A symbolic-numeric algorithm for genus computation

Mădălina Hodorog
Supervisor: Prof. Dr. Josef Schicho

Johann Radon Institute for Computational and Applied Mathematics,
Austrian Academy of Sciences,
Research Institute for Symbolic Computation,
Johannes Kepler University Linz, Austria

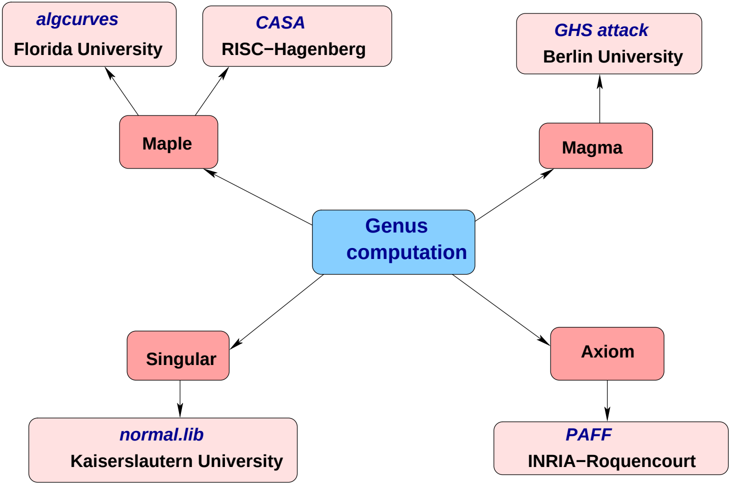
July 9, 2009

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Table of contents

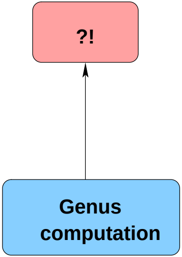
Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Symbolic Algorithms:



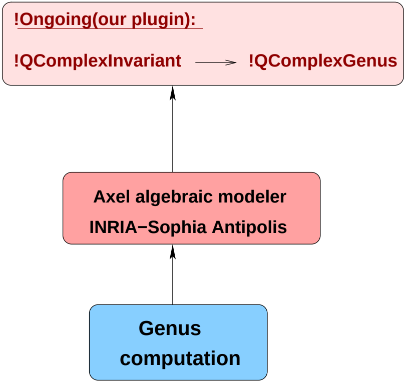
Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Numeric Algorithms:



Symbolic-Numeric Algorithms:

DK Project: Symbolic-Numeric techniques for genus computation and parametrization (project leader: Prof. Dr. Josef Schicho).



Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

What?

- Input:
  - $\mathbb{C}$ field of complex numbers;
  - $F \in \mathbb{C}[z, w]$ irreducible with coefficients of limited accuracy [1];
  - $C = \{(z, w) \in \mathbb{C}^2 | F(z, w) = 0\} =$
    $= \{(x, y, u, v) \in \mathbb{R}^4 | F(x + iy, u + iv) = 0\}$ complex algebraic curve (d is the degree, $Sing(C)$ is the set of singularities);
- Output:
  - approximate $genus(C)$ s.t.

$$genus(C) = \frac{1}{2}(d-1)(d-2) - \sum_{P \in Sing(C)} \delta\text{-invariant}(P);$$

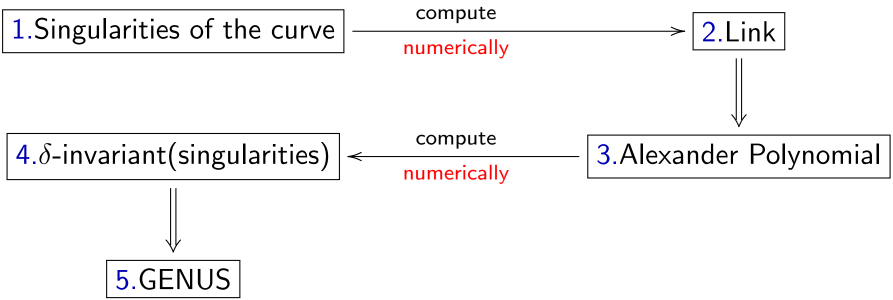
[1] For now: symbolic coefficients

Doctoral Program Computational Mathematics Numerical Analysis and Symbolic Computation

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

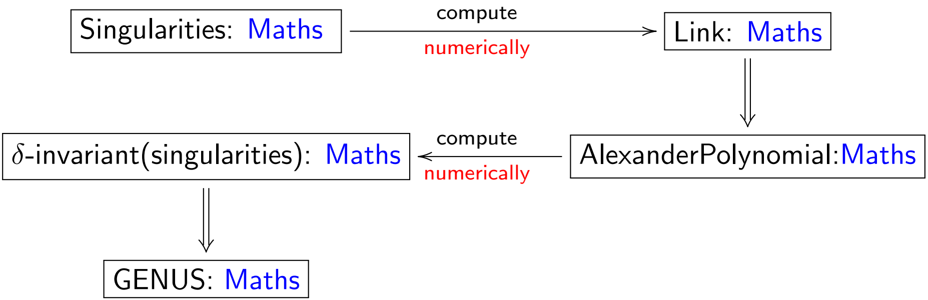
How?

- Strategy for computing the genus



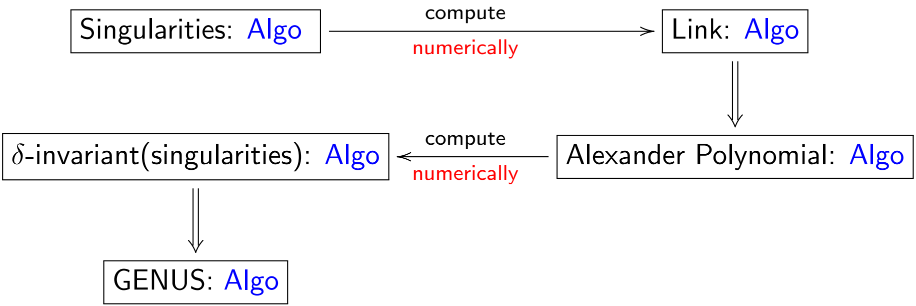
How?

- Method for computing the genus



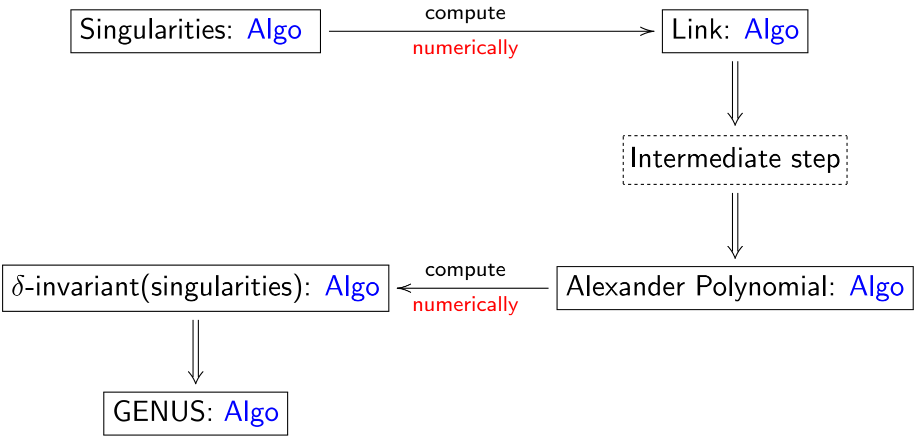
How?

- Algorithm for the method



How?

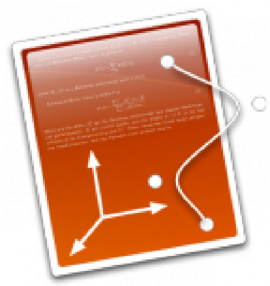
- Algorithm for the method



Solving the problem

Implementation of the algorithm

- *Axel* algebraic geometric modeler [a]
  - developed by *Galaad* team (INRIA Sophia-Antipolis);
  - written in Qt scripting language;
  - provides algebraic tools for:
    - implicit curves;
    - implicit surfaces.



[a]Acknowledgements: B. Mourrain, J. Wintz

Solving the problem

Implementation of the algorithm

- *Axel* algebraic geometric modeler [a]
    - developed by *Galaad* team (INRIA Sophia-Antipolis);
    - written in Qt scripting language;
    - provides algebraic tools for:
        - implicit curves;
        - implicit surfaces.

[a] Acknowledgements: B. Mourrain, J. 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 Qt scripting language;
    - provides algebraic tools for:
        - implicit curves;
        - implicit surfaces.

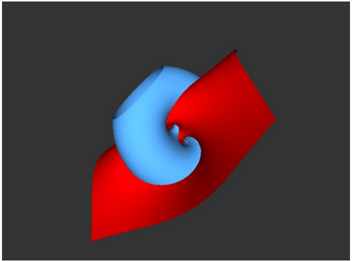
[a] Acknowledgements: B. Mourrain, J. Wintz

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Solving the problem

Implementation of the algorithm

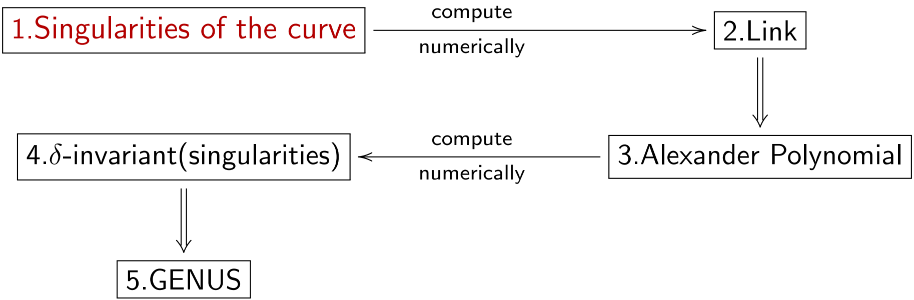
- *Axel* algebraic geometric modeler [a]
  - developed by *Galaad* team (INRIA Sophia-Antipolis);
  - written in Qt scripting language;
  - provides algebraic tools for:
    - implicit curves;
    - implicit surfaces.



[a]Acknowledgements: B. Mourrain, J. Wintz

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

First


1.Singularities of the curve
compute
numerically
2.Link
3.Alexander Polynomial
compute
numerically
4.δ-invariant(singularities)
5.GENUS

Computing the singularities of the curve

- Input:
  - $F \in \mathbb{C}[z, w]$
  - $C = \{(z, w) \in \mathbb{C}^2 | F(z, w) = 0\}$
- Output:
  - $Sing(C) = \{(z_0, w_0) \in \mathbb{C}^2 | F(z_0, w_0) = 0, \frac{\delta F}{\delta z}(z_0, w_0) = 0, \frac{\delta F}{\delta w}(z_0, w_0) = 0\}$

Method: $\Rightarrow$ solve overdeterminate system of polynomial equations in $\mathbb{C}^2$:

$$\begin{cases} F(z_0, w_0) = 0 \\ \frac{\delta F}{\delta z}(z_0, w_0) = 0 \\ \frac{\delta F}{\delta w}(z_0, w_0) = 0 \end{cases}, \tag{1}$$

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

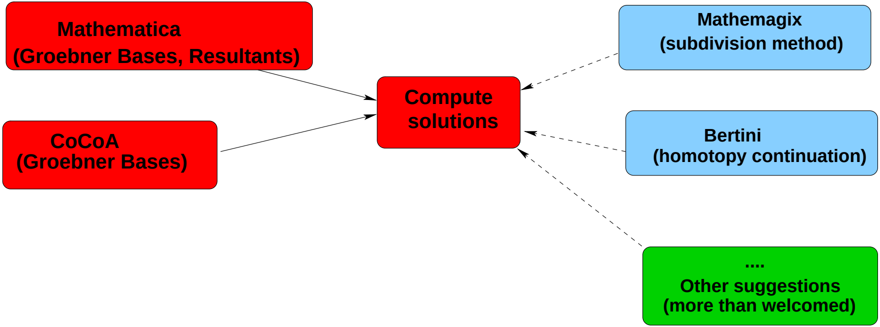
Computing the singularities of the curve

or in $\mathbb{R}^4$: $F(z,w) = F(x+iy, u+iv) = s(x,y,u,v) + it(x,y,u,v)$

$$\begin{cases} s(x_0,y_0,u_0,v_0) = 0 \\ t(x_0,y_0,u_0,v_0) = 0 \\ \\ \dfrac{\delta s}{\delta x}(x_0,y_0,u_0,v_0) = 0 \\ \\ \dfrac{\delta t}{\delta x}(x_0,y_0,u_0,v_0) = 0 \\ \\ \dfrac{\delta s}{\delta u}(x_0,y_0,u_0,v_0) = 0 \\ \\ \dfrac{\delta t}{\delta u}(x_0,y_0,u_0,v_0) = 0 \end{cases} , \qquad (2)$$

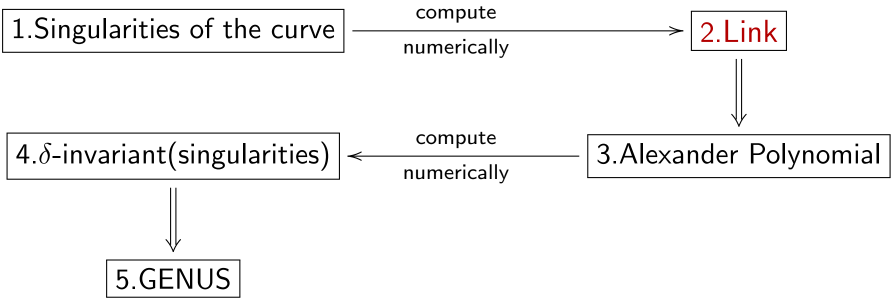
Computing the singularities of the curve

Using numeric input polynomials



Note: so far an open problem.

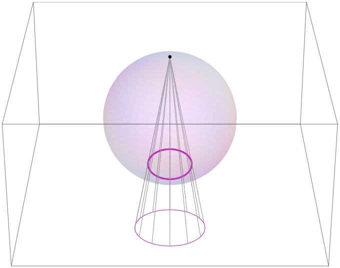
Next



Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Computing the link of the singularity

- Why the link of a singularity?
  - helps in understanding the topology of a complex curve near a singularity;
- How do we compute the link?
  - use stereographic projection;



•

Computing the link of the singularity

Method (based on Milnor's results)

1. Let $C = \{(x,y,u,v) \in \mathbb{R}^4 | F(x,y,u,v) = 0\}$ s.t. $(0,0,0,0) \in Sing(C)$

2. Consider $S_{(0,\epsilon)} := S = \{(x,y,u,v) \in \mathbb{R}^4 | x^2 + y^2 + u^2 + w^2 = \epsilon^2\}$,
   $X = C \bigcap S_{(0,\epsilon)} \subset \mathbb{R}^4$

3. For $P \in S \setminus C$ take $f : S \setminus \{P\} \to \mathbb{R}^3, f(x,y,u,v) = (\frac{x}{\epsilon - v}, \frac{y}{\epsilon - v}, \frac{u}{\epsilon - v})$,
   $f^{-1} : \mathbb{R}^3 \to S \setminus \{P\}$
   $f^{-1}(a,b,c) = (\frac{2a\epsilon}{1+a^2+b^2+c^2}, \frac{2b\epsilon}{1+a^2+b^2+c^2}, \frac{2c\epsilon}{1+a^2+b^2+c^2}, \frac{\epsilon(a^2+b^2+c^2-1)}{1+a^2+b^2+c^2})$

4. Compute $f(X) = \{(a,b,c,) \in \mathbb{R}^3 | F(...) = 0\} \Leftrightarrow$
   $f(X) = \{(a,b,c,) \in \mathbb{R}^3 | ReF(...) = 0, ImF(...) = 0\}$ and
   B for $f(X) = \{(a,b,c,) \in B \subset \mathbb{R}^3 | ReF(...) = 0, ImF(...) = 0\}$
   For small ϵ, $f(X)$ is a link

Note: A link is a closed loop in $\mathbb{R}^3$ that does not intersect itself.

Computing the link of the singularity

Why Axel?

It computes numerically the topology of implicit curves in $\mathbb{R}^3$

- For $C^4 = \{(z,w) \in \mathbb{C}^2 | z^3 - w^2 = 0\} \subset \mathbb{R}^4$ get
- $f(C^4 \cap S) := C =$
  $= \{(a,b,c) \in \mathbb{R}^3 | ReF(...) = 0, ImF(...) = 0\}$
- compute $Graph(C) = \langle \mathcal{V}, \mathcal{E} \rangle$ with
  $\mathcal{V} = \{p = (m,n,q) \in \mathbb{R}^3\}$
  $\mathcal{E} = \{(i,j) | i,j \in \mathcal{V}\}$
- s.t. $Graph(C) \cong_{isotopic} C$

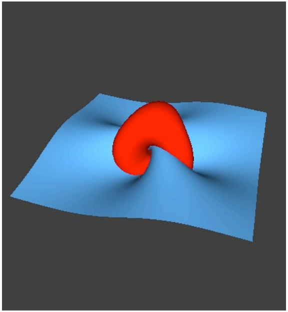
Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Computing the link of the singularity

Why Axel?

It computes numerically the topology of implicit curves in $\mathbb{R}^3$

- For $C^4 = \{(z,w) \in \mathbb{C}^2 | z^3 - w^2 = 0\} \subset \mathbb{R}^4$ get
- $f(C^4 \cap S) := C =$
  $= \{(a,b,c) \in \mathbb{R}^3 | ReF(...) = 0, ImF(...) = 0\}$
- compute $Graph(C) = \langle \mathcal{V}, \mathcal{E} \rangle$ with
  $\mathcal{V} = \{p = (m,n,q) \in \mathbb{R}^3\}$
  $\mathcal{E} = \{(i,j) | i,j \in \mathcal{V}\}$
- s.t. $Graph(C) \cong_{isotopic} C$

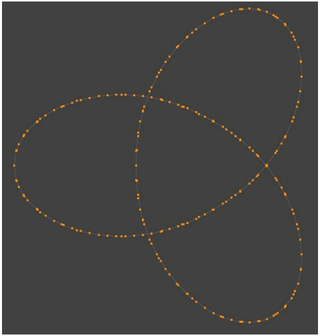


Computing the link of the singularity

Why Axel?

It computes numerically the topology of implicit curves in $\mathbb{R}^3$

- For $C^4 = \{(z, w) \in \mathbb{C}^2 | z^3 - w^2 = 0\} \subset \mathbb{R}^4$ get
- $f(C^4 \cap S) := C =$
  $= \{(a, b, c) \in \mathbb{R}^3 | ReF(...) = 0, ImF(...) = 0\}$
- compute $Graph(C) = \langle \mathcal{V}, \mathcal{E} \rangle$ with
  $\mathcal{V} = \{p = (m, n, q) \in \mathbb{R}^3\}$
  $\mathcal{E} = \{(i, j) | i, j \in \mathcal{V}\}$
- s.t. $Graph(C) \cong_{isotopic} C$

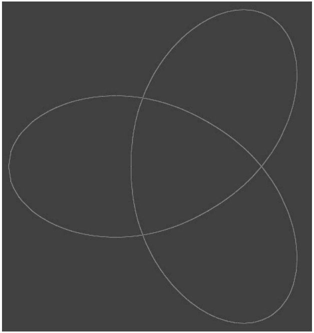


Computing the link of the singularity

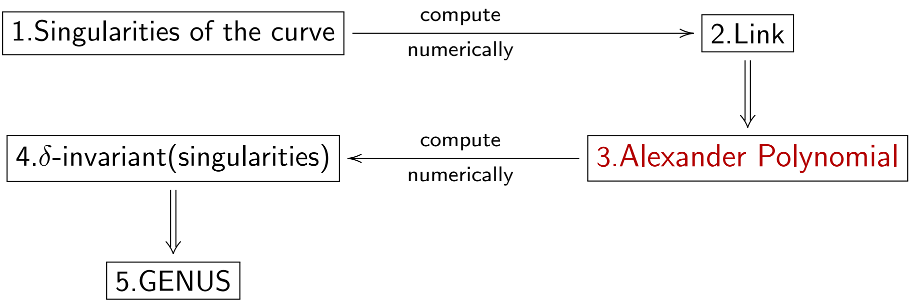
Why Axel?

It computes numerically the topology of implicit curves in $\mathbb{R}^3$

- For $C^4 = \{(z,w) \in \mathbb{C}^2 | z^3 - w^2 = 0\} \subset \mathbb{R}^4$ get
- $f(C^4 \cap S) := C =$
  $= \{(a,b,c) \in \mathbb{R}^3 | ReF(...) = 0, ImF(...) = 0\}$
- compute $Graph(C) = \langle \mathcal{V}, \mathcal{E} \rangle$ with
  $\mathcal{V} = \{p = (m,n,q) \in \mathbb{R}^3\}$
  $\mathcal{E} = \{(i,j) | i,j \in \mathcal{V}\}$
- s.t. $Graph(C) \cong_{isotopic} C$



Next


1.Singularities of the curve
compute
numerically
2.Link
3.Alexander Polynomial
compute
numerically
4.δ-invariant(singularities)
5.GENUS

Preliminaries

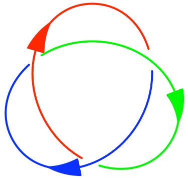
A double point of a projection is called a **crossing point**.

A **diagram** is the image under regular projection, together with the information on each crossing telling which branch goes over and which under.

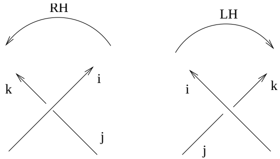
An **arc** is the part of a diagram between two undercrossings.

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

Diagram and arcs

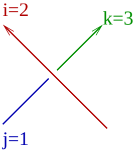


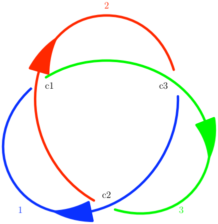
Crossings



Doctoral Program
Computational Mathematics

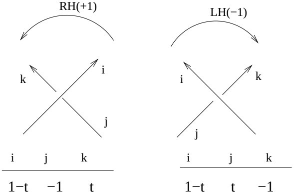
Computing the Alexander polynomial of the link





$$M(L) = \left(\begin{array}{c|cccc} & type & label_i & label_j & label_k \\ \hline c_1 & -1 & 2 & 1 & 3 \\ & & & & \end{array}\right)$$

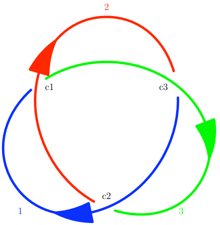
$$P(L) = \left(\quad \right)$$

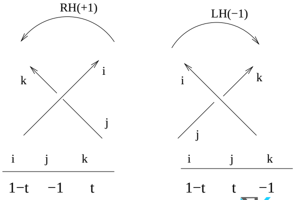


Computing the Alexander polynomial of the link

$$M(L) = \left(\begin{array}{c|cccc} & type & label_i & label_j & label_k \\ \hline c_1 & -1 & 2 & 1 & 3 \\ & & 1-t & t & -1 \\ & & & & \end{array}\right)$$

$$P(L) = \begin{pmatrix} 2 & 1 & 3 \\ 1-t & t & -1 \\ & & \end{pmatrix}$$

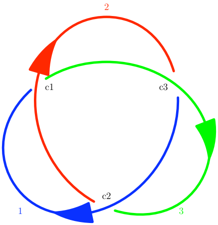


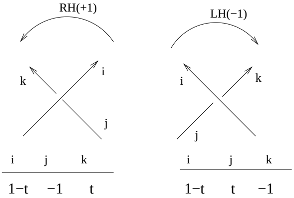


Computing the Alexander polynomial of the link

$$M(L) = \left(\begin{array}{c|cccc} & type & label_i & label_j & label_k \\ \hline c_1 & -1 & 2 & 1 & 3 \\ & & 1-t & t & -1 \\ & & & & \end{array} \right)$$

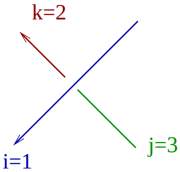
$$P(L) = \left(\begin{array}{ccc} 1 & 2 & 3 \\ t & 1-t & -1 \\ & & \end{array} \right)$$

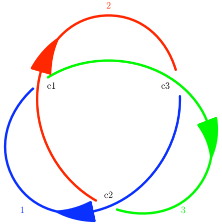




Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

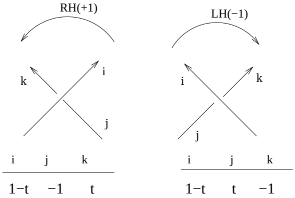
Computing the Alexander polynomial of the link





$$M(L) = \left(\begin{array}{c|cccc} & type & label_i & label_j & label_k \\ \hline c_2 & -1 & 1 & 3 & 2 \end{array} \right)$$

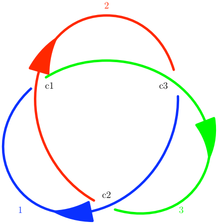
$$P(L) = \left(\quad \right)$$

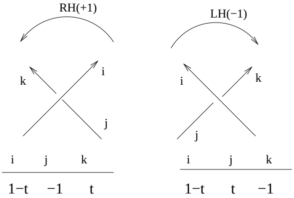


Computing the Alexander polynomial of the link

$$M(L) = \left(\begin{array}{c|cccc} & type & label_i & label_j & label_k \\ \hline c_2 & -1 & 1 & 3 & 2 \\ & & 1-t & t & -1 \end{array} \right)$$

$$P(L) = \left(\begin{array}{ccc} 1 & 3 & 2 \\ 1-t & t & -1 \end{array} \right)$$

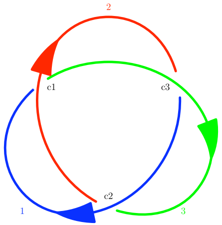


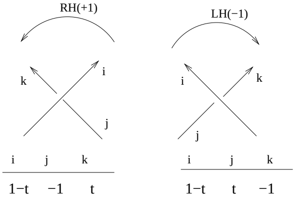


Computing the Alexander polynomial of the link

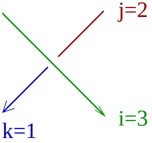
$$M(L) = \left(\begin{array}{c|cccc} & type & label_i & label_j & label_k \\ \hline c_2 & -1 & 1 & 3 & 2 \\ & & 1-t & t & -1 \end{array} \right)$$

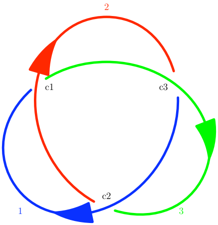
$$P(L) = \left(\begin{array}{ccc} 1 & 2 & 3 \\ 1-t & -1 & t \end{array} \right)$$





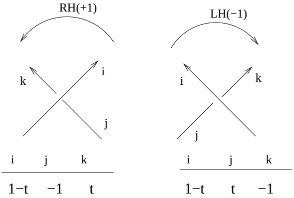
Computing the Alexander polynomial of the link





$$M(L) = \left(\begin{array}{c|cccc} & type & label_i & label_j & label_k \\ \hline & & & & \\ c_3 & -1 & 3 & 2 & 1 \end{array} \right)$$

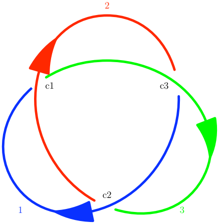
$$P(L) = \left(\quad \right)$$

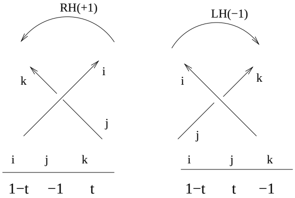


Computing the Alexander polynomial of the link

$$M(L) = \left(\begin{array}{c|cccc} & type & label_i & label_j & label_k \\ \hline & & & & \\ c_3 & -1 & 3 & 2 & 1 \\ & & 1-t & t & -1 \end{array}\right)$$

$$P(L) = \left(\begin{array}{ccc} & & \\ 3 & 2 & 1 \\ 1-t & t & -1 \end{array}\right)$$

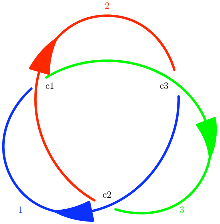


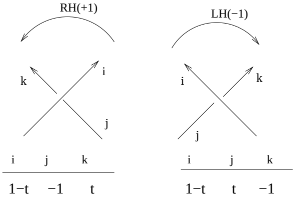


Computing the Alexander polynomial of the link

$$M(L) = \left(\begin{array}{c|cccc} & type & label_i & label_j & label_k \\ \hline & & & & \\ c_3 & -1 & 3 & 2 & 1 \\ & & 1-t & t & -1 \end{array}\right)$$

$$P(L) = \left(\begin{array}{ccc} & & \\ 1 & 2 & 3 \\ -1 & t & 1-t \end{array}\right)$$





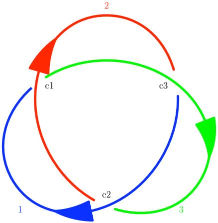
Computing the Alexander polynomial of the link

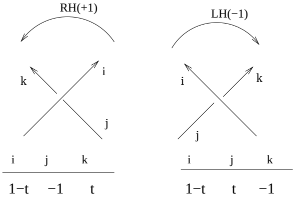
$$M(L) = \left(\begin{array}{c|cccc} & type & label_i & label_j & label_k \\ \hline c_1 & -1 & 2 & 1 & 3 \\ c_2 & -1 & 1 & 3 & 2 \\ c_3 & -1 & 3 & 2 & 1 \end{array}\right)$$

$$P(L) = \begin{pmatrix} t & 1-t & -1 \\ 1-t & -1 & t \\ -1 & t & 1-t \end{pmatrix}$$

$$det(P(L)) = -t^2 + t - 1$$

$$\Delta(L) := \Delta(t) = Normalise(det(P(L))) = t^2 - t + 1$$

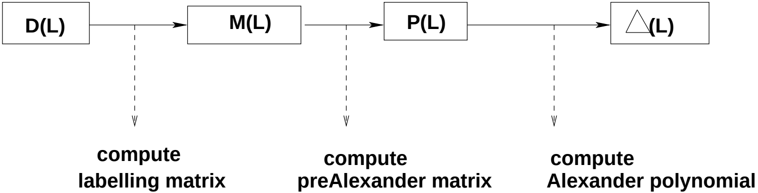




Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

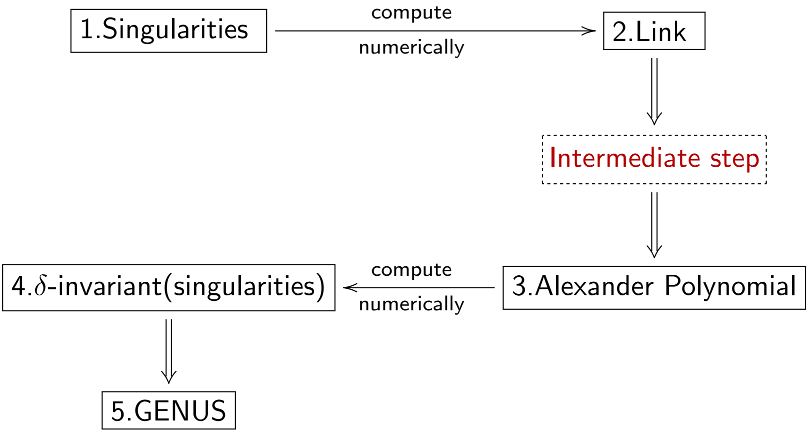
Computing the Alexander polynomial of the link

- Input:
  - $L = K_1 \cup ... \cup K_m$ with n - crossings
  - $D(L)$- oriented diagram of L
- Output:
  - $\Delta_L(t_1, ...t_m) \in \mathbb{Z}[t_1^{\pm 1}, ..., t_m^{\pm 1}]$
- Method: consists of several steps

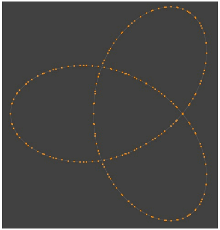


- Need D(L)!

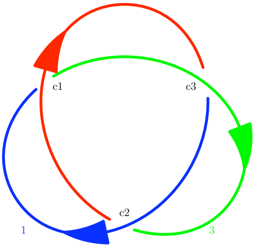
Next


1.Singularities
compute
numerically
2.Link
Intermediate step
3.Alexander Polynomial
compute
numerically
4.δ-invariant(singularities)
5.GENUS

Intermediate step

- 

$\overset{?}{\Rightarrow}$



- $G(L) = \langle P, E \rangle$

$D(L)$

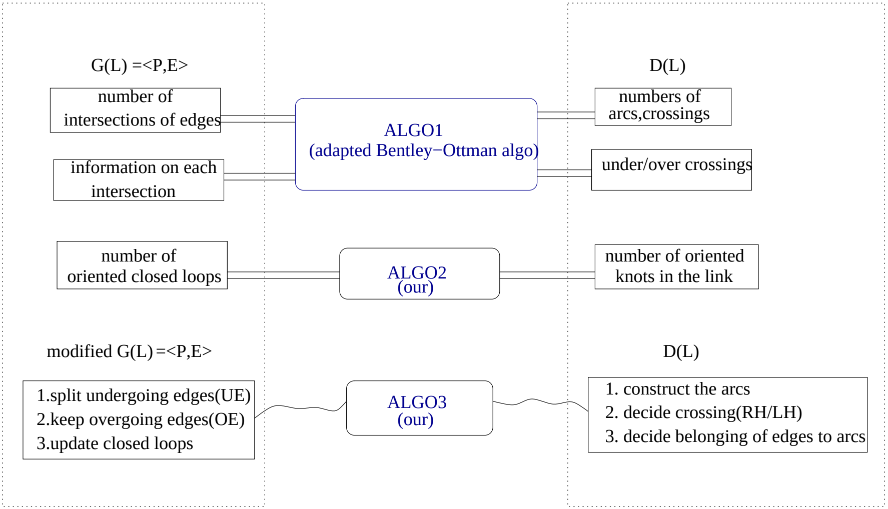
- p(index,x,y,z)

e(pointS, pointD)

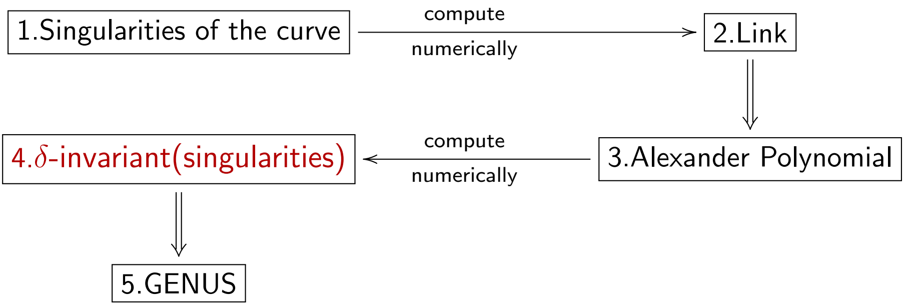
$\longrightarrow$ number of arcs, crossings
$\longrightarrow$ type of crossings (under, over)
$\longrightarrow$ number of knots in the link(orientation)

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Intermediate step



Next


1.Singularities of the curve
compute
numerically
2.Link
3.Alexander Polynomial
compute
numerically
4.δ-invariant(singularities)
5.GENUS

Computing the δ-invariant of the singularity

- Input:
  - $C \subset \mathbb{C}^2$ complex algebraic curve;
  - $z \in Singularities(C)$;
  - $\Delta(t_1, .., t_p)$ - Alexander polynomial of z;
  - $r =$ number of variables in Δ (branches of C through z);
  - $\mu =$ degree of Δ (multiplicity of z);
- Output:
  - $\delta_z > 0$ s.t. δ_z measures the number of double points of C at z.
- Method (based on Milnor's research)

$$\Delta(t_1, ..., t_p) : r, \mu$$

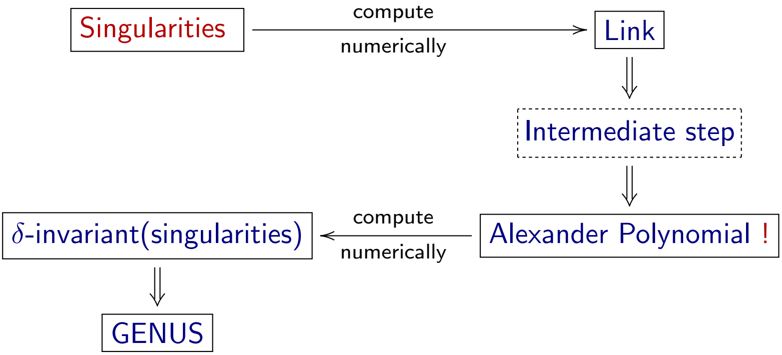
$$r \geq 2 \Rightarrow \delta = \frac{1}{2}(\mu + r)$$

$$r = 1 \Rightarrow \delta = \frac{1}{2}\mu$$

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Summary

- Present work: for symbolic coefficients



- Future work: tests for algorithm with numeric coefficients

1. Motivation

2. Describing the problem
   - What?

3. Solving the problem
   - How?

4. Current results

5. Conclusion and future work

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Conclusion

- first results (methods and algorithms) were presented;
- Future work:
  - deeper introspection into some mathematical aspects (i.e. numeric computation);
  - complete implementation of the algorithm;
  - correctness proof for the algorithm;
  - analysis of the algorithm.

Thank you for your attention.
Questions?