# Printing Algebraic Surfaces with Singularities

Daniel Brake

25 October, 2014

# Real Components

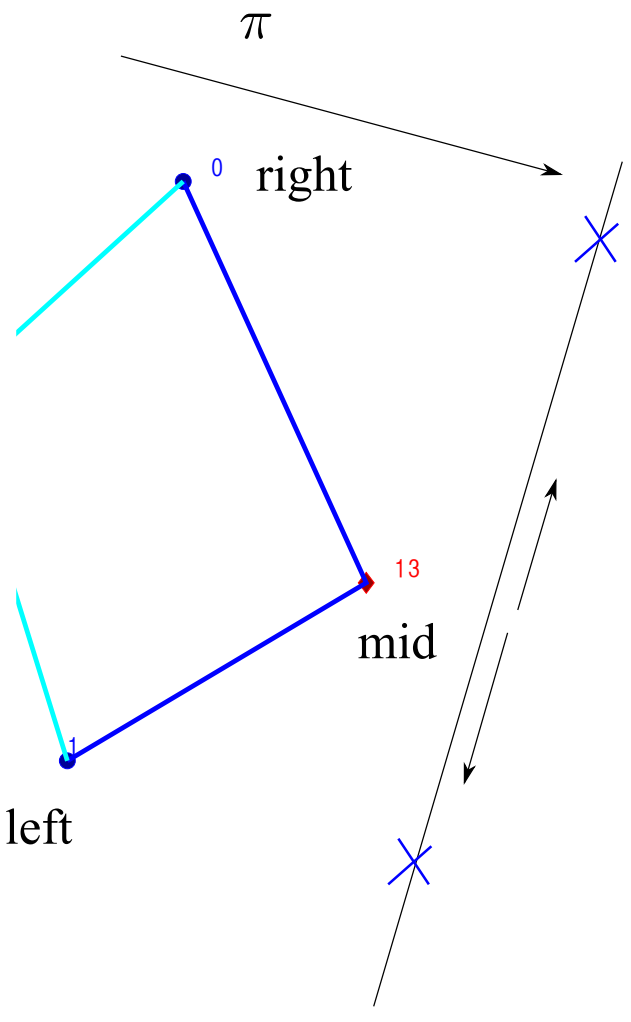
Numerical Cellular Decomposition for Real Components

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.

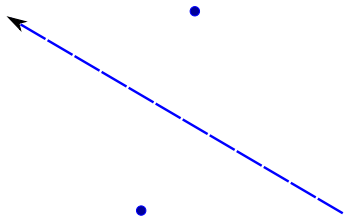
Our objective is to decompose the real part of $C$; i.e., $C \cap \mathbb{R}^N$

# Curve Cell

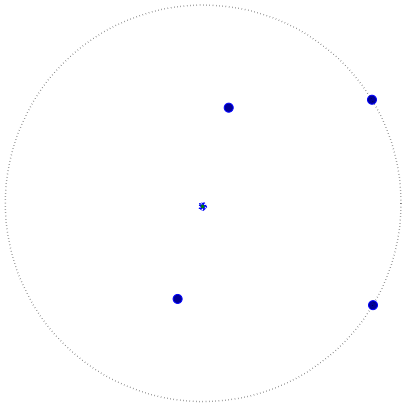


# Curves

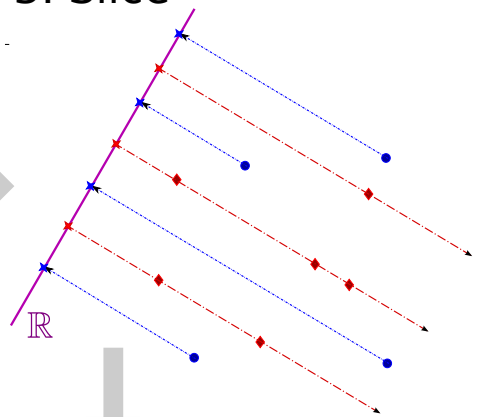
1. Find critical points



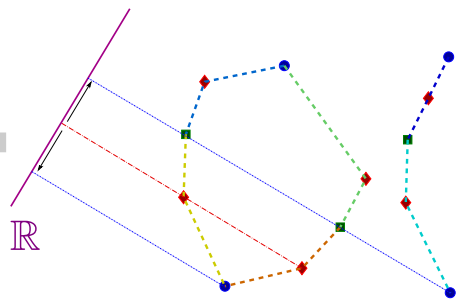
2. Intersect with sphere



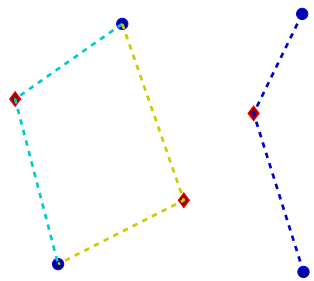
3. Slice



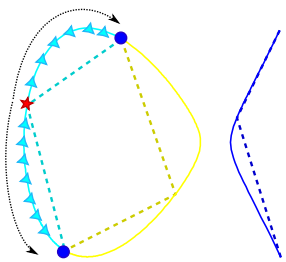

4. Connect the dots




5. Merge

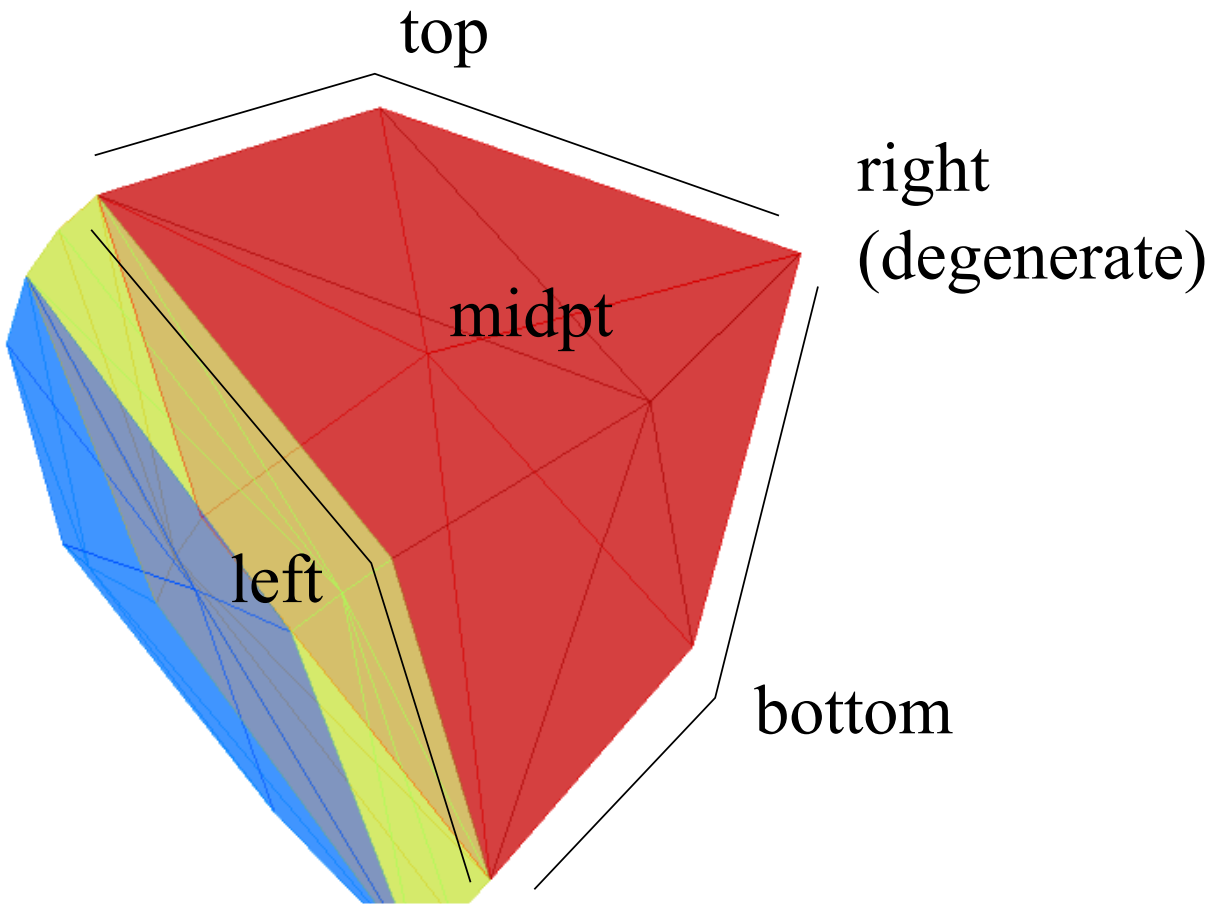


6. Refine



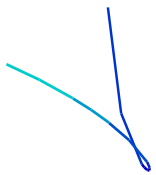
[Lu,Bates,Sommese,Wampler,2006]

# Surface Cell

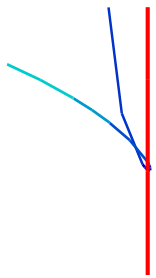


# Surface Decomposition

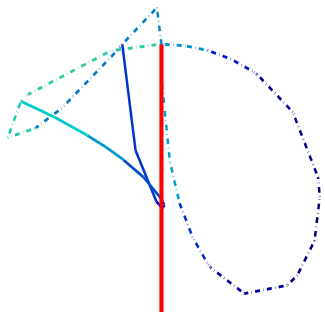
1. Decompose critical curve



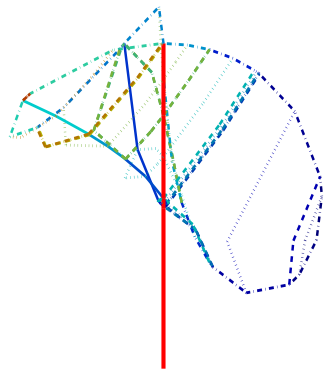
2. Decompose singular curves



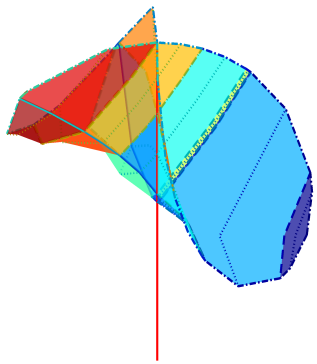
3. Intersect with sphere



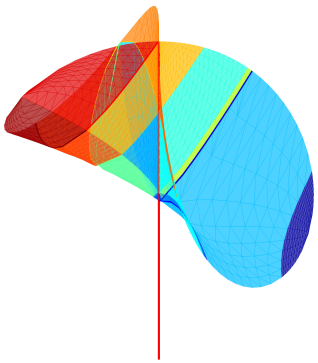
4. Slice



5. Connect the dots

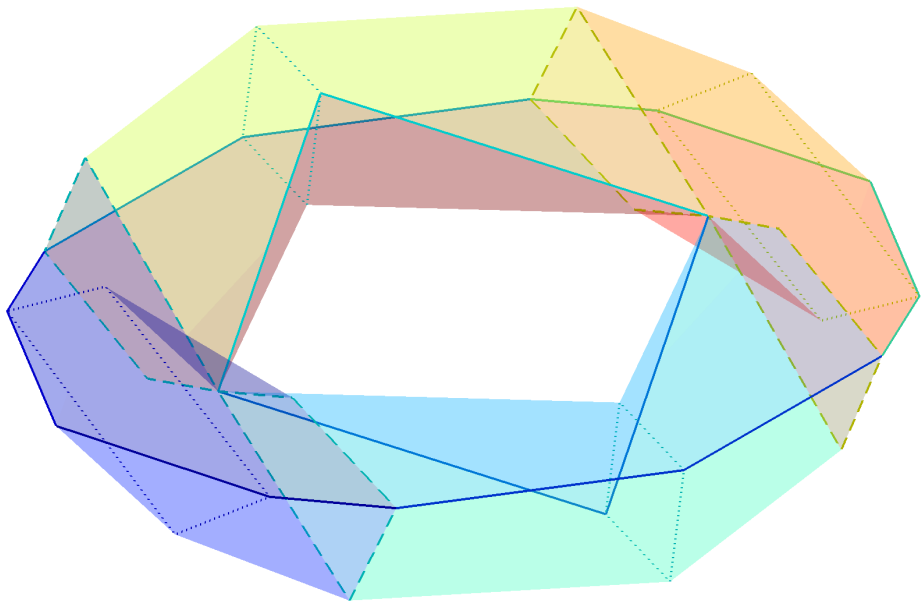


6. Refine



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

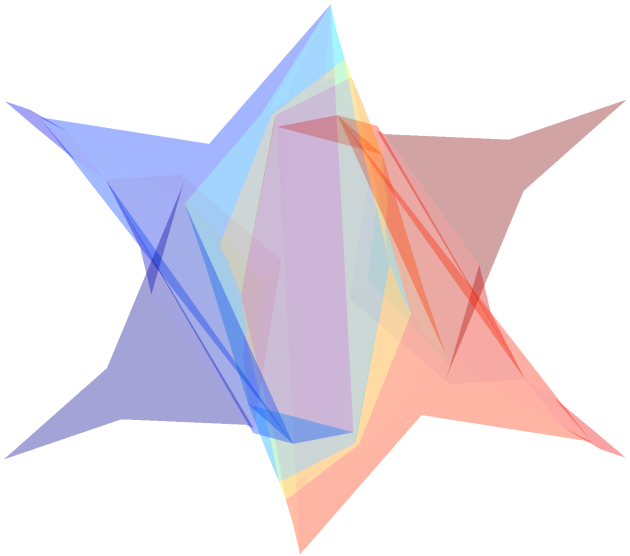
# Surface examples



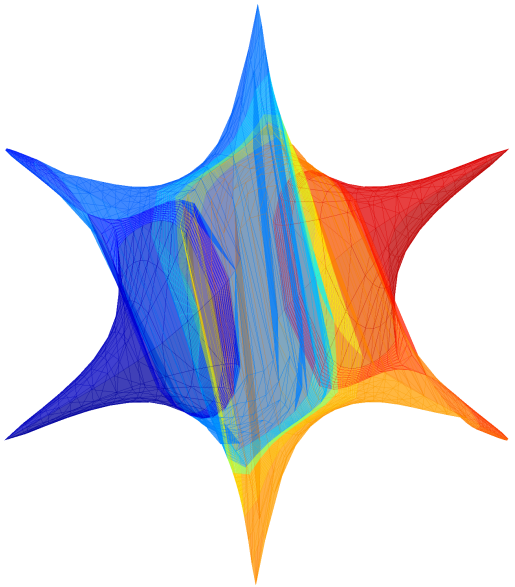
torus

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

# Surface examples



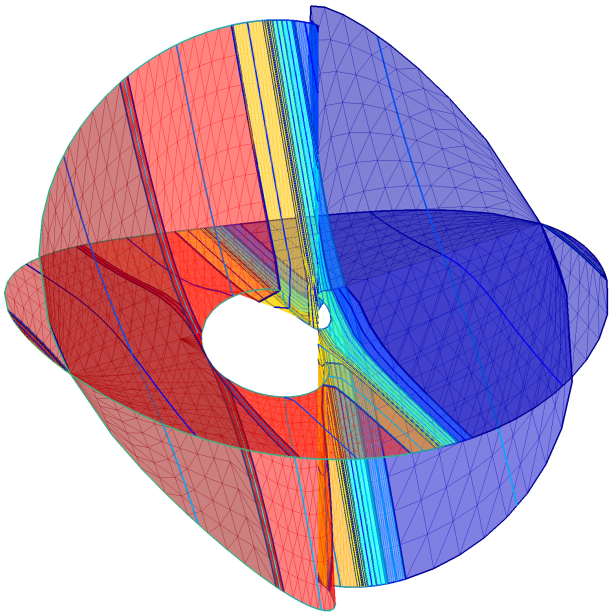
distel, unrefined



distel, refined

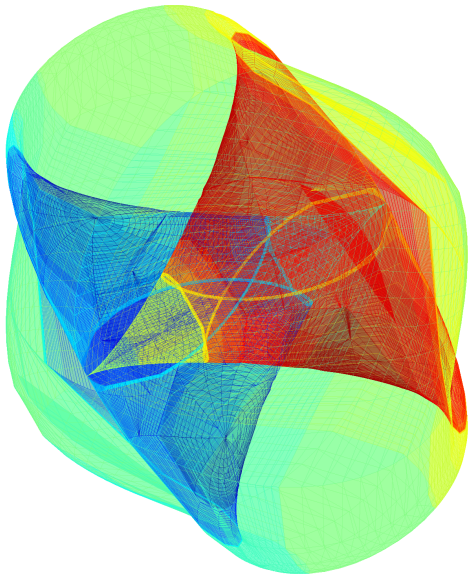
$$f(x,y,z) = x^2 + y^2 + z^2 + 1000(x^2 + y^2)(x^2 + z^2)(y^2 + z^2) - 1$$

# Surface examples



solitude

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



klein

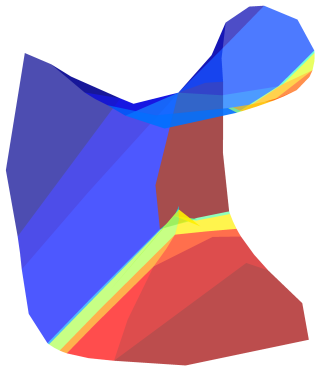
$$f(x,y,z,w) = \begin{bmatrix} w^2 + x^2 + y^2 + z^2 - 1 \\ 2wyz - x(y^2 + z^2) \end{bmatrix}$$

# 3D Printing

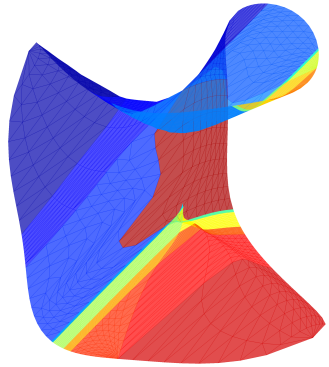
1. Run Bertini

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

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

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

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

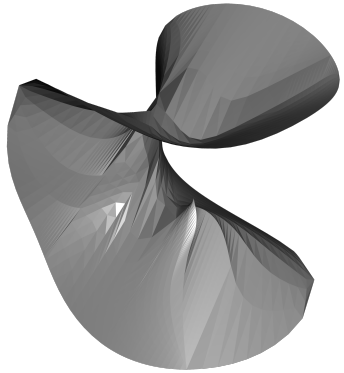
2. Run Bertini_real



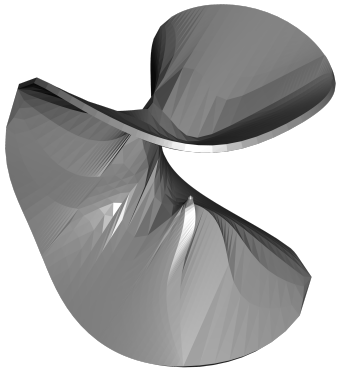
3. Refine



4. Process into `.stl`



5. Thicken surface



6. Print

# Challenges

Singularities are the main problem to solve for 3D printing of surfaces

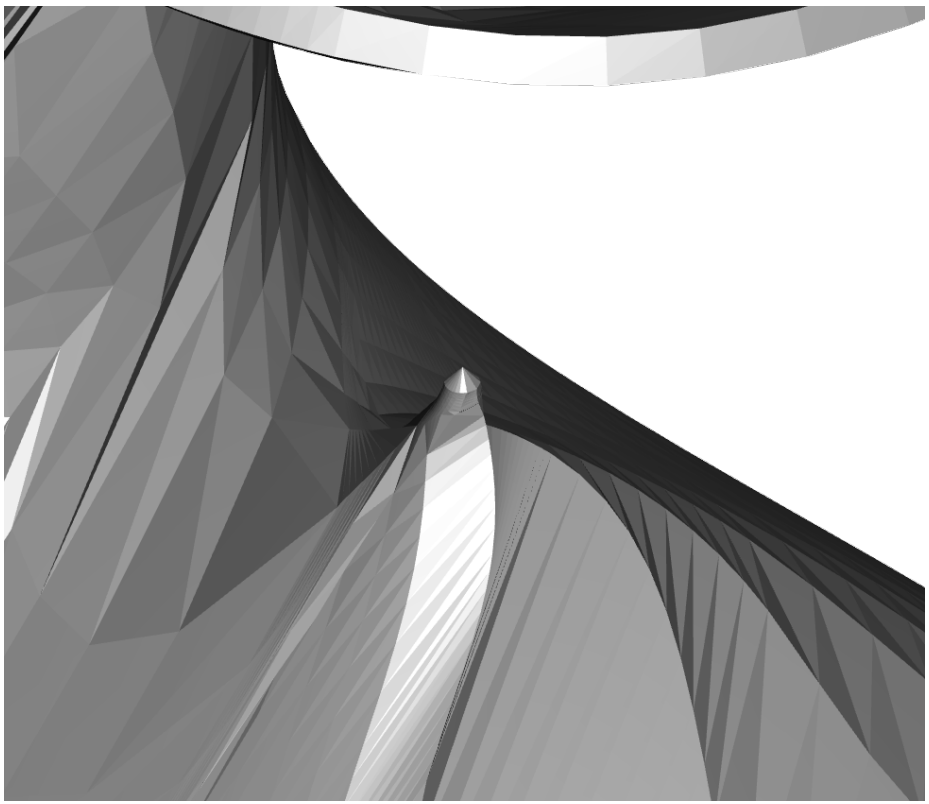
- point singularities – *e.g.* pinches
- curve singularities – *e.g.* cusps
- curve singularities – *e.g.* self-crossings

Non-compactness is a real challenge, too.

- surface has 0 volume

# Point Singularities

Pinch points are a real problem – thickening can cause self-crossings!

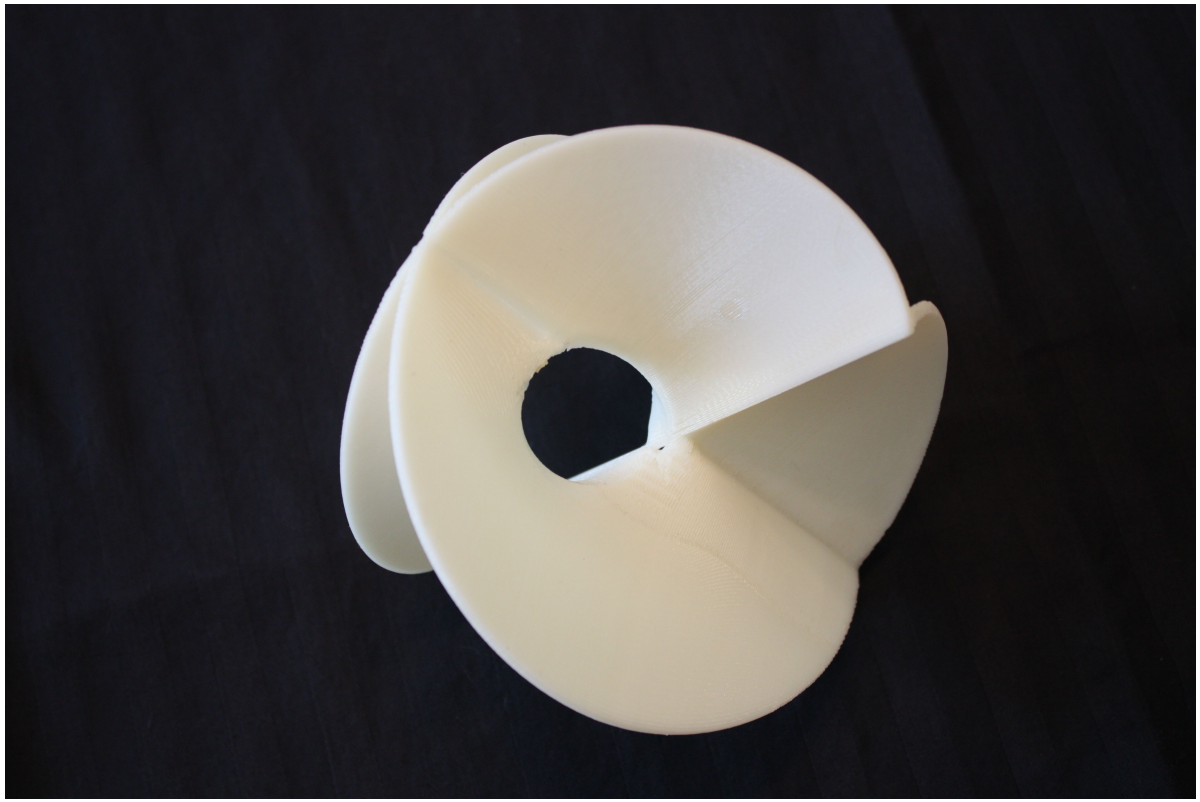


# Point Singularities

Pinch points are a real problem – you lose topological accuracy when printing!

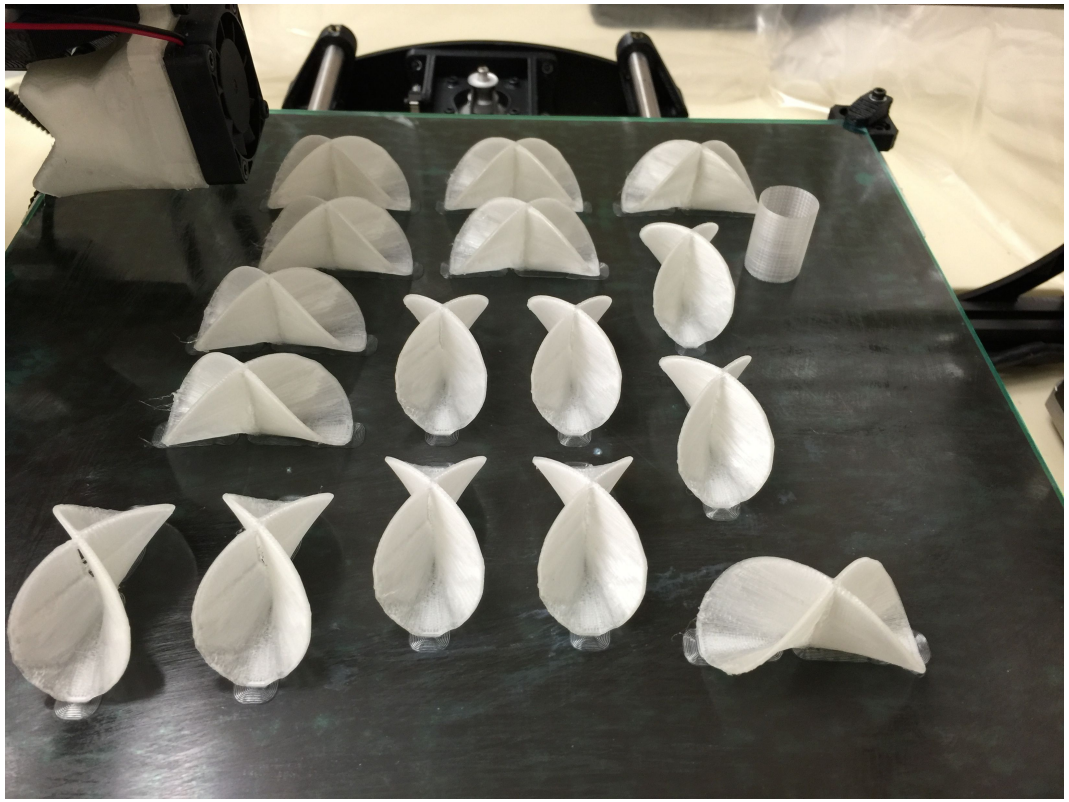
# Curve Singularities – cusps

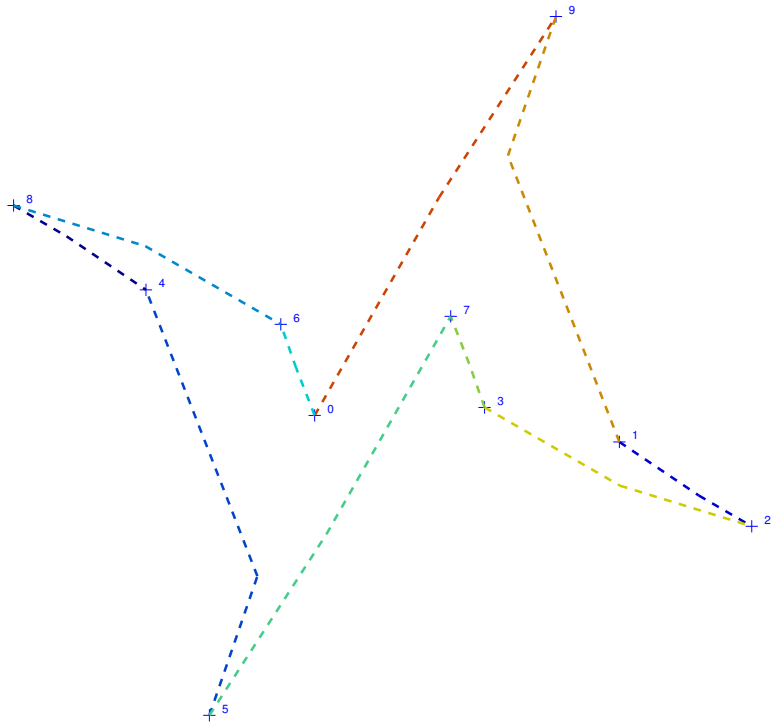
Cusps are a real problem – thickening can cause self-crossings!
Plus, sharp edges are hard to print!



# Curve Singularities – intersections

Self-intersections are a real problem – there may not be an orientation!






Thank you for your kind attention.

# Bertini_real Overview

Bertini_real is compiled command line software:

- performs almost purely numerical computations to produce a *cellular decomposition* of real algebraic components,
- uses Bertini as its homotopy continuation engine,
- uses Matlab for symbolic computations, such as deflation; as well as visualization,
- can decompose higher-dimensional curves and surfaces, including those with singularities,
- results are [almost] readily 3d printed.

# 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

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

- $\delta$ 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:

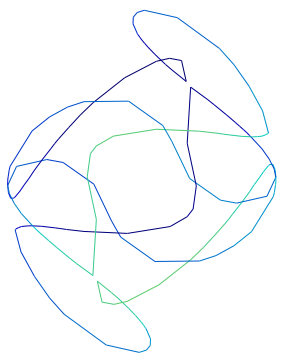
$$v = \begin{bmatrix} M_{1\neq j} \\ \vdots \\ M_{N\neq j} \\ \text{patch}_v \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

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

# 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:

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

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

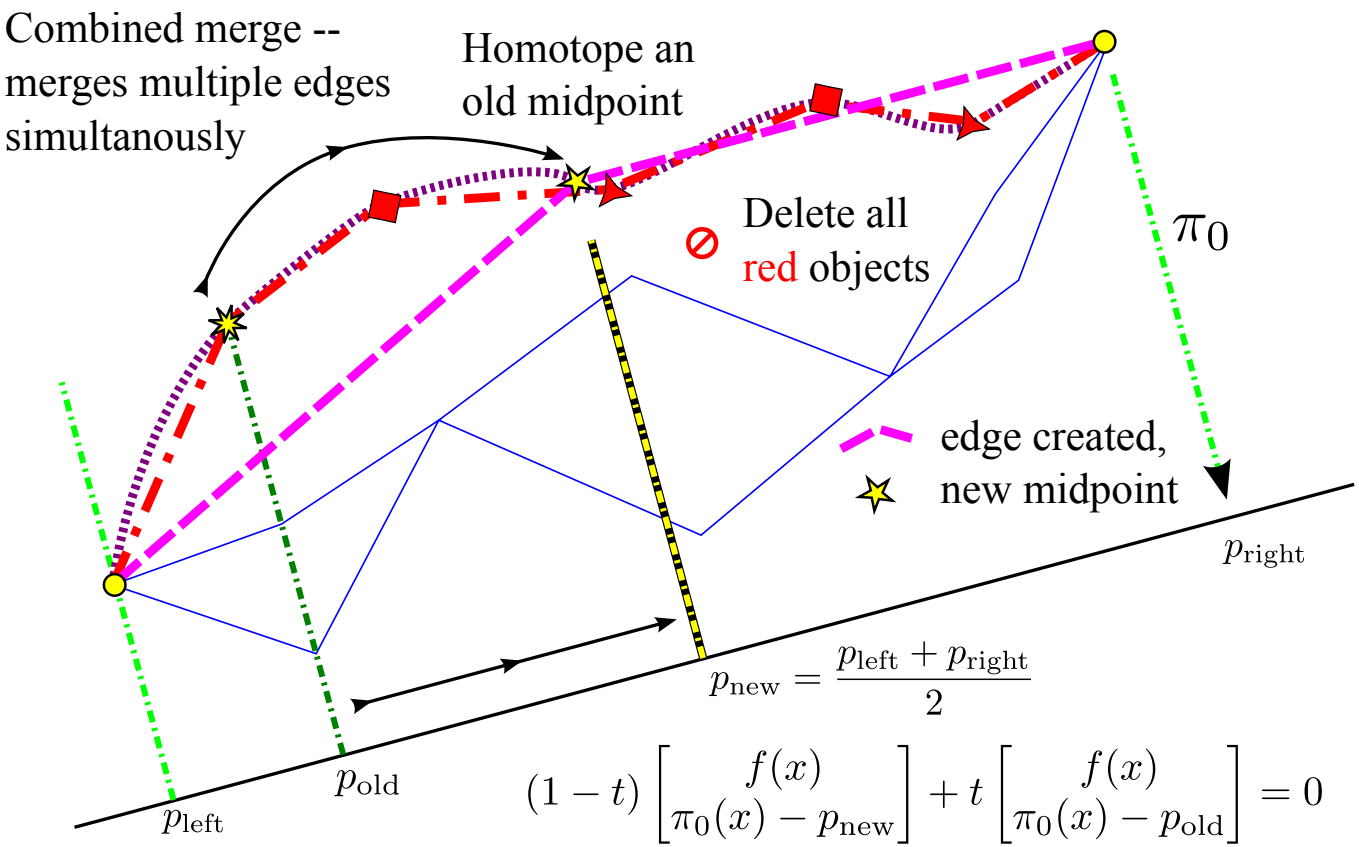
- Determinant operation produces high degree polynomials, contributing to numerical issues.
- Using this formulation for dimension 3 decompositions will be even worse w.r.t. degree and computational complexity.
- Fortunately, we can still use the nullspace method for finding critical points of this curve.

# Crit points of crit curve

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

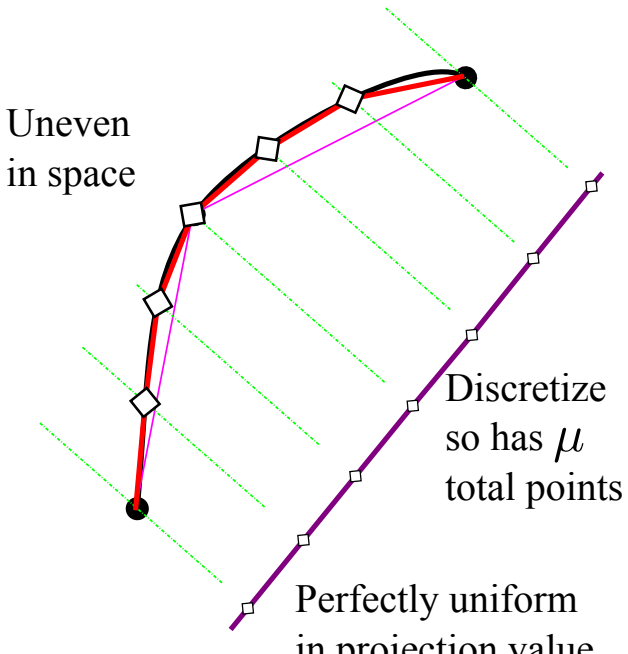
$$f_{\text{crit\_crit}} = \begin{bmatrix} f \\ \det \begin{pmatrix} Jf \\ J\pi_1 \\ J\pi_2 \end{pmatrix} \\ v^{\intercal} \cdot J \left( \det \begin{pmatrix} Jf \\ J\pi_1 \\ J\pi_2 \end{pmatrix} \right) \\ J\pi_1 \\ \text{patch}_v \end{bmatrix}$$

# Curve Merging

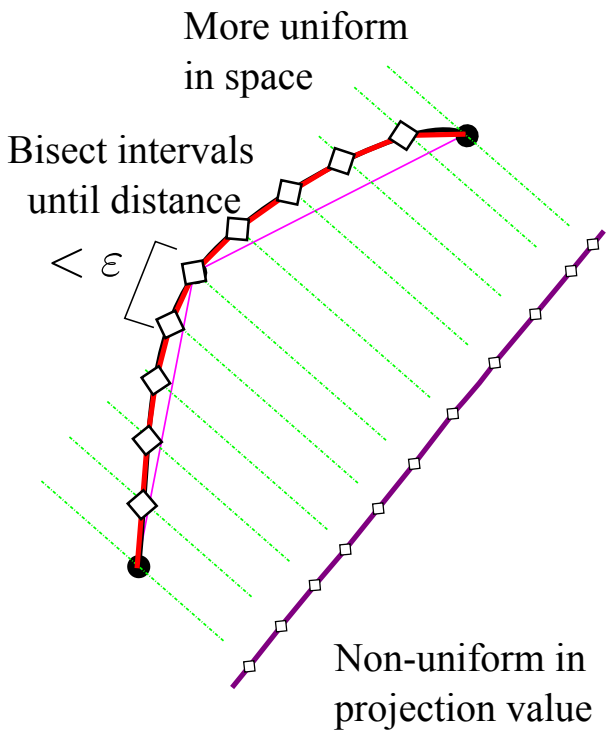


# Curve Sampling

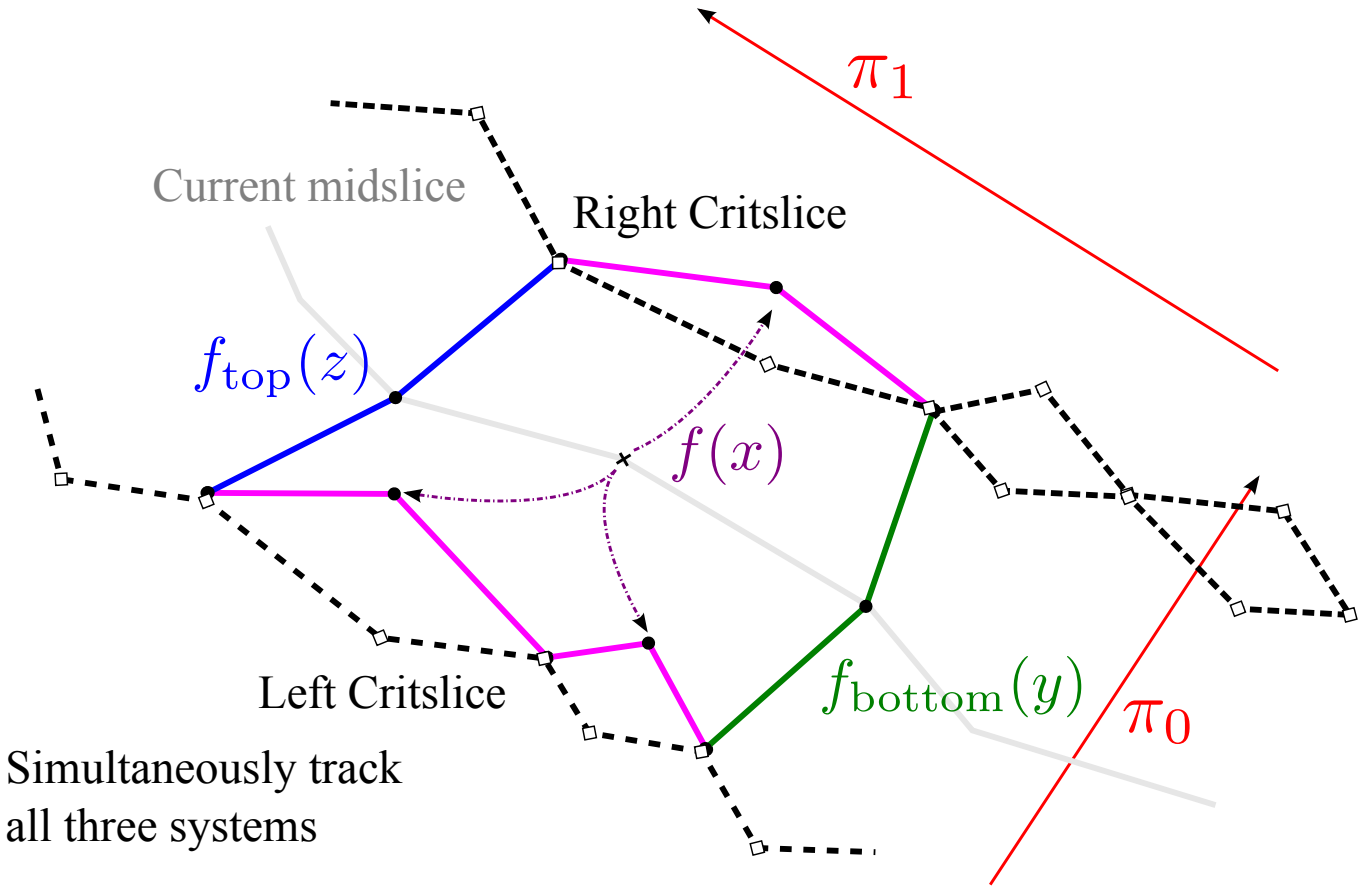
## Fixed-number sampling




## Adaptive sampling

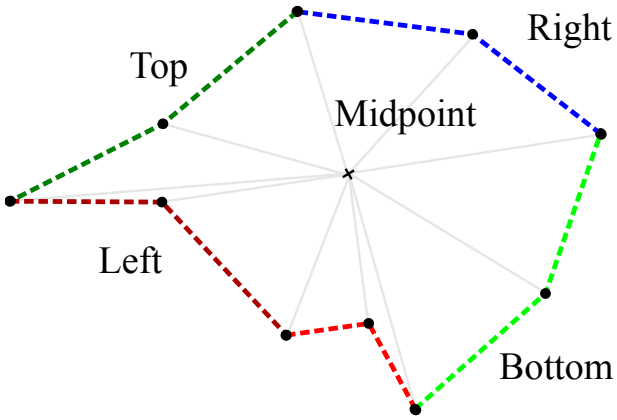


# Midtracking

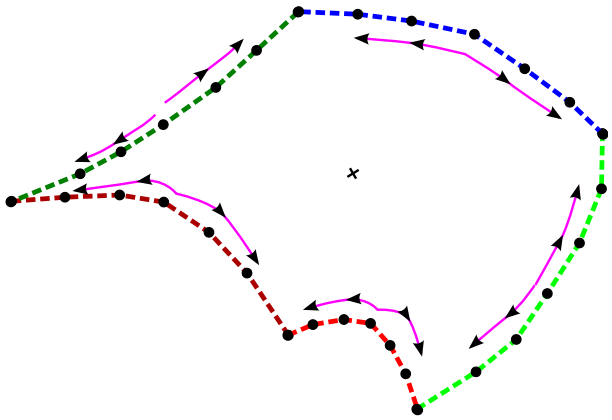


# Surface Refinement

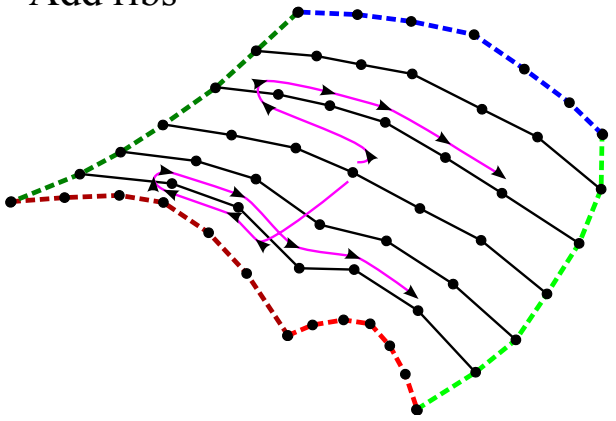
Raw face and edges



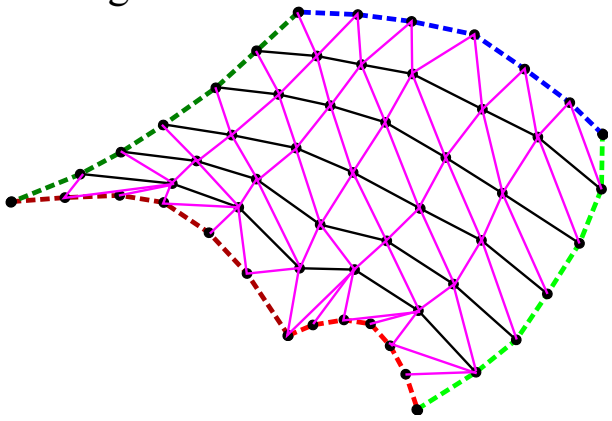

Refine the edges



Add ribs



Triangulate



# Generic Decomposition

**Input**: polynomial system $f$, witness set $W$, projections $\pi_1, ...\pi_d$.
**Output**: cell decomposition $D$.

1: Compute $W_K$, a witness set for the critical set of dimension $d$.
2: Compute witness sets for singular sets, sphere intersection.
3: Compute critical points of everything possible.
4: Decompose $S$, the singular set.
5: Decompose $K$, the critical set.
6: Intersect with sphere.
7: Slice using linears at all critical points.
8: Slice between all critical points.
9: Glue it all together.

Method not implemented or written up.