The genus computation problem and approximate algebraic computation

Mădălina Hodorog[1]
Supervisor: Josef Schicho[1]
Joint work with Bernard Mourrain[2]

[1]Johann Radon Institute for Computational and Applied Mathematics,
Austrian Academy of Sciences,
Research Institute for Symbolic Computation
Johannes Kepler University Linz, Austria
[2]INRIA Sophia-Antipolis, France

March 11, 2010

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Table of contents

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

What?

- Input:
  - $F \in \mathbb{C}[x, y]$ squarefree with coefficients of limited accuracy
  - $C = \{(x, y) \in \mathbb{C}^2 | F(x, y) = 0\} =$
    $= \{(a, b, c, d) \in \mathbb{R}^4 | F(a + ib, c + id) = 0\}$ complex algebraic curve (m is the degree of C);
  - $\epsilon \in \mathbb{R}^*_+$ a non-zero positive real number, the input parameter.
- Output:
  - approximate $genus(C)$, s.t.

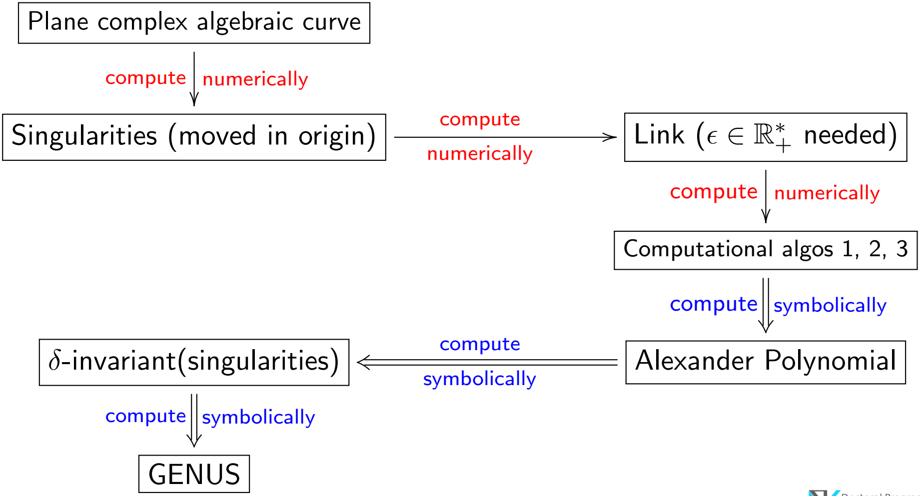
$$genus(C) = \frac{1}{2}(m-1)(m-2) - \sum_{P \in Sing(C)} \delta\text{-invariant}(P),$$

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

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

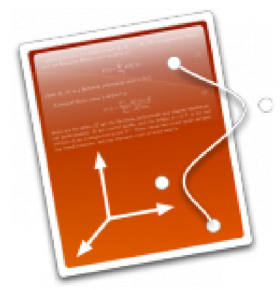
How?

- Strategy for computing the genus



How?

- For the implementation we use
  Axel algebraic geometric modeler [a]



[a] Acknowledgements: Julien Wintz

How?

- For the implementation we use *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

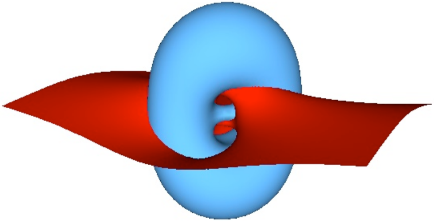
How?

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

[a] Acknowledgements: Julien Wintz

How?

- For the implementation we use *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;

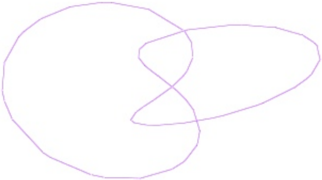


[a] Acknowledgements: Julien Wintz

Doctoral Program Computational Mathematics Numerical Analysis and Symbolic Computation

How?

- For the implementation we use *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

How?

- For the implementation we use *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:

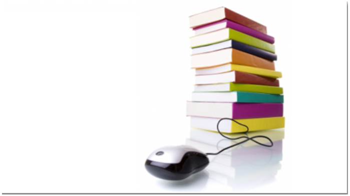
http://axel.inria.fr/

[a] Acknowledgements: Julien Wintz

Summary

At present:

- We have a symbolic-numeric algorithm, i.e. an approximate algorithm, for GENus cOMputation of plane Complex algebraiC Curves using Knot theory implemented in the GENOM3CK library.



- The algorithm is based on combinatorial techniques from knot theory, that allow us to analyze the singularities of the input curve and to compute the invariants: topology of singularities (algebraic link), Alexander polynomial, δ-invariant, genus. The algorithm depends on the parameter $\epsilon \in \mathbb{R}_+^*$.

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Summary

Next:

- The plane complex algebraic curves are defined by polynomials with coefficients of limited accuracy, i.e the coefficients

- For an arbitrary plane complex algebraic curve C defined by a polynomial with coefficients of limited accuracy, i.e $F(x, y) = x^3 - 1.865y^2 - y^3 + 0.0xy$, we want to compute the approximate $genus(C)$ using GENOM3CK.
  Important questions arise:

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Summary

Next:

- The plane complex algebraic curves are defined by polynomials with coefficients of limited accuracy, i.e the coefficients
  - are either exact data, i.e. integers or rational numbers: $1, -2, \frac{1}{2}$.
  - or inexact data, i.e. real numbers/floating point numbers: 1.865. For 1.865 we associate a tolerance of 10^{-3}, which means that the last digit is uncertain.
- For an arbitrary plane complex algebraic curve C defined by a polynomial with coefficients of limited accuracy, i.e $F(x, y) = x^3 - 1.865y^2 - y^3 + 0.0xy$, we want to compute the approximate $genus(C)$ using GENOM3CK.

  Important questions arise:

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Summary

Next:

- The plane complex algebraic curves are defined by polynomials with coefficients of limited accuracy, i.e the coefficients
  - are either exact data, i.e. integers or rational numbers: $1, -2, \frac{1}{2}$.
  - or inexact data, i.e. real numbers/floating point numbers: 1.865. For 1.865 we associate a tolerance of 10^{-3}, which means that the last digit is uncertain.
- For an arbitrary plane complex algebraic curve C defined by a polynomial with coefficients of limited accuracy, i.e $F(x, y) = x^3 - 1.865y^2 - y^3 + 0.0xy$, we want to compute the approximate $genus(C)$ using GENOM3CK.

  Important questions arise:
  - What does one mean by approximate genus?
  - How does one control the error in numerical computation?

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Preliminaries-Approximate algebraic computation

Objects of approximate algebraic computation[1]: polynomials with coefficients of limited accuracy, i.e. $F(x, y) = x^3 - 1.865y^2 - y^3 + 0.0xy$.

Basic questions

What happens when using approximate computation?

Why using approximate computation?

What is (one) of the aims of approximate computation?

[1]Thanks to the colleagues from the DK for their helpful discussions

Doctoral Program Computational Mathematics Numerical Analysis and Symbolic Computation

Preliminaries-Approximate algebraic computation

Objects of approximate algebraic computation[1]: polynomials with coefficients of limited accuracy, i.e. $F(x,y) = x^3 - 1.865y^2 - y^3 + 0.0xy$.

Basic questions

What happens when using approximate computation?

Why using approximate computation?

What is (one) of the aims of approximate computation?

Tiny perturbations in data input produce huge error in solution (ill-posed problems). We get failure of classical algorithms: Euclidean algorithm, root polynomial computation, genus computation, etc.

Definition (Hadamard). A problem is well posed if: it has a solution, the solution is unique, and the solution depends continuously on data and parameters.
Remark. If the solution of the problem depends in a discontinuous way on the data, then small errors can create large deviations, and the problem is called ill-posed.

[1]Thanks to the colleagues from the DK for their helpful discussions

Preliminaries-Approximate algebraic computation

Objects of approximate algebraic computation[1]: polynomials with coefficients of limited accuracy, i.e. $F(x,y) = x^3 - 1.865y^2 - y^3 + 0.0xy$.

Basic questions

What happens when using approximate computation?

Why using approximate computation?

What is (one) of the aims of approximate computation?

There is no other choice since the input data are only approximately known, because for example the coefficients of the polynomials come from experimental data.

[1] Thanks to the colleagues from the DK for their helpful discussions

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Preliminaries-Approximate algebraic computation

Objects of approximate algebraic computation[1]: polynomials with coefficients of limited accuracy, i.e. $F(x,y) = x^3 - 1.865y^2 - y^3 + 0.0xy$.

Basic questions
What happens when using approximate computation?
Why using approximate computation?
What is (one) of the aims of approximate computation?

To deal with ill-posed problems in numerical computation!

What should a numerical algorithm really do?

⇒ Naive answer: Compute solutions.

⇒ Z. Zeng, E. Kaltofen, H. Stetter: A numerical algorithm generates the exact solution of a nearby problem (related with regularization theory).

[1] Thanks to the colleagues from the DK for their helpful discussions

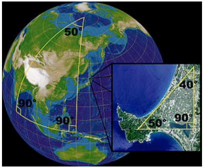
Genus computation - Approximate algebraic computation

Approximate algebraic computation to an ill-posed problem

- is based on **W. Kahan's** discovery: problems with certain solution structure form a "pejorative" manifold. The solution is lost when the problem leaves the manifold, but it is preserved when the problem stays on the manifold.

What is a manifold and its dimension?

- A manifold M is a topological space that is locally euclidean, i.e. around every point, there is a neighborhood that is topologically the same as the open unit ball in $\mathbb{R}^n$. n is the dimension of M. (any object that can be "charted" is a manifold.)



Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Genus computation - Approximate algebraic computation

Approximate algebraic computation to an ill-posed problem

- We partition the data input of the problem into pejorative manifolds. For given input we need to determine the nearby pejorative manifold of the highest codimension (i.e the smallest nearby pejorative manifold).

What does "nearby" means?

- "Nearby" depends on the input parameter ϵ.
- It is not precise what "nearby" means.

GENOM3CK and approximate algebraic computation

We consider the exact algorithm for genus computation as the function:

$$E : \mathbb{C}[x, y] \to \mathbb{Z}, F(x, y) \mapsto E(F(x, y)).$$

We consider the approximate algorithm from GENOM3CK for genus computation as the function:

$$A_\epsilon : \mathbb{C}[x, y] \times \mathbb{R}^*_+ \to \mathbb{Z}, F(x, y) \mapsto A_\epsilon(F(x, y)).$$

Remark: The output of A_ϵ: the Alexander polynomial (Δ), the δ-invariant (δ), and the genus (g).

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

GENOM3CK and approximate algebraic computation

Tests experiments performed with GENOM3CK indicate two important properties of A_ϵ:

Convergency

- we consider $F(x,y)$ with both exact and inexact coefficients; we compute $A_\epsilon(F(x,y))$ for different values of the parameter ϵ.
- for $x^3 - xy + y^2$, we know that the exact topology is the Hopf link;
- we notice that the approximate solution computed with A_ϵ converges to the exact solution as ϵ tends to 0: $\underset{F(x,y)}{\forall} \underset{\epsilon \to 0}{lim} A_\epsilon(F(x,y)) = E(F(x,y))$.

Equation and ϵ		Link	Alexander, δ invariants, genus		
$-x^3 - xy + y^2$	1.00	Trefoil knot	$\Delta(t_1) = t_1^2 - t_1 + 1$	$\delta = 1$	$g = 0$
$-x^3 - xy + y^2$	0.5	Trefoil knot	$\Delta(t_1) = t_1^2 - t_1 + 1$	$\delta = 1$	$g = 0$
$-x^3 - xy + y^2$	0.25	Hopf link	$\Delta(t_1, t_2) = 1$	$\delta = 1$	$g = 0$
$-x^3 - xy + y^2$	0.14	Hopf link	$\Delta(t_1, t_2) = 1$	$\delta = 1$	$g = 0$

GENOM3CK and approximate algebraic computation

Tests experiments performed with GENOM3CK indicate two properties of A_ϵ:
Continuity

- we consider $p(x, y)$ a polynomial with exact coefficients;
- for $\delta \in \mathbb{R}$ we consider $p_\delta(x, y)$ perturbations of p;
- perturbations of type I: $p_\delta(x, y) = p(x, y) + \delta$, where $\delta \in \mathbb{R}^*$.
- perturbations of type II: $p_\delta(x, y) = p(x, y) + \delta q(x, y)$, where $\delta \in \mathbb{R}^*$, $q(x, y) \in \mathbb{C}[x, y]$ is an arbitrary exact polynomial.
- we consider $F(x, y) := p_\sigma(x, y)$, and several values for ϵ. For each ϵ, we compute $A_\epsilon(F(x, y))$ for different values of δ.
- we observe that small changes on the input data produce small changes on the output solution:

$$\forall_{F(x,y)} \ \exists_{\eta>0} \text{ such that } \forall_{\epsilon<\eta} \ \exists_{\eta_1>0} \ \forall_{G(x,y)} \ G(x, y) \in I := (F(x, y) - \eta_1, F(x, y) + \eta_1)$$

$$A_\epsilon(G(x, y)) \text{ is constant in } I.$$

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

GENOM3CK and approximate algebraic computation

Continuity (next) small changes in the input produce small changes in the output:

Perturbations I and ϵ		$\sigma = 10^{-e}, e \in \mathbb{N}^*$	Link	Invariants
$-x^3 - xy + y^2 - 10^{-e}$	0.5	$\{10^{-2}, ..., 10^{-10}\}$	Trefoil knot	$\Delta(t_1) = t_1^2 - t_1 + 1$ $\delta = 1$ $g = 0$
$-x^3 - xy + y^2 - 10^{-e}$	0.25	$\{10^{-2}, ..., 10^{-10}\}$	Hopf link	$\Delta(t_1, t_2) = 1$ $\delta = 1$ $g = 0$

$p(x, y) = -x^3 - xy + y^2$ $q(x, y) = -x^3 - 2xy + y^2$;
$F(x, y) := p_\delta(x, y) = p(x, y) + \delta q(x, y) = -(1+10^{-e})x^3 - (1+2\cdot 10^{-e})xy + (1+10^{-e})y^2$
$\delta = 0.1 : F(x, y) = -1.1x^3 - 1.2x^2 + 1.1y^2$
$\delta = 0.01 : F(x, y) = -1.01x^3 - 1.02x^2 + 1.01y^2$, etc .

Perturbations II and ϵ		$\sigma = 10^{-e}, e \in \mathbb{N}^*$	Link	Invariants
$-(1+10^{-e})x^3 - (1+2\cdot 10^{-e})xy + (1+10^{-e})y^2$	0.15	$\{10^{-1}, ..., 10^{-10}\}$	Hopf link	$\Delta(t_1, t_2) = 1$ $\delta = 1$ $g = 0$
$-(1+10^{-e})x^3 - (1+2\cdot 10^{-e})xy + (1+10^{-e})y^2$	0.14	$\{10^{-1}, ..., 10^{-10}\}$	Hopf link	$\Delta(t_1, t_2) = 1$ $\delta = 1$ $g = 0$

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation

Conclusion and future work

Achieved goals:

- complete automatization for the steps of the approximate algorithm (in the library GENOM3CK);
- tests experiments show that the approximate algorithm has the continuity and convergency properties;

TO DO's:

Conclusion and future work

Achieved goals:

- complete automatization for the steps of the approximate algorithm (in the library GENOM3CK);
- tests experiments show that the approximate algorithm has the continuity and convergency properties;

TO DO's:

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

Conclusion and future work

Achieved goals:

- complete automatization for the steps of the approximate algorithm (in the library GENOM3CK);
- tests experiments show that the approximate algorithm has the continuity and convergency properties;
- the approximate algorithm computes discrete information from continuous data and it can be described using principles from regularization theory and approximate algebraic computation.

TO DO's:

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

Conclusion and future work

Achieved goals:

- complete automatization for the steps of the approximate algorithm (in the library GENOM3CK);
- tests experiments show that the approximate algorithm has the continuity and convergency properties;
- the approximate algorithm computes discrete information from continuous data and it can be described using principles from regularization theory and approximate algebraic computation.

TO DO's:

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

Thank you for your attention.
Questions?

Doctoral Program
Computational Mathematics
Numerical Analysis and Symbolic Computation