ECG

IST-2000-26473

Effective Computational Geometry for Curves and Surfaces

ECG Technical Report No. : *ECG-TR-122104-01*

Prototype implementation for the computation of intersection points and critical points on planar curves

Philippe Trebuchet

Deliverable:	12 21 04
Site:	Sophia Antipolis
Month:	12

Project funded by the European Community
under the "Information Society Technologies"
Programme (1998–2002)

1 Introduction

The aim of this program is to find the intersection points or the critical points of curves or surfaces. To reach this goal, we use an algorithm that based on an algrebraic approach, or more precisely on rewriting techniques, in order to be able to work in the quotient algebra $\mathcal{A} = \mathbf{K}[x_1 \ldots x_n]/I$ where I is the ideal generated by the equations we have to solve. From the knowledge of $\mathcal{A}$, we deduce the roots of our system.

2 Computing in the quotient algebra

To be able to work in the quotient algebra, one well known method is to compute a Grœbner basis. However one of the main problems with Grœbner bases is that prior to any computation and independently of any geometric proprety of the system of equations considered, we have to choose a monomial ordering, which induces artificial numerical instability. See [7] for an example of this phenomenon. In the new method that we propose, we choice function refining the degree. See [6] or [8] for a description of the whole algorithm.

3 Finding the points from the quotient structure.

In the previous section we have seen how to represent effectively the quotient algebra $\mathbf{K}[x_1 \ldots x_n]/I$, but this is only the first step in the resolution process. We have now to explicitly find the roots of the given 0-dimensional system. We will use the following theorem to recover the root from the multiplicative structure of $\mathcal{A} = \mathbf{K}[x_1 \ldots x_n]/I$ [2], [4], [5]:

Theorem 1 *Let a be a polynomial, we define M_a by:*

$$\begin{array}{rcl} M_a : \mathcal{A} & \rightarrow & \mathcal{A} \\ b & \mapsto & a\, b \end{array} \qquad \begin{array}{rcl} M_a^t : \widehat{\mathcal{A}} & \rightarrow & \widehat{\mathcal{A}} \\ \Lambda & \mapsto & a \cdot \Lambda = \Lambda \circ M_a. \end{array}$$

Denoting by $\zeta_1 \ldots \zeta_n$ the roots we are looking for, and by $\mathbb{1}_\zeta$ the linear form such that $\mathbb{1}_\zeta(p) = p(\zeta), p \in \mathbf{K}[x_1 \ldots x_n]$, we have then

- *The eigenvalues of the linear operator M_a (resp. M_a^t) are $\{a(\zeta_1), \ldots, a(\zeta_d)\}$.*
- *The common eigenvectors of $(M_a^t)_{a \in \mathcal{A}}$ are (up to a scalar) $\mathbb{1}_{\zeta_1}, \ldots, \mathbb{1}_{\zeta_d}$.*

This result reduces the numerical solving step to linear algebra computations that are numerically well understood.

For the eigenvalue and eigenvector computation, our implementation uses, the external library LAPACK [1], when the coefficients of type `float` or `double`.

4 Using the program

The implementation we propose of the previous algorithms is built using the environment SYNAPS, a framework for symbolic and numeric computations developped in the GALAAD project.

The implementation computes points of any 0-dimensional ideal. Of course, it is suitable to compute intersection points or critical points of curves of surfaces.

The statically linked executables are aviables on: `http://www-sop.inria.fr/galaad/collab/ecg`.

To use these programs, you just have to provide, in the command line, the following arguments in this order:

- First argument is the name of the file that contains the polynomials in the SYNAPS format [3].
- *optional* the name of the file where the results will be printed if no argument is given the results will be printed on the standart output.

Example 1 *let us consider the following system of equations(file execg):*

```
x0^2+x1^2-1;
2*x0^2+5*x1^2-10;
```

the following session computes the solution of this system using the macaulay like strategy and using double for the arithmetic.

```
solve_mac_double execg solexecg
```

The equations when reevaluated at the points found by our implementation are bounded in norm by 10^{-14} *which is as good as we can expect using the double precison arithmetic.*

Several executables with different strategies and precision are aviable. More generally the naming convention is as follows:

- The first token is the action the program performs. It can be one of the following
  - info, just computes the modular structure of the ideal and counts the number of points.
  - matrix, compute the multiplication matrices by all tthe variables.
  - solve, computes the multiplication matrices and perform a an eigenvector computation using `Lapack`.
- The second token is the strategy used to select the leading monomials. It can be one of the following:
  - mac, is a strategy that privilegies the monomials of high partial degree against the others.
  - naif, is a strategy that privilegies the monomials with high coefficients.
  - size, is a strategy that privilegies the monomials whose coefficients are of small memory size.
  - mix, is a strategy that chooses randomly a monomials either using the degre reverse lexicographical order either using the mac startegy.
- Finally, the last token is the arithmetic used. It can be one of the following:
  - fxxx, extended precision arithmetic using at least xxx bits.
  - rat, rational numbers.
  - double, long double, biult in C++ types.
  - mod modular arithmetic.

References

[1] E. Anderson, Z. Bai, C. Bischof, J.W. Demmel, J.J. Dongara, J. Du Croz, A. Greenbaum, S. Hammerling, A. McKenney, and D. Sorensen. LAPACK : A portable linear algebra library for high-performance computers. Technical Report CS-90-105, Comp. Sci. Dept, Univ. of Tennessee, Knoxville, (USA), 1990.

[2] W. Auzinger and H. J. Stetter. An elimination algorithm for the computation of all zeros of a system of multivariate polynomial equations. In *Proc. Intern. Conf. on Numerical Math.*, volume 86 of *Int. Series of Numerical Math*, pages 12–30. Birkhäuser Verlag, 1988.

[3] G. Dos Reis, B. Mourrain, and Ph. Trébuchet. An environment for symbolic and numeric computation. *submitted*, 2002.

[4] H. M. Möller. Systems of algebraic equations solved by means of endomorphisms. *Lect. Notes in Comp. Sci.*, 673:43–56, 1993.

[5] B. Mourrain. Computing isolated polynomial roots by matrix methods. *J. of Symbolic Computation, Special Issue on Symbolic-Numeric Algebra for Polynomials*, 26(6):715–738, Dec. 1998.

[6] B. Mourrain. A new criterion for normal form algorithms. In M. Fossorier, H. Imai, Shu Lin, and A. Poli, editors, *Proc. AAECC*, volume 1719 of *LNCS*, pages 430–443. Springer, Berlin, 1999.

[7] B. Mourrain and Ph. Trébuchet. Solving projective complete intersection faster. In C. Traverso, editor, *Proc. Intern. Symp. on Symbolic and Algebraic Computation*, pages 231–238. New-York, ACM Press., 2000.

[8] B. Mourrain and Ph. Trébuchet. Algebraic methods for numerical solving. In *Proc. of the 3rd International Workshop on Symbolic and Numeric Algorithms for Scientific Computing'01 (Timisoara, Romania)*, 2002.