

The genus computation problem: symbolic-numeric solutions and beyond

Mădălina Hodorog¹

Supervisor: Josef Schicho¹

Joint work with Bernard Mourrain²

¹Johann Radon Institute for Computational and Applied Mathematics,
Austrian Academy of Sciences,
Research Institute for Symbolic Computation
Johannes Kepler University Linz, Austria

²INRIA Sophia-Antipolis, France

October 1, 2009

Table of contents

- 1 Motivation
- 2 Describing the problem
What?
- 3 Solving the problem
How?
- 4 Current results
- 5 Conclusion and future work

1 Motivation

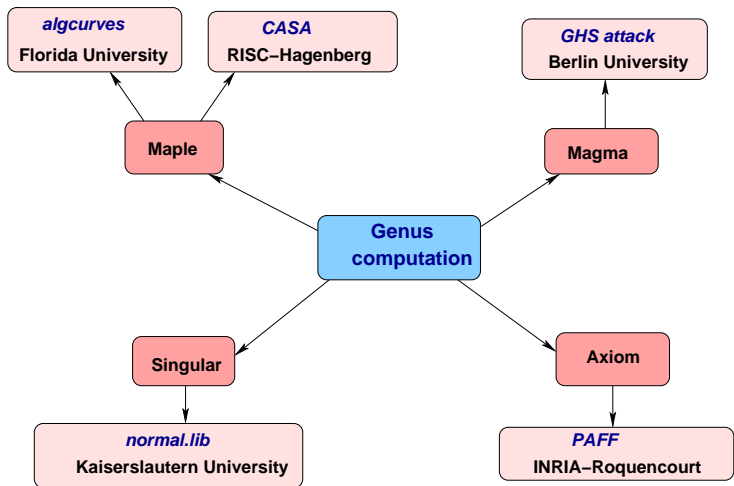
2 Describing the problem What?

3 Solving the problem How?

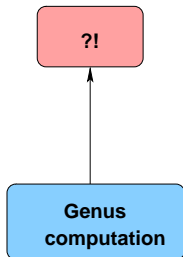
4 Current results

5 Conclusion and future work

Symbolic Algorithms:

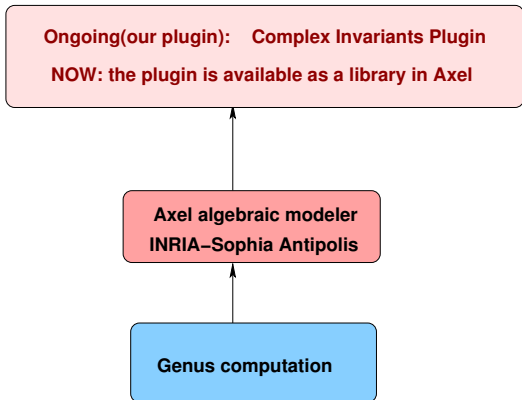


Numeric Algorithms:



Symbolic-Numeric Algorithms:

DK Project: Symbolic-Numeric techniques for genus computation and parametrization (initiated by Prof. Josef Schicho).



① Motivation

② Describing the problem
What?

③ Solving the problem
How?

④ Current results

⑤ Conclusion and future work

What?

- Input:

- \mathbb{C} field of complex numbers;
- $F \in \mathbb{C}[z, w]$ irreducible with coefficients of **limited accuracy**¹;
- $C = \{(z, w) \in \mathbb{C}^2 \mid F(z, w) = 0\} = \{(x, y, u, v) \in \mathbb{R}^4 \mid F(x + iy, u + iv) = 0\}$ complex algebraic curve (d is the degree);

- Output:

- **approximate** $genus(C)$ s.t.

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

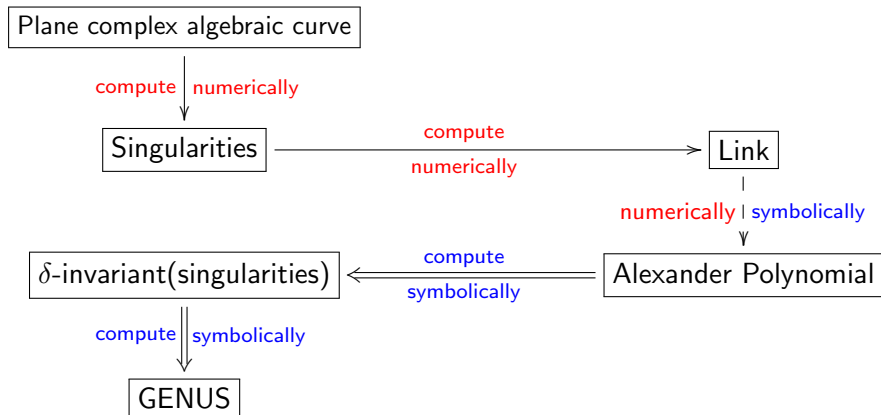
where $Sing(C)$ is the set of singularities of the curve C .

¹For now: symbolic coefficients

- ① Motivation
- ② Describing the problem
What?
- ③ Solving the problem
How?
- ④ Current results
- ⑤ Conclusion and future work

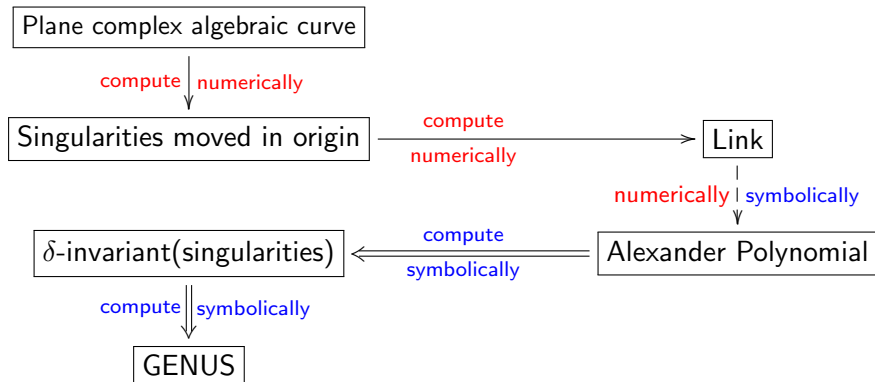
How?

- Strategy for computing the genus



How?

- Strategy for computing the genus



Solving the problem

Implementation of the algorithm

- *Axel* algebraic geometric modeler ^a
 - developed by *Galaad* team (INRIA Sophia-Antipolis);
 - in C++, Qt scripting language;
 - provides algebraic tools for:
 - implicit surfaces;
 - implicit curves.
 - free, available at:



^aAcknowledgements: Julien Wintz

Solving the problem

Implementation of the algorithm

- *Axel* algebraic geometric modeler ^a
 - developed by *Galaad* team (INRIA Sophia-Antipolis);
 - in C++, Qt scripting language;
 - provides algebraic tools for:
 - implicit surfaces;
 - implicit curves.
 - free, available at:

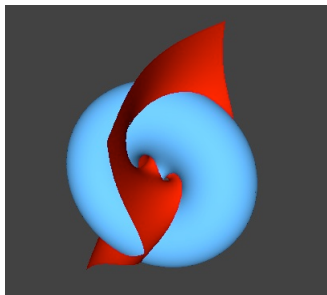


^aAcknowledgements: Julien Wintz

Solving the problem

Implementation of the algorithm

- *Axel* algebraic geometric modeler ^a
 - developed by *Galaad* team (INRIA Sophia-Antipolis);
 - in C++, Qt scripting language;
 - provides algebraic tools for:
 - implicit surfaces;
 - implicit curves.
 - free, available at:



^aAcknowledgements: Julien Wintz

Solving the problem

Implementation of the algorithm

- *Axel* algebraic geometric modeler ^a
 - developed by *Galaad* team (INRIA Sophia-Antipolis);
 - in C++, Qt scripting language;
 - provides algebraic tools for:
 - implicit surfaces;
 - implicit curves.
 - free, available at:

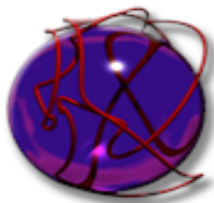


^aAcknowledgements: Julien Wintz

Solving the problem

Implementation of the algorithm

- *Axel* algebraic geometric modeler ^a
 - developed by *Galaad* team (INRIA Sophia-Antipolis);
 - in C++, Qt scripting language;
 - provides algebraic tools for:
 - implicit surfaces;
 - implicit curves.
 - free, available at:

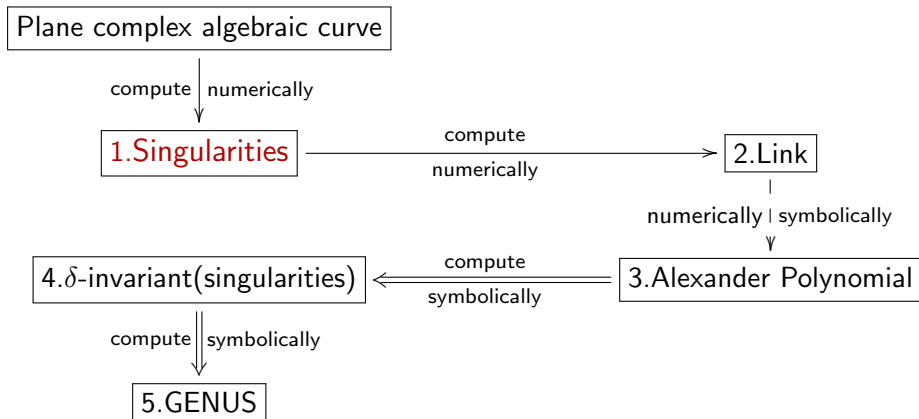


<http://axel.inria.fr/>

^aAcknowledgements: Julien Wintz

- 1 Motivation
- 2 Describing the problem
What?
- 3 Solving the problem
How?
- 4 Current results
- 5 Conclusion and future work

First



Computing the singularities of the curve

- **Input:**

- $F \in \mathbb{C}[z, w]$
- $C = \{(z, w) \in \mathbb{C}^2 \mid F(z, w) = 0\}$

- **Output:**

- $Sing(C) = \{(z_0, w_0) \in \mathbb{C}^2 \mid F(z_0, w_0) = 0, \frac{\partial F}{\partial z}(z_0, w_0) = 0, \frac{\partial F}{\partial w}(z_0, w_0) = 0\}$

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

$$\left\{ \begin{array}{l} F(z_0, w_0) = 0 \\ \frac{\partial F}{\partial z}(z_0, w_0) = 0 \\ \frac{\partial F}{\partial w}(z_0, w_0) = 0 \end{array} \right. , \quad (1)$$

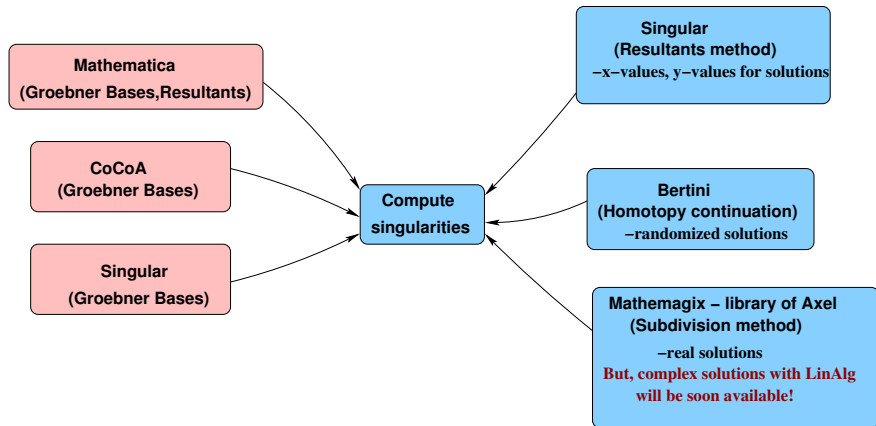
Computing the singularities of the curve

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

$$\left\{ \begin{array}{l} s(x_0, y_0, u_0, v_0) = 0 \\ t(x_0, y_0, u_0, v_0) = 0 \\ \frac{\partial s}{\partial x}(x_0, y_0, u_0, v_0) = 0 \\ \frac{\partial t}{\partial x}(x_0, y_0, u_0, v_0) = 0 \\ \frac{\partial s}{\partial u}(x_0, y_0, u_0, v_0) = 0 \\ \frac{\delta t}{\delta u}(x_0, y_0, u_0, v_0) = 0 \end{array} \right. , \quad (2)$$

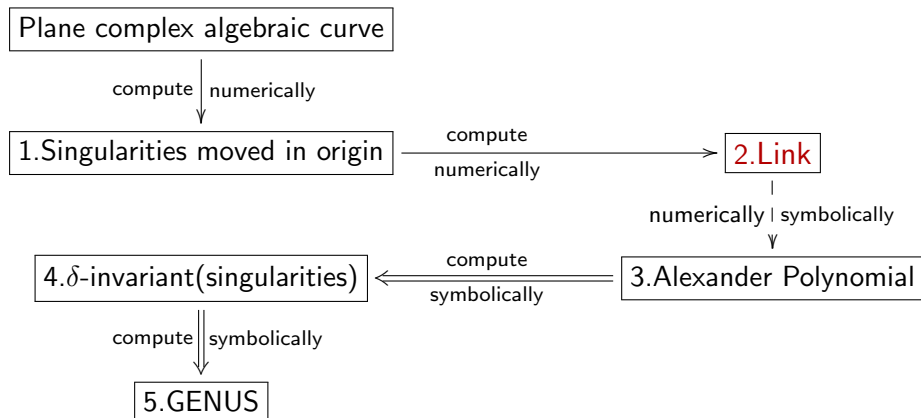
Computing the singularities of the curve

For input polynomials with numeric coefficients



Note: so far this is an open problem.

Next

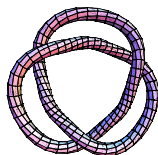


Knot theory - preliminaries

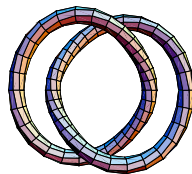
- A **knot** is a simple closed curve in \mathbb{R}^3 .
- A **link** is a finite union of disjoint knots.
- Links resulted from the intersection of a given curve with the sphere are called **algebraic links**.

Note: Alexander polynomial is a complete invariant for the algebraic links (Yamamoto 1984).

Trefoil Knot

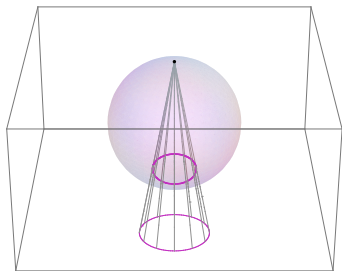


Hopf Link



Computing the link of the singularity

- Why the link of a singularity?
 - helps to study 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 $C = \{(x, y, u, v) \in \mathbb{R}^4 \mid F(x, y, u, v) = 0\}$ s.t. $(0, 0, 0, 0) \in \text{Sing}(C)$
2. Consider $S_{(0, \epsilon)} := S = \{(x, y, u, v) \in \mathbb{R}^4 \mid x^2 + y^2 + u^2 + v^2 = \epsilon^2\}$,
 $X = C \cap S_{(0, \epsilon)} \subset \mathbb{R}^4$
3. For $P \in S \setminus C$ take $f : S \setminus \{P\} \rightarrow \mathbb{R}^3$, $f(x, y, u, v) = (\frac{x}{\epsilon - v}, \frac{y}{\epsilon - v}, \frac{u}{\epsilon - v})$,
 $f^{-1} : \mathbb{R}^3 \rightarrow S \setminus \{P\}$
 $f^{-1}(a, b, c) = (\frac{2a\epsilon}{1+a^2+b^2+c^2}, \frac{2b\epsilon}{1+a^2+b^2+c^2}, \frac{2c\epsilon}{1+a^2+b^2+c^2}, \frac{\epsilon(a^2+b^2+c^2-1)}{1+a^2+b^2+c^2})$
4. Compute $f(X) = \{(a, b, c) \in \mathbb{R}^3 \mid F(\dots) = 0\} \Leftrightarrow$
 $f(X) = \{(a, b, c) \in \mathbb{R}^3 \mid \text{Re}F(\dots) = 0, \text{Im}F(\dots) = 0\}$
For small ϵ , $f(X)$ is a link.

Computing the link of the singularity

Why Axel?

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

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

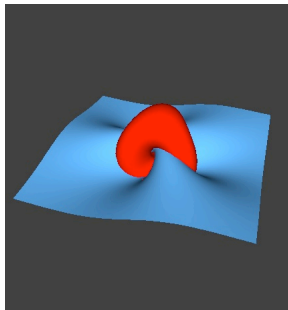


Computing the link of the singularity

Why Axel?

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

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

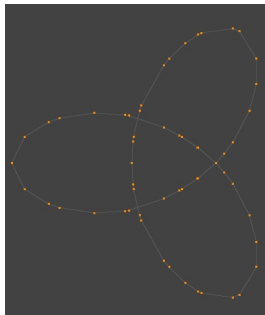


Computing the link of the singularity

Why Axel?

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

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



Computing the link of the singularity

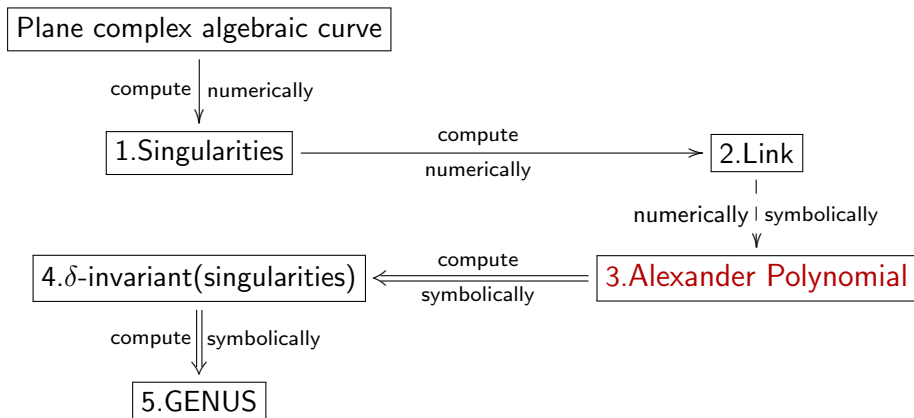
Why Axel?

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

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

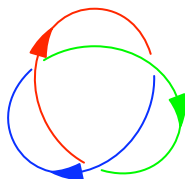


Next



Knot theory - preliminaries

Diagram and arcs



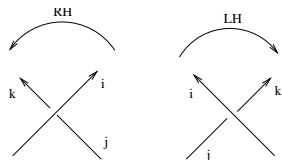
A **diagram** is the image under regular projection, together with the information on each crossing telling which branch goes over and which under.

A crossing is:

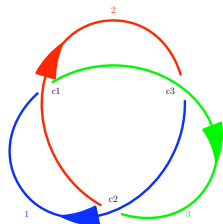
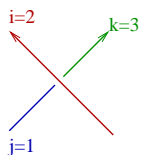
-**righthanded** if the underpass traffic goes from right to left.

-**lefthanded** if the underpass traffic goes from left to right.

Crossings

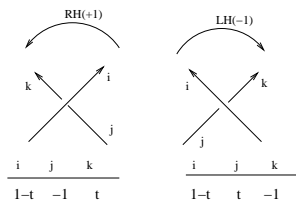


Computing the Alexander polynomial of the link



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

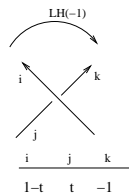
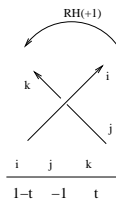
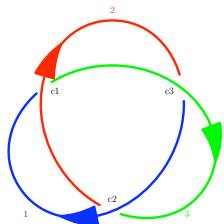
$$P(L) = \left(\begin{array}{c} \\ \\ \\ \end{array} \right)$$



Computing the Alexander polynomial of the link

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

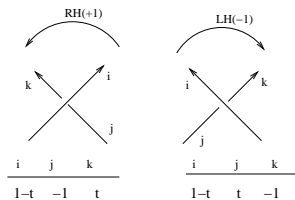
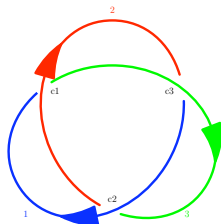
$$P(L) = \begin{pmatrix} 2 & 1 & 3 \\ 1-t & t & -1 \end{pmatrix}$$



Computing the Alexander polynomial of the link

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

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



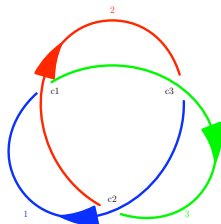
Computing the Alexander polynomial of the link

For a link with $K = 1$ knot:

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

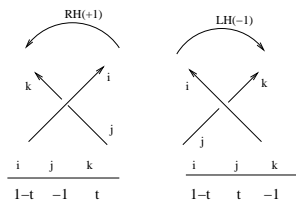
$$D := \det(\text{minor}(P(L))) = -t^2 + t - 1$$

$$\Delta(L) := \Delta(t) = \text{Normalise}(D) = t^2 - t + 1$$



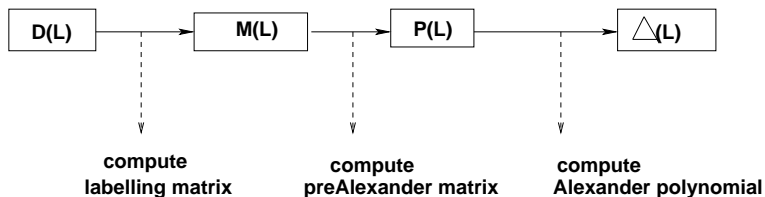
For a link with $K > 1$ knots and n crossings $\Delta(L)$ is the *gcd* of all the $(n-1) \times (n-1)$ minor determinants of $P(L)$.

Note: The Alexander polynomial is $\Delta(L)$.



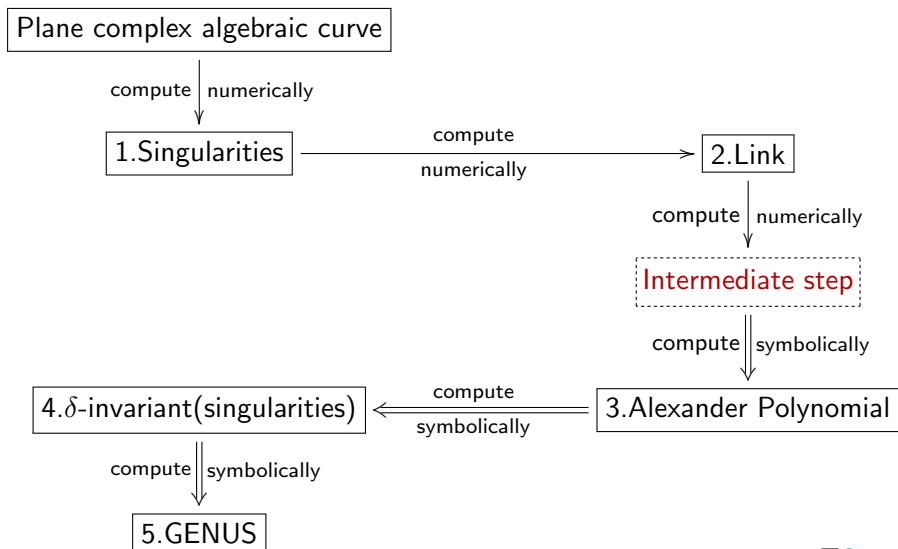
Computing the Alexander polynomial of the link

So, the Alexander polynomial is computed in several steps:

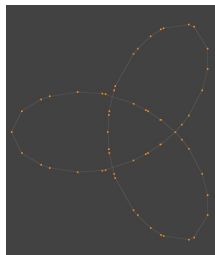


In order to compute it, we need $D(L)$!

Next



Intermediate step



•

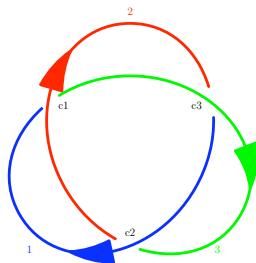
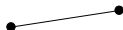
• $G(L) = \langle P, E \rangle$

•

$p(\text{index}, x, y, z)$

•

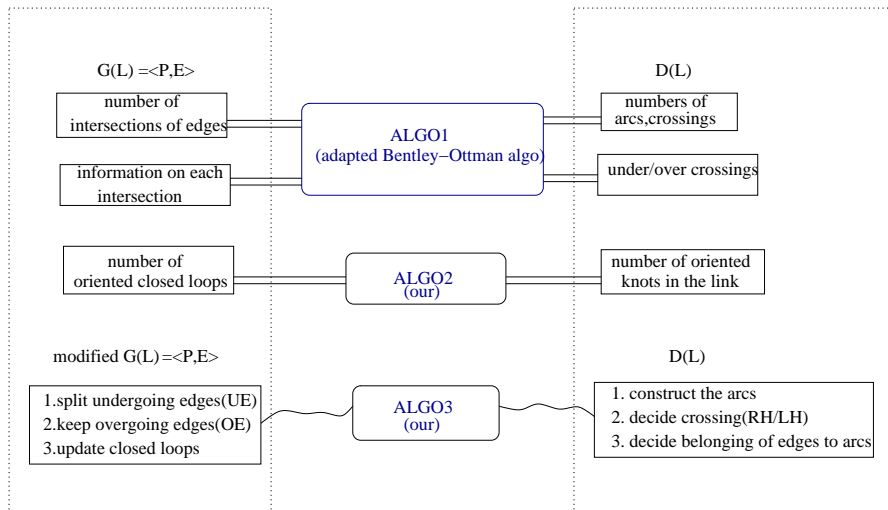
$e(\text{pointS}, \text{pointD})$



$D(L)$

- number of arcs, crossings
- type of crossings (under, over)
- number of knots in the link(orientation)

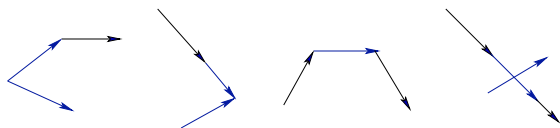
Intermediate step



Algorithm 1 - Adapted version of Bentley-Ottman

- Input

- S a set of "short" edges ordered from left to right:

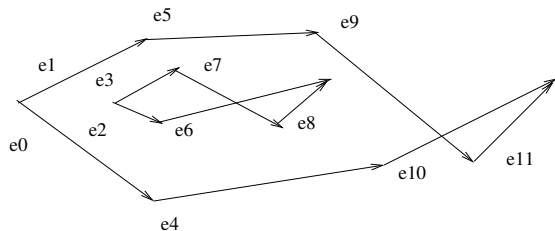


- Output

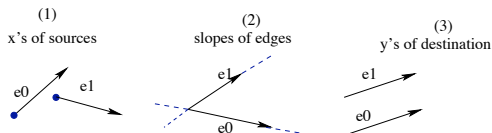
- I - the set of all intersections among edges of S and
- for each $p \in I$, the "arranged" pair of edges (e_i, e_j) such that $p = e_i \cap e_j$.

Note: (e_i, e_j) is an "arranged" pair of edges if and only if for $p = e_i \cap e_j$, e_i is below e_j in \mathbb{R}^3 .

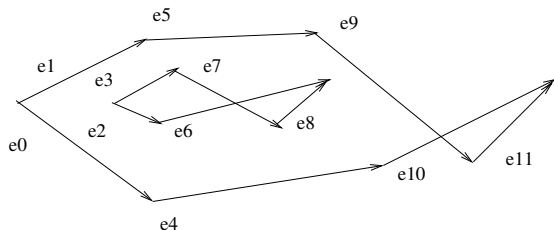
Algorithm 1 - Adapted version of Bentley-Ottman



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

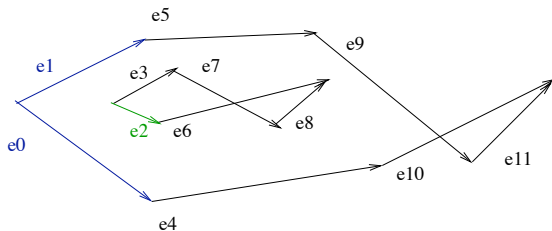


Algorithm 1 - Adapted version of Bentley-Ottman



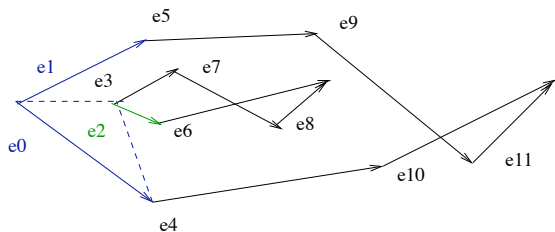
- we consider l a sweep line
- we keep track of two lists:
 $E = \{e_0, e_1, \dots, e_{11}\}$ the list of ordered edges
 $Sw = \{?\}$ the list of event points
- while traversing E we insert the edges in Sw in the "right" position
- That is...

Algorithm 1 - Adapted version of Bentley-Ottman



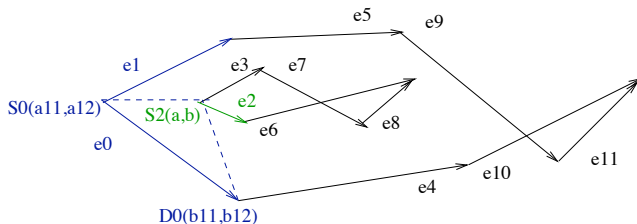
- $E = \{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$
- $Sw = \{e_0, e_1\}$

Algorithm 1 - Adapted version of Bentley-Ottman



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

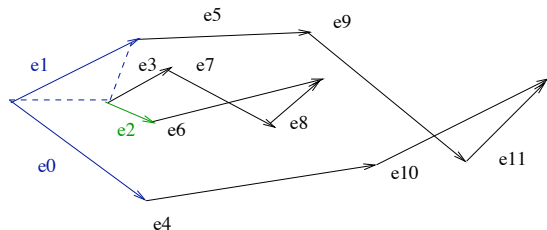
Algorithm 1 - Adapted version of Bentley-Ottman



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

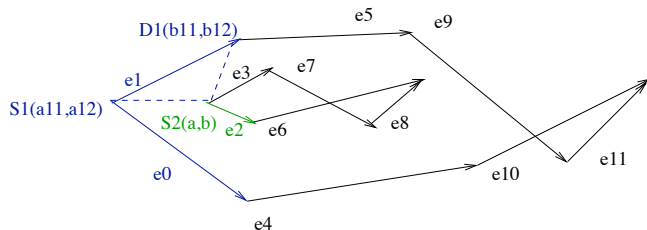
$$\det(e_2, e_0) = \begin{pmatrix} a_{11} & a_{12} & 1 \\ b_{11} & b_{12} & 1 \\ a & b & 1 \end{pmatrix} > 0 \Rightarrow e_2 \text{ after } e_0 \text{ in } Sw$$

Algorithm 1 - Adapted version of Bentley-Ottman



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

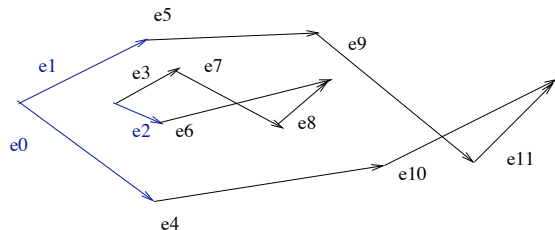
Algorithm 1 - Adapted version of Bentley-Ottman



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

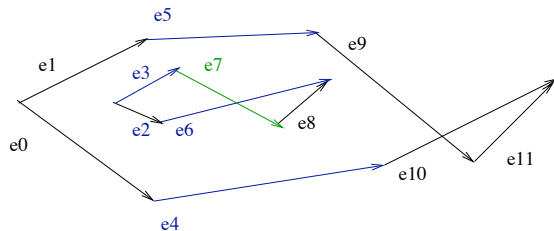
$$\det(e_2, e_1) = \begin{pmatrix} a_{11} & a_{12} & 1 \\ b_{11} & b_{12} & 1 \\ a & b & 1 \end{pmatrix} < 0 \Rightarrow e_2 \text{ before } e_1 \text{ in } Sw$$

Algorithm 1 - Adapted version of Bentley-Ottman



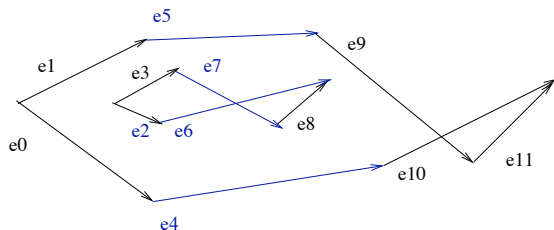
- $E = \{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$
- $Sw = \{e_0, e_2, e_1\}$
- Test $e_0 \cap e_2$? No!
Test $e_2 \cap e_1$? No!
- $I = \emptyset$
 $E_I = \emptyset$

Algorithm 1 - Adapted version of Bentley-Ottman



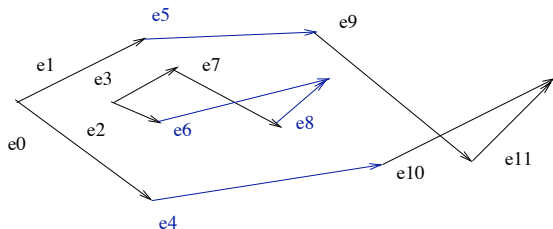
- $E = \{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$
- $Sw = \{e_4, e_6, e_3, e_5\}$

Algorithm 1 - Adapted version of Bentley-Ottman



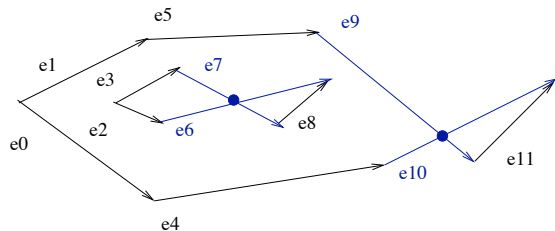
- $E = \{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$
- $Sw = \{e_4, e_6, e_7, e_5\}$
- Test $e_6 \cap e_7 = ?$ Yes!
Test $e_7 \cap e_5 = ?$ No! $\Rightarrow I = \{(a_1, b_1)\}$ $E_I = \{(e_6, e_7)\}$
 $Sw = \{e_4, e_7, e_6, e_5\}$

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
 $Sw = \{e_4, e_5\}$

Algorithm 1 - Adapted version of Bentley-Ottman



- Final output

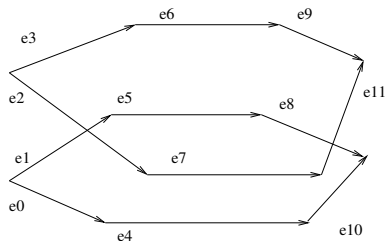
- $I = \{i_1 = (a_1, b_1), i_2 = (a_2, b_2)\}$

$$E_I = \{(e_6, e_7), (e_{10}, e_9)\} \text{ with}$$

- e_6 below e_7 in \mathbb{R}^3 and
- e_{10} below e_9 in \mathbb{R}^3

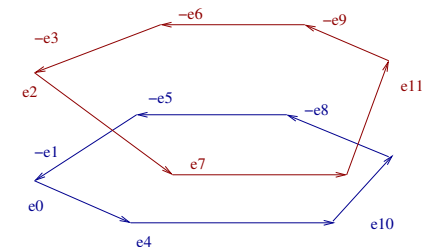
Specific algorithms

- Algorithm for constructing the loops produces:



- E ordered by (1),(2),(3)

\Rightarrow

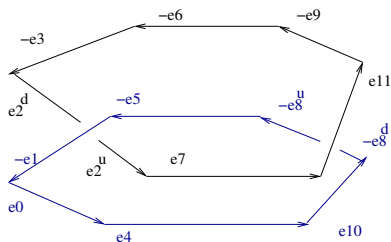
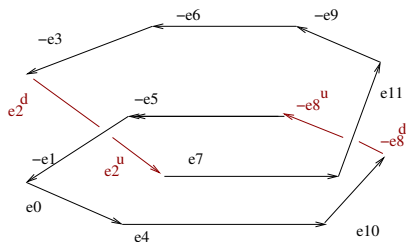


$$L_0 = \{e_0, e_4, e_{10}, -e_8, -e_5, -e_1\}$$

$$L_1 = \{e_2, e_7, e_{11}, -e_9, -e_6, -e_3\}$$

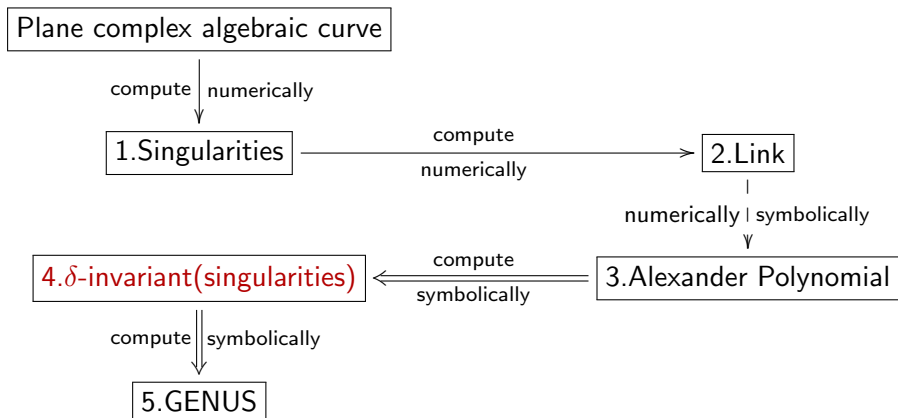
Specific algorithms

- Algorithm for constructing the arcs produces:



- $E_I = \{(e_2, e_1), (e_8, e_{11})\} \Rightarrow a_0 = \{-e_8^u, -e_5, -e_1, e_0, e_4, e_{10}, -e_8^d\}$
- $L_0 = \{e_0, \dots, -e_8^d, -e_8^u, \dots, -e_1\} \quad a_1 = \{e_2^u, e_7, e_{11}, -e_9, -e_6, -e_3, e_2^d\}$
- $L_1 = \{e_2^d, e_2^u, e_7, \dots, -e_3\}$
- An arc contains all the edges between 2 undergoing crossings

Next



Computing the δ -invariant of the singularity

From the Alexander polynomial, we derive the formulae for the δ -invariant:
(based on Milnor's research)

$C \subset \mathbb{C}^2$ complex curve, $z \in \text{Sing}(C)$



$\Delta(t_1, \dots, t_p) : r - \text{number of variables}, \mu - \text{degree}$

$r \geq 2$

$$\delta_z = \frac{1}{2}(\mu + r)$$

$r = 1$

$$\delta_z = \frac{1}{2}\mu$$

Summary

Using our library **QComplexInvariants** in Axel we get the results:

Equation	Link	Alex poly, δ -invariant, genus
$z^2 - w^2, \epsilon = 1.0$	Hopf link	$\Delta(t_1) = 1, \delta = 1, g = -1$
$z^2 - w^3, \epsilon = 1.0$	Trefoil knot	$\Delta(t_1) = t_1^2 - t_1 + 1, \delta = 1, g = 0$
$z^2 - w^4, \epsilon = 1.0$	2-knots link	$\Delta(t_1, t_2) = t_1 t_2 + 1, \delta = 2, g = 1$
$z^2 - w^5, \epsilon = 1.0$	1-knot link	$\Delta(t_1) = t_1^4 - t_1^3 + t_1^2 - t_1 + 1, \delta = 2, g = 4$
$(z - 2)^3 - (w - 1)^3, \epsilon = 1.0$	3-knots link	$\Delta(t_1, t_2, t_3) = -t_1 t_2 t_3 + 1, \delta = 3, g = -2$
$z^4 + z^2 w + w^5, \epsilon = 0.25$	3-knots link	$\Delta(t_1, t_2, t_3) = -t_1^2 t_2^2 t_3 + 1, \delta = 4, g = 2$

- ① Motivation
- ② Describing the problem
What?
- ③ Solving the problem
How?
- ④ Current results
- ⑤ Conclusion and future work

Conclusion

Present work:

- all the steps of the algorithm are now completely automatized;
- together with its main functionality to compute the genus,
- the symbolic-numeric algorithm provides also tools for computation:
 - in knot theory (i.e. diagram of links, Alexander polynomial);
 - in algebraic geometry (i.e. delta invariant, singularities of plane complex algebraic curve);

Conclusion

Future work:

- Analyze the algorithm for numeric input:
 - How to control the error in numerical computation?
 - How to improve the representation to our problem?
- Need to make investigations at the frontier between symbolic and numeric computation.



Thank you for your attention.
Questions?