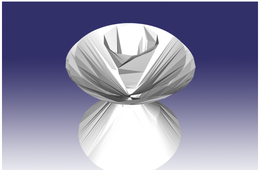
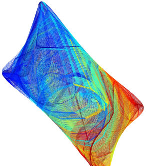
Numerically decomposing algebraic surfaces with an infinite number of singularities

Daniel Brake
with
Dan Bates, Wenrui Hao, Jon Hauenstein,
Andrew Sommese, and Charles Wampler

18 November, 2014





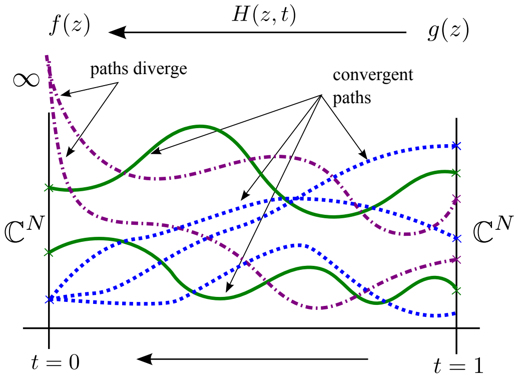
Polynomial Systems

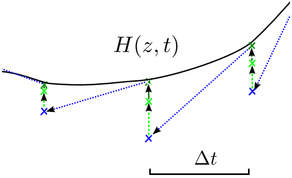
Polynomials are everywhere!

- chemical reaction networks
- kinematics
- dynamical systems
- optimization
- many more

How to solve a system $f(z)$? That's **hard**!

Homotopy Methods





Start Systems

There are lots of ways to make start system g.

- Roots of unity - $g_i = (z_i - 1)_i^d$
- Total degree - $g_i = \prod^{d_i} \mathcal{L}(\vec{x})$
- Multi-homogeneous - $g_i = \prod \mathcal{L}(\vec{x}_\sigma) \cdot \prod \mathcal{L}(\vec{x}_\delta) \cdot \ldots$
- Polyhedral - g_i determined by the Newton Polytope

The canonical linear homotopy:

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

Witness Sets

Witness set - Numerical Algebraic Geometry's representation of positive dimensional components C.

Consists of three things:

- points x – generic, nonsingular. $\# = \text{degree}(C)$.
- linears L – generic. $\# = \dim(C)$.
- system f – possibly randomized, Jf is full rank on each x.

Positive dimensional $\mathbb{C}$ components are nice - degree and structure are same almost everywhere.

$\mathbb{R}$ not so much...

Real Components

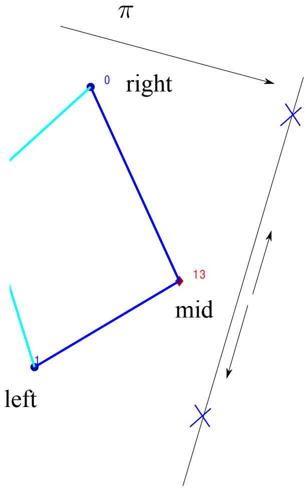
Bertini_real – Numerical Cellular Decomposition

Setup

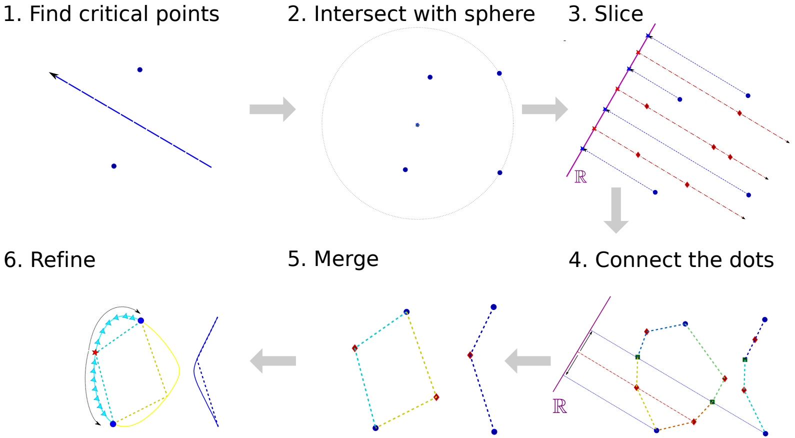
- Let f be a polynomial system with $\mathbb{R}$ coefficients, and $f : \mathbb{C}^N \to \mathbb{C}^n$.
- Let $V(f)$ be the variety of f.
- Consider $C \subseteq V(f)$ be a component of dimension k.
- If f is overdetermined, replace f by a randomized version of itself.

Our objective is to decompose the real part of C; i.e., $C \cap \mathbb{R}^N$

Curve Cell

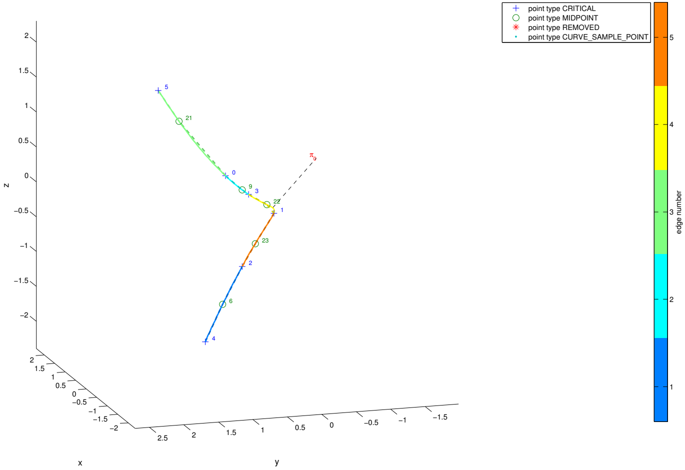


Curves



[Lu,Bates,Sommese,Wampler,2006]

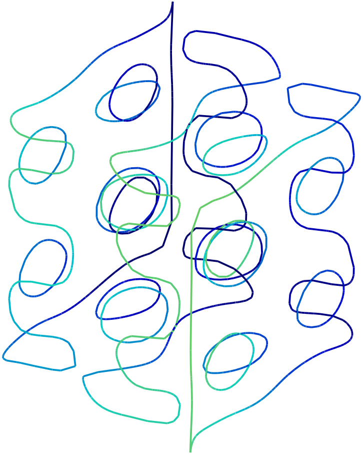
Curves



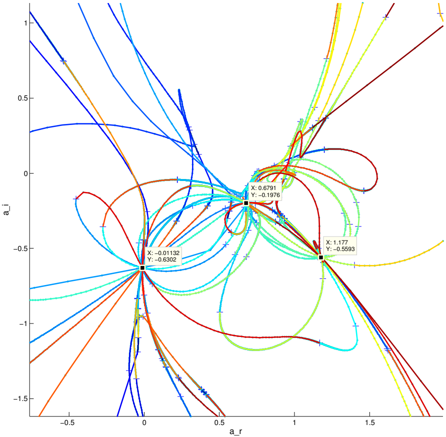
twisted cubic

$$f(x,y,z) = \begin{bmatrix} y - x^2 \\ z - x^3 \\ y^2 - xz \end{bmatrix}$$

Curves



Curves



Burmester 3-3 curve
dimension 14. In 2d projection, of degree 128.

Critical points of curves

Computing critical points of curves is easy.

Since f defines a dimension-one component, the working witness set comes with one random linear:

$$\begin{bmatrix} f \\ \mathcal{L}_1 \end{bmatrix}$$

Then we use regeneration to solve the system

$$\begin{bmatrix} f \\ \det \begin{pmatrix} Jf \\ J\pi_1 \end{pmatrix} \\ \text{patch}_v \end{bmatrix}$$

[Hauenstein,Sommese,Wampler, 2009]

Regeneration to find crit

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

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

- δ is the maximum degree any polynomial in f.

We have a new result supporting this computation – a single homotopy of this form will find all isolated solutions in terms of x variables, regardless of the fiber dimension.

Building a start system

- To form the start system and solutions, we move the given random complex $\mathcal{L}(x)$ to each $L_{j,i}$ one at a time, to find the x coordinates:

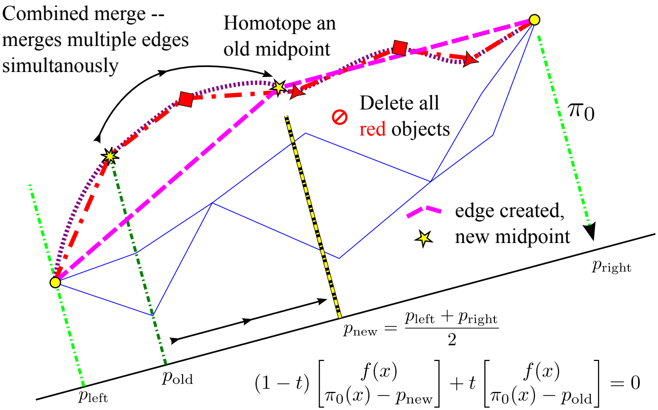
$$H(x;t) = (1-t)\begin{bmatrix} f \\ L_{j,i} \end{bmatrix} + t\begin{bmatrix} f \\ \mathcal{L}_1 \end{bmatrix} = 0$$

- Since only one $L_{j,i}$ vanishes for each start point, the remainder of the start functions must vanish due to $M_j(v)$, which are solved for v start values by matrix inversion:

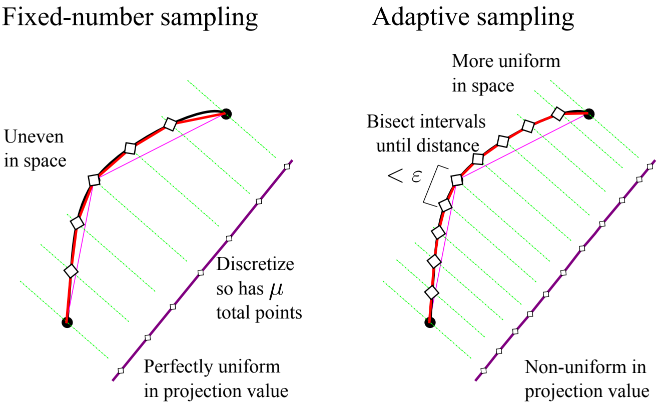
$$v = \begin{bmatrix} M_{1\neq j} \\ \vdots \\ M_{N\neq j} \\ \text{patch}_v \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

Then we take the x solution from the top, and v from bottom, and concatenate to form the start point.

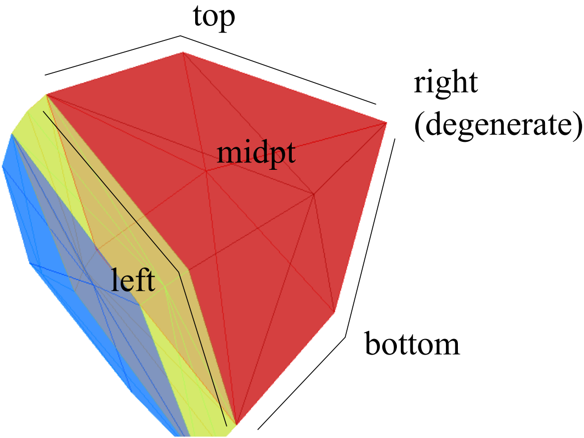
Curve Merging



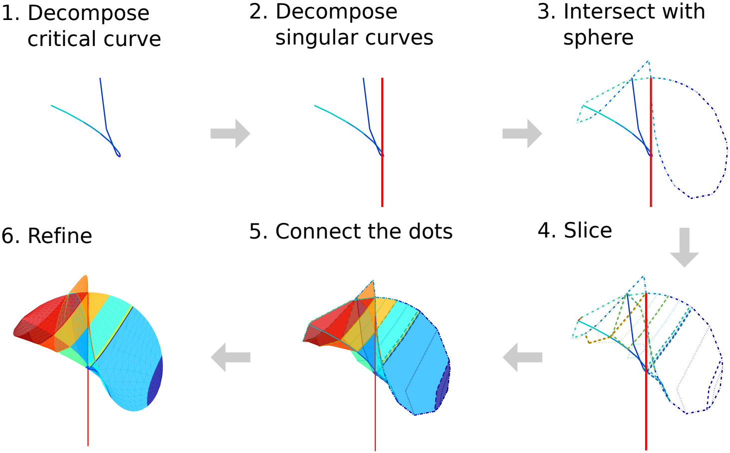
Curve Sampling



Surface Cell

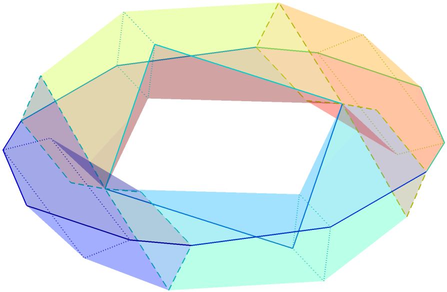


Surface Decomposition



[Besana,Di Rocco,Hauenstein,Sommese,Wampler, 2013]

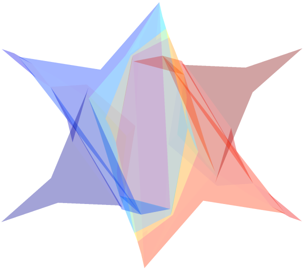
Surface examples



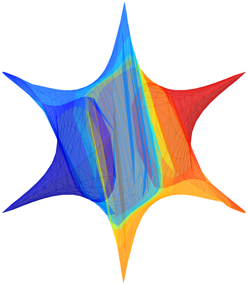
torus

$$f(x,y,z) = (x^2 + y^2 + z^2 + 2^2 - \frac{1}{2}^2)^2 - 16(x^2 + y^2)$$

Surface examples



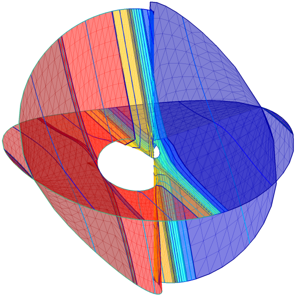
distel, unrefined



distel, refined

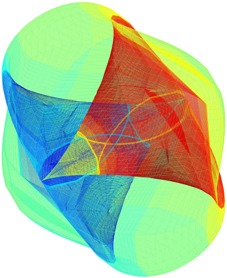
$$f(x,y,z) = x^2 + y^2 + z^2 + 1000(x^2 + y^2)(x^2 + z^2)(y^2 + z^2) - 1$$

Surface examples



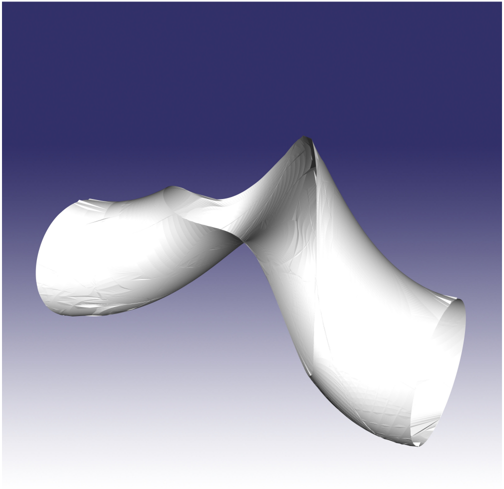
solitude

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



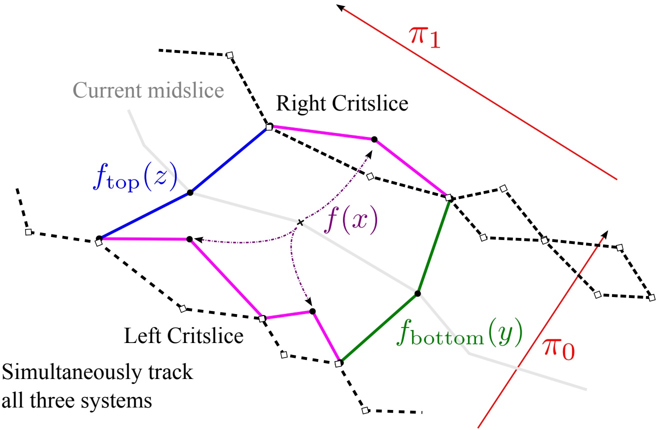
klein

$$f(x,y,z,w) =$$
$$\begin{bmatrix} w^2 + x^2 + y^2 + z^2 - 1 \\ 2wyz - x(y^2 + z^2) \end{bmatrix}$$



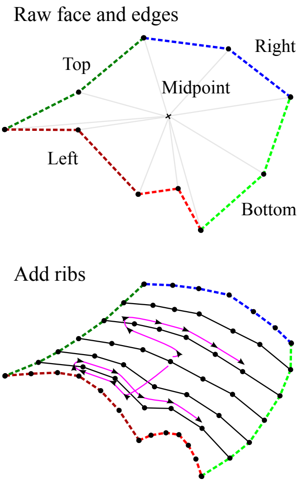
Fivebar mechanism kinematics
Six-dimensional trigonometric system,
passed though an atan2 projection

Midtracking



Simultaneously track
all three systems

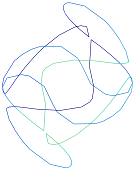
Surface Refinement



Crit points of surfaces

Computing critical points of surfaces is hard.

- Surfaces have *critical curves*.
- Surfaces also have critical points themselves.



Current surface critical method

Using the *determinantal* form of the criticality conditions:

witness set:

$$f_{\text{crit\_curve}} = \begin{bmatrix} f \\ \det \begin{pmatrix} Jf \\ J\pi_1 \\ J\pi_2 \end{pmatrix} \\ \mathcal{L}_1 \end{bmatrix}$$

$$f_{\text{crit\_crit}} = \begin{bmatrix} f \\ \det \begin{pmatrix} Jf \\ J\pi_1 \\ J\pi_2 \end{pmatrix} \\ \det \begin{pmatrix} J \begin{pmatrix} f \\ \det \begin{pmatrix} Jf \\ J\pi_1 \\ J\pi_2 \end{pmatrix} \end{pmatrix} \\ J\pi_1 \end{pmatrix} \\ \text{patch}_v \end{bmatrix}$$

- Determinant operation produces high degree polynomials, contributing to numerical issues.
- Using this formulation for dimension 3 decompositions will be even worse w.r.t. degree and computational complexity.
- Fortunately, we can still use the nullspace method for finding critical points of this curve.

Crit points of crit curve

The actual system Bertini_real currently solves to obtain crit points of crit curve, by regeneration:

$$f_{\text{crit\_crit}} = \begin{bmatrix} f \\ \det \begin{pmatrix} Jf \\ J\pi_1 \\ J\pi_2 \end{pmatrix} \\ v^\intercal \cdot J \begin{pmatrix} f \\ \det \begin{pmatrix} Jf \\ J\pi_1 \\ J\pi_2 \end{pmatrix} \\ J\pi_1 \end{pmatrix} \\ \text{patch}_v \end{bmatrix}$$

How to find singular curves

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

Deflation

Deflation

- making the untrackable, trackable.
- Removing singularity of points.

Multiple flavors of deflation:

- Nullspace - adds **lots** of variables, but no functions

$$g_i(x, \xi_1, \xi_2, \ldots, \xi_d) = \begin{bmatrix} f(x) \\ Jf(x) \cdot (p_1 + A\xi_1) \\ \vdots \\ Jf(x) \cdot (p_d + A\xi_d) \end{bmatrix}$$

Notation - d is dimension of nullspace of Jf.

[Hauenstein, Wampler '13]

Deflation

- Determinental - adds no variables, but **lots** of functions

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

- σ_i is an index set of an $(N-d+1) \times (N-d+1)$ submatrix.
- $m = \binom{n}{N-d+1} \cdot \binom{n}{N-d+1}$

For our decompositions, we use *isosingular deflation*, and the *determinental* formulation

[Hauenstein, Wampler '13]

Isosingular Deflation

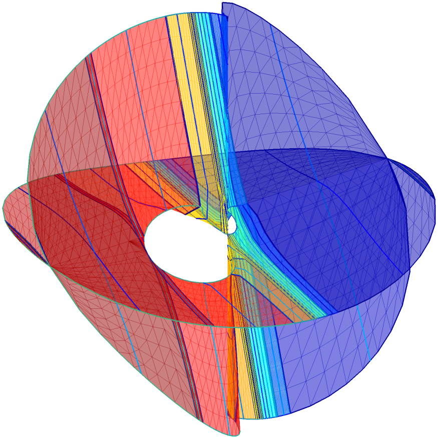
Separate points based on *deflation sequence.*
- dimension of nullspace after r^{th} deflation:

$$\{d_r(f,V)\}_{r=0}^{\infty}$$

Method

1. Compute witness points for critical curve – some may be singular.
2. Deflate surface WRT a singular witness point - get sequence.
3. Test other witness points to see if satisfy deflated system.
4. GOTO 2, for each non-deflated point.
5. Decompose curves, using deflated system.

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

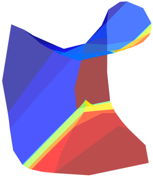
$$
\mathcal{D}^2 = \left[\begin{array}{c}
x^2yz + xy^2 + y^3 + y^3z - x^2z^2 \\
\\
y^2/4 - (xz^2)/2 + (xyz)/2 \\
(xy)/2 + (x^2z)/4 + (3y^2z)/4 + (3y^2)/4 \\
(x^2y)/4 - (x^2z)/2 + y^3/4 \\
\\
(y^2z^2)/8 - (x^2z^3)/24 + (y^2z^3)/8 - (y^3z^2)/8 - (y^3z)/8 + y^3/24 + (xy^2z)/24 + (x^2yz^2)/24 \\
(x^2z^3)/12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

3D Printing

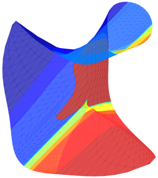
1. Run Bertini

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

| dimension | components | classified
-----------------------------------------
|   2        |   1         |   4
-----------------------------------------
=========
******* Decomposition by Degree *********

Dimension 2: 1 classified component
-----------------------------------------
   degree 4: 1 component

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

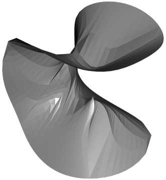
2. Run Bertini_real



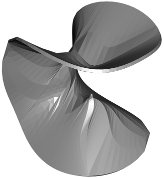
3. Refine



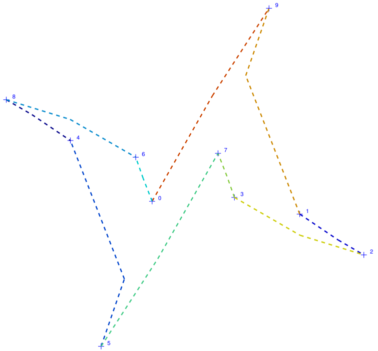
4. Process into `.stl`



5. Thicken surface



6. Print



Thank you for your kind attention.

Acknowledgements

- `Bertini_real` is joint work with Dan Bates, Jon Hauenstein, Wenrui Hao, Charles Wampler, and Andrew Sommese.
- We have been graciously funded by the NSF, and DARPA.