The Complexity Analysis For An Algorithm To Compute Adjoints For Curves Due To Jeremy Teitelbaum

Josef Schicho, Bogdan Mătăsaru

January 8, 1999

Abstract

The method of adjoints is used in many CAD/CAM applications where the parameterization of curves is needed. Also an increasing number of applications deal with algebraic surfaces, for which efficient algorithm are not yet discovered. Though, there is an algorithm due to Schicho, and there are possibilities to improve it. The aim of this paper is to give an estimation of the complexity of computing adjoints for curves.

1 Introduction

In this paper we will present an algorithm proposed by Jeremy Teitelbaum for computing the resolution of plane curve singularities and, derived from this, an algorithm to compute the adjoints for curves.

In the beginning there will be a short presentation of the problem of computing adjoints. Then, several existent algorithms will be enumerated. More emphasis will be put on the algorithm proposed by Titelbaum. The next subsections will give some useful definitions and will introduce the notion of Gröbner bases for power series rings. Finally, various algorithms which lead to the algorithm for computing the adjoints and the algorithm for computing adjoints itself will be shown and also the number of field operations required.

2 The Problem

The computation of the adjoints is the only way to solve the problem of parameterization for curves and surfaces. The parameterization and rational parameterization are of an increasing importance in recent times because they constitute the most difficult step in computing NURBS(**N**on-**u**niform **r**ational **B**-**s**pline)-representations. The NURBS-representations are used in many CAD/CAM applications.

The parameterization of curves is not the only domain which heavily uses adjoints. Various problems in the theory of curves classification are such an example.

3 Known Algorithms

There are several algorithms which compute adjoints for curves.

- A good implementation can be found in the CASA package developed at RISC. It is available for download on the ftp site **ftp://ftp.risc.uni-linz.ac.at/pub/casa**. Many attempts to make a complexity analysis for the algorithm implemented in CASA were done. A coarse bound found was exponential, but a polynomial behavior is expected.
- Jeremy Teitelbaum proposed an algorithm for computing the resolution of plane curve singularities, and based on this, an algorithm for computing the adjoints can be derived. It also made a complexity analysis of his algorithm. Unfortunately, the bound is polynomial in one of the outputs, which is not very interesting from the complexity analysis's point of view. We will present later this algorithm.

4 Teitelbaum's Algorithm

In the following section we will present Teitelbaum's algorithm for computing the adjoints. First of all we will give some usefull definitions. Next, the algorithm itself and the analysis of complexity made by Teitelbaum will be described. In the end, a slight refinement of the bound will be presented.

4.1 Definitions

4.1.1 Preliminaries

In the following we will consider a field $\mathbb{K}$. A plane curve singularity over $\mathbb{K}$ is defined by a power series $F(x, y) \in A = \mathbb{K}[[x, y]]$ such that A/FA is reduced. In order to have a reduced quotient field, F should be square free.

Lemma 1. *Given K a field of characteristic 0, there is no algorithm for deciding whether or not a power series $F \in K[[x, y]]$ is square free.*

Proof. Suppose there is such an algorithm. Let $f : \mathbb{N} \to \mathbb{K}$ be a computable function and $F = y^2 + \sum_{i=0}^{\infty} f(i) \cdot x^i$. Then F is square free if and only if $f(i) = 0, \forall i \in \mathbb{N}$. Thus, we can determine whether f is identically 0 or not by applying the algorithm to F. But we cannot determine whether f is identically 0. $\square$

Since there is no algorithm to decide whether a power series is square free, we will assume that the power series we will be working with are square free.

Of course, for some special classes of power series, like polynomials, the problem is computable.

Let $p \in \mathbb{K}[T]$ be square free and let $D(p) = \mathbb{K}[T]/p$. A power series $F \in D(p)[[x, y]]$ represents finitely many plane curve singularities. Such a power series defines the following invariants of the ring $D(p)[[x, y]]/F$:

Definition 1. Let $A = D(p)[[x, y]$ and let $F \in A$ be a square free power series. Let $B = A/FA$. Then we define the following invariants:

1. The *multiplicity* $m_F = \max\{i \in \mathbb{N} | F \in (x, y)^i\}$,
2. The *Milnor number* $\mu_F = dim_{\mathbb{K}} A/(\partial F/\partial x, \partial F/\partial y)$,
3. The *conductor* $c_F = 2dim_{\mathbb{K}} B^*/B$, where B^* denotes the total integral closure of B,
4. The *degree* $\delta_F = min\{i \in \mathbb{N} | (x, y)^i \subset (\partial F/\partial x, \partial F/\partial y)\}$.

Lemma 2. *The following relationships between the above invariants hold:*

$$m_F - 1 \leq \delta_F \leq \mu_F \leq c_F \leq 2\mu_F \leq C deg(p)\delta_F^2.$$

These local invariants and the relationships between them will be used further in order to measure the complexity of computing the adjoints. They are finite because F is square free.

A plane curve singularity is defined by a power series $F(x, y) \in D(p)[[x, y]]$. The following lemma shows that such a singularity depends on only finitely many coefficients of F. Moreover, the number of necessary coefficients is polynomially bounded by the Milnor number μ_F.

Lemma 3. *Let $\mathbb{K}$ be a field of characteristic zero, let $p \in \mathbb{K}[T]$ be square free, and let F be an element of the ideal (x, y) in $D(p)[[x, y]]$. Let $I_F = (\partial F/\partial x, \partial F/\partial y)$ and G also a power series in $D(p)[[x, y]]$ such that $G \equiv F(mod\ (x, y)I_F^2)$. Then there are power series*

$$u(x, y) = x + \cdots, \quad v(x, y) = y + \cdots,$$

such that

$$\begin{aligned} u &\equiv x(mod\ I(x, y)), \\ v &\equiv y(mod\ I(x, y)), \\ G(x, y) &= F(u(x, y), v(x, y)). \end{aligned}$$

Lemma 4. *Let F be a square free power series in $D(p)[[x, y]]$. We construct the polynomial G from F by dropping all the terms with degree greater than $2\delta_F$. Let $I_G = (\partial F/\partial x, \partial F/\partial y)$. Then $I_G = I_F$ and there is an automorphism $\theta : D(p)[[x, y]] \to D(p)[[x, y]]$ such that $\theta(F) = G$. This automorphism induces the identity map on $A/I_F = A/I_G$.*

The number of monomials in a polynomial of degree 2δ is $\mathcal{O}(\delta^2)$ and, by Lemma 2, this is bounded by $\mathcal{O}(\mu_F^2)$, which is polynomial in μ_F.

Lemma 4 tells us that an analytical equivalent polynomial G can be constructed if we know the value of δ_F and the amount of data necessary to describe a plane curve singularity is polynomially bounded by $\mathcal{O}(\mu_F^2)$.

In order to compute the values of δ_F and μ_F we need to adapt the algorithm for computing Gröbner bases from polynomial rings to power series rings.

4.1.2 Gröbner bases in power series rings

We will consider the lexicographically within the degree ordering on the monomials in x and y given by:

$$1 > x > y > x^2 > xy > y^2 > \cdots.$$

Definition 2. Let $f \in D(p)[[x, y]]$ be a power series. We define the *leading term* $l(f, p)$ to be the largest monomial in f with nonzero coefficient and the *leading coefficient* $c(f, p)$ will be its coefficient.

Definition 3 (Gröbner basis). Suppose p is be a prime polynomial over $\mathbb{K}$, and that $I \subset D(p)$ is an ideal. Let $B = \{f_1, \ldots, f_n\}$ be a subset of $D(p)$ such that $l(f_1, p) > l(f_2, p) > \ldots > l(f_n, p)$ and $< B >= I$. Let $B^* = \{l(f_1, p), \ldots, l(f_n, p)\}$ be the set of leading terms of B and $I^* = \{l(f, p) | f \in I\}$. Then B is a *Gröbner basis* for I if and only if

$$I^* =< B^* > .$$

Hironaka gave a theorem about the uniqueness of polynomial decomposition in polynomial rings. In our case $D(p)$ contains divisors of zero, thus the uniqueness property is not true anymore. This can be avoided by forcing the condition that the leading coefficients of all f_i's are units of $D(p)$.

Definition 4. Let $p(T)$ be a square free polynomial over $\mathbb{K}$, and let $I \subset D(p)[[x, y]]$ be an ideal. Let $B = \{f_1, \ldots, f_n\}$ be a geneating set for I. Then B is called a *Gröbner basis of I over $D(p)$* if and only if $I^* =< B^* >$ and all $c(f_i, p)$ are units in $D(p)$, where B^* and I^* have the same meaning as in Definition 3.

By this supplementry constraint on the leading coefficients the uniqueness is insured.

Lemma 5. *Let $p(T)$ be a square free polynomial over $\mathbb{K}$, and let $I \subset D(p)[[x, y]]$ be an ideal. Let $B = \{f_1, \ldots, f_n\}$ be a Gröbner basis of I over $D(p)$, with $f_1 > f_2 > \ldots > f_n$. Then every element $g \in D(p)[[x, y]]$ has a unique representation as*

$$g = \sum_{i=1}^{n} a_i \cdot f_i + a_{n+1}, \quad a_i \in D(p),$$

where no monomial m occuring in a_{n+1} with nonzero coefficient is divisible by any $l(f_i, p)$, and if m is a monomial with nonzero coefficient in any of a_i, then $m \cdot l(f_i, p)$ is not divisible by $l(f_j, p)$ for any $j < i$. Such a representation is called a normal form *for g.*

An immediate corollary of this lemma tells us the index of an ideal is the same as the index of the monomial ideal generated by the leading terms of the elements of a Gröbner basis. In the above notation this means

$$\mu_F = dim_{\mathbb{K}} D(p)[[x,y]]/B^* D(p)[[x,y]] = deg(p) dim_{\mathbb{K}} \mathbb{K}[[x,y]]/B^* \mathbb{K}[[x,y]].$$

4.1.3 Computing Gröbner bases in power series rings

In polynomial rings there is the *Buchberger algorithm* which computes a Gröbner basis of the ideal. It uses the notion of *S-polynomial*.

Definition 5 (S-polynomial). Let $g, h \in D(p)[[x,y]]$ be two power series, and let m be the least common multiple of g and h. Then the S-polynomial of g and h is defined by

$$S(g,h) = \frac{mc(h,p)}{l(g,p)} g - \frac{mc(g,p)}{l(h,p)} h.$$

The normal form of a power series is obtained by the *reduction algorithm.*

Definition 6. Let $G = g_1, \ldots, g_n$ be a set of elements of $D(p)[[x,y]]$ such that the leading coefficients $c(g_i, p) = 1$ for all i. Let h be a power series in $D(p)[[x,y]]$.

The R-operation is defined as

$$R(h,G,p) = \lim_{n\to\infty} R_n(h,G,p),$$

where R_n is given by the following rules

1. If $l(g_i,p) \not| l(h,p), \forall 1 \le i \le n$, then $R_1(h,G,p) = h$.
2. Else $g = \max\{g \in G | l(g,p) | l(h,p)\}$ and $R_1(h,G,p) = S(h,g)$.
3. $R_n(h,G,p) = R_1(R_{n-1}(h,G,p),G,p)$.

The Buchberger's algorithm uses a criterion to decide whether we reached a Gröbner basis or not. This criterion is derived from the *Main Theorem of Gröbner Bases Theory.* For the case of Gröbner bases in power series rings it needs a slight modification.

Theorem 6. *Let $I \subseteq A(p)$ be an ideal and let $B = \{f_1, \ldots, f_n\} \subset A(p)$ be a set of power series which generates I, such that $l(f_1,p) > l(f_2,p) > \ldots > l(f_n,p)$ and $c(f_i,p) = 1, \forall 1 \le i \le n$. Then B is a*

Gröbner basis for I over $D(p)$ if and only if for all $1 \leq i \neq j \leq n$, there exist $a_1^{(ij)}, \ldots, a_n^{(ij)} \in A(p)$ such that

$$S(f_i, f_j) = \sum_{k=1}^{n} a_k^{(ij)} f_k$$

and all monomials m with nonzero coefficient in $a_k^{(ij)}$ satisfy

$$m \cdot l(f_k, p) < \operatorname{lcm}(l(f_i, m), l(f_j, m)).$$

4.2 The Algorithm

The algorithm proposed by Teitelbaum follows two guidelines:

1. It relies only on local data regarding the singularity to be resolved. The estimation of the complexity of the algorithm will be based on local invariants, such as the Milnor number. Although a singularity is described by a power series, some results from the deformation theory show that we can discard all but finitely many coefficients of the power series and still a description of the singularity will be preserved.
2. It uses only arithmetic operations on polynomials over a field, and it does not invoke any algorithms for factoring polynomials. There will be succesive extensions of Artinian rings instead of successive field extensions.

The first three algorithms (Algorithm 7, Algorithm 9, Algorithm 11) are generalizations of standard algorithms for computing the greatest common divisor, square free factorization and primitive elements to polynomials over $D(p)$ when p is a square free polynomial.

Algorithm 7 (EGCD). The *extended gcd* splits $D(p)$ up into factor rings and computes the generator of an ideal (u, v) in $D(p)[x]$.
Input :

$p(T) \in \mathbb{K}$ a square free polynomial
$u = \sum_{i=0}^{m} u_{m-i}x^i$, $v = \sum_{i=0}^{n} v_{n-i}x^i \in \mathbb{K}[T,x]/p(T)$ two polynomials. This list satisfies the conditions:

$$\begin{aligned} p &= \prod_j q_j, \\ q_j & \quad \text{pairwise relatively prime,} \\ u &\equiv a_j w_j \bmod q_j,\ \forall j, \\ v &\equiv b_j w_j \bmod q_j,\ \forall j. \end{aligned}$$

Output :
A list of quadruples (q_j, a_j, b_j, w_j), where $q_j \in \mathbb{K}$ and $a_j, b_j, w_j \in \mathbb{K}[T,x]/p(T)$
Step 0 :
`If` $v = 0$ `Return` $\{(p, 1, 0, u)\}$
Step 1 :
$d := \gcd(p, v_0)$,
Find a, b such that $av_0 + bp = d$,
$r := 0$.
Step 2 :
Using arithmetic mod p/d do:
$t := \deg(u) - \deg(v)$
`While` $t \geq 0$ `do` (Division)
  $u := u - (u_0/v_0)x^t v$
  $r := r + (u_0/v_0)x^t$
  $t := \deg(u) - \deg(v)$
$u' := v$
$v' := u$
$LIST_1 := \mathbf{EGCD}(p/d, u', v')$
$LISTU := \{\}$
`For each` $(q, a, b, d) \in LIST_1$ `do` (Shift)
  $LISTU := LISTU \cup \{(q, b + ar \bmod q, a, d)\}$
$LISTZ := \{\}$
`If` $\deg(d) > 0$
  $LISTZ := \mathbf{EGCD}(d, u \bmod d, q \bmod d)$
Step 3 :
`Return` $LISTU \cup LISTZ$

Lemma 8. *Algorithm 7 uses*

$$\mathcal{O}(deg(p)^2 deg(u) deg(v))$$

field operations.

Algorithm 9 (SQFR). The *square free factorization* computes the square free part of an element $q \in D(p)[x]$ using the algorithm 7.
Input :
$p \in \mathbb{K}[T]$ a square free polynomial.
$q(T, x) \in \mathbb{K}[T, x]$ a polynomial
Output :
A list of pairs (p_i, q_i) such that:

$$\begin{array}{lcl} p & = & \prod_i p_i, \\ p_i & & \text{pairwaise relatively prime,} \\ q_i & & \text{is the square free part of the image of } q \text{ in } D(p_i),\ \forall i. \end{array}$$

Step 0 :
$LIST := \mathbf{EGCD}(p, q, \partial q/\partial x)$
$RESULT := \{\}$
Step 1 :
`For each` $(p_i, a_i, b_i, d_i) \in LIST$
$RESULT := RESULT \cup \{(p_i, a_i)\}$

Step 2 :
`Return` $RESULT$

Lemma 10. *Algorithm 9 uses*

$$\mathcal{O}(deg(p)^2 deg(u) deg(v))$$

field operations.

Algorithm 11 (PrimElem). Finds the *primitive elements* for the ring $D(p)[W]/f$.
Input :
$p(T) \in \mathbb{K}[T]$ a square free polynomial
$F(T, W) \in D(p)[T, W]$ a square free polynomial
Output :
A list of triples $(q_i(T), u_i(T), v_i(T))$ such that the map

$$\begin{array}{rcl} \mathbb{K}/(p(T), f(T, W)) & \mapsto & \prod_i \mathbb{K}[T]/q_i(T), \\ T & \mapsto & u_i(T), \\ W & \mapsto & v_i(T) \end{array}$$

is an isomorphism.

Step 0 :
$c := 0$
$RESULT := \{\}$
Step 1 :
$g(T,Z) := f(T, Z - cT)$
Step 2 :
$r(Z) := res_T(g(T,Z), p(T))$
`If` $r(Z)$ is not square free
$c := c+1$
`Goto` *Step 1*
Step 3 :
$LIST := EGCD(r,p,g)$
Step 4 :
`For each` $(p_i, a_i, b_i, d_i) \in LIST$ `do`
$d_i(Z,T) := a(Z)T - b(Z)$
$u(T) := b(T)/a(T) \bmod p_i(T)$
$RESULT := RESULT \cup \{(p_i(T), u(T), Z - cu(T)\})$
Step 5 :
`Return` $RESULT$

Lemma 12. *Let $p(T) \in \mathbb{K}[T]$ and $f(T,W) \in D(p)[T,W]$ be square free polynomials. Let*

$$n = \dim \mathbb{K}[T,W]/(p,f).$$

Then the algorithm 11 needs

$$\mathcal{O}(n^4)$$

field operations.

Algorithm 13 (BlowUp).
Input :
$p(T) \in \mathbb{K}$ a square free polynomial
$F(x,y) \in D(p)[x,y]$ a square free polynomial with the leading coefficient invertible
Output :
A list of quintuples representing the blowup of $A(p)/F$
Step 0 :
$m := \mathbf{mult}_{(x,y)}(F)$
$\tilde{F}_1 := x^{-m} F(x, xy)$
$f(W) = F_1(0,W)$
Step 1 :

$LIST := SQFF(p(T), f(W))$

Step 2 :

$RESULT := \{\}$

Step 3 :

`For each` $(p_i, f_i) \in LIST$ `do`

$PRIMLIST :=$ **PrimElem**(p_i, f_i)

`For each` $(q_{ij}, u_{ij}, v_{ij}) \in PRIMLIST$ `do`

Let G_{ij} be the polynomial obtained from $\tilde{F}_1$ by substituting:

$$\begin{aligned} T &\mapsto v_{ij}(T), \\ y &\mapsto y + u_{ij}(T), \end{aligned}$$

and computing modulo $q_{ij}(T)$.

$RESULT := RESULT \cup \{(x, G_{ij}, q_{ij}, u_{ij}, v_{ij})\}$

Step 4 :

`If` $a_{0,m} = 0$

$G := y^{-m} F(x, y)$

$RESULT := RESULT \cup \{(y, G, p, 0, T)\}$

Step 5 :

`Return` $RESULT$

Lemma 14. *Let m be the multiplicity of F at the ideal (x, y), let r be the degree of the coefficient polynomial $p(T)$, and let d be the degree of the polynomial F representing the singularity. Then the algorithm 13 requires*

$$\mathcal{O}(m^4 r^4 + m^2 r^2 d^4)$$

field operations.

Algorithm 15 (Resolution).

Input :

p a square free polynomial

$F \in D(p)$ a square free power series

Output :

A list of lists representing the resolution of singularity of F at the origin

Step 0 :

Apply the standard basis algorithm to F and p. The result is a tree of standard bases. From this tree we obtain a factorization $p(T) = \prod p_i(T)$ such that F has constant multiplicity on each factor $A(p_i)$ of $A(p)$, as well as the values of δ_i.

Step 1 :

`For each` i, let G_i be the polynomial obtained by dropping all terms of F_i of total degree greater than or equal to $2\delta + 1$.

Step 2 :

$RESOLUTION := \{\}$

Step 3 :

`For each` i `do`

$RES_i := \{\}$

`If` mult(G_i)¿1

$NEWPTS :=$ **BlowUp**(G_i) (with the Algorithm 13)

`For each` $(w_{ij}, G_{ij}, q_{ij}, u_{ij}, v_{ij}) \in NEWPTS$ `do`

$RES_{ij} :=$ **Resolution**(q_{ij}, G_{ij})

$RES_i := \{(w_{ij}, G_{ij}, q_{ij}, u_{ij}, v_{ij}), RES_{ij}\} \cup RES_i$

$RESOLUTION := RESOLUTION \cup \{p_i, RES_i\}$

Step 4 :

`Return` $Resolution$

Lemma 16. *Let $p \in \mathbb{K}[T]$ be a square free polynomial, $F \in D(p)[[x, y]]$ a power series and let c be the index of the conductor of F. Then the algorithm 15 requires*

$$\mathcal{O}(\deg(p)^2(1 + c^6))$$

field operations.

Algorithm 17 (Adjoints).

Input :

p a square free polynomial

$F \in D(p)$ a square free power series

$RESOLUTION$ the output from the algorithm 15

Output :

A list of linear equations describing the linear subspace of the adjoints

Step 0 :

$RESLIST := \{\}$

Step 1 :

`For each` $\{q, LIST\} \in RESOLUTION$ `do`

$Q := \{(i,j) | x^i y^j \notin I_F\}$

Construct a "generic" polynomial

$$E = \sum_{(i,j)\in Q} \left(\sum_{k=0}^{\deg(q)-1} a_{ijk} T^k \right) x^i y^j$$

so that a_{ijk} are unknowns and $Q \subset A(q)/I_F$

$QLIST :=$ **Reduction**$(E, \{q, LIST\})$ (with the Algorithm 18).

$RESLIST := RESLIST \cup QLIST$

Step 2 :

`Return` $RESLIST$

Algorithm 18 (Reduction).

Input :

E a polynomial in x, y and T with the coefficients a_{ijk} that are linear functions

$\{p, LIST\}$ an element from the resolution list

Output :

A list of linear eqations in a_{ijk}

Step 0 :

$EQLIST := \{\}$

Step 1 :

$m :=$ the multiplicity of the point $\{q, LIST\}$

`If` $m \leq 1$ `return`

`Else`

`For each` term $a_{ijk} T^k x^i y^j$ occuring in E with $i + j < m - 1$ `do`

$EQLIST := EQLIST \cup \{a_{ijk} = 0\}$

Delete this term from E

Step 2 :

`For each` element $\{(w, G, q, u, vP), RES\} \in LIST$ make the change of coordinates represented by the quintuple in E

$E := w^{1-m} E$

`For each` $\{p, LIST\}$

Reduce E mod p

$EQLIST := EQLIST \cup$ **Reduction**$(E \bmod p, \{p, LIST\})$

Step 3 :

`Return` $EQLIST$

Lemma 19. *The algorithm 17 requires*

$$\mathcal{O}(\deg(p)^2 c^6)$$

field operations.

References

[1] J. Teitelbaum, *The Computational Complexity of the Resolution of Plane Curve Singularities*, Mathematics of Computation, vol. 54, number 190, April 1990, 797-837

[2] F. Winkler, *Polynomial Algorithms in Computer Algebra*, Springer-Verlag Wien New York, 1996

[3] B. Buchberger, *Introduction to Gröbner Bases*, The Gröbner Bases Course, Research Institute for Symbolic Computation, Linz, 12-30 January, 1998