Topology analysis of complex curves singularities using knot theory

Mădălina Hodorog[1], Bernard Mourrain[2], Josef Schicho[1]

[1]Johann Radon Institute for Computational and Applied Mathematics,
Doctoral Program "Computational Mathematics"
Johannes Kepler University Linz, Austria

[2]INRIA Sophia-Antipolis, France

7^{th} Conference on Curves and Surfaces, France
June 24, 2010

Table of contents

1. Motivation

2. Topology of plane complex curves singularities
   - Describing the problem
   - Solving the problem

3. A library for topology of plane complex curves singularities

4. Conclusion and future work

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Motivation

Why study the topology of a complex curve singularity?
What is the topology of a singularity?
How to compute the topology?
Why using knot theory?

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Motivation

Why study the topology of a complex curve singularity?
What is the topology of a singularity?
How to compute the topology?
Why using knot theory?

☞ To compute the genus of plane complex curves!

Motivation

Why study the topology of a complex curve singularity?
What is the topology of a singularity?
How to compute the topology?
Why using knot theory?

☛ To compute the genus of plane complex curves!
☞ The algebraic link of the singularity!

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Motivation

Why study the topology of a complex curve singularity?
What is the topology of a singularity?
How to compute the topology?
Why using knot theory?

- ☛ To compute the genus of plane complex curves! We use
- ☛ The algebraic link of the singularity!
- ☞ We propose a symbolic-numeric algorithm for this purpose!

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Motivation

Why study the topology of a complex curve singularity?
What is the topology of a singularity?
How to compute the topology?
Why using knot theory?

- ☛ To compute the genus of plane complex curves!
- ☛ The algebraic link of the singularity!
- ☛ We propose a symbolic-numeric algorithm for this purpose!
- ☞ The proposed algorithm is stable w.r.t. small perturbations!

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

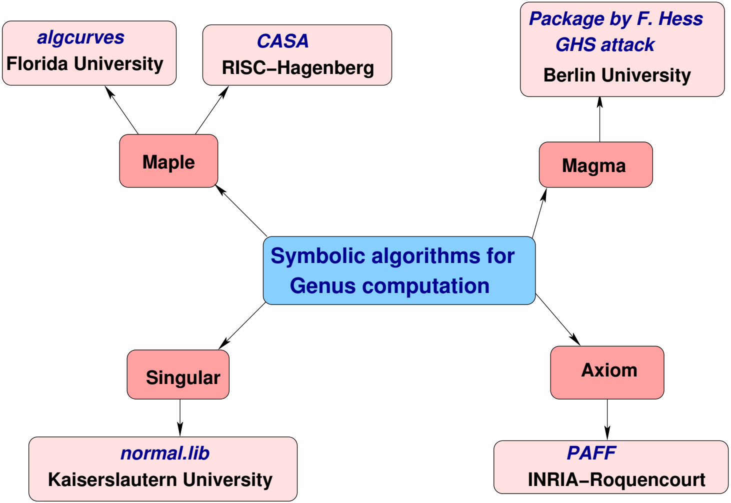
Motivation

Why is this proposed symbolic-numeric algorithm "special"?

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Motivation

At present, there exists several...



Motivation

But...

For moderate symbolic input data: they are too expensive

↑

Disadvantages of Symbolic algorithms for Genus computation

↓

For numeric input data: they are unusable

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Motivation

For instance, in Maple using algcurves package...

```
> with(algcurves);
[AbelMap, Siegel, Weierstrassform, algfun_series_sol, differentials, genus,
    homogeneous, homology, implicitize, integral_basis, is_hyperelliptic,
    j_invariant, monodromy, parametrization, periodmatrix, plot_knot,
    plot_real_curve, puiseux, singularities]
```

> $f := x^2\, y + y^4$

$$f := x^2\, y + y^4$$

> $genus(f, x, y)$

$$-1$$

> $g := 1.02 \cdot x^2\, y + 1.12 \cdot y^4$

$$g := 1.02\, x^2\, y + 1.12\, y^4$$

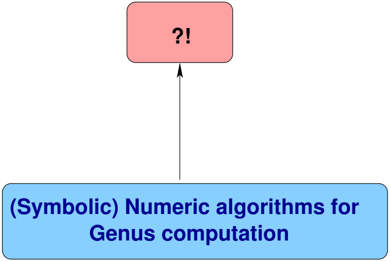
> $genus(g, x, y)$

```
Error, (in content/polynom) general case of floats not handled
```

>

Motivation

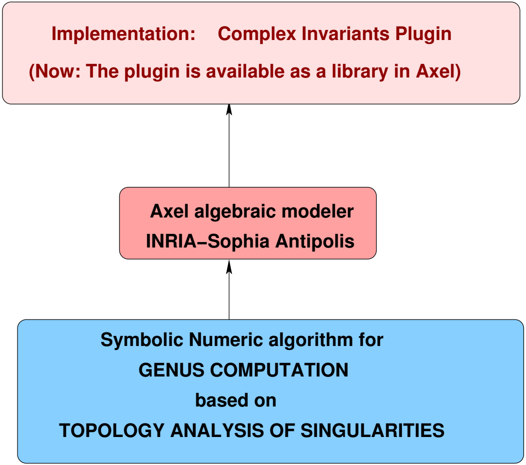
Thus we need...



Motivation

Hopefully...

Project: Symbolic-Numeric techniques for genus computation (initiated by J. Schicho).



Other numeric method was reported (in the group of R. Sendra).

Doctoral Program Computational Mathematics Numerical Analysis and Symbolic Computation

What?

- Input:
  - $F \in \mathbb{C}[x, y]$ squarefree with coefficients of limited accuracy:
  - $C = \{(x, y) \in \mathbb{C}^2 | F(x, y) = 0\}$ complex algebraic curve of degree m.
  - $\epsilon \in \mathbb{R}_+^*$ a non-zero positive real number, the input parameter.
- Output:
  - the algebraic link/topology of each singularity $s \in Sing(C)$, where $Sing(C)$ is the set of singularities of the curve C.

What?

- Input:
  - $F \in \mathbb{C}[x, y]$ squarefree with coefficients of limited accuracy:
    - integers or rational numbers: $1, -2, \dfrac{1}{2}$.
    - or real numbers. For 1.001 we associate a tolerance of $\sigma = 10^{-3}$.
  - $C = \{(x, y) \in \mathbb{C}^2 | F(x, y) = 0\}$ complex algebraic curve of degree m.
  - $\epsilon \in \mathbb{R}_+^*$ a non-zero positive real number, the input parameter.
- Output:
  - the algebraic link/topology of each singularity $s \in Sing(C)$, where $Sing(C)$ is the set of singularities of the curve C.

How?

- Strategy for computing the topology of all the singularities of the curve

$$\boxed{\text{Plane complex algebraic curve}}$$

$$\Big\downarrow \text{compute} \;\text{numerically}$$

$$\boxed{\text{Singularities}}$$

$$\Big\downarrow \text{compute} \;\text{numerically-symbolically}$$

$$\boxed{\text{Algebraic link and operations on it } (\epsilon \text{ needed})}$$

$$\Big\downarrow \text{compute} \;\text{symbolically}$$

$$\boxed{\text{Alexander polynomial, } \delta\text{-invariant, GENUS}}$$

How?

- Strategy for computing the topology of all the singularities of the curve

$$\begin{array}{c}
\boxed{\text{Plane complex algebraic curve}} \\
\text{compute} \Big\downarrow \text{numerically} \\
\boxed{\text{Singularities moved in origin}} \\
\text{compute} \Big\downarrow \text{numerically-symbolically} \\
\boxed{\text{Algebraic link and operations on it } (\epsilon \text{ needed})} \\
\text{compute} \Big\downarrow \text{symbolically} \\
\boxed{\text{Alexander polynomial, } \delta\text{-invariant, GENUS}}
\end{array}$$

Solving the problem
Implementation of the algorithm

- *Axel* algebraic geometric modeler [a]

[a] Acknowledgements: Julien Wintz

Solving the problem
Implementation of the algorithm

- *Axel* algebraic geometric modeler [a]
    - developed by *Galaad* team (INRIA Sophia-Antipolis);

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

[a]Acknowledgements: Julien Wintz

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

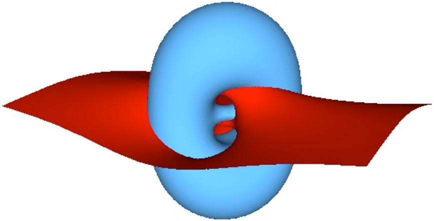
Solving the problem
Implementation of the algorithm

- *Axel* algebraic geometric modeler [a]
    - developed by *Galaad* team (INRIA Sophia-Antipolis);
    - written in *C++*, Qt Script for Applications (QSA);

[a] Acknowledgements: Julien Wintz

Solving the problem
Implementation of the algorithm

- *Axel* algebraic geometric modeler [a]
  - developed by *Galaad* team (INRIA Sophia-Antipolis);
  - written in $C{+}{+}$, Qt Script for Applications (QSA);
  - provides algebraic tools for:
    - implicit surfaces;



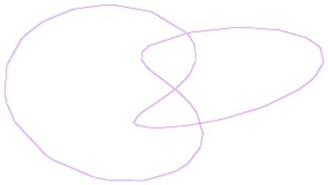
[a] Acknowledgements: Julien Wintz

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Solving the problem

Implementation of the algorithm

- *Axel* algebraic geometric modeler [a]
  - developed by *Galaad* team (INRIA Sophia-Antipolis);
  - written in *C++*, Qt Script for Applications (QSA);
  - provides algebraic tools for:
    - implicit surfaces;
    - implicit curves.



[a] Acknowledgements: Julien Wintz

Solving the problem

Implementation of the algorithm

- *Axel* algebraic geometric modeler [a]
  - developed by *Galaad* team (INRIA Sophia-Antipolis);
  - written in *C++*, Qt Script for Applications (QSA);
  - provides algebraic tools for:
    - implicit surfaces;
    - implicit curves.
  - free, available at:

http://axel.inria.fr/

[a] Acknowledgements: Julien Wintz

First

$$\begin{tikzcd}
\boxed{\text{Plane complex algebraic curve}} \arrow[d, "\text{compute}"', "\text{numerically}"] \\
\boxed{\text{Singularities}} \arrow[d, "\text{compute}"', "\text{numerically-symbolically}"] \\
\boxed{\text{Algebraic link and operations on it } (\epsilon \text{ needed})} \arrow[d, dashed, "\text{compute}"', "\text{symbolically}"] \\
\boxed{\text{Alexander polynomial, } \delta\text{-invariant, GENUS}}
\end{tikzcd}$$

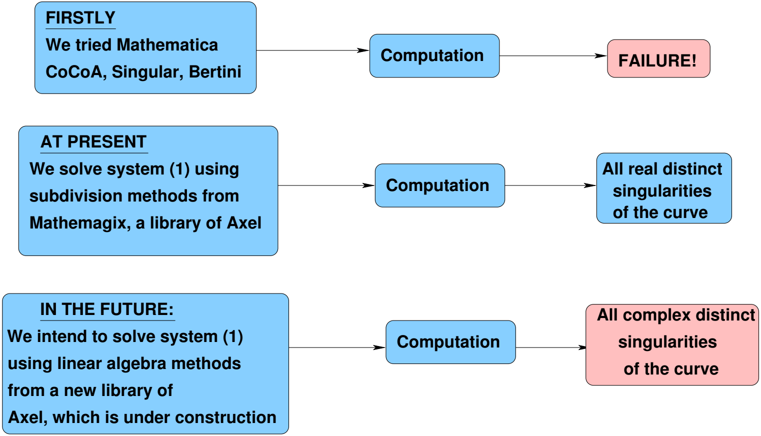
Computing the singularities of the curve

- Input:
  - $F(x,y) \in \mathbb{C}[x,y]$ squarefree with coefficients of limited accuracy.
  - $C = \{(x,y) \in \mathbb{C}^2 | F(x,y) = 0\}$ complex algebraic curve of degree m.
- Output:
  - $Sing(C) = \{(x_0,y_0) \in \mathbb{C}^2 | F(x_0,y_0) = 0, \frac{\partial F}{\partial x}(x_0,y_0) = 0, \frac{\partial F}{\partial y}(x_0,y_0) = 0\}$
- Method: We solve the overderminate system of polynomial equations with coefficients of limited accuracy in $\mathbb{C}^2$:

$$\begin{cases} F(x_0,y_0) = 0 \\ \frac{\partial F}{\partial x}(x_0,y_0) = 0 \\ \frac{\partial F}{\partial y}(x_0,y_0) = 0 \end{cases}, \tag{1}$$

Computing the singularities of the curve

For input polynomials with coefficients of limited accuracy:



Note: We assume the subdivision methods return all the singularities.

Next

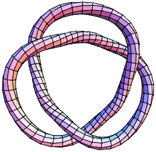
$$\begin{array}{c}
\boxed{\text{Plane complex algebraic curve}} \\
\text{compute} \Big\downarrow \text{numerically} \\
\boxed{\text{Singularities moved in origin}} \\
\text{compute} \Big\downarrow \text{numerically-symbolically} \\
\boxed{\text{Algebraic link and operations on it } (\epsilon \text{ needed})} \\
\text{compute} \Big\downarrow \text{symbolically} \\
\boxed{\text{Alexander polynomial, } \delta\text{-invariant, GENUS}}
\end{array}$$

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

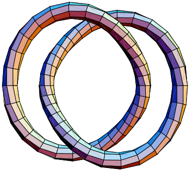
Knot theory - preliminaries

- A **knot** is a piecewise linear or a differentiable simple closed curve in $\mathbb{R}^3$.
- A **link** is a finite union of disjoint knots.
- Links resulted from the intersection of a given curve with the sphere are called **algebraic links**.

Trefoil Knot

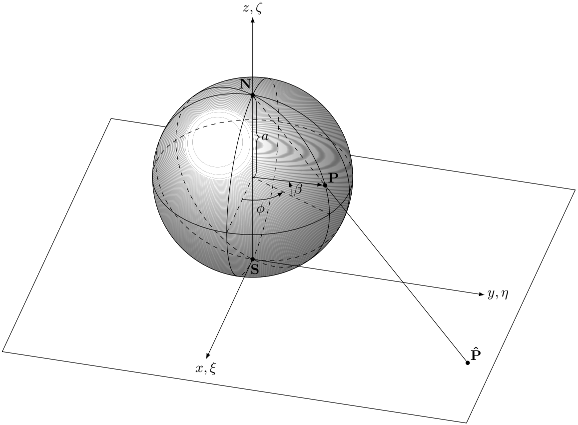


Hopf Link



Computing the algebraic link of the singularity

- Why the algebraic link of a singularity?
  - helps to study the topology of a complex curve near a singularity;
- How do we compute the algebraic link?
  - use the generalization of the stereographic projection;



Computing the link of the singularity

Method (based on Milnor's results)

1. Let $C = \{(a,b,c,d) \in \mathbb{R}^4 | F(a,b,c,d) = 0\}$ s.t. $(0,0,0,0) \in Sing(C)$
2. Consider $S_{(0,\epsilon)} := S = \{(a,b,c,d) \in \mathbb{R}^4 | a^2 + b^2 + c^2 + d^2 = \epsilon^2\}$,
   $X = C \bigcap S_{(0,\epsilon)} \subset \mathbb{R}^4$
3. For $P \in S \setminus C$, $f : S \setminus \{P\} \to \mathbb{R}^3, (a,b,c,d) \mapsto (u = \frac{a}{\epsilon - d}, v = \frac{b}{\epsilon - d}, w = \frac{c}{\epsilon - d})$,
   $f^{-1} : \mathbb{R}^3 \to S \setminus \{P\}$
   $(u,v,w) \mapsto (a = \frac{2u\epsilon}{n}, b = \frac{2v\epsilon}{n}, c = \frac{2w\epsilon}{n}, d = \frac{\epsilon(u^2+v^2+w^2-1)}{n})$, where $n = 1 + u^2 + v^2 + w^2$.
4. Compute $f(X) = \{(u,v,w) \in \mathbb{R}^3 | F(\frac{2u\epsilon}{n}, \frac{2v\epsilon}{n}, ...) = 0\} \Leftrightarrow$
   $f(X) = \{(u,v,w) \in \mathbb{R}^3 | ReF(...) = 0, ImF(...) = 0\}$
   $f(X)$ is an implicitly defined algebraic curve!
   For small ϵ, $f(X)$ is a link (a differentiable algebraic link).

Computing the link of the singularity

We use Axel for implementation. Why Axel?

It is the only system to contain the implementation of a method for certified topology of smooth implicit curves in $\mathbb{R}^3$!

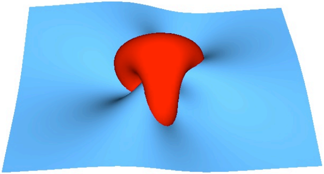
- For $C = \{(x,y) \in \mathbb{C}^2 | x^3 - y^2 = 0\} \subset \mathbb{R}^4$ we compute with the previous method in Axel:

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Computing the link of the singularity

We use Axel for implementation. Why Axel?
It is the only system to contain the implementation of a method for certified topology of smooth implicit curves in $\mathbb{R}^3$!

- For $C = \{(x,y) \in \mathbb{C}^2 | x^3 - y^2 = 0\} \subset \mathbb{R}^4$
  we compute with the previous method in Axel:
- $f(C \cap S) = f(X) := L =$
  $= \{(u,v,w) \in \mathbb{R}^3 | ReF(...) = 0, ImF(...) = 0\}$

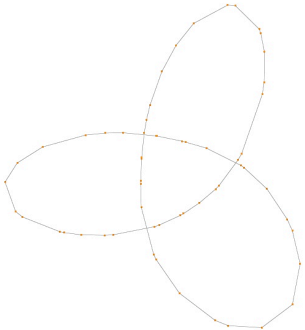


Computing the link of the singularity

We use Axel for implementation. Why Axel?

It is the only system to contain the implementation of a method for certified topology of smooth implicit curves in $\mathbb{R}^3$!

- For $C = \{(x,y) \in \mathbb{C}^2 | x^3 - y^2 = 0\} \subset \mathbb{R}^4$ we compute with the previous method in Axel:
- $f(C \cap S) = f(X) := L =$
  $= \{(u,v,w) \in \mathbb{R}^3 | ReF(...) = 0, ImF(...) = 0\}$
- $Graph(L) = \langle \mathcal{V}, \mathcal{E} \rangle$ with
  $\mathcal{V} = \{p = (m,n,q) \in \mathbb{R}^3\}$
  $\mathcal{E} = \{(i,j) | i,j \in \mathcal{V}\}$

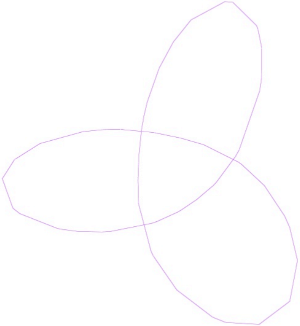


Computing the link of the singularity

We use Axel for implementation. Why Axel?

It is the only system to contain the implementation of a method for certified topology of smooth implicit curves in $\mathbb{R}^3$!

- For $C = \{(x, y) \in \mathbb{C}^2 | x^3 - y^2 = 0\} \subset \mathbb{R}^4$ we compute with the previous method in Axel:
- $f(C \cap S) = f(X) := L =$
  $= \{(u, v, w) \in \mathbb{R}^3 | ReF(...) = 0, ImF(...) = 0\}$
- $Graph(L) = \langle \mathcal{V}, \mathcal{E} \rangle$ with
  $\mathcal{V} = \{p = (m, n, q) \in \mathbb{R}^3\}$
  $\mathcal{E} = \{(i, j) | i, j \in \mathcal{V}\}$
- s.t. $Graph(L) \cong_{isotopic} L$

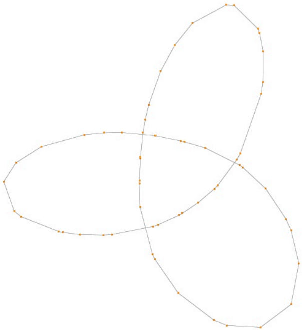


Computing the link of the singularity

We use Axel for implementation. Why Axel?

It is the only system to contain the implementation of a method for certified topology of smooth implicit curves in $\mathbb{R}^3$!

- For $C = \{(x, y) \in \mathbb{C}^2 | x^3 - y^2 = 0\} \subset \mathbb{R}^4$ we compute with the previous method in Axel:
- $f(C \cap S) = f(X) := L =$
  $= \{(u, v, w) \in \mathbb{R}^3 | ReF(...) = 0, ImF(...) = 0\}$
- $Graph(L) = \langle \mathcal{V}, \mathcal{E} \rangle$ with
  $\mathcal{V} = \{p = (m, n, q) \in \mathbb{R}^3\}$
  $\mathcal{E} = \{(i, j) | i, j \in \mathcal{V}\}$
- s.t. $Graph(L) \cong_{isotopic} L$
- $Graph(L)$ is the topology of L,
  a piecewise linear approximation
  for the differentiable algebraic link L;



Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Computing the link of the singularity

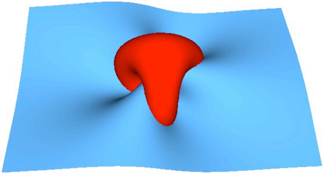
We use Axel for the implementation. Why Axel?
It is the only system to contain the implementation of a method for certified topology of smooth implicit curves in $\mathbb{R}^3$!

- For $C = \{(x, y) \in \mathbb{C}^2 | x^3 - y^2 = 0\} \subset \mathbb{R}^4$

Computing the link of the singularity

We use Axel for the implementation. Why Axel?
It is the only system to contain the implementation of a method for certified topology of smooth implicit curves in $\mathbb{R}^3$!

- For $C = \{(x, y) \in \mathbb{C}^2 | x^3 - y^2 = 0\} \subset \mathbb{R}^4$
- and $L =$
  $= \{(u, v, w) \in \mathbb{R}^3 | ReF(...) = 0, ImF(...) = 0\}$



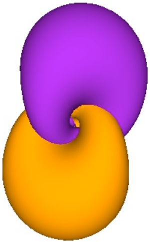
Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Computing the link of the singularity

We use Axel for the implementation. Why Axel?

It is the only system to contain the implementation of a method for certified topology of smooth implicit curves in $\mathbb{R}^3$!

- For $C = \{(x,y) \in \mathbb{C}^2 | x^3 - y^2 = 0\} \subset \mathbb{R}^4$
- and $L =$
  $= \{(u,v,w) \in \mathbb{R}^3 | ReF(...) = 0, ImF(...) = 0\}$
- we also compute (for visualization reasons)
  $S^{'} = \{(u,v,w) \in \mathbb{R}^3 | ReF(...) + ImF(...) = 0\}$
  $S^{''} = \{(u,v,w) \in \mathbb{R}^3 | Re(F) - ImF(...) = 0\}$

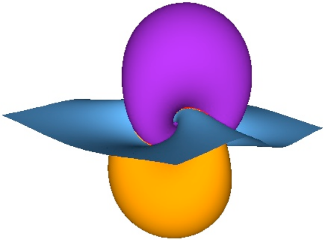


Doctoral Program Computational Mathematics
Numerical Analysis and Symbolic Computation

Computing the link of the singularity

We use Axel for the implementation. Why Axel?
It is the only system to contain the implementation of a method for certified topology of smooth implicit curves in $\mathbb{R}^3$!

- For $C = \{(x, y) \in \mathbb{C}^2 | x^3 - y^2 = 0\} \subset \mathbb{R}^4$
- and $L =$
  $= \{(u, v, w) \in \mathbb{R}^3 | ReF(...) = 0, ImF(...) = 0\}$
- we also compute (for visualization reasons)
  $S^{'} = \{(u, v, w) \in \mathbb{R}^3 | ReF(...) + ImF(...) = 0\}$
  $S^{''} = \{(u, v, w) \in \mathbb{R}^3 | Re(F) - ImF(...) = 0\}$
- L is the intersection of any 2 of the surfaces:
  $ReF(...), ImF(...)$
  $ReF(...) + ImF(...), ReF(...) - ImF(...)$



Next

Plane complex algebraic curve

↓ compute numerically

Singularities moved in origin

↓ compute numerically-symbolically

Algebraic link and operations on it (ϵ needed)

↓ compute symbolically

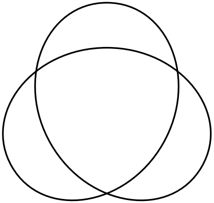
Alexander polynomial, δ-invariant, GENUS

Knot theory - preliminaries

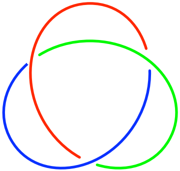
A knot projection is a **regular projection** if no three points on the knot project to the same point, and no vertex projects to the same point as any other point on the knot.
A double point of a regular projection is a **crossing point**.

A **diagram** is the image under regular projection, together with the information on each crossing telling which branch goes over/under.
An **arc** is the part of a diagram between two undercrossings.

Regular projection



Diagram

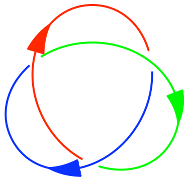


Knot theory - preliminaries

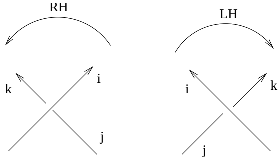
A diagram together with a given orientation of the link is called an **oriented diagram**.

A crossing is:
-**righthanded** if the underpass traffic goes from right to left.
-**lefthanded** if the underpass traffic goes from left to right.

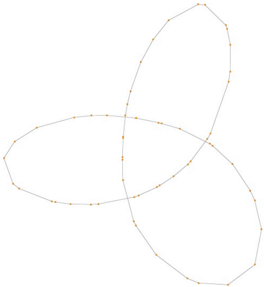
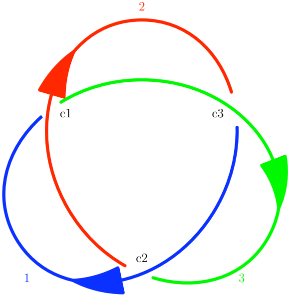
Oriented diagram



Crossings



Computing operations on the algebraic link

-  $\overset{?}{\Rightarrow}$ 

- $G(L) = \langle P, E \rangle$ $\qquad\qquad$ $D(L)$
- p(index,x,y,z)

  e(indexS, indexD)

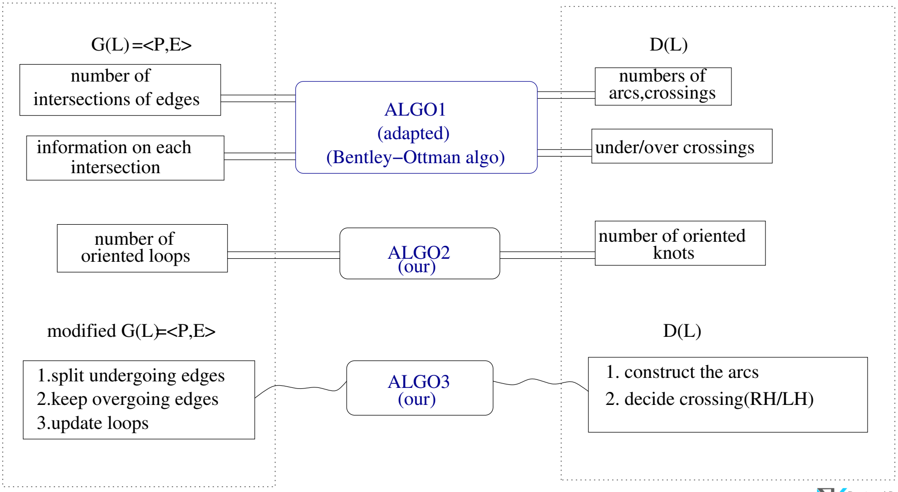
$\longrightarrow$ number of arcs, crossings

$\longrightarrow$ type of crossings (under, over)

$\longrightarrow$ number of knots in the link(orientation)

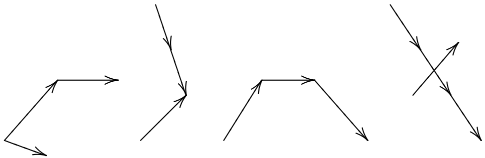
Computing operations on the algebraic link

By performing operations on $G(L)$ we obtain the elements of $D(L)$!



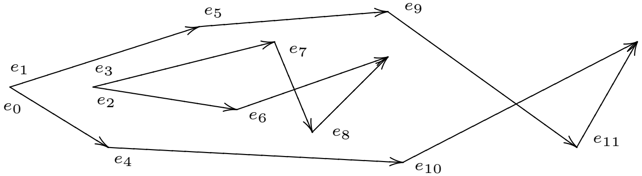
Algorithm 1 - Adapted version of Bentley-Ottman

- Input: S a set of "short" edges ordered from left to right:
  - A "short" edge is an edge whose projection contains at most one crossing point.

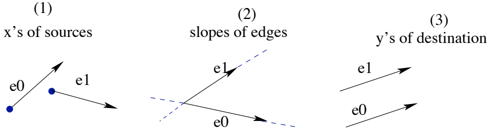


- Output: I - the set of all intersections among edges of S and
  - for each $p = e_i \cap e_j \in I$, the "arranged" pair of edges (e_i, e_j), i.e e_i is below e_j in $\mathbb{R}^3$

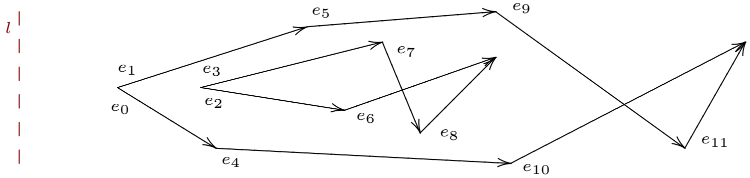
Algorithm 1 - Adapted version of Bentley-Ottman



- First: the edges are ordered by criteria (1),(2),(3):

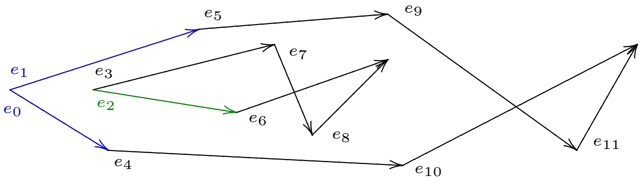


Algorithm 1 - Adapted version of Bentley-Ottman



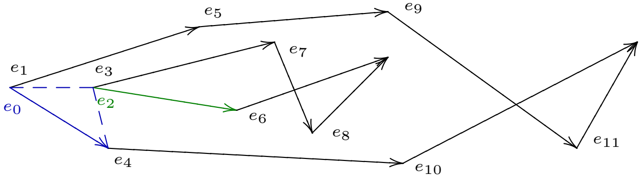
- we consider l a sweep line
- we keep track of two lists:
  $E = \{e_0, e_1, ..., e_{11}\}$ the list of ordered edges
  $Sw = \{?\}$ the list of event points
- while traversing E we insert the edges in Sw in the "right" position
- That is...

Algorithm 1 - Adapted version of Bentley-Ottman



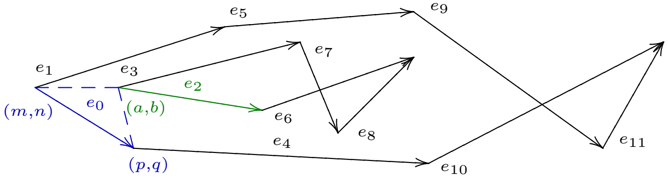
- $E = \{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$
- $Sw = \{e_0, e_1\}$

Algorithm 1 - Adapted version of Bentley-Ottman



- $E = \{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$
- $Sw = \{e_0, e_1\}$

Algorithm 1 - Adapted version of Bentley-Ottman

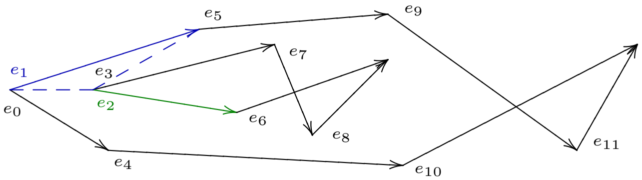


- $E = \{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$
- $Sw = \{e_0, e_1\}$; compute:

$$det(e_2, e_0) = \begin{pmatrix} m & n & 1 \\ p & q & 1 \\ a & b & 1 \end{pmatrix} > 0 \Rightarrow e_2 \text{ after } e_0 \text{ in } Sw$$

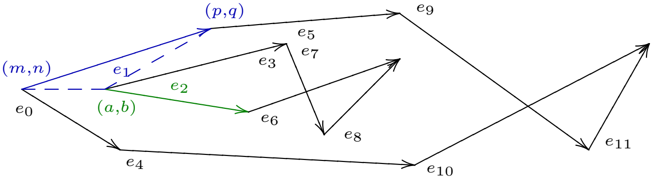
Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Algorithm 1 - Adapted version of Bentley-Ottman



- $E = \{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$
- $Sw = \{e_0, e_1\}$

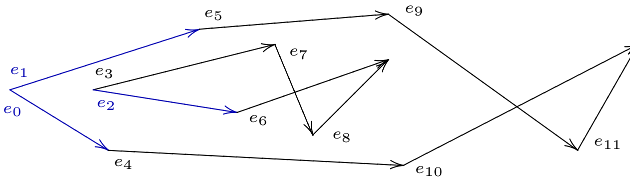
Algorithm 1 - Adapted version of Bentley-Ottman



- $E = \{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$
- $Sw = \{e_0, e_1\}$; compute:

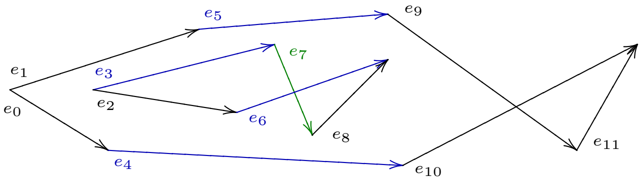
$$det(e_2, e_1) = \begin{pmatrix} m & n & 1 \\ p & q & 1 \\ a & b & 1 \end{pmatrix} < 0 \Rightarrow e_2 \text{ before } e_1 \text{ in } Sw$$

Algorithm 1 - Adapted version of Bentley-Ottman



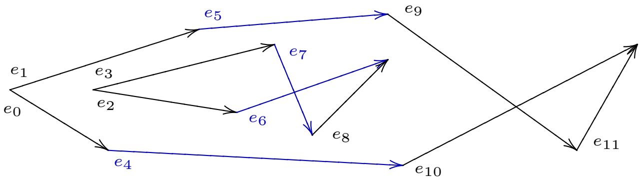
- $E = \{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$
- $Sw = \{e_0, e_2, e_1\}$
- Test $e_0 \cap e_2$? No!
  Test $e_2 \cap e_1$? No!
- $I = \emptyset$
  $E_I = \emptyset$

Algorithm 1 - Adapted version of Bentley-Ottman



- $E = \{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$
- $Sw = \{e_4, e_6, e_3, e_5\}$

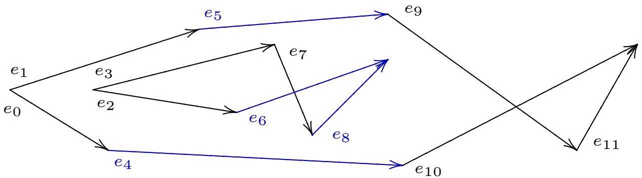
Algorithm 1 - Adapted version of Bentley-Ottman



- $E = \{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$
- $Sw = \{e_4, e_6, e_7, e_5\}$
- Test $e_6 \cap e_7 =?$ Yes!
  Test $e_7 \cap e_5 =?$ No! $\Rightarrow I = \{(a_1, b_1)\}\ E_I = \{(e_6, e_7)\}$
  $Sw = \{e_4, e_7, e_6, e_5\}$

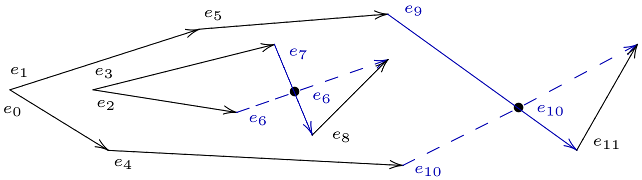
Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Algorithm 1 - Adapted version of Bentley-Ottman



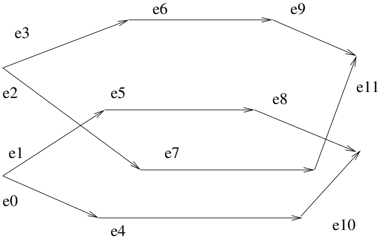
- $E = \{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$
- $Sw = \{e_4, e_8, e_6, e_5\}$
- Test $e_4 \cap e_8 =$?No!
  Test $e_8 \cap e_6 =$?No!
- Test $dest(e_4) = dest(e_8)$? No!
  Test $dest(e_8) = dest(e_6)$? Yes! $\Rightarrow Sw = \{e_4, \cancel{e_8}, \cancel{e_6}, e_5\} = \{e_4, e_5\}$

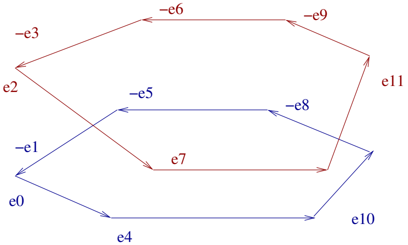
Algorithm 1 - Adapted version of Bentley-Ottman



- The adapted Bentley-Ottman algorithm produces the final output:
  $I = \{i_1 = (x_1, y_1), i_2 = (x_2, y_2)\}$
  $E_I = \{(e_6, e_7), (e_{10}, e_9)\}$ with
  - e_6 below e_7 in $\mathbb{R}^3$ and
  - e_{10} below e_9 in $\mathbb{R}^3$

Algorithm 2 - Constructing the loops

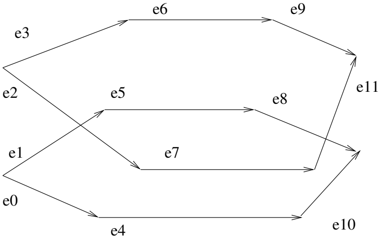


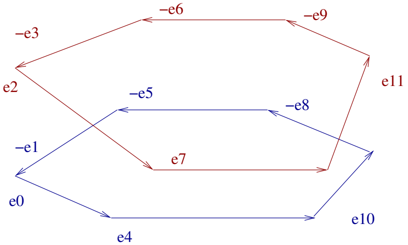


- E ordered by (1),(2),(3) $\Rightarrow$

$$L_0 = \{e_0, e_4, e_{10}, -e_8, -e_5, -e_1\}$$
$$L_1 = \{e_2, e_7, e_{11}, -e_9, -e_6, -e_3\}$$

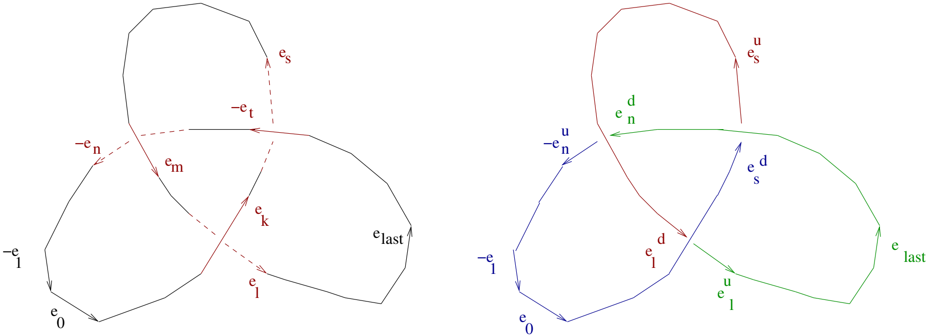
Algorithm 2 - Constructing the loops





- E ordered by (1),(2),(3) $\Rightarrow$ $L_0 = \{e_0, e_4, e_{10}, -e_8, -e_5, -e_1\}$
  $L_1 = \{e_2, e_7, e_{11}, -e_9, -e_6, -e_3\}$
- Here, we introduce:
  ✏ the positive edges ($\xrightarrow{e}$): $x.dest(e) > x.source(e)$
  ✏ the negative edges ($\xleftarrow{-f}$): $x.dest(-f) < x.source(-f)$

Algorithm 3 - Constructing the arcs



- $E = \{e_0, ..., e_{last}\}$
- $E_I = \{(-e_n, e_m), (e_l, e_k), (e_s, -e_t)\}$ $\Rightarrow$
- $L_0 = \{e_0, ..., e_s, e_l, ..., -e_1\}$

$a_0 = \{e_n^u, .., -e_1, e_0, .., e_k, .., e_s^d\}$

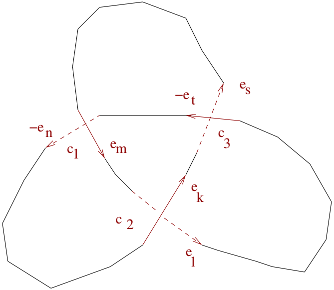
$a_1 = \{e_l^u, .., -e_t, .., -e_n^d\}$

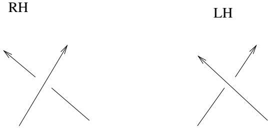
$a_2 = \{e_s^u, ..., e_m, ..., e_l^d\}$

- While constructing the arcs we also decide the type of crossings (RH or LH).

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Algorithm 3 - Deciding the type of crossing





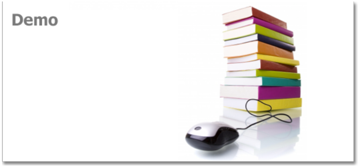
- For instance $c_1 = (-e_n, e_m)$ is LH since:
  - x.source$(-e_n)>x$.dest$(-e_n)$,
  - x.source$(e_m)<x$.dest(e_m),
  - slope$(e_m)<$slope$(-e_n)$
- $c_2 = (e_l, e_k)$ is LH, $c_3 = (e_s, -e_t)$ is LH.

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

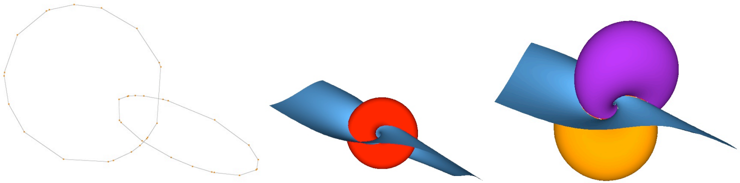
Summary

We have a symbolic-numeric algorithm (i.e. approximate algorithm) for performing operations on a plane complex algebraic curve, implemented in the library GENOM3CK.
About GENOM3CK: `http://people.ricam.oeaw.ac.at/m.hodorog/software.html`

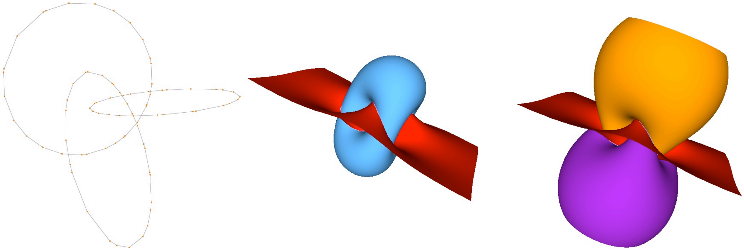
Equation	Link
$x^2 - y^2, \epsilon = 1.0$	Hopf link
$x^2 - y^3, \epsilon = 1.0$	Trefoil knot
$x^3 - y^3, \epsilon = 1.0$	3-knots link
$x^2 - y^4, \epsilon = 1.0$	2-knots link
$x^2 - y^5, \epsilon = 1.0$	1-knot link
$x^4 + x^2y + y^5, \epsilon = 0.5$	3-knots link



Summary (pictures made with GENOM3CK in Axel)



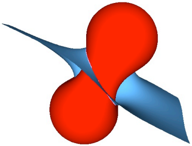
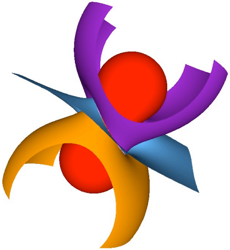
$x^2 - y^2$



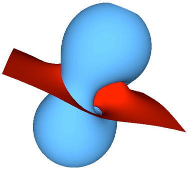
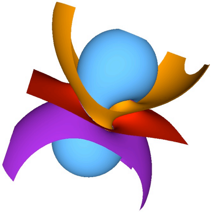
$x^3 - y^3$

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Summary (pictures made with GENOM3CK in Axel)

$x^2 - y^4$

$x^2 - y^5$

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Conclusion and future work

✓ DONE:

- complete automatization of the approximate algorithm (in GENOM3CK); we compute the singularities, topology/algebraic link, Alexander polynomial, delta-invariant, genus;
- experiments show the output is unique and continuously depends on the data;

✗ TO DO's:

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Conclusion and future work

✓ DONE:

- complete automatization of the approximate algorithm (in GENOM3CK); we compute the singularities, topology/algebraic link, Alexander polynomial, delta-invariant, genus;
- experiments show the output is unique and continuously depends on the data;

X TO DO's:

- prove the properties of the approximate algorithm (i.e. convergency, continuity);

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Conclusion and future work

✓ DONE:

- complete automatization of the approximate algorithm (in GENOM3CK); we compute the singularities, topology/algebraic link, Alexander polynomial, delta-invariant, genus;
- experiments show the output is unique and continuously depends on the data;
- we can describe it with principles from regularization theory, approximate algebraic computation.

✗ TO DO's:

- prove the properties of the approximate algorithm (i.e. convergency, continuity);

Conclusion and future work

✓ DONE:

- complete automatization of the approximate algorithm (in GENOM3CK); we compute the singularities, topology/algebraic link, Alexander polynomial, delta-invariant, genus;
- experiments show the output is unique and continuously depends on the data;
- we can describe it with principles from regularization theory, approximate algebraic computation.

X TO DO's:

- prove the properties of the approximate algorithm (i.e. convergency, continuity);
- make precise the meaning of the computed output with the approximate algorithm.

Thank you for your attention.
Questions?