

On the Exact Computation of the Topology of Real Algebraic Curves *

(Exploiting a little more Geometry and a little less Algebra)

Raimund Seidel
Universität des Saarlandes
66123 Saarbrücken, Germany
rseidel@cs.uni-sb.de

Nicola Wolpert
Max-Planck-Institut für Informatik
66123 Saarbrücken, Germany
wolpert@mpi-sb.mpg.de

ABSTRACT

We consider the problem of computing a representation of the plane graph induced by one (or more) algebraic curves in the real plane. We make no assumptions about the curves, in particular we allow arbitrary singularities and arbitrary intersection. This problem has been well studied for the case of a single curve. All proposed approaches to this problem so far require finding and counting real roots of polynomials over an algebraic extension of \mathbb{Q} , i.e. the coefficients of those polynomials are algebraic numbers. Various algebraic approaches for this real root finding and counting problem have been developed, but they tend to be costly unless speedups via floating point approximations are introduced, which without additional checks in some cases can render the approach incorrect for some inputs.

We propose a method that is always correct and that avoids finding and counting real roots of polynomials with non-rational coefficients. We achieve this using two simple geometric approaches: a triple projections method and a curve avoidance method. We have implemented our approach for the case of computing the topology of a single real algebraic curve. Even this prototypical implementation without optimizations appears to be competitive with other implementations.

Categories and Subject Descriptors

I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—*geometric algorithms*; G.1.5 [Numerical Analysis]: Roots of Nonlinear Equations—*methods for polynomials*; D.m [Software]: Miscellaneous—*robust geometric computation*

*Partially supported by the IST Programme of the European Union as a Shared-cost RTD (FET Open) Project under Contract No IST-2000-26473 (ECG – Effective Computational Geometry for Curves and Surfaces)

©ACM, 2005. This is the authors' version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version was published in the *Proceedings of the 21st Annual Symposium on Computational Geometry* (SCG 2005), <http://doi.acm.org/10.1145/1064092.1064111>

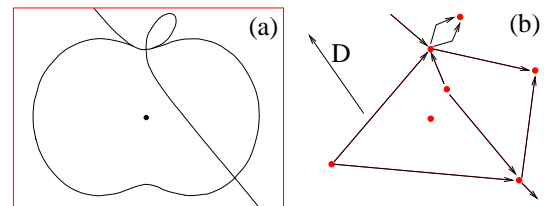


Figure 1: A curve and its plane graph

General Terms

Algorithms, Performance, Reliability

Keywords

Computational geometry, algebraic curves, robustness, exact geometric computation

1. INTRODUCTION

The real zero set $C_f = \{(x, y) \in \mathbb{R}^2 \mid f(x, y) = 0\}$ of a bivariate polynomial f with rational coefficients is usually referred to as a real algebraic curve. Such a curve may have special points, called singularities, where the tangent is not well defined, e.g. isolated points, self-intersections, or cusps. If we fix a direction D as “vertical” then there are further special points, namely points whose tangent line has direction D . All these special points are usually referred to as *critical* points. If the direction D is good, in the sense that no line with direction D is contained in C_f or contains more than one critical point of C_f , then the critical points partition the curve into “monotone” arcs.

We are interested in the following problem: Given such a polynomial f and a good direction D compute a geometric-combinatorial description of the plane graph that is induced by C_f .

As an input example consider the polynomial

$$x^3y^4 - 3x^5 + 2x^5y^2 + 2y^4 + xy^4 - x^4 - 6x^2y^4 + x^3y^2 + 4x^2 + y^2 + 10x^2y^2 - 3x^4y^2 - 3y^6 + x^7 - 4x^3 + y^7 + 2y^5 + y^3x - y^2x - y^5x - 12x^2y + 4x^3y + 2x^2y^3 - 2x^3y^3 + 2x^2y^5 + 3x^4y - x^5y + x^4y^3 - 3y^3.$$

Its curve is depicted inside the box in Figure 1(a). The desired output is indicated in Figure 1(b). The orientation of the edges is “left-to-right” with respect to “vertical” direction D .

A slightly more general problem concerns computing the plane graph induced by two or more algebraic curves. This introduces as additional critical points the intersections between different curves. It turns out that dealing with such points is very similar to dealing with self-intersections.¹ As a matter of fact, situations such as higher order tangential intersections, which cause difficulties in other approaches, pose no particular problem for our method. Dealing with the complexity caused by a large number of curves is well understood in computational geometry (sweep line algorithms, randomized incremental construction). For these reasons we will concentrate in this abstract on the case of a single curve.

2. PREVIOUS WORK

There has been a fair number of papers, e.g. [1, 3, 10, 6, 7, 9, 8] dealing with the problem of determining the topology of a real algebraic curve defined by a polynomial $f(x, y) \in \mathbb{Q}[x, y]$. They all follow the same kind of scheme:

Phase 1: Compute $R(x)$, the discriminant of f with respect to y and determine its real roots $\alpha_1, \dots, \alpha_m$. Each vertical line defined by $x = \alpha_i$ will contain a critical point of curve C_f .

Phase 2: For every α_i compute the real roots of the polynomials $g_i(y) = f(\alpha_i, y)$, i.e. the set $P_i = \{p_{i1}, \dots, p_{in_i}\}$ of intersection points of C_f and the line $x = \alpha_i$.

Phase 3: For each i and each $p \in P_i$ determine $\ell(p)$, the number of branches of C_f entering p from the left, and $r(p)$, the number of branches entering p from the right, and $a(p)$, the number of branches of C_f crossing the line $x = \alpha_i$ above p .

From this information a discrete description of the topology of C_f can be recovered.

In some papers this is all preceded by a Phase 0, which via a linear change of coordinates ensures that f is generic in the sense that the vertical direction is good for f , i.e. C_f contains no vertical line and no vertical line contains two critical points of C_f . In this case for each i there is only one $p \in P_i$ with $(\ell(p), r(p)) \neq (1, 1)$, and it is really on this p for which the values $\ell(p)$, $r(p)$, and $a(p)$ need to be determined.

Notice that the polynomials $g_i(y)$ in Phase 2 in general do not have rational coefficients. This makes real root finding non-trivial. To this end maybe the most elegant approach was proposed by M. Coste and M.F. Roy [3], who employed so-called Thom's codes of real roots. This approach was later on carried further by L. González-Vega and M. El Kahoui [7], who even managed to prove a worst case running time of $O(n^{16} \log^5 n)$ for their approach. Here n is essentially the maximum of the degree of f and the size of the bit-description of the coefficients of f . This bound is the lowest worst case running time bound for this problem known to date. However, as far as we know this algorithm has been of theoretical interest only.

H. Hong [9] solved the real root finding problem of Phase 2 by employing so-called subresultant polynomial remainder sequences. This allows to produce what we call "insulating boxes" for the points $p \in P_i$. In Phase 3 one then needs to

¹This is not surprising as intersections of C_f and C_g are self-intersections of $C_{f \cdot g}$. Considering the eventual computational effort, such a reduction to $C_{f \cdot g}$ is not advisable, though.

count the number of intersections of the boundaries of those boxes with C_f . But since those boxes have rational descriptions this reduces to real root counting for polynomials with rational coefficients. Hong implemented his approach and reported good success with using floating point arithmetic to quickly arrive at good guesses of subsolutions that are subsequently verified using exact arithmetic.

The most recent paper appears to be the one written by L. González-Vega and I. Necula [8]. They invoke Phase 0, which results in exactly one critical point q_i per P_i . They employ so-called Sturm-Habicht sequences, from which they can derive various pieces of information. In particular, if $q_i = (\alpha_i, \beta_i)$ then they can represent β_i as a rational function of α_i , which in a way obviates Phase 2. Moreover, from the Sturm-Habicht sequence they can determine the multiplicity k_i of the root β_i of $g_i(y) = f(\alpha_i, y)$. As, because of Phase 0, β_i is the only multiple root of g_i they get that $G_i(y) = g_i(y)/(y - \beta_i)^{k_i}$ has exactly one fewer real root than g_i has. Therefore comparing the number of real roots of G_i with the number of real roots of $f((\alpha_i + \alpha_{i+1})/2, y)$ allows them to determine $r(q_i)$ and analogously $\ell(q_i)$, thus completing Phase 3.

This last approach is algebraically very elegant and was implemented in MAPLE. However, note that in general the polynomials G_i do not have rational coefficients. Approximating those coefficients by floating point numbers, as proposed by the authors of that paper, will undoubtedly work in most cases, but can (and does) produce incorrect results in some cases. Presumably this floating point approach may work in all cases if the precision of those numbers is chosen appropriately. However, at this point no analysis is available that prescribes how this choice of precision is to be made.

It is worth noting that our approach does not need an a priori precision analysis. Our algorithm adapts automatically to the precision needed.

3. OUR APPROACH, OVERVIEW

Our approach also assumes that a Phase 0 has made f generic. Thus there are no two critical points on the same vertical line. Phase 1 proceeds as in the other algorithms producing the set $A = \{\alpha_1 < \alpha_2 < \dots < \alpha_m\}$ of roots of $R(x)$. Now each of the vertical lines $x = \alpha_i$ contains exactly one critical point, $q_i = (\alpha_i, \beta_i)$. Notice that geometrically Phase 1 is simply a projection of the critical points onto the x -axis. This suggests that the β_i 's could be determined by projecting the critical points onto the y -axis. We proceed this way and determine the set B of real roots of the appropriate discriminant polynomial $S(y)$. Now let K be the set of critical points of f . Unfortunately, knowing just the projections A and B of K we cannot recover K . We just know $K \subset A \times B$. However, if we project K into a third direction and if this direction is non-degenerate in the sense that no two points of $A \times B$ have the same projection, then we can indeed recover K from $A \times B$.

We can view this triple projection approach as a set of m vertical lines and a set n of horizontal lines (horizontal need not be nice) defining a grid of mn points, and a third set of parallel lines selecting m of those grid points (see Figure 2).

Of course in general we cannot represent real roots of polynomials explicitly as single numbers. We will instead use the usual representation of a real root α by isolating intervals I_α which contain the root α and can be made as small as desired. Thus the projection lines discussed in the previous

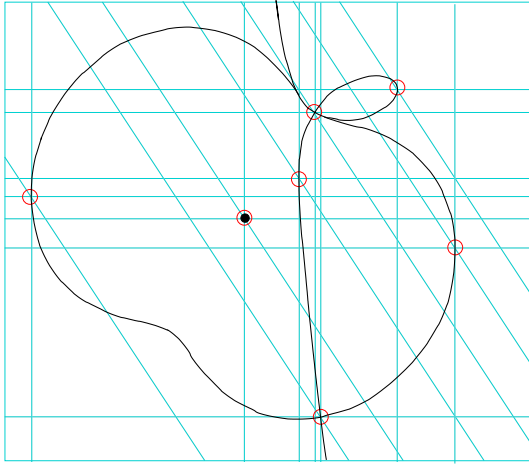


Figure 2: The multiple projection method

paragraph turn into projection stripes. By making these stripes sufficiently narrow the construction described in the previous paragraph can be carried over and made to produce m boxes $I_\alpha \times I_\beta$, each containing one critical point in K . We call those boxes *identifying boxes*. This triple projection method forms Phases 1 and 2 of our approach. The reader may wonder whether computing three projections, i.e. three resultants is not too expensive. In experiments with our implementation we found this not to be true: in almost all cases Phase 3 takes the longest.

This Phase 3 must contract the identifying boxes to *insulating boxes*. An insulating box for critical point q is an axis-parallel rectangle B containing q whose boundary is only intersected by branches of C_f that are incident to q , each such branch intersects B 's boundary exactly once, and all those intersections occur on the vertical sides of B .

Note that from such an insulating box B for q the desired numbers $\ell(q)$, $r(q)$, and $a(q)$ can be computed by counting intersections of C_f and the vertical sides of B (and the vertical lines through those sides). Since those lines and sides will be specified by rational numbers this reduces to real root counting for polynomials with rational coefficients.

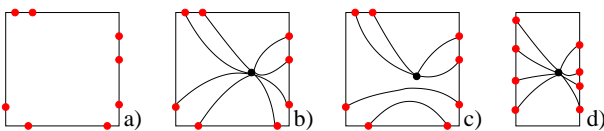


Figure 3: How can we make a box insulating?

How do we shrink identifying boxes to insulating boxes? To appreciate the problem consider Figure 3. Since we cannot “look inside” a box B but can only look at the intersection pattern of f with the boundary of B , we are really faced with a situation of an opaque box depicted in part a). However, the situation inside B may be as indicated in part d), where B already is an insulating box, or as in parts b) and c) where further shrinking is necessary. How can we tell the difference?

Let q be a critical point of curve f and let B be its identifying box. Let ℓ be the vertical line containing q . Note that in general we do not know ℓ . Suppose we had a curve h

that does not contain q and moreover separates q from the other intersections of f and ℓ in the sense that if one walks along ℓ starting at q one always encounters h before f . If one now shrinks B to B' so that B' does not intersect h (but still contains q), then the situation of an arc simply cutting across B' as depicted in Figure 3 c) cannot occur any more. Shrinking B' further to a box B'' whose top and bottom sides do not intersect f then yields the desired insulating box (see Figure 3 d)).

This leaves us with two questions: (i) How do we obtain such a magic curve h ? (ii) How do we test whether B intersects C_h .

Regarding (i) it turns out that one can use $h = f_{y^k}$, the k -th derivative of f with respect to y , where k has to be chosen appropriately according to the “multiplicity” of the special point q . To be precise, using the notation of the previous section, if $q = (\alpha_i, \beta_i)$, then k must be the multiplicity of the root β_i of $g_i(y)$. This k can be determined using subresultants.

Problem (ii) can be solved as follows: Let $B = I_x \times I_y$. To test, whether B intersects C_h , the zero set of h , simply evaluate $h(I_x, I_y)$ using interval arithmetic. The resulting interval I is a superset of $\{h(x, y) \mid (x, y) \in B\}$. Thus $0 \notin I$ implies B does not intersect C_h . If critical point q does indeed not lie on C_h as asserted, then for some sufficiently shrunk version of its insulating box this interval arithmetic based test will succeed.

4. OUR APPROACH, DETAILS

4.1 Preliminaries

The objects we consider and manipulate in our work are *algebraic curves*. A polynomial $f \in \mathbb{Q}[x, y]$ defines such a curve C_f via its zero-set, i.e. $C_f = \{(\alpha, \beta) \in \mathbb{R}^2 \mid f(\alpha, \beta) = 0\}$.

We assume without loss of generality that every polynomial f defining a curve is *square-free*. Geometrically this means that the curve f has no multiple components. Square-freeness can be easily established. See the next section.

For a curve f the *gradient vector* of f is defined to be

$$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right) =: (f_x, f_y) \in (\mathbb{Q}[x, y])^2.$$

With the help of the gradient vector we characterize a point $(\alpha, \beta) \in \mathbb{R}^2$ lying on a curve f . It is named a *singular point* of f if $\nabla f(\alpha, \beta) = (0, 0)$, otherwise *non-singular*. The non-singular points of f are exactly the ones that admit a unique tangent line to the curve. The tangent line at a non-singular point (α, β) is perpendicular to $\nabla f(\alpha, \beta)$. We call a non-singular point (α, β) of f *extreme* if it has a vertical tangent, i.e. if $\nabla f(\alpha, \beta) = c \cdot (1, 0)$ with c being a non-zero constant. We refer to singularities and extreme points of f collectively as the *critical points* of f . The set of these critical points is exactly the set of intersection points of the curves f and f_y , i.e. a point (α, β) is critical if $f(\alpha, \beta) = f_y(\alpha, \beta) = 0$.

Note that all these definitions carry over to possible complex parts of the curve f . Although we are not really interested in computing complex singularities or extreme points of f , these complex points do influence our computations at various points.

We call f *generic* if the vertical direction is good for f , in other words, if no two critical points (including complex

ones) have the same x -coordinate and if f contains no vertical line and has no vertical asymptote. In addition C_f must not have an inflection point with vertical tangent. For our computations we will assume that f is generic. This can be achieved by skewing the plane, i.e. considering the curve $\bar{f}(x, y) = f(x + \lambda y, y)$, where λ is an appropriately chosen real. Such a skewing leaves the topological structure of the curve unchanged. How can λ be chosen “appropriately?” A random choice is appropriate with probability 1. However, there are also deterministic methods for producing such an appropriate choice, e.g. [7, 5].

4.2 Mathematical Tools

In our algorithms we will need the following basic algebraic tools: We will need to perform simple operations on polynomials such as taking derivatives, dividing, and performing gcd-computations on uni- and bivariate polynomials. Recall that one can make a polynomial $g \in \mathbb{Q}[x]$ square-free by dividing g by $\gcd(g, g_x)$. This square-free polynomial has the same roots as g and all the roots are simple. Note that a square-free $g \in \mathbb{Q}[x]$ changes sign at every real root. Similarly $g \in \mathbb{Q}[x, y]$ can be made square-free by a bivariate gcd-computation.

We are interested in the real roots of $g \in \mathbb{Q}[x]$. In general those roots are not rational numbers but algebraic numbers. Such a root α can be represented by a pair $([a, b], \tilde{g})$, where $[a, b] \ni \alpha$ is an *isolating interval*, i.e. a closed interval with rational endpoints that contains no other root of g , and \tilde{g} is g made square-free. Note that since \tilde{g} changes sign at α this representation allows to shrink the containing interval $[a, b]$ to any desired positive width by repeated bisection. Two algebraic numbers α and β with representations $([a, b], g)$ and $([c, d], h)$ can be tested for equality by testing whether $[a, b] \cap [c, d]$ contains a root of $\gcd(g, h)$. The inequality $\alpha < \beta$ can then be trivially tested after shrinking the containing intervals until they are disjoint.

Given a polynomial $g \in \mathbb{Q}[x]$ the set of all of its real roots in the representation outlined above can be determined for instance by Uspenky’s algorithm [2].

Please note that “computing” the intersections of a curve f with a rational straight line ℓ reduces exactly to such root finding. This means such intersections are represented as disjoint intervals on ℓ and these intervals can be reduced to any positive size desired and they can also be ordered along ℓ . The same can be said about “computing” the intersections of f with the boundary of a box B .

Sometimes we need to count the number of intersections of a curve f with a segment of a rational straight line. This reduces to counting the number of real roots of a polynomial P with rational coefficients that are contained in a certain interval I . This can either be done by isolating all the real roots of P and determining which ones lie in I , or sometimes more efficiently by applying the method of so-called Sturm sequences (see e.g. [12, chapter 7]).

For computing the critical points of a single curve and for computing the intersection points of two curves the theory of resultants is all important. All results described in this section are well-known and can be found e.g. in [4, 11].

Consider bivariate polynomials $f, g \in \mathbb{Q}[x, y]$ as polynomials in y with coefficients in $\mathbb{Q}[x]$: $f(x, y) = \sum_{0 \leq i \leq n} f_i(x)y^i \in \mathbb{Q}[x, y]$ with $f_n(x) \not\equiv 0$ and $g(x, y) = \sum_{0 \leq j \leq m} g_j(x)y^j \in \mathbb{Q}[x, y]$ with $g_m(x) \not\equiv 0$. We consider only square-free and generic polynomials.

We can compute a rational polynomial $\text{res}(f, g, y) \in \mathbb{Q}[x]$ called the *resultant* of f and g with respect to y (of course we can do the same with respect to x). The degree of $\text{res}(f, g, y)$ is bounded from above by $\deg(f) \cdot \deg(g)$. The resultant has the following property:

PROPOSITION 1. *Let $f, g \in \mathbb{Q}[x, y]$ and let $f_n, g_m \in \mathbb{Q}[x]$ be as defined before. If $\text{res}(f, g, y) \in \mathbb{Q}[x]$ vanishes for $\alpha \in \mathbb{R}$, then*

1. *either $f_n(\alpha) = g_m(\alpha) = 0$ and we call α an extraneous root*
2. *or there is $\beta \in \mathbb{C}$ such that $f(\alpha, \beta) = g(\alpha, \beta) = 0$.*

If in addition f is generic and $g = f_y$, then there is exactly one $\beta \in \mathbb{R}$ such that $0 = f(\alpha, \beta) = g(\alpha, \beta)$.

Thus determining the coordinates of intersection points reduces to the problem of real root isolation.

If α is not an extraneous root of the resultant, then the vanishing of the resultant at $x = \alpha \in \mathbb{R}$ implies that the polynomials $f(\alpha, y), g(\alpha, y) \in \mathbb{R}[y]$ have a common factor. As we will see, for generic curves we are also interested in the degree of this common factor. We can compute a sequence of polynomials $\text{sres}_i(f, g, y) \in \mathbb{Q}[x]$, $0 \leq i \leq \min(n, m)$, called the *subresultants* of f and g with respect to y (sometimes this is also called the principal subresultant coefficient). We have $\text{sres}_0(f, g, y) = \text{res}(f, g, y)$ and moreover

PROPOSITION 2. *Let $f, g \in \mathbb{Q}[x, y]$ be generic and $1 \leq k \in \mathbb{N}$. Then $\alpha \in \mathbb{R}$ is a root of $\text{sres}_i(f, g, y) \in \mathbb{Q}[x]$ for $0 \leq i \leq k - 1$ if and only if $f(\alpha, y), g(\alpha, y) \in \mathbb{R}[y]$ have a common factor of degree at least k .*

It is easy to see that using gcd-computation of the subresultants we can factor the resultant $\text{res}(f, g, y) = r_1 \cdot r_2 \dots r_{\min(m, n)}$ such that for each root α of $r_i(x) \in \mathbb{Q}[x]$ the polynomials $f(\alpha, y), g(\alpha, y)$ have a gcd of degree i . We call i the *y-degree* of α with respect to f and g : $i =: d_{f, g}(\alpha) = \deg(\gcd(f(\alpha, y), g(\alpha, y)))$.

Finally a technical remark: In the following when we speak about resultants and subresultants we will always mean the square-free versions of these polynomials.

4.3 Finding Identifying Boxes

Given a generic curve f we would like to find identifying boxes for its critical points. Recall that those points are exactly the intersections of f and f_y .

Compute the resultants $R(x) = \text{res}(f, f_y, y)$ and $S(y) = \text{res}(f, f_y, x)$ and determine isolating intervals for the real roots of these two polynomials. The intervals for R define m vertical stripes in the plane. By the fact that f is generic every one of those stripes contains exactly one critical point of f (and it is real). Similarly the intervals for S define n horizontal stripes. They are not necessarily in 1-1 correspondence to the critical points of f . Shrink all the intervals (and hence the stripes) so that the collection of mn boxes formed by the intersections of the vertical and horizontal stripes satisfies the following property: For each of the $\binom{mn}{2}$ pairs of distinct boxes B, B' the set of “forbidden” directions, i.e. directions subtending lines that intersect both B and B' is “small,” which is to mean when directions are expressed in terms of polar angles the forbidden directions

form an interval of size $\pi/(mn)^2$ at most.² Then the set of directions that is forbidden by some pair of boxes has measure at most $\pi/2$, which means that at least “half” of the possible directions are good in the sense that no line with that direction intersects two of the boxes.

Now by repeated random choice find a rational direction D into which the mn boxes have disjoint projections. We would like to compute stripes in direction D similar to the vertical ones that cover all critical points. This can be achieved by shearing the plane so that D becomes the vertical direction and then computing a resultant. To be specific, assume $(\lambda, 1)$ is a vector with direction D . Simply compute the resultant $R^{(\lambda)}(x) = \text{res}(f^{(\lambda)}, f_y^{(\lambda)}, y)$, where $f^{(\lambda)}(x, y) := f(x - \lambda y, y)$ and $f_y^{(\lambda)}(x, y) := f_y(x - \lambda y, y)$. The real roots of $R^{(\lambda)}$ are then the projections of the intersection points of f and f_y along the line $x + \lambda y$ onto the x -axis. The isolating intervals for those roots of $R^{(\lambda)}$ induce the desired stripes in the good direction D .

There will always be at least m such stripes. If exactly m stripes were produced this way, then sufficiently narrowed down, they pick out from the mn boxes exactly those that contain a critical point of f , i.e. the set of identifying boxes was produced.

If more than m stripes were produced, then this was very unlikely bad luck because in this case D happens to be the direction of a straight line contained in f or of an asymptote of f or some complex singularity of f happens to project along D into the reals. (For the complex singularity (w, z) the chosen λ happened to be such that $w + \lambda z$ is real.) However if f has degree d then it can contain at most d straight lines or asymptotes, each making one direction bad, and there can be at most $\binom{d}{2}$ complex singularities each making exactly one (real) direction bad. Thus simply retrying will eventually produce a successful direction D .

4.4 Finding Insulating Boxes

Once we have found the identifying boxes for the critical points of f we have to shrink the boxes until they are insulating. How far do we need to shrink them? To this end consider the following definition: Let $q = (\alpha, \beta)$ be a critical point of f and let ℓ be the vertical line $x = \alpha$ through q . Let C be a curve not containing q and let L_q be the interval on ℓ formed by the connected component of $\ell \setminus C$ that contains q . We call a curve C a *separating curve* for q , if $q \notin C$ and L_q contains no root of $g(y) = f(\alpha, y)$ except for β . In other words, if $\beta_1 < \beta_2 < \dots < \beta_r$ are the roots of $g(y)$ and $\beta = \beta_j$, then C does not intersect ℓ in β but C intersects ℓ between β and β_{j+1} (if it exists) and C intersects ℓ between β and β_{j-1} (if it exists).

The following is obvious:

PROPOSITION 3. *Let C be a separating curve for critical point q of f , and let B be an identifying box of q that does not intersect C and whose boundary is intersected by C_f only along the vertical sides.*

Then B is an insulating box for q .

Thus if a separating curve C for q and an identifying box B for q is available, then an insulating box for q can be

²It is not hard to prove that this can be for instance achieved by reducing the widths of all the stripes, and hence the side lengths of all boxes to something less than $D/(mn)^2$, where D is the smallest distance between two parallel stripes.

obtained by first shrinking B until it does not intersect C and then shrinking it horizontally until the horizontal sides do not intersect C_f .

The following proposition tells how a separating curve can be obtained for every critical point q of f :

PROPOSITION 4. *Let $q = (\alpha, \beta)$ be a critical point of f . If β is a root of the polynomial $g(y) = f(\alpha, y)$ of multiplicity exactly k , then C_h is a separating curve for q , where $h = f_{y^k}(x, y)$, the k -th derivative of f with respect to y .*

PROOF. For $0 \leq i \leq k$ let $g_i(y) = f_{y^i}(\alpha, y)$. Note that $g_0 = g$ and that for each $i > 0$ the polynomial g_i is the derivative of g_{i-1} . Moreover, since β is a root of g_0 of multiplicity exactly k , we get that β is also a root of g_i for $0 \leq i < k$. We need to show that β is separated from every other root of g_0 by a root of g_k . Let $b_0 \neq \beta$ be a root of g_0 . By the mean value theorem β and b_0 are separated by a root b_1 of g_1 . If $k = 1$ we are done. Otherwise β is a root of g_1 and hence by the mean value theorem β and b_1 are separated by a root b_2 of g_2 . This b_2 also separates β from b_0 . Repeat this argument up to g_k . \square

This leaves us with the problem of determining for each critical point $q = (\alpha, \beta)$ the exact multiplicity k of β as a root of $g(y) = f(\alpha, y)$. Here genericity of f is important. By genericity of f no two critical points of f are covertical, which means β is the only multiple root of g and hence β has multiplicity exactly k iff g and its derivative have a common factor of degree $k-1$, in other words, d_{f, f_y} , the y -degree of α with respect to f and f_y , is $k-1$. Thus, following the remark after Proposition 2 it suffices to determine the factor r_i of the resultant factorization $R(x) = \text{res}(f, f_y, y) = r_1(x) \cdot r_2(x) \cdot \dots$ of which α is a root.

With a separating curve C_h available for q we want to shrink an identifying box of q until it does not intersect h . How can we test whether a box $B = I_x \times I_y$ intersects C_h ? A simple way to achieve this is to evaluate the polynomial $h(x, y)$ in interval arithmetic using I_x and I_y as arguments. If 0 is not contained in the result interval, then B does not intersect C_h . Note that this test may err, but only in one direction: it may report an intersection, although there is not any. But for B small enough this test will succeed. Shrinking the resulting box horizontally until its top and bottom sides do not intersect C_f is then straightforward.

Note that no separation bounds and no a priori precision analysis are necessary for this approach to work.

After an insulating box $B = [b, c] \times [d, e]$ of a critical point q has been determined, the desired numbers $\ell(q)$ and $r(q)$ can be determined by counting the number of real roots of $f(b, y)$ and $f(c, y)$ in the interval $[d, e]$, and $a(q)$ is the number of real roots of $f(b, y)$ that are bigger than e .

It is possible to achieve increased efficiency by treating extreme points specially. We leave the details of this to the full version of the paper.

5. ARRANGEMENTS OF CURVES

For computing the arrangement of several algebraic curves the basic subproblem is to compute the topology of the union of two algebraic curves. With a solution to this subproblem one can apply one of the standard computational geometry approaches such as the sweep-line paradigm or randomized incremental construction to compute the arrangement of a larger number of curves. This is relatively straightforward,

although note that a fair amount of book-keeping is necessary, especially if more than two curves intersect in a common point.

So assume f and g are two generic polynomials defining curves C_f and C_g . Compute the sets Q_f and Q_g of critical points of f and g along the lines outlined above, where each critical point is represented by an insulating box. In order to compute the set P of intersection points of C_f and C_g apply the triple projection method. This is possible since the projections of those intersection points are given by the appropriate resultant of the polynomials f and g . This produces an identifying box for each intersection point $p \in P$.

Now let p be an intersection point in P and let B its identifying box.

If p is neither in Q_f nor in Q_g (this can be tested via gcd's of resultants), then shrinking B until its (vertical) boundary intersects C_f and C_g exactly twice produces an insulating box for q .

If p is in one of Q_f or Q_g , say Q_f , and B' is the insulating box of p for f , then shrinking B' so that its (vertical) boundary intersects C_g twice produces an insulating box of p for f and g .

If p is in Q_f and in Q_g and B_f and B_g are the respective insulating boxes, then shrinking $B_f \cap B_g$ horizontally until its horizontal boundary intersects neither C_f nor C_g produces an insulating box of p for f and g .

The vertical order in which branches of C_f and of C_g “enter” p from the left or “leave” to the right can then be determined from the intersection pattern of C_f and C_g with the boundary of the insulating box.

6. EXPERIMENTS

We have written a prototypical MAPLE implementation `insulate` of our method for computing the topology of a single real algebraic curve given by a bivariate polynomial f with rational coefficients. It uses exact arithmetic and follows the approach outlined in this abstract closely. For transforming f into a sheared generic version we could use some code that was supplied to us by M’hammed El Kahoui.

We could compare the performance of our implementation with the performance of a MAPLE program `top` that was supplied to us by Laureano González-Vega and Ioana Necula, which implements their algorithm presented in [8]. This program uses floating point computations extensively and allows the user to specify a starting precision for those computations. If the program finds this starting precision to be insufficient then it keeps increasing the precision until a solution can be produced.

The results so far of this performance comparison can be summarized as follows: On examples with well separated critical points where the running time is less than a second `top` outperforms `insulate`, typically by a factor of about 2 or 3. On examples with crowded critical points where the running time is substantially more than a second `top` can produce incorrect results if the starting precision is not high enough, whereas `insulate` has always produced a correct result in those cases. Moreover, on many of those examples `insulate` outperforms `top` even if the latter is started with high enough precision to produce the correct result.

We give a few illustrative examples of runs of `insulate` in the appendix. The implementation can be obtained from us on request.

Acknowledgments

We would like to thank Chee Yap and M’hammed El Kahoui for extensive discussions. We thank M’hammed El Kahoui and also Laureano González-Vega and Ioana Necula for supplying their code.

7. REFERENCES

- [1] D. S. Arnon and S. McCallum. A polynomial time algorithm for the topological type of a real algebraic curve. *Journal of Symbolic Computation*, 5:213–236, 1988.
- [2] G. E. Collins and R. Loos. Real zeros of polynomials. In B. Buchberger, G. E. Collins, and R. Loos, editors, *Computer Algebra: Symbolic and Algebraic Computation*, pages 83–94. Springer-Verlag, New York, NY, 1982.
- [3] M. Coste and M.-F. Roy. Thom’s lemma, the coding of real algebraic numbers and the computation of the topology of semi-algebraic sets. *Journal of Symbolic Computation*, 5:121–129, 1988.
- [4] D. Cox, J. Little, and D. O’Shea. *Ideals, Varieties, and Algorithms*. Springer, New York, 1997.
- [5] M. El Kahoui. Computing with real algebraic curves in generic position. in preparation, 2004.
- [6] H. Feng. *Decomposition and Computation of the Topology of Plane Real Algebraic Curves*. The Royal Institute of Technology, Stockholm, 1992. PhD Thesis.
- [7] L. González-Vega and M. El Kahoui. An improved upper complexity bound for the topology computation of a real algebraic plane curve. *Journal of Complexity*, 12:527–544, 1996.
- [8] L. González-Vega and I. Necula. Efficient topology determination of implicitly defined algebraic plane curves. *Computer Aided Geometric Design*, 19:719–743, 2002.
- [9] H. Hong. An efficient method for analyzing the topology of plane real algebraic curves. *Mathematics and Computers in Simulation*, 42:571–582, 1996.
- [10] T. Sakkalis. The topological configuration of a real algebraic curve. *Bulletin of the Australian Mathematical Society*, 43:37–50, 1991.
- [11] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 1999.
- [12] C. K. Yap. *Fundamental Problems of Algorithmic Algebra*. Oxford University Press, New York, Oxford, 2000.

APPENDIX

These are the running times for `insulate` for five example inputs. Examples 4 and 5 were intended to produce crowded critical points. The time for making the input function generic is included in the running time. The running times are averages over 5 runs. Running times are reported in seconds.

The program was executed under MAPLE 9.01 on a PC with a 2.66 GHz Pentium 4 processor with 512 MBytes of memory under Windows 2000 Professional.

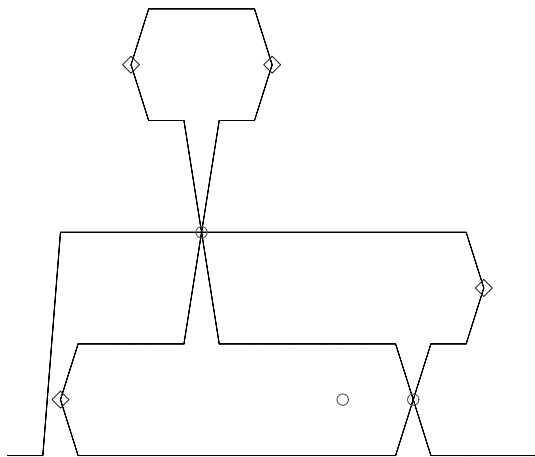
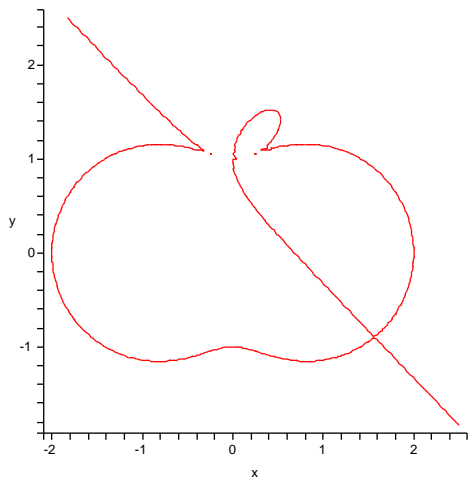
input	insulate
f_1	2.169
f_2	0.156
f_3	2.731
f_4	13.528
f_5	466.541

Some of the example polynomials are presented in a factorized form. Note that `insulate` at this point is not able to take advantage of such a factorization but works simply with the expanded version of the polynomial. Below we show the example polynomials along with a drawing produced by the MAPLE command `implicitplot` as well as with the output produced by `insulate`. Beware that `insulate` uses a shear to make f generic.

A. EXAMPLE 1

$$f_1 = (x^3 + x - 1 - y * x + 3 * y - 3 * y^2 + y^3) * (x^4 + 2 * y^2 * x^2 - 4 * x^2 - y^2 + y^4)$$

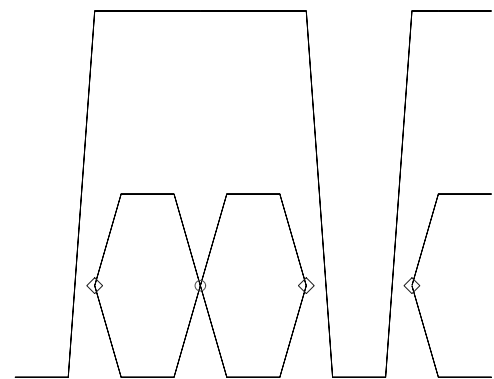
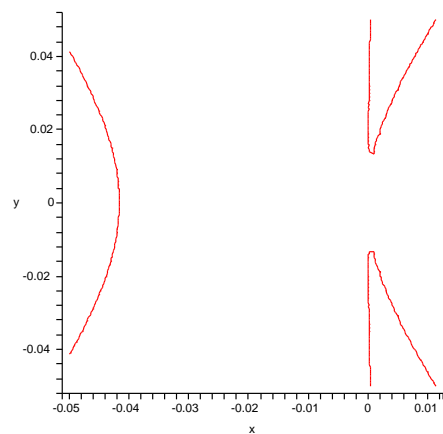
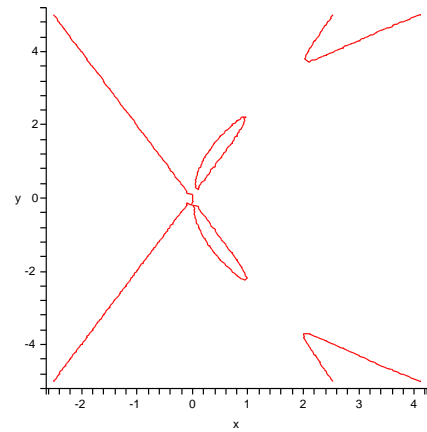
This polynomial defines the “apple” which was already introduced in the introduction.



B. EXAMPLE 2

$$f_2 = y^4 - 6 * y^2 * x + x^2 - 4 * y^2 * x^2 + 24 * x^3$$

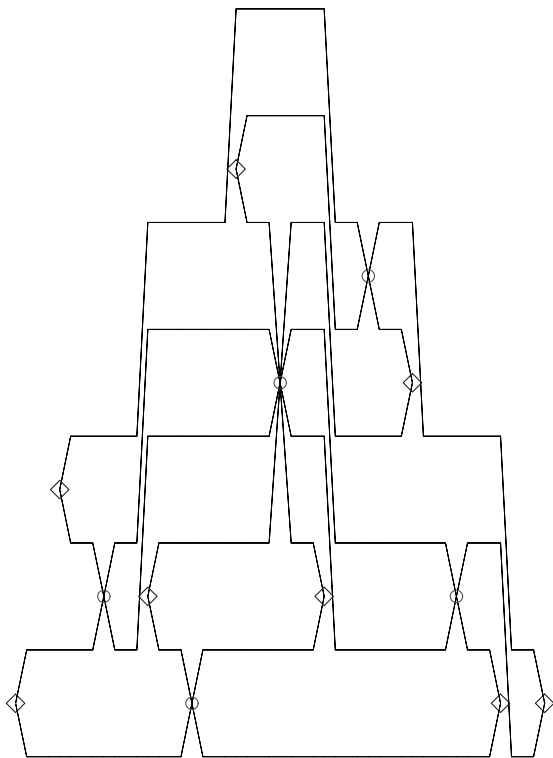
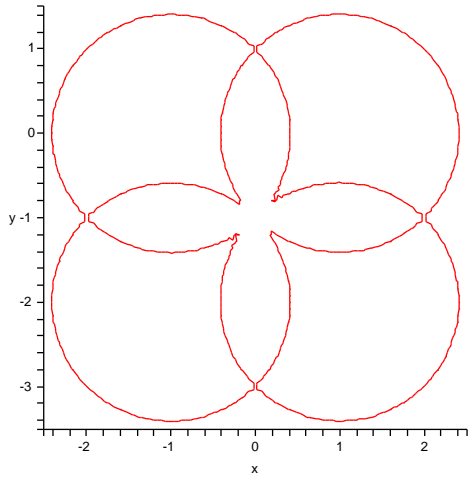
The curve defined by f_2 has one singular point in the origin. We have produced two drawings for this curve where the second one is a more detailed plot around $(0,0)$. By our computation it turns out that the number of branches entering the singular point is four.



C. EXAMPLE 3

$$f_3 = ((x-1)^2 + y^2 - 2) * ((x+1)^2 + y^2 - 2) * ((x-1)^2 + (y+2)^2 - 2) * ((x+1)^2 + (y+2)^2 - 2)$$

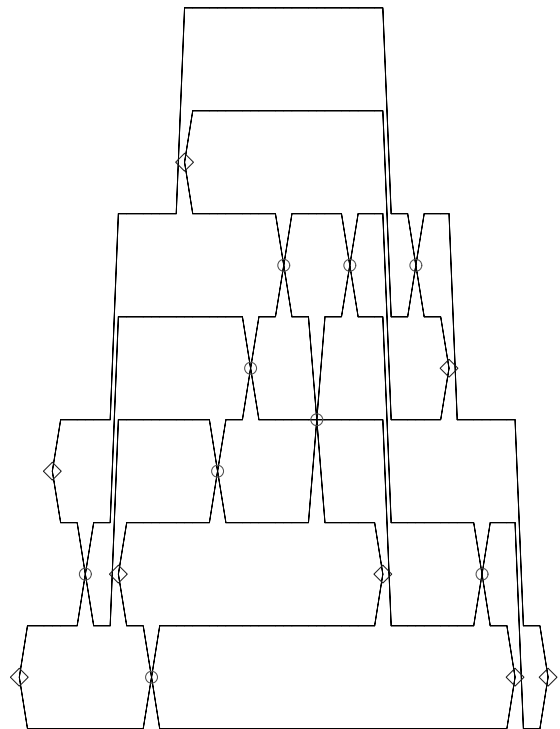
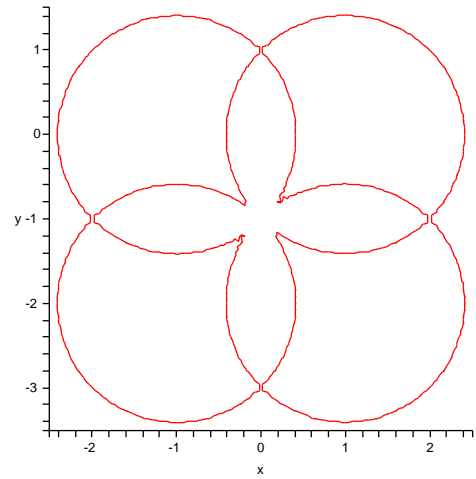
The curve defined by f_3 consists of four circles intersecting in the point $(0, -1)$.



D. EXAMPLE 4

$$f_4 = ((x-1)^2 + y^2 - 2) * ((x+1)^2 + y^2 - 2) * ((x-1)^2 + (y+2)^2 - 2) * ((x+1)^2 + (y+2)^2 - 2 - (1/10^5))$$

This curve is obtained from the previous one by making the last of the four circles a little bit larger. This causes 4 new singular points close to $(0, -1)$.



E. EXAMPLE 5

$$f_5 = ((x+1)^4 + (y+1)^4 - 2) * \\ * ((x+1)^4 + (y-1)^4 - 2) * \\ * ((x-1)^4 + (y+1)^4 - 2) * \\ * ((x-1)^4 + (y-1)^4 - 2 - (1/10^5))$$

This example is constructed the same way as the one before. Again we take four circles intersecting in the point $(0, 0)$, but now with respect to the L_4 norm. Going from the L_2 norm to the L_4 norm increases the degree of the curve from 8 to 16. We then enlarge the last circle a bit causing 4 new singularities close to the origin.

