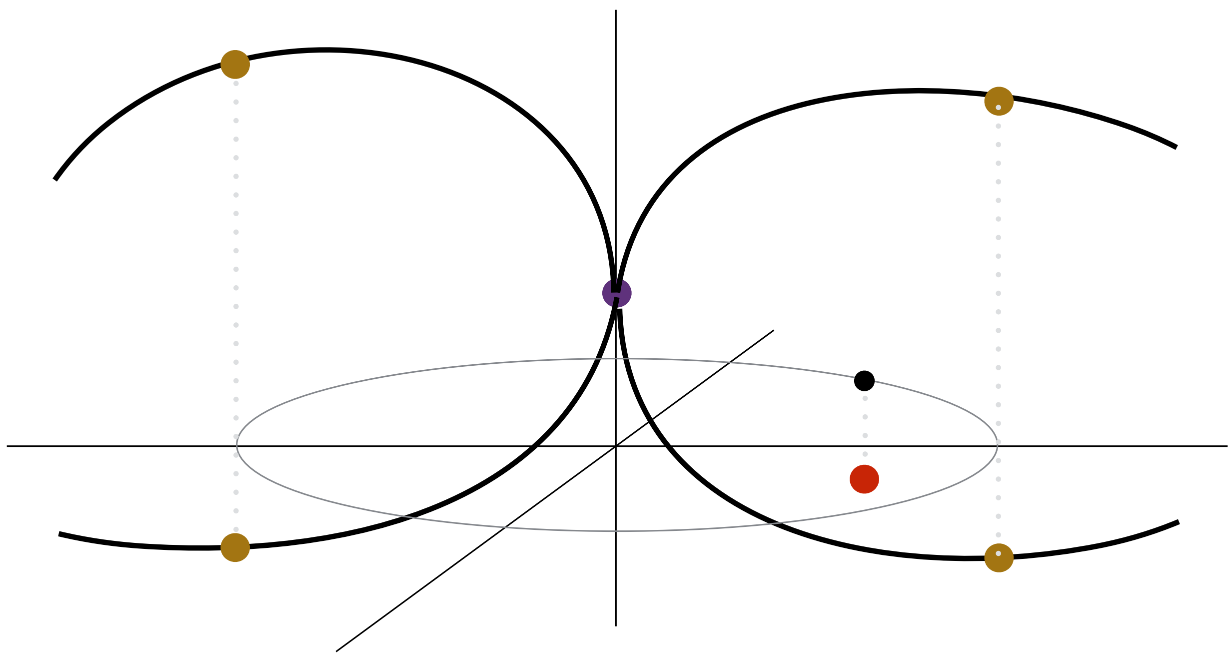
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

Endgames

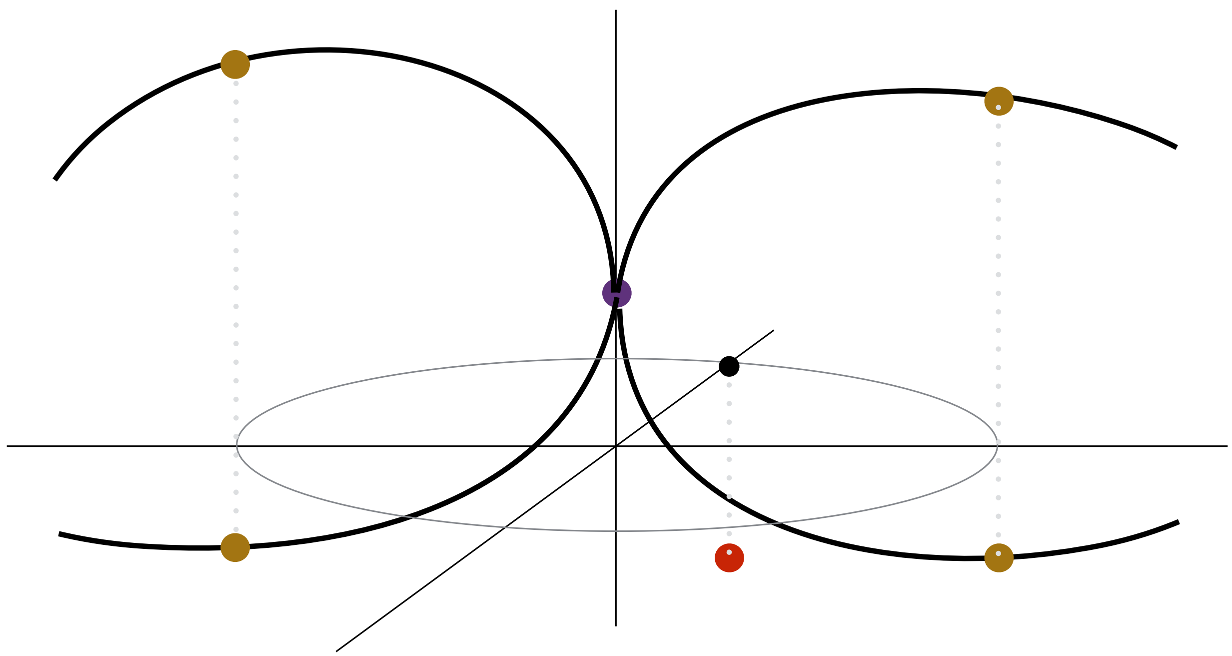
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

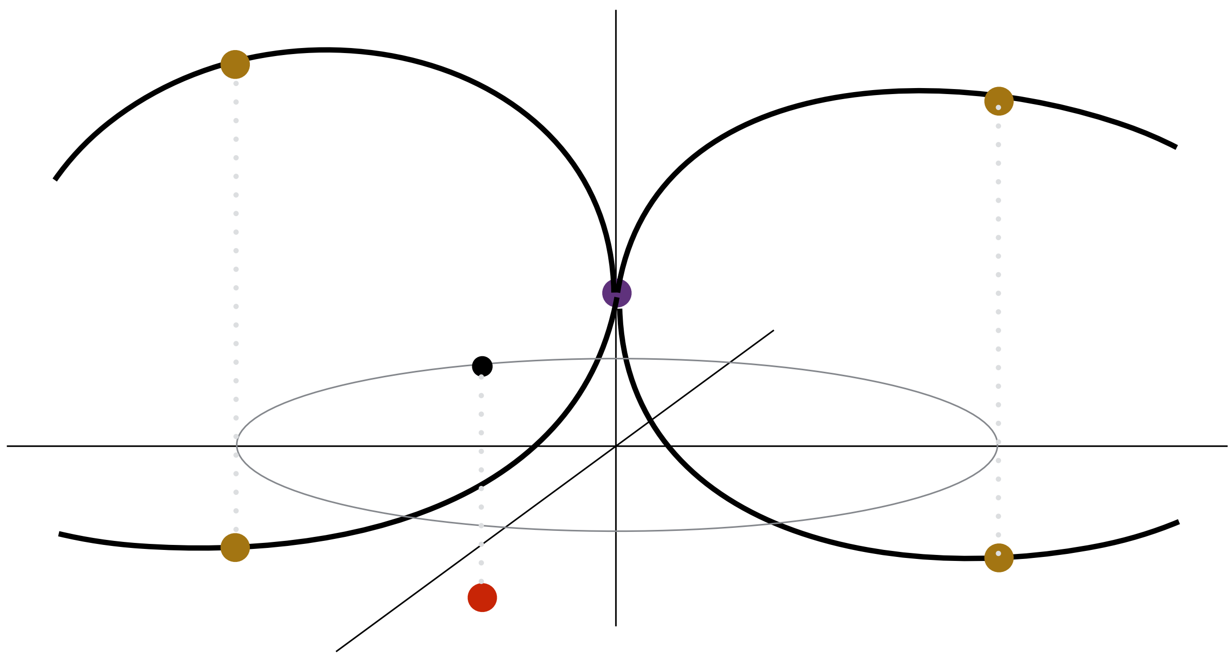
Endgames

We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame
2. Cauchy endgame



Finding isolated solutions

Endgames

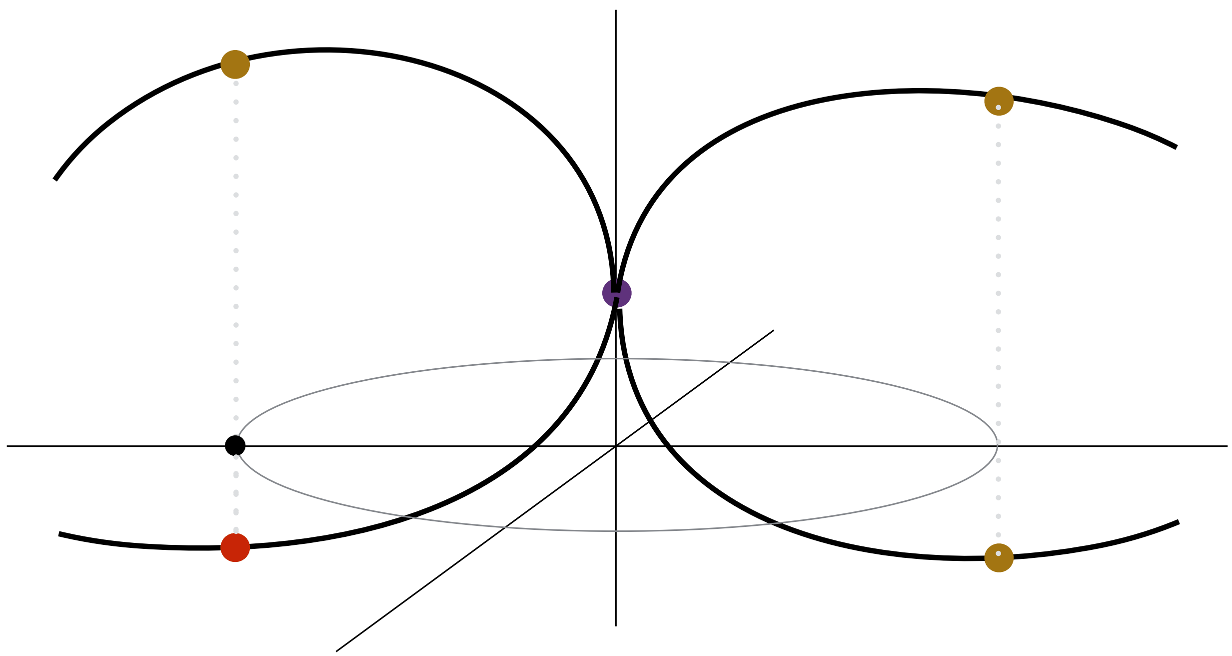
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

Endgames

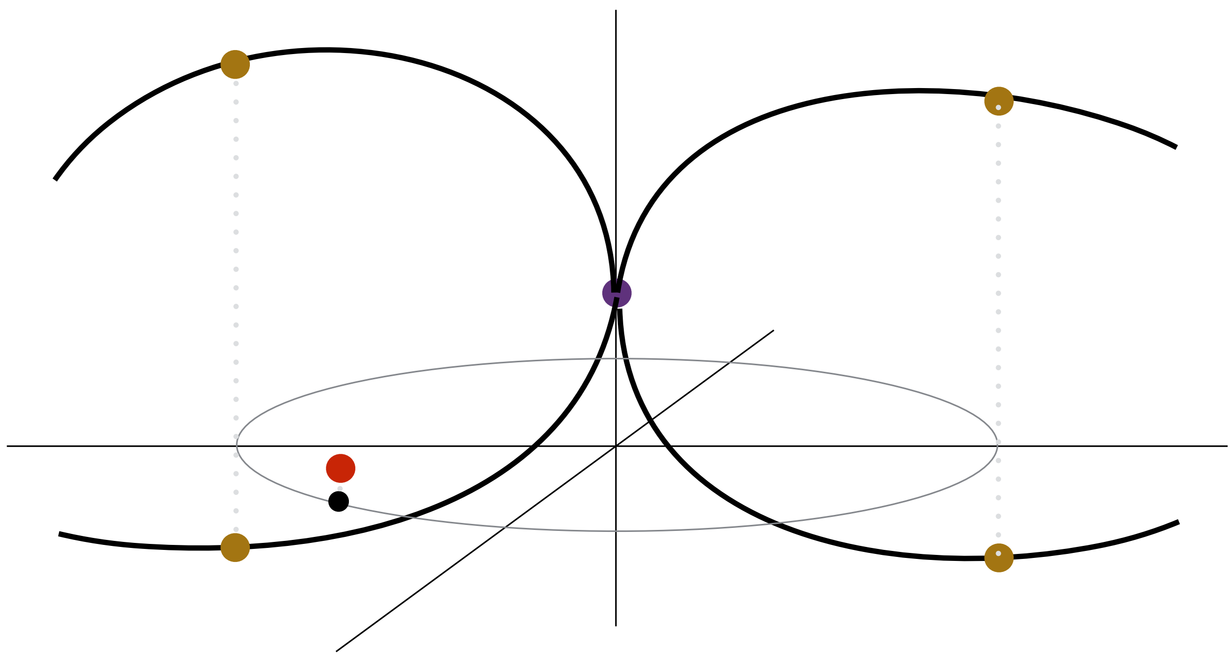
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

Endgames

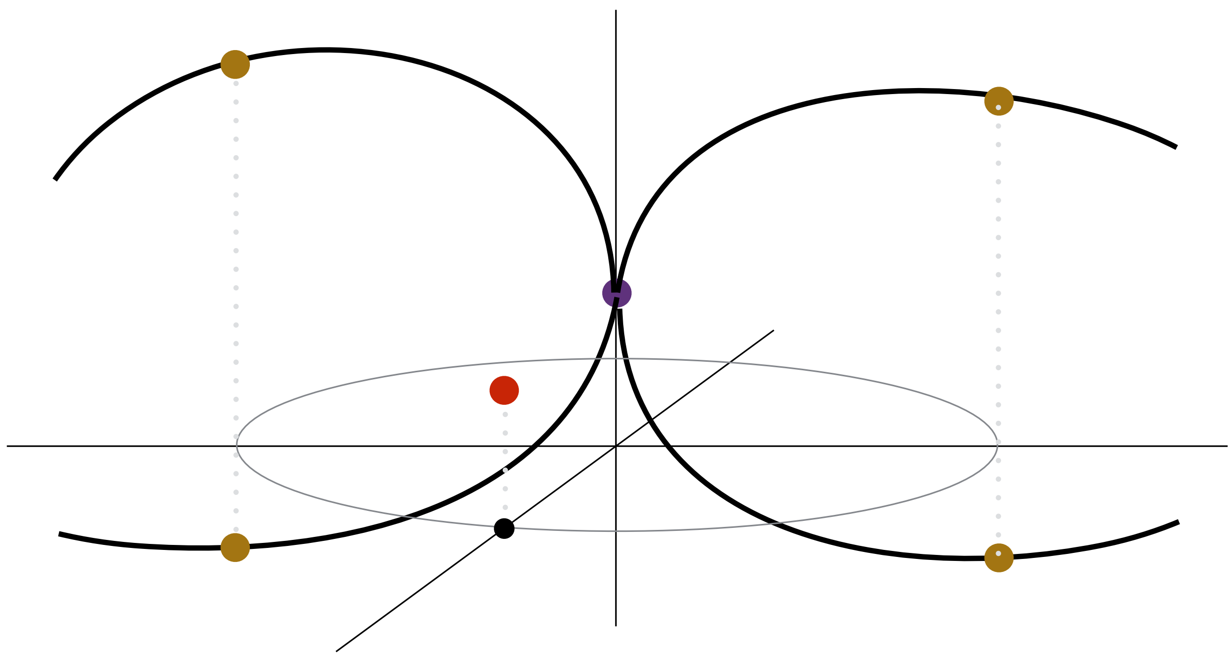
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

Endgames

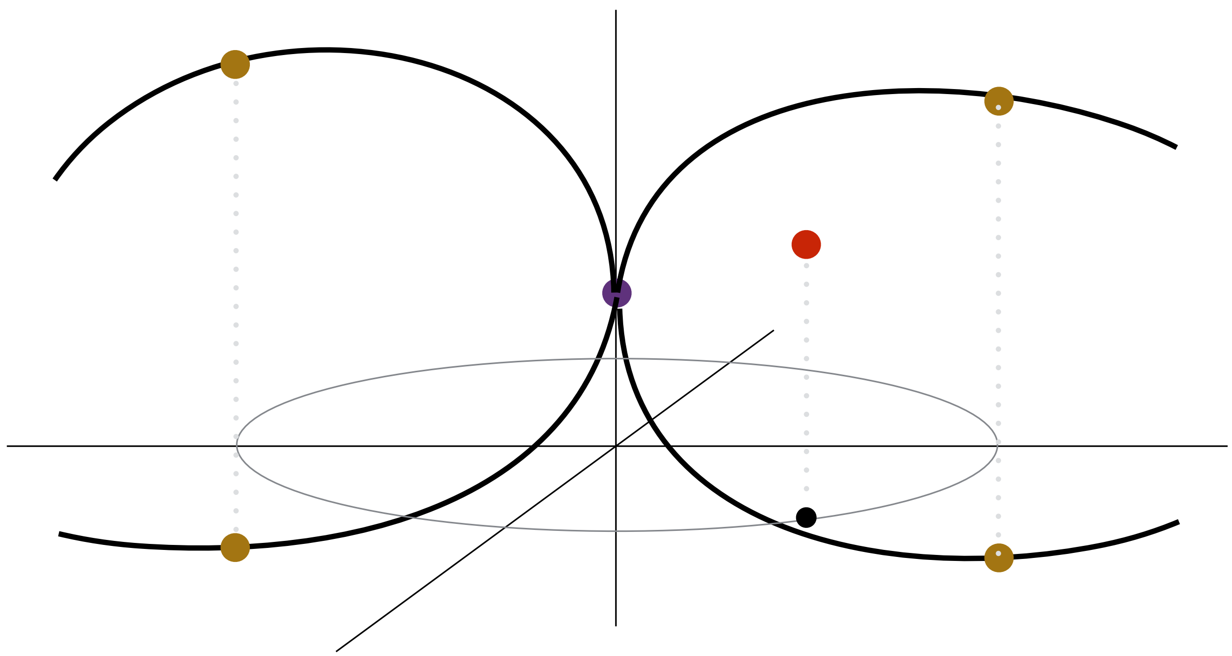
We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame

2. Cauchy endgame



Finding isolated solutions

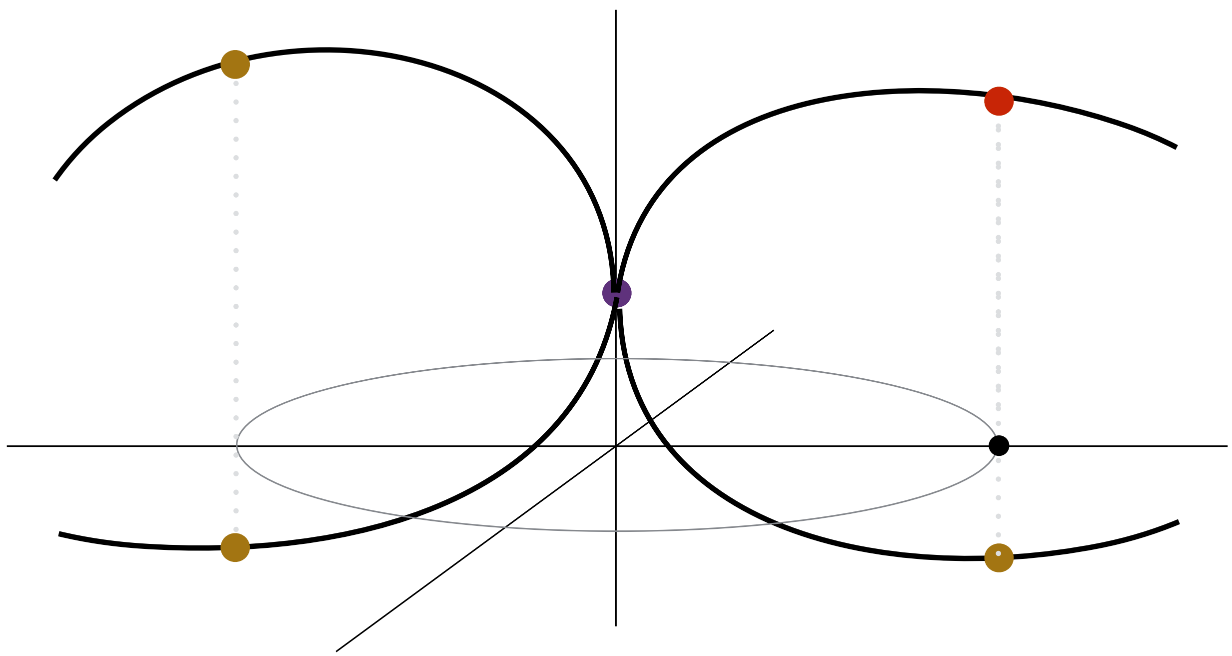
Endgames

We cannot avoid singularities and ill-conditioning at $t = 0$.

Interesting systems often have singular solutions, but we cannot track into $t = 0$. We use endgames built from interpolation or complex analysis.

Two main options:

1. Power series endgame
2. Cauchy endgame



Finding isolated solutions

<u>Bertini</u>

Bertini is free to download and use, and the source code is publicly available. It may be downloaded (source code or binary) at <u>bertini.nd.edu</u>.

2002: Project started
2006: Initial beta release, during IMA special year.
2006-2015: Various extensions, releases.
2015: Latest version (1.5.2).
~ 8 downloads/day
~ 300 citations

Bates Brake Hauenstein Sommese Wampler

<u>**Next up**</u>: Bertini 2.0 (b2)
Large development team (you??), C++, Boost, Eigen, GPL3, modules, etc.
More on that soon….

Finding isolated solutions

Bertini Classic I/O

$$h(x,y) = \begin{bmatrix} x^2 + y^2 - 1 \\ x^2 + (y+c)^2 - 1 \end{bmatrix}$$

```
Dans-MacBook-Pro-2:tmp_19may16 bates$ more input


variable_group x,y;
function f,g;

f = x^2 + y^2 - 1;
g = x^2 + (y+1)^2 - 1;

END;

Dans-MacBook-Pro-2:tmp_19may16 bates$ bertini input
```

Finding isolated solutions

(screen output)

```
Finite Solution Summary

NOTE: nonsingular vs singular is based on condition number and identical endpoints

                | Number of real solns |  Number of non-real solns     |  Total
--------------------------------------------------------------------------------------------
Non-singular    |       2              |                0              |   2
Singular        |       0              |                0              |   0
--------------------------------------------------------------------------------------------
Total           |       2              |                0              |   2


Finite Multiplicity Summary

  Multiplicity  |  Number of real solns |  Number of non-real solns
--------------------------------------------------------------------------------------------
        1       |       2               |        0



--------------------------------------------------------------------------------------------
The following files may be of interest to you:

main_data:              A human-readable version of the solutions - main output file.
raw_solutions:          A list of the solutions with the corresponding path numbers.
raw_data:               Similar to the previous, but with the points in Bertini's homogeneous
                          coordinates along with more information about the solutions.
real_finite_solutions: A list of all real finite solutions.
finite_solutions:       A list of all finite solutions.
nonsingular_solutions: A list of all nonsingular solutions.
singular_solutions:     A list of all singular solutions.
--------------------------------------------------------------------------------------------

Paths Tracked: 4
 Truncated infinite paths: 2 - try adjusting SecurityMaxNorm or set SecurityLevel to 1 in the input file
   Please see 'failed_paths' for more information about these paths.
```

Finding isolated solutions

(main_data file)

```
Number of variables: 2
Variables:  x y
Rank: 2
-------------------------
Solution 0 (path number 1)
Estimated condition number: 5.439468605656218e+00
Function residual: 2.694578982019247e-16
Latest Newton residual: 5.945183495476102e-17
T value at final sample point: 3.906250000000000e-04
Maximum precision utilized: 52
T value of first precision increase: 0.000000000000000e+00
Accuracy estimate, internal coordinates (difference of last two endpoint estimates):  3.233556711006548e-13
Accuracy estimate, user's coordinates (after dehomogenization, if applicable):  5.993731656525824e-13
Cycle number: 1
8.660254037844386e-01 8.326672684688674e-17
-4.999999999999999e-01 -1.734723475976807e-17
Paths with the same endpoint, to the prescribed tolerance:
Multiplicity: 1
-------------------------
Solution 1 (path number 3)
Estimated condition number: 7.927793214493077e+00
Function residual: 4.002966042486721e-16
Latest Newton residual: 1.540038302682865e-16
T value at final sample point: 3.906250000000000e-04
Maximum precision utilized: 52
T value of first precision increase: 0.000000000000000e+00
Accuracy estimate, internal coordinates (difference of last two endpoint estimates):  9.267175966288035e-13
Accuracy estimate, user's coordinates (after dehomogenization, if applicable):  1.300027528966579e-12
Cycle number: 1
-8.660254037844386e-01 5.551115123125783e-17
-5.000000000000000e-01 -2.775557561562891e-17
Paths with the same endpoint, to the prescribed tolerance:
Multiplicity: 1
-------------------------
At tol=1.000000000000e-10, there appear to be 2 solutions.
```

Game plan

1. Polynomial systems and their solution sets
2. Finding isolated solutions (homotopy continuation)
3. **Advanced topics for isolated solutions**
4. Finding positive-dimensional solution sets (briefly)

Game plan

3. **Advanced topics for isolated solutions**
   A. Parameter homotopies
   B. Certification
   C. Regeneration

Advanced topics for isolated solutions

Parameter homotopies

Given N polynomials, $f = (f_1(z,q)\ldots, f_N(z,q))$ in N variables $z \in \mathbb{C}^N$ and k parameters $q \in \mathbb{C}^k$, suppose we want to solve $f = 0$ at M points in parameter space, $q_1, \ldots q_M \in \mathbb{C}^k$, for some M>>0.

Option A (naïve): Solve at each of the M points from scratch, using a standard homotopy (such as a total degree homotopy).

Let P = # paths required to solve at each point in parameter space,
L = lower bound on # divergent paths for each run from scratch.

paths: MP
wasted: ML

Advanced topics for isolated solutions

Parameter homotopies

Option B (parameter homotopy):

1. Solve $f = (f_1(z, q') \ldots, f_N(z, q'))$ for randomly chosen q' $\in \mathbb{C}^k$.
2. Homotope from q' to q_i for i=1,…,M, following all finite solutions from the Step 1 solve at q'.

paths: P + M(P-L)
wasted: P

See Li-Sauer-Yorke, *SIAM J. Num. An.* '89 and Morgan-Sommese, Appl. Math. and Comp. '89.

Advanced topics for isolated solutions

Parameter homotopies

Example: P = 10, L = 5, M = 1,000,000:

Option A
paths: MP = 10,000,000
wasted: ML = 5,000,000

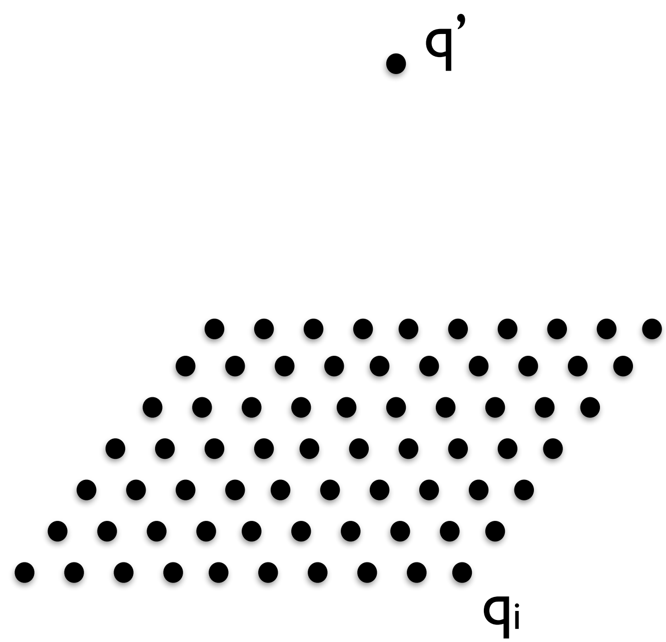
Option B
paths: P + M(P-L) = 5,000,010
wasted: P = 10

Of course, if L=0 (not common), Option A is marginally better, at least in terms of # paths.

Advanced topics for isolated solutions

Parameter homotopies

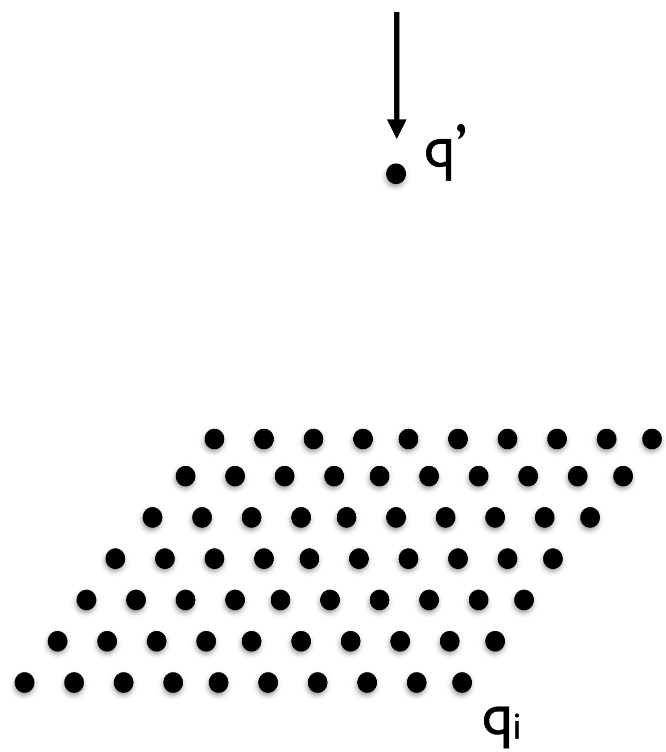
Schematic



Advanced topics for isolated solutions

Parameter homotopies

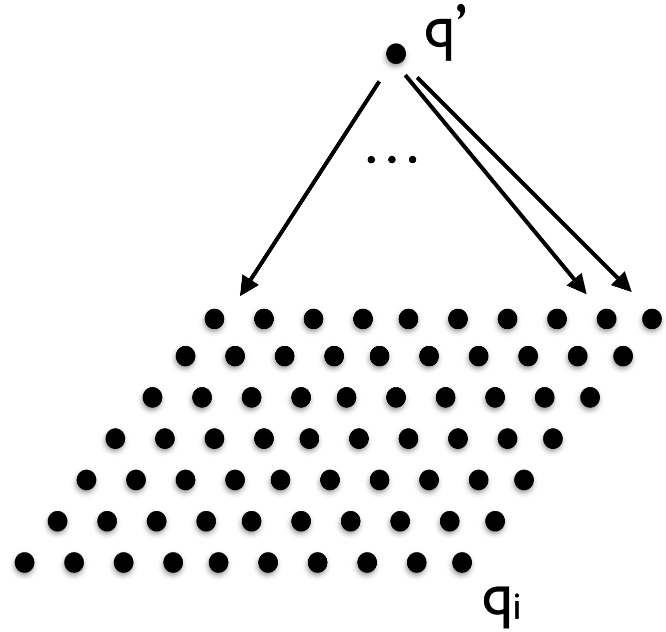
Schematic: Step 1



Advanced topics for isolated solutions

Parameter homotopies

Schematic: Step 2



Advanced topics for isolated solutions

Parameter homotopies

Problem: Standard NAG software options include parameter homotopy capabilities, but:

* parallelization is within the run (not between runs),
* failure mitigation is not automated (a headache), and
* each run includes startup costs, e.g., parsing.

Solution:

Paramotopy

D. Bates, D. Brake (Notre Dame), and M. Niemerg (IBM/Oak Ridge)

www.paramotopy.com

Free to use, source code publicly available

C++, Boost

Uses Bertini as a library

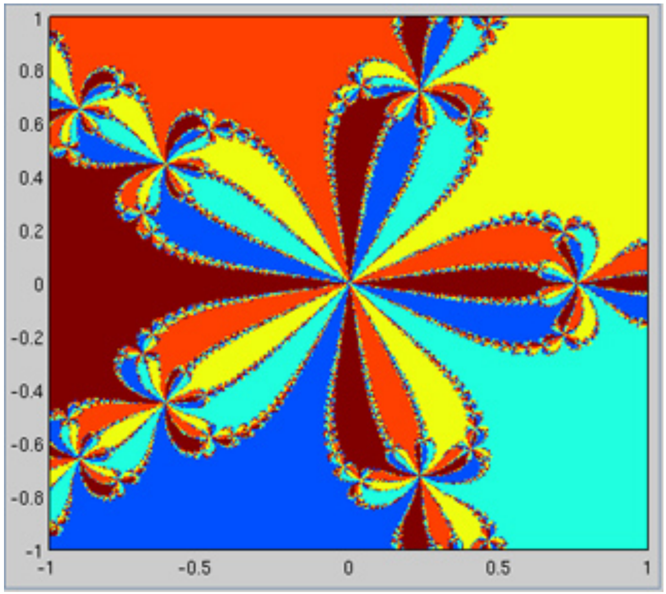
To be incorporated into Bertini 2.x in some form.

Advanced topics for isolated solutions

Certification

Given some approximation of a nonsingular isolated solution of a polynomial system, Newton's method will:

- diverge,
- converge slowly, or
- converge quadratically.



Basins of attraction for $z^5 + 1 = 0$

from MIT Open Courseware (Intro to Matlab Programming)

Advanced topics for isolated solutions

Certification

Given some approximation of a nonsingular isolated solution of a polynomial system, Newton's method will:

diverge,
converge slowly, or
converge quadratically.

The region of quadratic convergence is quite small.

It is possible to *prove* that an approximation is within this region of quadratic convergence for some solution. This is the crux of alpha theory, though the certifiable region is extremely small!

Advanced topics for isolated solutions

Certification

Two basic options (more introduced recently):

1. Certify as you go through a homotopy (Beltran-Leykin, NAG4M2) or
2. Post-certify (Hauenstein-Sottile, alphaCertified).

See the upcoming talk of Nick Hein & Alan Liddell.

Advanced topics for isolated solutions

Regeneration

This is the latest equation-by-equation solver. Replaces standard homotopy continuation with a number of simpler homotopies.

$$\begin{bmatrix} f_1 \\ L_2^{(1)} \\ L_3^{(1)} \\ \vdots \\ L_N^{(1)} \end{bmatrix} \to \begin{bmatrix} f_1 \\ L_2^{(2)} \\ L_3^{(1)} \\ \vdots \\ L_N^{(1)} \end{bmatrix} \to \cdots \to \begin{bmatrix} f_1 \\ L_2^{(d_2)} \\ L_3^{(1)} \\ \vdots \\ L_N^{(1)} \end{bmatrix} \qquad \begin{bmatrix} f_1 \\ \prod_{j=1}^{d_2} L_2^{(j)} \\ L_3^{(1)} \\ \vdots \\ L_N^{(1)} \end{bmatrix} \qquad \begin{bmatrix} f_1 \\ \prod_{j=1}^{d_2} L_2^{(j)} \\ L_3^{(1)} \\ \vdots \\ L_N^{(1)} \end{bmatrix} \to \begin{bmatrix} f_1 \\ f_2 \\ L_3^{(1)} \\ \vdots \\ L_N^{(1)} \end{bmatrix}$$

See Hauenstein-Sommese-Wampler, *Math. Comp.* '10 for more details.

Game plan

1. Polynomial systems and their solution sets
2. Finding isolated solutions (homotopy continuation)
3. Advanced topics for isolated solutions
4. **Finding positive-dimensional solution sets (briefly)**

Game plan

4. **Finding positive-dimensional solution sets (briefly)**

   A. Slicing and the numerical irreducible decomposition

   B. Bertini Classic I/O

   C. Sampling

   D. Real solutions

Finding positive-dimensional solution sets

Recall: $Z = \mathcal{V}(f) = \bigcup_{i=0}^{D} Z_i = \bigcup_{i=0}^{D} \bigcup_{j \in \Lambda_i} Z_{i,j}$, where:

D is the dimension of Z,
i cycles through possible dimensions of irreducible components,
j is an index within dimension i, and the
$Z_{i,j}$ are the irreducible components.

(This is the irreducible decomposition of Z.)

For each positive-dimensional irreducible component, $Z_{i,j}$, we aim to find numerical approximations to *some number* of generic points on $Z_{i,j}$.

Finding positive-dimensional solution sets

Key fact: Given irreducible component $Z_{i,j}$ of dimension i, for almost every choice of linear space L of codimension i, $Z_{i,j}$ intersects L in a set of a particular number of points. That number is the **degree** of $Z_{i,j}$.

So, to find deg($Z_{i,j}$) points on $Z_{i,j}$, we can append i linears to f. We refer to this operation as **slicing**.

To find points on all components, we can just loop through for all i.

Finding positive-dimensional solution sets

Problem 1: We could pick up points on higher-dimensional components.
Problem 2: We could find points on multiple i-dimensional components.

Example: Suppose there are two curves and a surface. When we slice for the curves, we will find points on both curves and also on the surface.

Solution 1: Start at the top dimension and work your way down. Use a **membership test** on points in lower dimensions to see if they sit on the higher-dimensional components already found.

Solution 2: Carry out an equidimensional decomposition, using **monodromy** and the **trace test**.

Finding positive-dimensional solution sets

In fact, there is a clever way to string the homotopies together, called a cascade of homotopies. (There are more recent approaches, too.)

All told, the goal is to have deg $Z_{i,j}$ witness points on each component $Z_{i,j}$, yielding witness point set

$$W_{i,j} = Z_{i,j} \cap L_i$$

For each component, put the linear functions, the witness points, and the original functions together and you have a witness set for the component:

$$\mathcal{W}_{i,j} = (f, L_i, W_{i,j})$$

Then, the numerical irreducible decomposition is the union of all such sets for all irreducible components:

$$\mathcal{W} = \bigcup_{i=0}^{D} \mathcal{W}_i = \bigcup_{i=0}^{D} \bigcup_{j \in \Lambda_i} \mathcal{W}_{i,j}$$

Recall Example 8 again:

Finding positive-dimensional solution sets

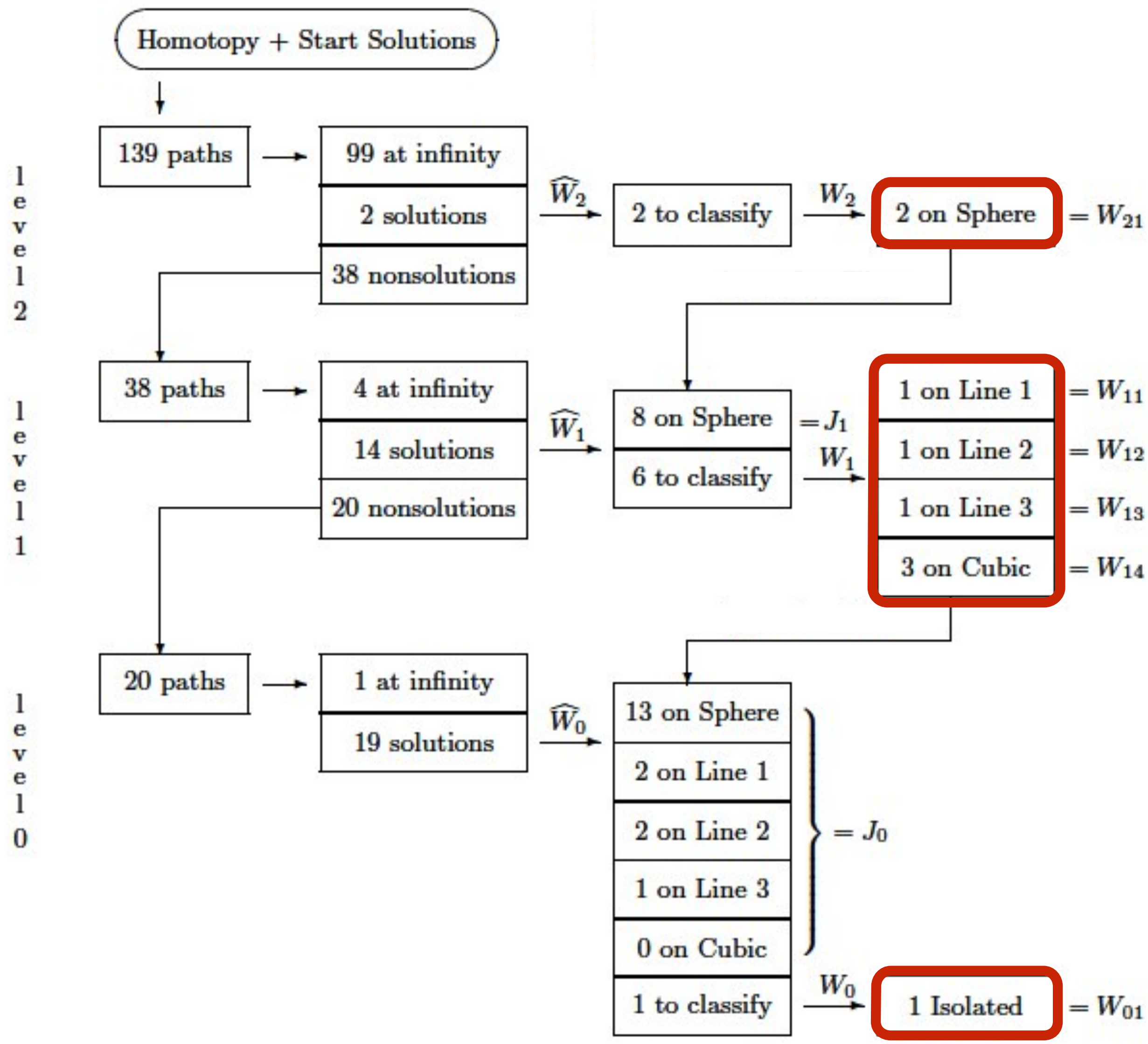
Example 8:

$$\begin{bmatrix} (y-x^2)(x^2+y^2+z^2-1)(x-2) \\ (z-x^3)(x^2+y^2+z^2-1)(y-2) \\ (z-x^3)(y-x^2)(x^2+y^2+z^2-1)(z-2) \end{bmatrix}$$

Solutions:

Dimension 2: One surface
Dimension 1: Three lines and one cubic curve
Dimension 0: One point

Finding positive-dimensional solution sets



From Sommese-Verschelde-Wampler, *SIAM J. Num. An.* '01.

Cascade of homotopies for computing the numerical irreducible decomposition of the illustrative example.

Notice that we find the degree number of points on each component.

Finding positive-dimensional solution sets

Bertini Classic I/O

(screen output)

```
Dans-MacBook-Pro-2:tmp_19may16 bates$ more input
CONFIG

TrackType: 1;

END;

INPUT

variable_group x,y,z;
function f,g,h;

f = (y-x^2)*(x^+y^2+z^2-1)*(x-2);
g = (z-x^3)*(x^+y^2+z^2-1)*(y-2);
h = (z-x^3)*(y-x^2)*(x^+y^2+z^2-1)*(z-2);

END;
```

```
************* Witness Set Decomposition *************

| dimension | components | classified | unclassified
-----------------------------------------------------
|   2       |   1        |   2        |   0
|   1       |   4        |   6        |   0
|   0       |   1        |   1        |   0
-----------------------------------------------------

************** Decomposition by Degree **************

Dimension 2: 1 classified component
-----------------------------------------------------
   degree 2: 1 component

Dimension 1: 4 classified components
-----------------------------------------------------
   degree 1: 3 components
   degree 3: 1 component

Dimension 0: 1 classified component
-----------------------------------------------------
   degree 1: 1 component

*****************************************************
```

Finding positive-dimensional solution sets

Sampling

By following the solutions as we move the linears from a witness set, we pick up more points on the same irreducible component. This is called **component sampling**.

In this way, you can find many points on a curve (or other irreducible component) quite rapidly.

Finding positive-dimensional solution sets

Real solutions

For *real* applications, people typically want *real* solutions!

This is very difficult. There are several recent, limited techniques using homotopy methods:

* Khovanskii-Rolle continuation for fewnomials (Bates-Sottile, KhRo)

* Seidenberg-like methods (Hauenstein)

* Real cellular decompositions (Lu-Bates-Sommese-Wampler, Besana-Di Rocco-Hauenstein-Sommese-Wampler, Bates-Brake-Hao-Hauenstein-Sommese-Wampler, BertiniReal)

There are non-homotopy numerical methods, too (cellular exclusion, cylindrical decomposition, etc.).

This is still a major open problem!

Finding positive-dimensional solution sets

Here's a sketch of how Bertini Real works: Curves first:

Given

1. a witness set for complex curve Z and
2. randomly chosen projection $\pi = Az$ (A random),

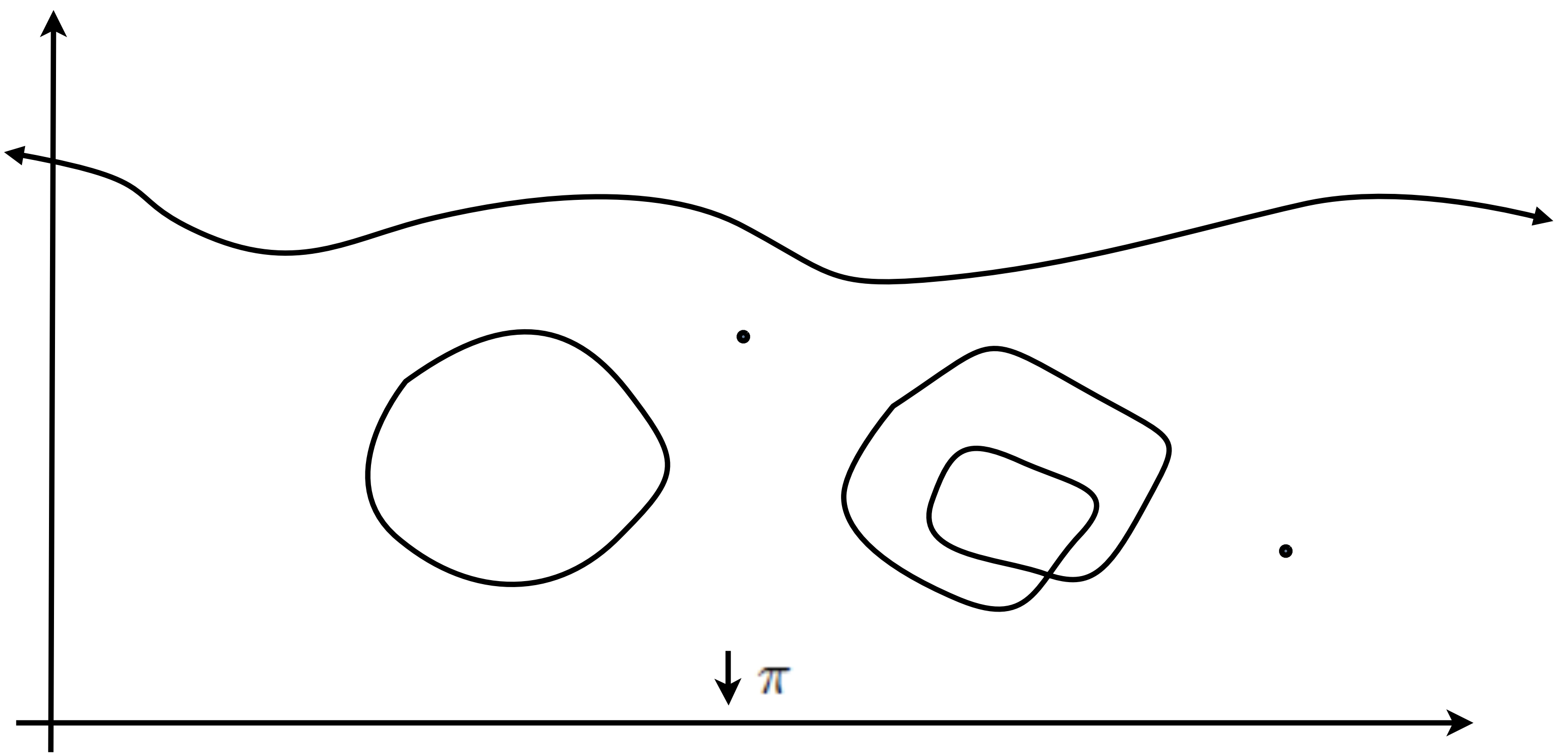
we want to find real points or curves inside Z.

The isolated real solutions and the critical points of any real curve will be isolated real solutions of

$$\mathrm{Crit}(Z,\pi) := Z \cap V\left(\det \begin{bmatrix} Jf \\ A \end{bmatrix}\right)$$

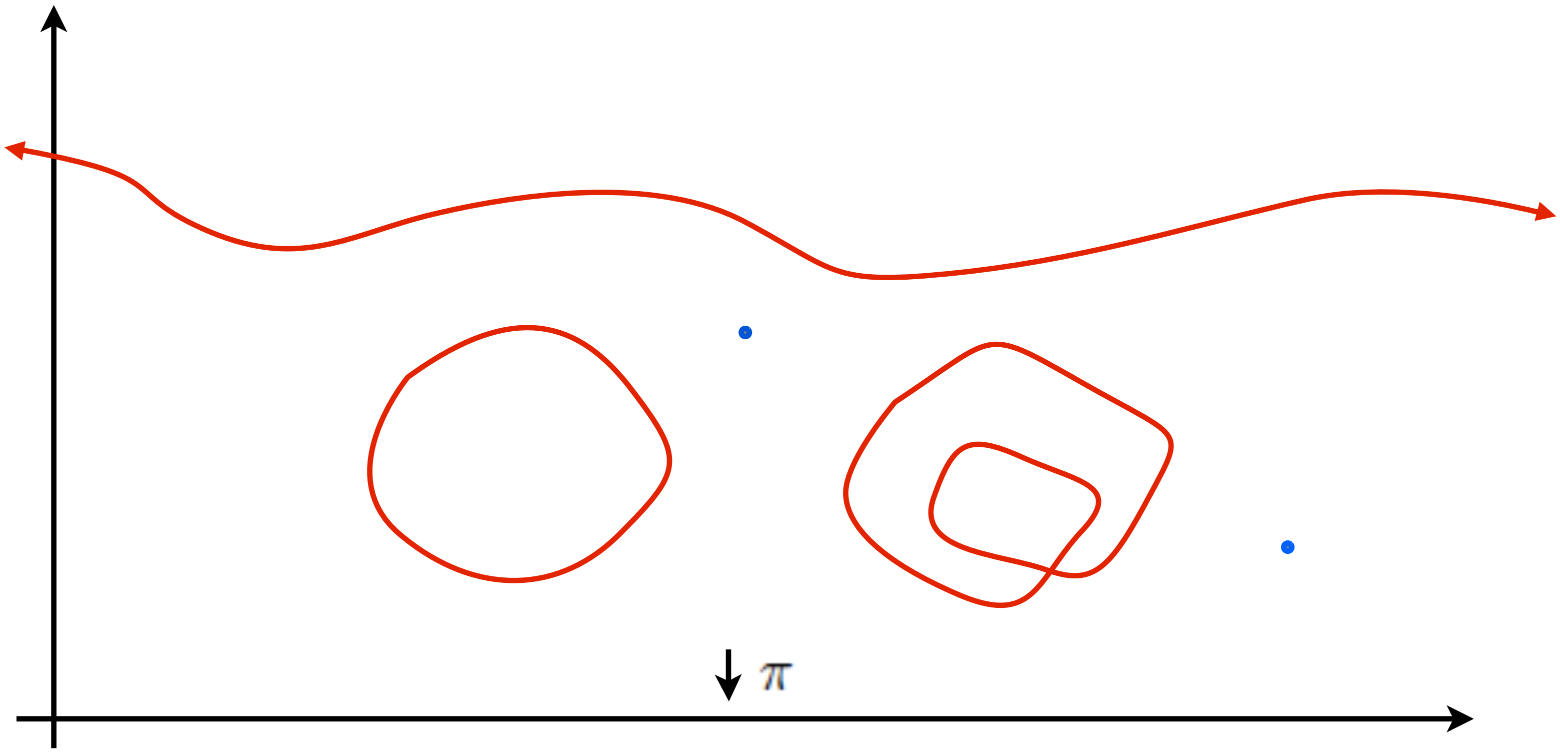
Finding positive-dimensional solution sets

This last bit is a polynomial system that we can solve. Let's switch to a schematic drawing, in $\mathbb{R}^2$ for simplicity:



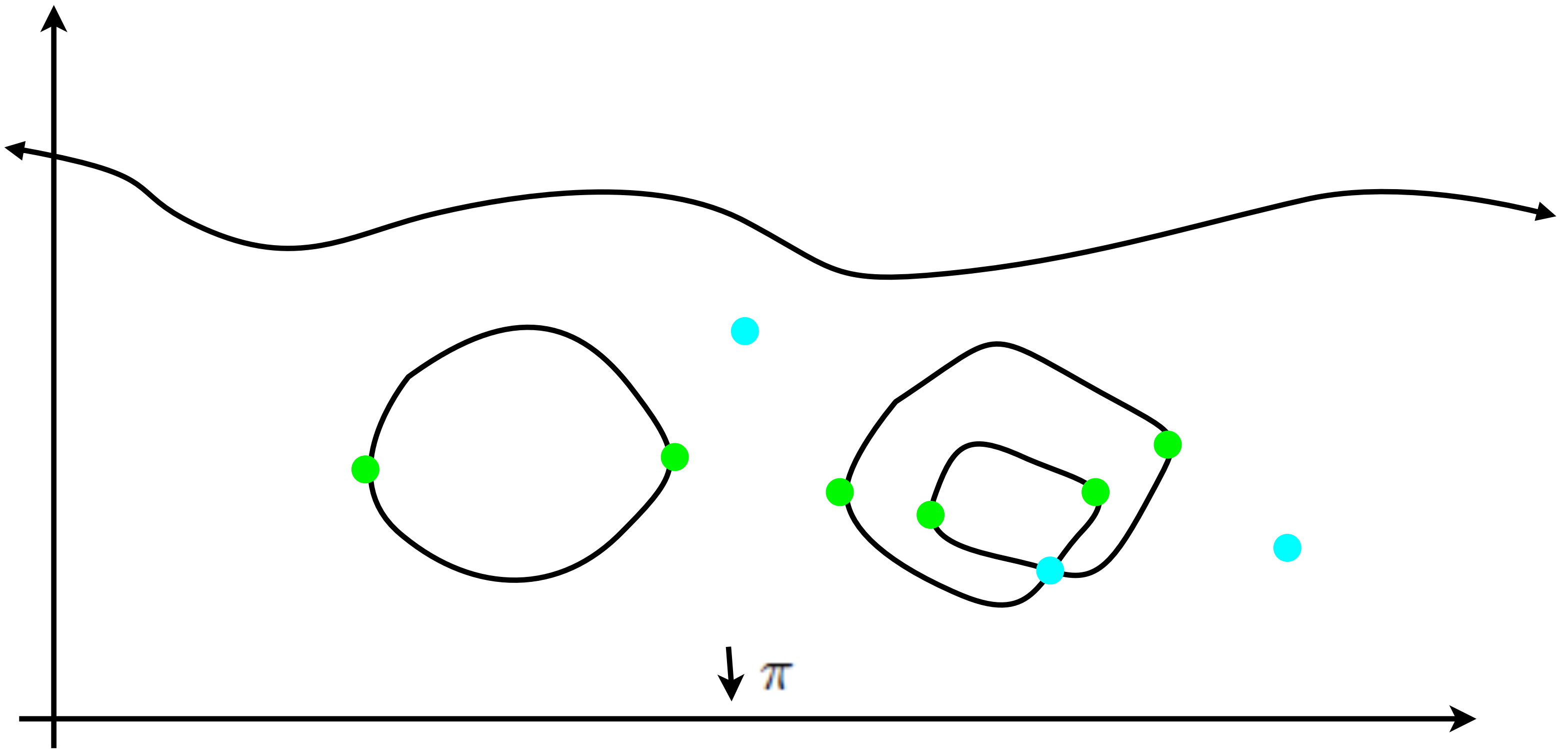
Finding positive-dimensional solution sets

Notice that there are three real curves (two compact) and two points that we want to find.



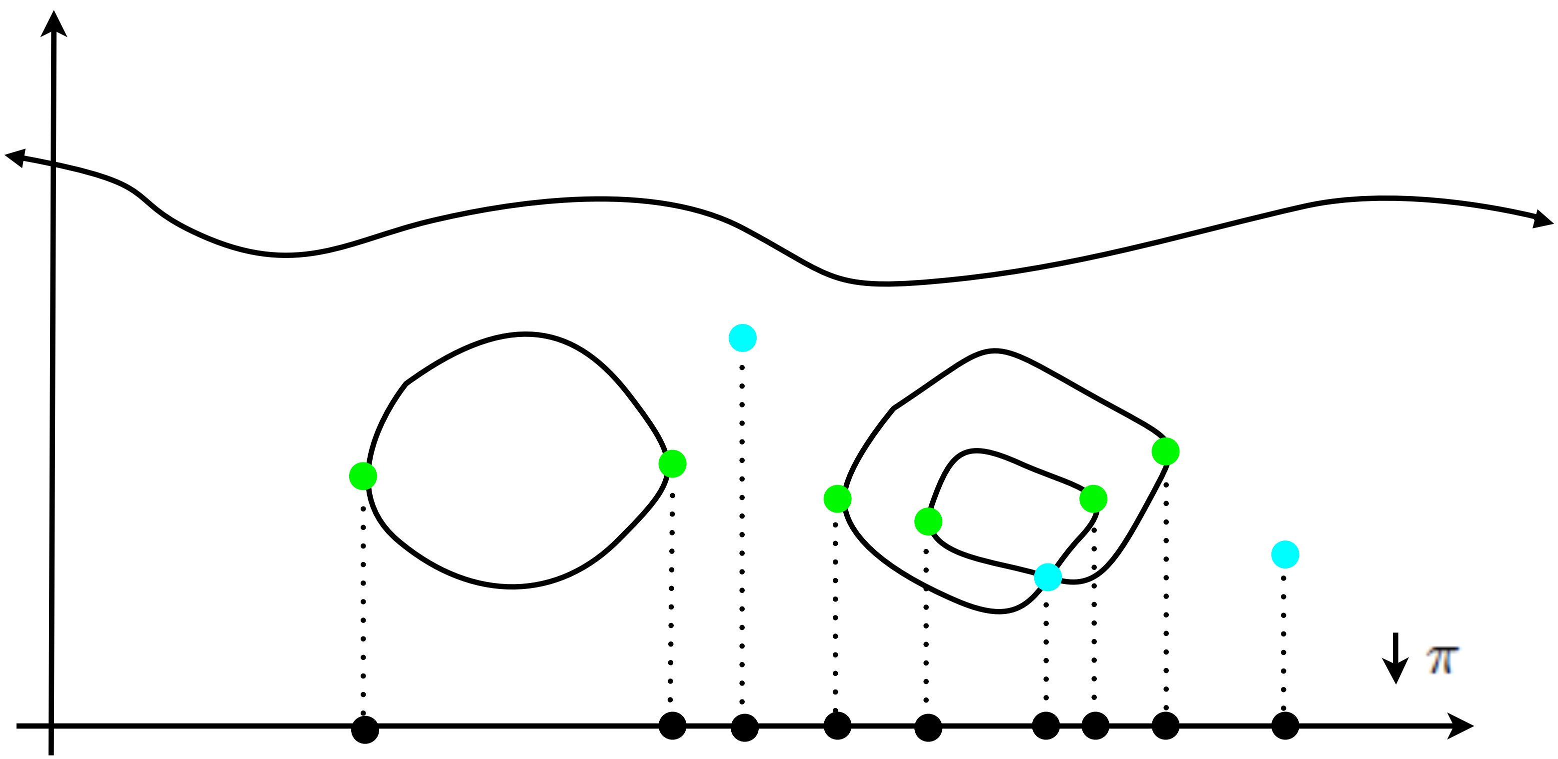
Finding positive-dimensional solution sets

The polynomial system three slides back will find all isolated real points and singular points of the projection (where topology changes occur).



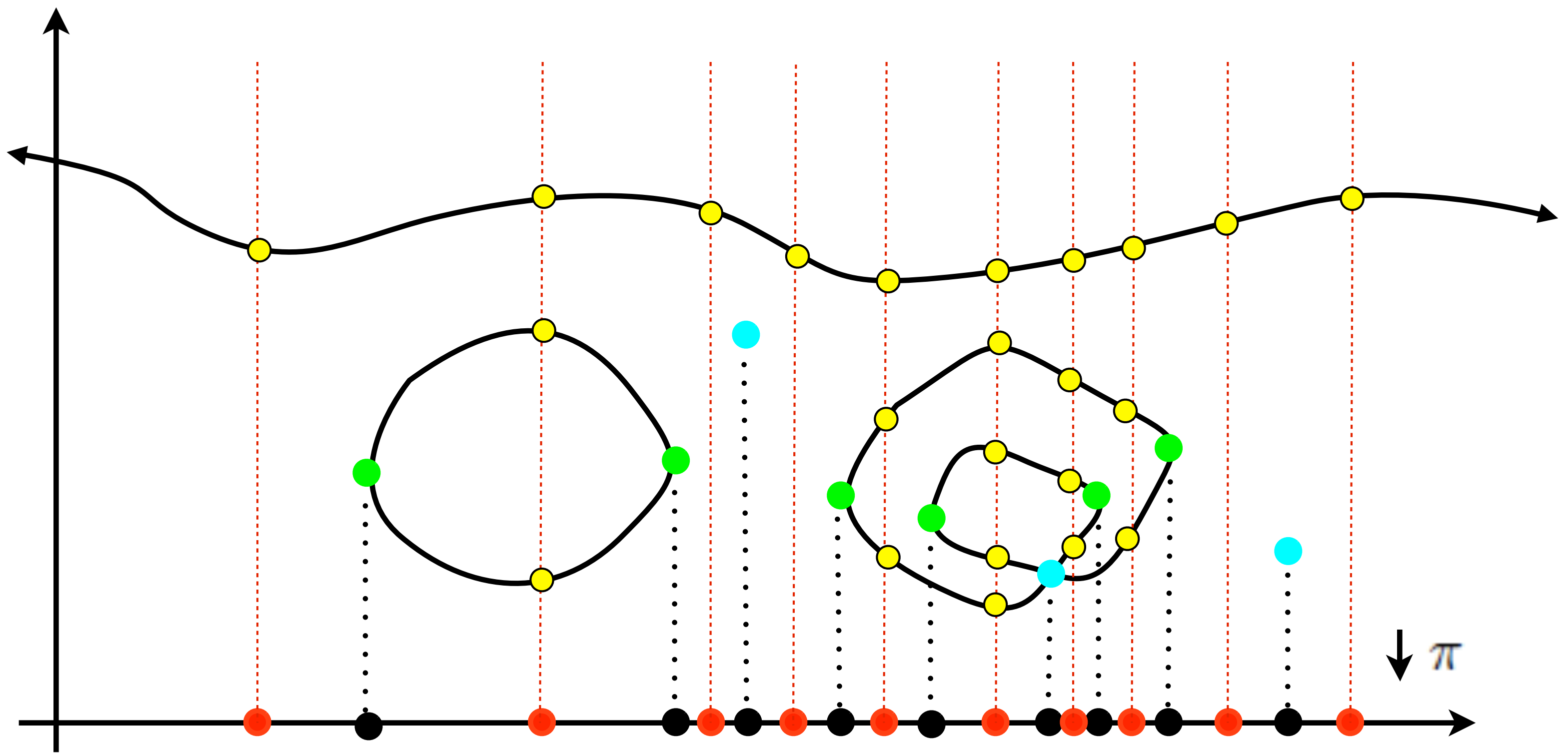
Finding positive-dimensional solution sets

Projecting these down via π (just function evaluation), we get points "downstairs" with fibers containing singular points. Nothing "interesting" happens between these points.



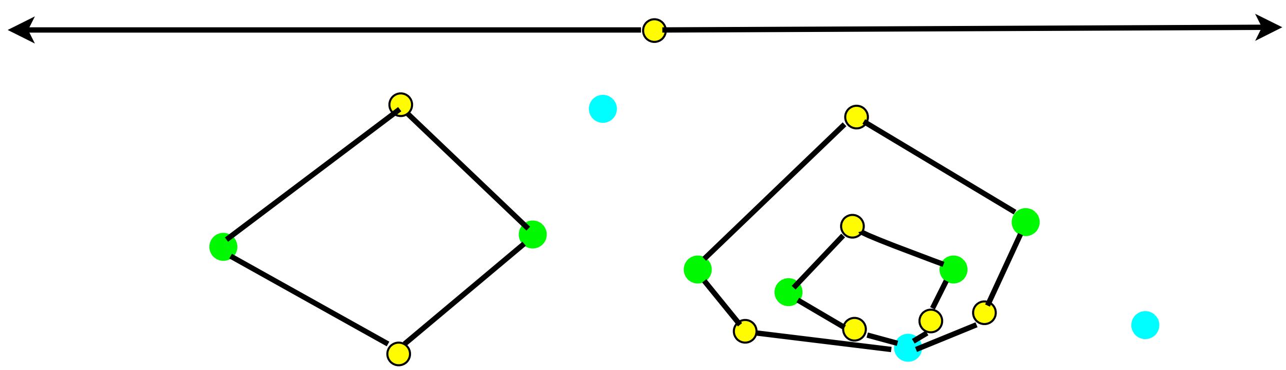
Finding positive-dimensional solution sets

Choosing midpoints between these points "downstairs," we may find all points in the fiber by moving the slices of the witness set for Z to the appropriate linears. There is a special (parameter) homotopy for this move.



Finding positive-dimensional solution sets

Moving these fiber points back and forth (moving the midpoints downstairs to the left and right to the black points), we play connect-the-dots and get a schematic representation of the curve. If the real curve is singular deflation is necessary.



Finding positive-dimensional solution sets

Now for the surface case:

The surface case relies heavily on the curve case.

In the *curve* case, we find projections of critical points, then connect the dots in the fiber.

In the *surface* case, we find projections of critical points, construct critical lines (giving critical curves), then connect the 2-D patches to the critical curves.

Here's a picture:

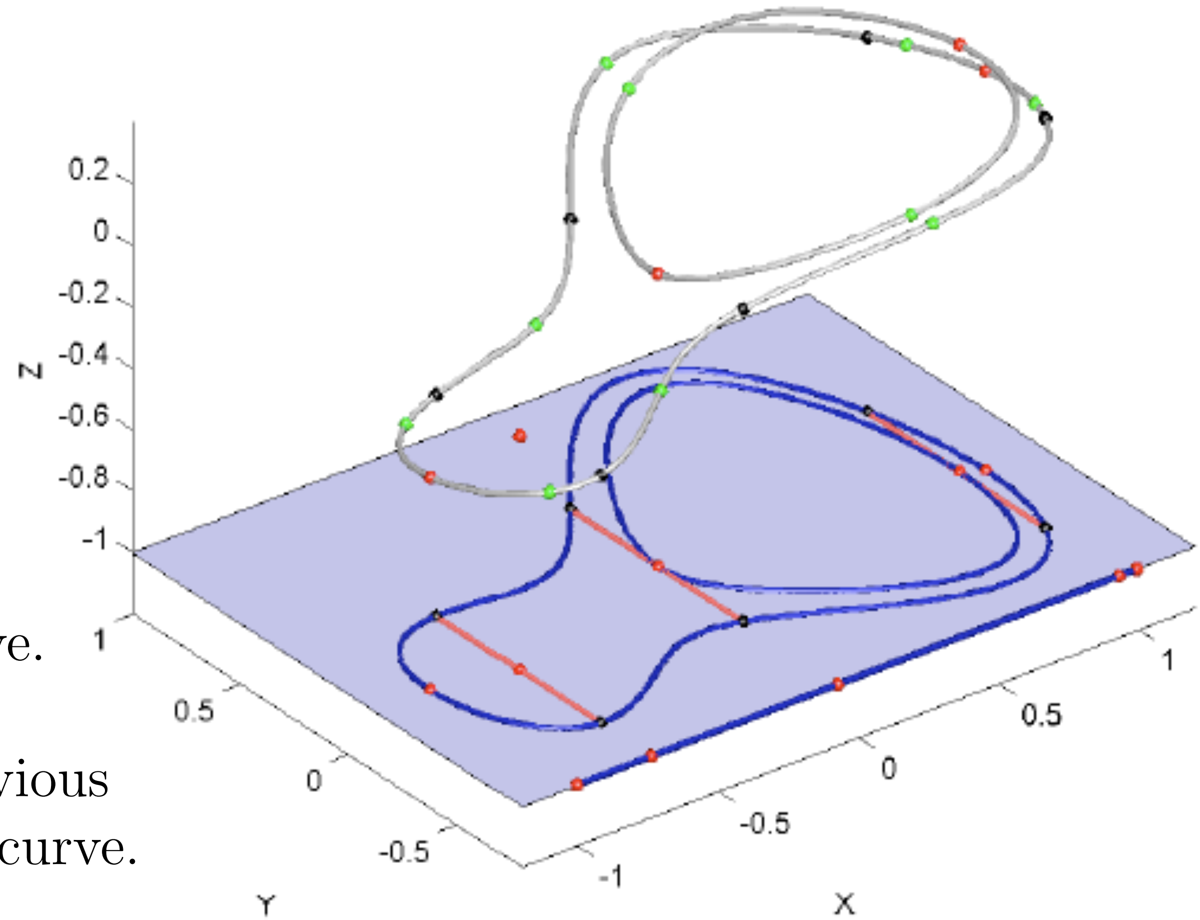
Finding positive-dimensional solution sets

This surface has a hole and a pinchpoint.

Red dots on x-axis correspond to topology changes upstairs.

The blue curve on the floor is the projection of the critical curve.

We use the curve algorithm (previous slides) to decompose the critical curve.

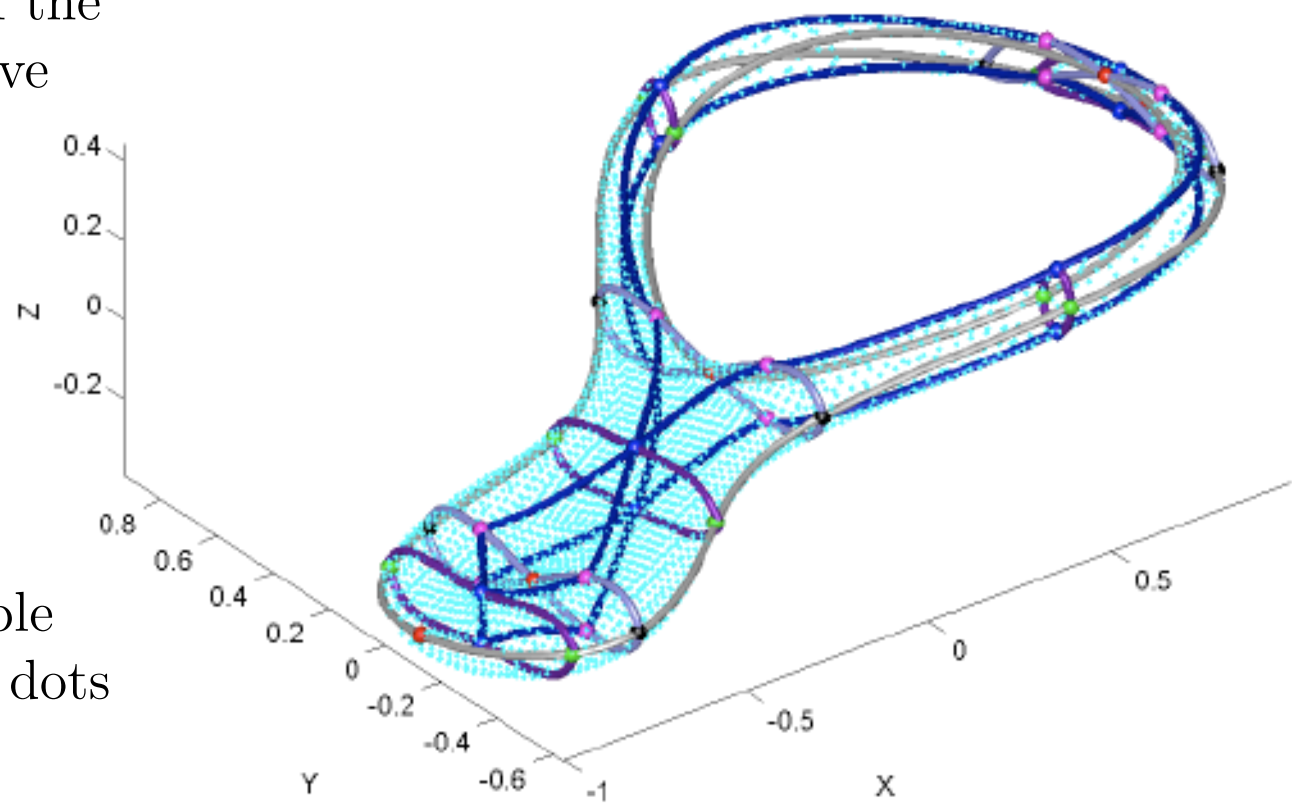


Finding positive-dimensional solution sets

Here, the gray arcs lie above the critical slices in the floor of the previous figure. Again, the curve algorithm is used for this.

The purple arcs lie above undrawn midpoints between critical points.

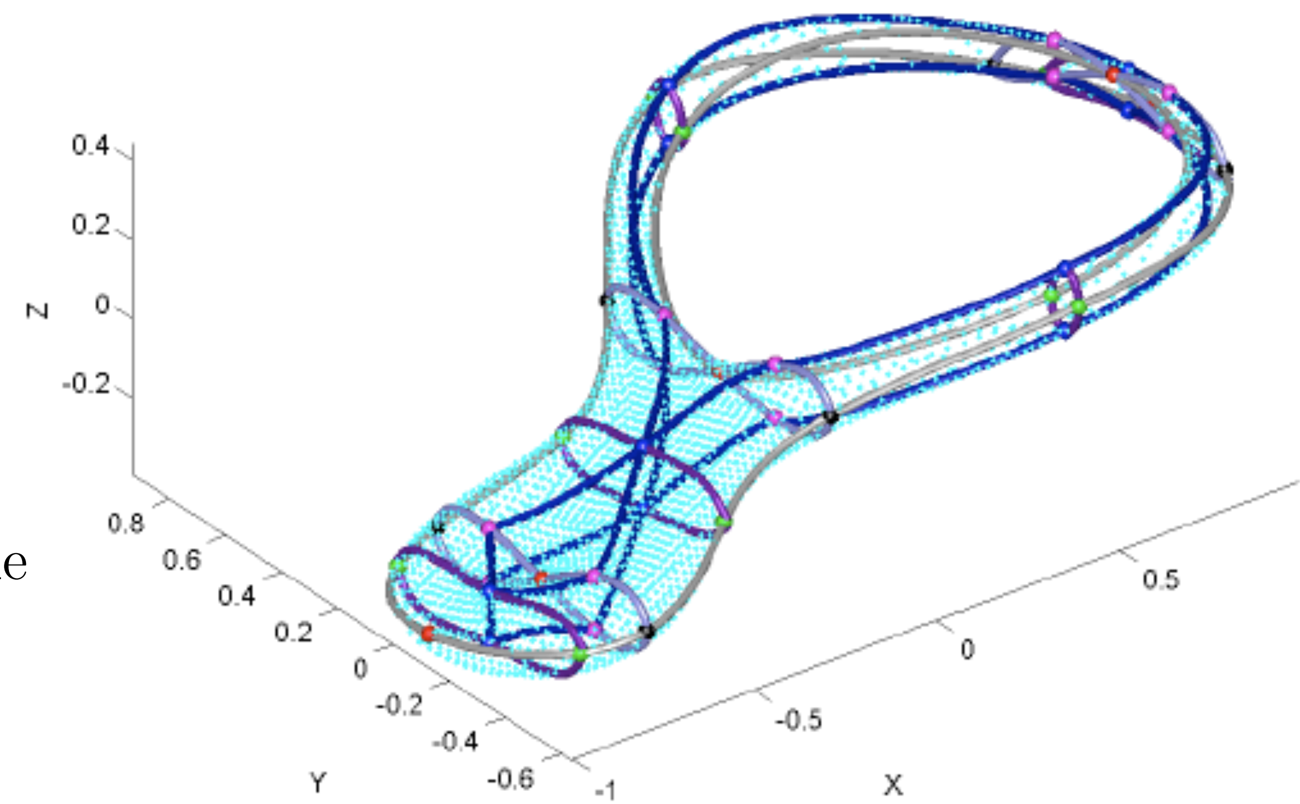
The blue arcs show how the midpoints of the gray and purple arcs connect up. All light blue dots were found via sampling.



Finding positive-dimensional solution sets

The hardest part of the surface case is playing "connect-the-edges."

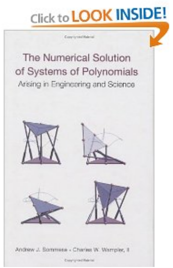
For this, we build a homotopy that makes critical points stay on the critical curve and moves the 2D cell midpoint to the edge midpoints.



See Dan Brake's talk for how to make use of this….

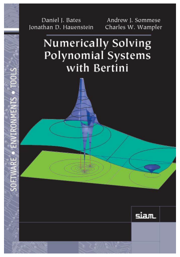
Finding positive-dimensional solution sets

<u>General references for numerical algebraic geometry</u>



Sommese-Wampler, *Numerical solution of polynomial systems arising in science and engineering,* World Scientific, 2005.

Bates-Hauenstein-Sommese-Wampler, *Numerically solving polynomial systems with Bertini,* SIAM, 2013.



<u>More specific references</u>:

<u>Path-tracking</u>: Allgower-Georg, SIAM, 2004.
<u>Polyhedral methods</u>: Li, *Acta Numerica*, 2003.
<u>Algebraic kinematics</u>: Sommese-Wampler, *Acta Numerica*, 2013.
Many more — just ask!

THANKS!!!

(These slides are already posted on my website.)

Partially supported by NSF ACI-1440467, FA8650-13-1-7317, and the IMA.