New bivariate system solver and topology of algebraic curves

Yacine Bouzidi, Sylvain Lazard, Marc Pouget, Fabrice Rouillier

New bivariate system solver and topology of algebraic curves

Yacine Bouzidi* Sylvain Lazard* Marc Pouget* Fabrice Rouillier†

Abstract

We present a new approach for solving polynomial systems of two bivariate polynomials with rational coefficients. We first use González-Vega and Necula approach [3] based on sub-resultant sequences for decomposing a system into subsystems according to the number of roots (counted with multiplicities) in vertical lines. We then show how the resulting triangular subsystems can be efficiently solved by computing lexicographic Gröbner basis and Rational Univariate Representations (RURs) of these systems. We also show how this approach can be performed using modular arithmetic, while remaining deterministic.

Finally we apply our solver to the problem of computing the topology of algebraic curves using the algorithm Isotop [2]. We show that our approach yields a substantial gain of a factor between 1 to 10 on curves of degree up to 28 compared to directly computing a Gröbner basis and RUR of the input system, and how it leads to a very competitive algorithm compared to the other state-of-the-art implementations.

1 Bivariate system solving algorithm

We present in this section our new bivariate solver and discuss in Section 2 its application to the computation of the topology of plane algebraic curves.

The input is a system $S = \{P, Q\}$ where P and Q are two bivariate polynomials in $\mathbb{Q}[x, y]$. We solve the system, that is the output is a set of boxes isolating the solutions of S, i.e., pairwise-disjoint axis-parallel boxes in the xy-plane, each containing exactly one root.

Our algorithm proceeds in two steps. First we decompose the system S into a set of triangular systems according to the sub-resultant sequence of P and Q (Section 1.1). This decomposition is essentially identical to that of González-Vega and Necula [3]; for completeness, we describe this decomposition in Section 1.2 together with our extension for the treatment of solutions for which the leading coefficient of P or Q (seen as a polynomial in y) vanishes. In the second step, we compute a Gröbner basis and a Rational Univariate Representation (RUR) [4] of every system of the decomposition (Sections 1.3). We show how the specific form of the input triangular systems permits to trivially compute a Gröbner basis, and yields improvements to the algorithm of [4] for computing a RUR. The RUR is then used in a standard way to solve these systems.

We show in Section 1.4 how these two steps can be performed with modular arithmetic in a deterministic way, yielding a very efficient algorithm.

It should be stressed that we do not compute the multiplicity of the roots in the input system. However, for every root (α, β) of a system $\{F, \frac{\partial F}{\partial y}\}$, we obtain the multiplicity of β as a root of $F(\alpha, y)$ (Section 2). These multiplicities are needed in our algorithm for computing the topology of plane curves, and this is a key feature of our solving algorithm.

1.1 Preliminaries : sub-resultant sequences

Recall that P and Q are two polynomials in $\mathbb{Q}[x, y]$. Considering P and Q as polynomials in y, let $P = \sum_{i=0}^{p} a_i y^i$, and $Q = \sum_{i=0}^{q} b_i y^i$ with a_i, b_i in $\mathbb{Q}[x]$. Assume without loss of generality that $p \leqslant q$.

The Sylvester matrix of P and Q is the $(p+q) \times (p+q)$ matrix where the k-th row consists, for $1 \leqslant k \leqslant q$, of $k-1$ zeros, followed by the coefficients of P, $a_p, \ldots, a_0$, and completed by $q-k$ zeros, and, for $q+1 \leqslant k \leqslant p+q$, of $k-q-1$ zeros, followed by the coefficients of Q, $b_q, \ldots, b_0$, and completed by $p+q-k$ zeros. One denotes by $Sylv_i$, $i \leqslant q$, the $(p+q-2i) \times (p+q-i)$ matrix obtained from the Sylvester matrix of P and Q by deleting the i last rows of the coefficients of P, the i last rows of the coefficients of Q, and the i last columns.

Definition 1 ([1]) *The i-th polynomial* sub-resultant *of P and Q, denoted by $Sres_i(P, Q)$, is the polynomial*

$$\det(M_{p+q-2i})y^i + \det(M_{p+q-2i+1})y^{i-1} + \cdots + \det(M_{p+q-i})$$

where M_j is the square sub-matrix of $Sylv_i$ of size $p+q-2i$ and consisting of the $p+q-2i-1$ first columns and the j-th column of $Sylv_i$, $j \in \{p+q-2i, \ldots, p+q-i\}$. The sub-resultant sequence *of P and Q is the sequence $Sres_i(P, Q)$ for i from 0 to q.*

$Sres_i(P, Q)$ is a polynomial of degree at most i in y, and the coefficient of the monomial of degree i, denoted by $sres_i(P, Q)$, is called the i-th *principal sub-resultant coefficient*. Note that $Sres_0(P, Q) =$

*INRIA Nancy Grand Est, LORIA laboratory, Nancy, France. Firstname.Name@loria.fr

†INRIA Paris-Rocquencourt and LIP6 (Université Paris 6, CNRS), Paris, France. Firstname.Name@lip6.fr

$sres_0(P,Q)$ is the *resultant* of P and Q. We thus have

$$Sres_i(P,Q) = sres_i(P,Q)y^i + R_i(x,y) \qquad (1)$$

where the degree in y of R_i is at most $i-1$, and its degree in x is at most the product of the total degrees of P and Q (the degree in x is generically maximal for $i=0$ which is that of the resultant).

The polynomial sub-resultants of P and Q are equal to either 0 or to (up to a constant) polynomials in the remainder sequence of P and Q [1, p.308]. For efficiency, the computations of sub-resultant sequences are usually performed by computing polynomial remainder sequences using some variants of Euclid algorithm.

1.2 Triangular decomposition

First, we project the solutions of S on the x-axis; algebraically, this amounts to computing the resultant of P and Q with respect to the variable y. The roots of that resultant, denoted $Res_y(P,Q) \in \mathbb{Q}[x]$, are the x-coordinates of the solutions of S, and the common roots of their leading coefficients a_p and $b_q \in \mathbb{Q}[x]$. Hence, the resultant of P and Q provides the x-coordinates of the points where the corresponding curves intersect or have a common vertical asymptote. For each such x-coordinate, the difficulty is to compute the y-coordinates of the corresponding intersection points. This can be done with the sub-resultant sequence based on the following fundamental proposition.

Proposition 1 ([3]) *Let $P,Q \in \mathbb{Q}[x,y]$ be two square-free polynomials and consider their sub-resultant sequence with respect to the variable y. Let $\alpha \in \mathbb{R}$ be a root of $Res_y(P,Q)$ such that $a_p(\alpha) \neq 0$ and $b_q(\alpha) \neq 0$. Then*

$$\begin{cases} sres_0(\alpha) = 0, \cdots, sres_{k-1}(\alpha) = 0 \\ sres_k(\alpha) \neq 0 \end{cases}$$
$$\Leftrightarrow Gcd(P(\alpha,y),Q(\alpha,y)) = Sres_k(\alpha,y).$$

We decompose the input system S into a set of triangular systems with respect to the degree of $Gcd(P(\alpha,y),Q(\alpha,y))$ according to Prop. 1. Throughout the decomposition we consider the square-free part of the resultant of P and Q, $Res_y(P,Q) = Sres_0(P,Q)$, and we denote it by $sqrfree(Sres_0(x))$. Taking the square-free part simplifies the computation but it does not preserve the multiplicities of the roots. However, this is actually critical in our algorithm for computing the multiplicities in the fibers needed for the topology computation, as mentioned earlier (Section 2).

According to Prop. 1, we first consider only x-coordinates on which a_p and b_q do not vanish. Geometrically, the roots of a_p correspond to vertical asymptotes of the curve $\mathcal{C}_P$ defined by $P=0$; similarly for b_q and the curve $\mathcal{C}_Q$. We denote by $Fres(x)$ the polynomial $sqrfree(Sres_0(x))$ "without these roots":

$$Fres(x) = \frac{sqrfree(Sres_0(x))}{Gcd(sqrfree(Sres_0(x)), a_p(x)b_q(x))}. \qquad (2)$$

Solutions outside the vertical asymptotes. The solutions of S that do not lie on a vertical asymptote of $\mathcal{C}_P$ or $\mathcal{C}_Q$ have their x-coordinates solutions of $Fres(x)$. The idea is to factorize $Fres(x)$ with respect to the degree of the $Gcd(P(\alpha,y),Q(\alpha,y))$ for α root of $Fres(x)$.

Let $G_1(x) = Gcd(Fres(x), sres_1(x))$. We split $Fres(x)$ in two factors G_1 and $F_1(x) = \frac{Fres(x)}{G_1(x)}$, and we consider the system $S_1 = \begin{cases} F_1(x) \\ Sres_1(x,y) = sres_1(x)y + R(x) \end{cases}$. The roots of S_1 are exactly the roots (α,β) of S such that the degree of the $Gcd(P(\alpha,y),Q(\alpha,y))$ is 1. In other words, α is a root of $Fres(x)$ such that S admits a unique root (α,y) counted with multiplicity.

According to Prop. 1, we split again G_1 with respect, this time, to the polynomial $sres_2(x)$. Let $G_2 = Gcd(G_1, sres_2)$, $F_2 = G_1/G_2$, the system $S_2 = \{F_2(x), Sres_2(x,y)\}$ encodes the solutions (α,β) of S such that the degree of the $Gcd(P(\alpha,y),Q(\alpha,y))$ is 2. In other words, α is a root of $Fres(x)$ such that S admits two roots (α,y) counted with multiplicity.

We go ahead recursively until we decompose completely $Fres(x)$ with respect to the polynomials $sres_1(x), \ldots, sres_q(x)$. At the k-th step the roots of $F_k(x)$ are exactly the roots of $Fres(x)$ verifying $sres_1(x) = 0, \ldots, sres_{k-1}(x) = 0$, and $sres_k(x) \neq 0$.

The result is a set of triangular systems $S_k = \{F_k(x), Sres_k(x,y)\}$ such that $Gcd(F_k(x), sres_k(x)) = 1$.

Solutions on vertical asymptotes. We now consider the solutions of S on an asymptote $x=\alpha$, with α a root of a_p or b_q. Our algorithm differs here from that of [3] in which they detect when such a situation occurs and shear the coordinate system. For sake of brevity, we only detail the case of solutions on a common asymptote, that is $x=\alpha$ with α a root of $D(x)$ the square-free part of $Gcd(a_p,b_q)$. Let P_1 , Q_1 be defined as follows:

$P_1 = \sum_{i=1}^{p}(a_i mod D(x))y^i$, $Q_1 = \sum_{i=1}^{q}(b_i mod D(x))y^i$, where $U(x) \mod D(x)$ is the remainder of the Euclidean division of $U(x)$ by $D(x)$ in $\mathbb{Q}[x]$. It is clear that P_1, Q_1 coincide with P and Q above the roots of $D(x)$, i.e., $\forall \alpha \in \mathbb{R} : D(\alpha) = 0 \Longrightarrow P_1(\alpha,y) = P(\alpha,y)$ and $Q_1(\alpha,y) = Q(\alpha,y)$. We decompose the system $\{P_1,Q_1\}$ as described above but keeping in the resultant only the roots of $D(x)$: $Fres(x) = sqrfree(Gcd(Sres_0(P_1,Q_1), D(x)))$. We thus recover in the resulting systems, solutions located on the common asymptotes.

Output. The output is a set of triangular systems $S_k = \begin{cases} F_k(x) = 0 \\ Sres_k(x,y) = 0 \end{cases}$ such that $Gcd(F_k(x), sres_k(x)) = 1$ and all F_k are square-free. For every k there might be several systems S_k: depending on whether the solution of S_k lies on a vertical asymptote or not, $Sres_k$ denotes the k-th subresultant of P and Q, or of some reductions of P and Q modulo some factors of their leading coefficients a_p and b_q in $\mathbb{Q}[x]$.

1.3 Groebner basis and RUR computation

A specific property of the decomposition obtained above is that $Gcd(F_k(x), sres_k(x)) = 1$. By Bézout identity, there exists $U, V \in \mathbb{Q}[x]$ such that $UF_k + Vsres_k = 1$. Thus, since $Sres_k(x,y) = sres_k(P,Q)y^k + R_k(x,y)$ (Eq. (1)), the system S_k is equivalent to $\tilde{S}_k : \begin{cases} F_k(x) = 0 \\ y^k + \tilde{R}_k(x,y) = 0 \end{cases}$ where $\tilde{R}_k$ is the reduction of VR_k modulo F_k (that is each coefficient is reduced modulo $F_k(x)$); hence $\tilde{R}_k$ has degree in y strictly less than k and strictly less than that of F_k in x. The resulting system thus is a (reduced) lexicographic Gröbner basis [1].

At this stage, we can use a standard RUR algorithm for solving each of these systems [4]. Recall that given a bivariate system, a RUR defines a parameterization of its solutions with four polynomials $f, g_x, g_y, g \in \mathbb{Q}[t]$ such that the solutions of the system are $(\frac{g_x(t_i)}{g(t_i)}, \frac{g_y(t_i)}{g(t_i)})$ for the roots t_i of f. It should be stressed that a RUR preserves the multiplicity of the roots of the system, that is, the multiplicity of a root t_i of f is the multiplicity of the root $(\frac{g_x(t_i)}{g(t_i)}, \frac{g_y(t_i)}{g(t_i)})$ in $\tilde{S}_k$, and thus in S_k. The fact that F_k is square-free yields a geometric interpretation of the multiplicities is the system S_k that will be exploited in our application on the topology of curves.

The particular structure of the systems we obtain yields improvements to the algorithm of Rouillier [4] for computing a RUR. The critical properties of our input systems S_k is that (i) they are Gröbner basis *for the lexicographic order*, which implies that one of the polynomials is univariate, and (ii) there are only two polynomials in the basis. Most of the RUR computations are performed using linear algebra in the quotient algebra $\mathbb{Q}[x,y]/S_k$ (where S_k denote here the ideal associated to the system). Property (ii) implies that a basis of this algebra is simply $\{x^i y^j\}$, for $0 \leqslant i < \text{degree}(F_k)$ and $0 \leqslant j < k$. One important step of the algorithm is to compute the product of all pairs of elements of that basis, reduced modulo the system S_k; this step is substantially simplified by the structure of the basis and by using Property (i). The complexity of this step, and actually of the whole RUR computation, is this way reduced from $O(D^3)$ to $O(D^2)$ where D is the size of the algebra basis.

1.4 Deterministic modular method

The bitsize of the coefficients in the RUR is reasonable in the sense that these coefficients have more or less the same size as those in the resultant [2, §4.2]. However, the size of the coefficients of the lexicographic Gröbner basis in the intermediate computations may be quite large. A standard approach for avoiding such intermediate growth is to use modular computations and the Chinese Remainder Theorem [1]. The difficulty being the design of a deterministic algorithm (rather than a Monte-Carlo algorithm). We show how this approach can be applied here.

Without loss of generality, we can assume that the two input polynomials P and Q have integer coefficients (rather than rational). For brevity, we only describe how to compute the roots outside the vertical asymptotes; the roots on the vertical asymptotes are treated similarly. Let μ be a prime number larger than pq, $\mathbb{Z}_\mu = \frac{\mathbb{Z}}{\mu\mathbb{Z}}$, and let ϕ_μ be the canonical surjection $\mathbb{Z}[x,y] \to \mathbb{Z}_\mu[x,y]$ that transforms the polynomial coefficients modulo μ. We first check that (i) $\phi_\mu(a_p) \neq 0$ and $\phi_\mu(b_q) \neq 0$, then we compute $Fres(P,Q)$ and $Fres(\phi_\mu(P), \phi_\mu(Q))$ (Eq. (2)) and test whether (ii) $\phi_\mu(Fres(P,Q)) = Fres(\phi_\mu(P), \phi_\mu(Q))$ and (iii) $degree(\phi_\mu(Fres(P,Q))) = degree(Fres(P,Q))$. We consider prime numbers in turn until all these properties are satisfied. Then, considering polynomials in $\mathbb{Z}_\mu[x,y]$, we solve the system $\{\phi_\mu(P), \phi_\mu(Q)\}$ as described in Sections 1.2 and 1.3.

Lemma 2 *The sum of the degrees of the systems (the degree of a system being the number of its complex roots counted with multiplicity) in the decomposition applied to $\phi_\mu(P)$ and $\phi_\mu(Q)$ and performed in $\mathbb{Z}_\mu[x,y]$ is greater than or equal to the sum of the degrees of the systems in the decomposition of P and Q performed over the rationals.*

An important remark is that since the RUR preserves multiplicities, the degree of a system is the degree of the first polynomial of its RUR, even when considered over $\mathbb{Z}_\mu[x,y]$, as soon as $\mu > pq$ [4]. If the sum of the degrees of the systems obtained by a computation modulo μ is strictly greater than the sum of the degrees of the systems obtained by the computation over the rationals, we say that μ is an *unlucky prime*. The difficulty is of course to detect unlucky primes.

Our algorithm selects a set of primes μ (as described above), decomposes the system $\{\phi_\mu(P), \phi_\mu(Q)\}$, and determines a RUR for each S_k over $\mathbb{Z}_\mu[x,y]$. If the sums of the degrees of S_k are not the same for all considered primes, the primes for which this sum is not minimal are unlucky, and we discard these primes. Otherwise, either all primes are lucky or they are all unlucky. We then use the Chinese Remainder Theorem to lift the RURs of the systems S_k.

We then check whether the roots of the set of resulting RURs (f, g_x, g_y, g) of each S_k are indeed roots of the system $\{P, Q\}$ with the correct multiplicities. Checking whether the roots are correct is done by substituting the rational coordinates $x = \frac{g_x(t)}{g(t)}$ and $y = \frac{g_y(t)}{g(t)}$ in P and Q and checking that f divides its numerator. A root of multiplicity m corresponding to a root t of f can be verified similarly by substituting the RUR coordinates in the i-th derivatives of P and Q with respect to y (for i from 1 to $m-1$), and checking that t is a root of their gcd with f (more precisely, let $f = \Pi_i f_i^i$ be the decomposition with respect to multiplicities of the first poly of the RUR then t is a root of f_m, and one must check that f_m divides the numerators of the derivatives of P and Q after substitution).

Similarly, the multiplicity m of a root (α, β) of the system (corresponding to a root t of f) can be checked to be correct by substituting the RUR coordinates in the i-th derivatives of P and Q with respect to y (for i from 1 to $m-1$), and checking that t is a root of their gcd with f. Checking that t is not a root of the m-th derivative is unnecessary because m is necessarily larger than or equal to the correct multiplicity of the root (by the proof of Lemma 2).

If, for all S_k, all the roots of the RURs are indeed roots of $\{P, Q\}$, and if their multiplicities are correct, the set of roots (with multiplicities) is necessarily correct by Lemma 2. Otherwise we add some new primes and lift again the result. (We omit here some details, including that a prime may also be unlucky for the computation of the RURs.)

2 Application to the topology of plane curves

We now consider the problem of computing efficiently the topology of a real plane algebraic curve $\mathcal{C}_f$ defined by a bivariate polynomial f in $\mathbb{Q}[x, y]$. Isotop [2] is an algorithm that uses Gröbner basis and RUR computation as a black box for isolating the critical points of $\mathcal{C}_f$, that is the points that are solutions of $\{f, f_y\}$ with $f_y = \frac{\partial f}{\partial y}$. Compared to other algorithms, Isotop is particularly efficient on non-generic curves (i.e., curves with several critical points on some vertical line) because they are treated in the original coordinate system without shearing. However, Isotop is less efficient on some types of *generic* curves, in particular for (i) "random" curves for which the multiplicity of all critical points in their fiber is 2, and (ii) generic curves such that the multiplicity (in the fiber) of some critical points is high. In case (i), the Gröbner basis computation is expensive compared to the sub-resultant sequence decomposition approach of [3] because the decomposition leads to a unique system S_1. In case (ii), the RUR computation (performed as a black box on a Gröbner basis of $\{f, f_y\}$) turns out to also be expensive compared to the decomposition approach because multiplicities in the initial system are kept (refer to [2] for details).

Our contribution is to change the Gröbner basis and RUR black box by the solver presented in section 1. The idea being to decompose the input system when possible while keeping RUR approach for solving these systems. This yields two main changes to the Isotop algorithm: the way to compute critical points, and to determine multiplicities in fibers.

Computing critical points of the curve $\mathcal{C}_f$. Critical points of $\mathcal{C}_f$ are the solutions of the system $S = \{f, f_y\}$. We isolate these solutions by decomposing S in triangular systems S_i and computing their RURs according to the algorithm presented in Sect. 1.

Computing the multiplicities of critical points in their fibers. For a point $\mathbf{p} = (\alpha, \beta)$ on the curve $\mathcal{C}_f$, the **multiplicity of p in its fiber** denoted by $mult(f(\alpha, y), \beta)$ is defined as the multiplicity of β in the univariate polynomial $f(\alpha, y)$. Solving the systems S_i with RURs enables to preserve the multiplicities and, hence, to compute the multiplicities in the fibers even if there are several critical points with the same x-coordinate:

Proposition 3 *If $\mathbf{p}$ is a critical point of $\mathcal{C}_f$ then it is a solution of some system S_i, and its multiplicity in its fiber is its multiplicity is S_i plus one.*

Preliminary experiments. We have computed the ratio of running times of the original Isotop algorithm with its new version, Isotop2, using our solver as explained in Section 2. As expected the ratio is large for generic curves (a ratio between 5 and 14 on curves of degree between 10 and 20) and we also have a significant gain for non-generic curves (a ratio between 1.5 and 15 for degrees between 12 and 28). These preliminary tests hence confirm the theoretical analysis that our approach automatically selects the best strategy in any case.

References

[1] S. Basu, R. Pollack, and M.-R. Roy. *Algorithms in Real Algebraic Geometry*, Springer-Verlag, 2nd edition, 2006.

[2] J. Cheng, S. Lazard, L. Peñaranda, M. Pouget, F. Rouillier, and E. Tsigaridas. On the topology of real algebraic plane curves. *Mathematics in Computer Science*, 4:113–137, 2010. 10.1007/s11786-010-0044-3.

[3] L. González-Vega and I. Necula. Efficient topology determination of implicitly defined algebraic plane curves. *Computer Aided Geometric Design*, 19(9), 2002.

[4] F. Rouillier. Solving zero-dimensional systems through the rational univariate representation. *J. of Applicable Algebra in Engineering, Communication and Computing*, 9(5):433–461, 1999.