

The Calculation of Radical Ideals in Positive Characteristic

Gregor Kemper

IWR, Universität Heidelberg, Im Neuenheimer Feld 368

69 120 Heidelberg, Germany

email `Gregor.Kemper@iwr.uni-heidelberg.de`

June 13, 2002

Abstract

We propose an algorithm for computing the radical of a polynomial ideal in positive characteristic. The algorithm does not involve polynomial factorization.

Introduction

The computation of the radical \sqrt{I} of a given ideal $I \subseteq K[x_1, \dots, x_n]$ in a polynomial ring is one of the basic tasks in computational commutative algebra. For example, radical computation is an ingredient in de Jong's normalization algorithm (see de Jong [18], Decker et al. [8], Matsumoto [22]). Moreover, algorithms for primary decomposition often start by forming the radical ideal (see Becker and Weispfenning [2, Algorithm 8.6]).

A very common approach for computing the radical of I is by reducing to the case where I is zero-dimensional (see Gianni et al. [14], Krick and Logar [20], Alonso et al. [1], Becker and Weispfenning [2]). A nice presentation of this method can be found in Becker and Weispfenning [2, Section 8.7]. The main idea is to choose a subset of the indeterminates which is maximally independent modulo I . After renumbering, let this subset be $\{x_1, \dots, x_d\}$. Then the main step is to pass to the ideal $\tilde{I} := IK(x_1, \dots, x_d)[x_{d+1}, \dots, x_n]$ generated by I in the polynomial ring over the rational function field $K(x_1, \dots, x_d)$, and to compute the radical $\sqrt{\tilde{I}}$. Note that \tilde{I} is zero-dimensional. In Alonso et al. [1] some variants and customizations of this idea can be found, all of which reduce to the zero-dimensional case. Therefore everything hinges on the feasibility of the computation of zero-dimensional radicals over a rational function field over the original ground field. It should be mentioned that Eisenbud et al. [10] gave an algorithm for computing radicals which does not reduce to the zero-dimensional case. However, the limitation of this algorithm is that it requires the ground field K to be of characteristic 0, or that $K[x_1, \dots, x_n]/I$ is generated by elements whose index of nilpotency is less than $\text{char}(K)$ (see Theorem 2.7 in [10]).

The aim of this note is to present a new algorithm for the computation of zero-dimensional radicals which works over any field K which is finitely generated over a perfect field. For the case that K is perfect, there are good algorithms for this purpose given by Seidenberg [24] (see also Becker and Weispfenning [2, Algorithm 8.3]) and Gianni and Mora [13, Algorithm 5.4]. Therefore the main focus here lies on the case where K is not perfect. This case is highly relevant since it occurs whenever we want to compute the radical of a positive-dimensional ideal in positive characteristic (see above). To the best of the author's knowledge, the only algorithm for the computation of zero-dimensional radicals over non-perfect fields which has so far appeared in the literature was given by Gianni et al. [14]. In that paper the authors proposed to compute the radical as the intersection of the associated primes. Consequently, the algorithm requires factorization of (univariate) polynomials over a field which is finitely generated over its prime field. This approach poses several practical problems:

- The problem of factorizing a polynomial over a field which is finitely generated over its prime field is quite hard in terms of the computational cost.
- Although methods to solve the factorization problem were given by Davenport and Trager [6], these methods are quite involved. The author was not able to find a computer algebra system where factorization over arbitrary finitely generated fields is implemented. Hermann [17] also gave methods for factorizing polynomials over a finitely generated field. These methods are similar to those from [6], but they do not cover the case of positive characteristic. (They fail, for example for $f = x^p - t \in \mathbb{F}_p(t, \alpha)[x]$ with $\alpha^p = t$.)
- The factorization method given by Davenport and Trager [6] requires the pre-computation of the squarefree part of a polynomial. This is straight-forward in characteristic 0, but in positive characteristic there may be a problem. In fact, the authors of [6] refer to Davenport [7, p. 182] for this task, but the algorithm given in [7] only works over prime fields.

These problems make it very hard to implement the algorithm suggested by Gianni et al. [14]. In fact, prior to the appearance of the preprint version of this note, there existed no implementation of radical computation in positive characteristic in any of the major computer algebra systems¹. It is therefore highly desirable to find an algorithm for zero-dimensional radical computation over non-perfect fields which avoids factorization. Such an algorithm is given in this paper. As a special case we obtain an algorithm which computes the squarefree part of a polynomial, since $\sqrt{(f)} = (\text{sqrfree}(f))$. This gives an answer to the third point raised above.

Our algorithm is very similar to the one given by Seidenberg [24]. In fact, the problem with applying Seidenberg's method in positive characteristic is that squarefree polynomials do not always remain squarefree if the ground field is extended. Our remedy to this shortcoming is to substitute the squarefree part of a polynomial by the separable part. The difficulty then is that the separable part does not live over the original ground field, but instead over a purely inseparable extension. In Section 1 we give an algorithm which computes the separable part of a polynomial. The separable part is then used in Algorithm 6, the main algorithm for computing zero-dimensional radicals, which is presented in Section 2. This algorithm is probably of worse complexity than the characteristic zero counterpart since it involves additional variables and the computation of an elimination ideal. On the other hand, one may argue that in positive characteristic the coefficient arithmetic becomes cheaper. Algorithm 6 is quite simple, easy to implement, and only uses features that are available in almost all computer algebra systems which support commutative algebra, such as CoCoA [5], MACAULAY (2) [15], MAGMA [3], or SINGULAR [16]. The author has implemented Algorithm 6 in MAGMA. It is planned to include this implementation into the standard distribution of MAGMA soon. An implementation in SINGULAR was written by Gerhard Pfister. This is incorporated into Version 2.0 and beyond of SINGULAR.

While this paper was in the refereeing process, Matsumoto published an article [23] which gives an entirely different algorithm for calculating the radical of an ideal in a polynomial ring in positive characteristic. Therefore a third section has been added to this article which contains a brief discussion of Matsumoto's algorithm and some comparisons of performance.

Acknowledgments. I thank Gerhard Pfister, Lorenzo Robbiano, and Wolmer Vasconcelos for helpful conversations and for their comments on the preprint version of this note. I am also thankful to the anonymous referees for suggesting improvements and pointing out some typing errors.

1 The separable part

Let K be a field and $f \in K[x]$ a non-zero polynomial with coefficients in K . We call f **separable** if f has no multiple roots in a splitting field $L \geq K$ (see Lang [21, Chapter VII, § 4]). This is

¹Now implementations in SINGULAR (Version 2.0) and in MAGMA exist.

equivalent with $\gcd(f, f') = 1$ (see Becker and Weispfenning [2, Proposition 7.33]). If

$$f = c \cdot \prod_{i=1}^m (x - \alpha_i)^{e_i}$$

with $c \in K \setminus \{0\}$ and $\alpha_i \in L$ pairwise distinct roots of f , we write

$$\text{sep}(f) := c \cdot \prod_{i=1}^m (x - \alpha_i) \in L[x]$$

for the **separable part** of f . Notice that this coincides with the squarefree part of f if K is perfect. For $f, g \in K[x]$ not both equal to zero, we denote the monic greatest common divisor by $\gcd(f, g)$. It is well known that in the case $\text{char}(K) = 0$ one has $\text{sep}(f) = f / \gcd(f, f')$. The following algorithm calculates $\text{sep}(f)$ in the case $K = k(t_1, \dots, t_m)$ with t_i indeterminates and k a perfect field of characteristic $p > 0$. We work with the tacit assumption that we can compute p -th roots of elements from k , or at least all those p -th roots that are required in the algorithm. Algorithm 1 is essentially contained in Proposition 3.7.12 from Kreuzer and Robbiano [19]. We present the algorithm and its proof here for the reader's convenience.

Algorithm 1 (Separable part).

Input: A non-zero polynomial $f \in k(t_1, \dots, t_m)[x]$ with k a perfect field of positive characteristic p .

Output: The separable part of f as a polynomial in $k(\sqrt[p]{t_1}, \dots, \sqrt[p]{t_m})[x]$ with q a p -power.

- (1) Set $h := \gcd(f, f')$.
- (2) Set $g_1 := f/h$.
- (3) Set $\tilde{h} := \gcd(h, h')$.
- (4) If $\tilde{h} = h$, go to (6).
- (5) Set $h := \tilde{h}$ and go to (3).
- (6) If $h = 1$ then return g_1 .
- (7) Write $h = u(x^p)$ with $u \in k(t_1, \dots, t_m)[x]$. (This is possible since $h' = 0$.)
- (8) Form $v \in k(\sqrt[p]{t_1}, \dots, \sqrt[p]{t_m})[x]$ from u by replacing every t_i occurring in u with $\sqrt[p]{t_i}$ and every $a \in k$ in u with $\sqrt[p]{a} \in k$. (Thus $v^p = h$.)
- (9) Compute $g_2 := \text{sep}(v)$ by a recursive call.
- (10) Compute $g_3 := \text{sep}(g_1 g_2)$ by a recursive call and return g_3 .

Remark 2. The computation of greatest common divisors in steps (1) and (3) of the algorithm can be performed by the Euclidean algorithm (see Geddes et al. [12, Section 2.4]), and is therefore much cheaper than factorization of the polynomials. \triangleleft

Proposition 3. Algorithm 1 terminates after finitely many steps and correctly calculates $\text{sep}(f)$.

Proof. We proceed by induction on the (x) -degree of f .

Choose a field extension $L \geq K := k(t_1, \dots, t_m)$ containing all roots of f and write

$$f = c \cdot \prod_{i=1}^r (x - \alpha_i)^{d_i} \cdot \prod_{i=1}^s (x - \beta_i)^{pe_i},$$

where r , s , d_i and e_i are non-negative integers with $0 < d_i < p$ and $e_i > 0$, and c , α_i and β_i are elements in L with $\alpha_i \neq \alpha_j$ and $\beta_i \neq \beta_j$ if $i \neq j$. Thus

$$f' = \sum_{i=1}^r d_i \frac{f}{x - \alpha_i}$$

and

$$\gcd(f, f') = \prod_{i=1}^r (x - \alpha_i)^{d_i - 1} \cdot \prod_{i=1}^s (x - \beta_i)^{pe_i}.$$

The polynomial g_1 formed in line (2) of the algorithm is therefore

$$g_1 = c \cdot \prod_{i=1}^r (x - \alpha_i),$$

and the polynomial h in line (6) is

$$h = \prod_{i=1}^s (x - \beta_i)^{pe_i}.$$

Hence the algorithm is correct if $s = 0$. On the other hand, if $s > 0$, then v in line (8) is

$$v = \prod_{i=1}^s (x - \beta_i)^{e_i},$$

and $\deg(v) < \deg(h) \leq \deg(f)$. By induction, the recursive call in line (9) terminates and yields

$$g_2 = \prod_{i=1}^s (x - \beta_i).$$

We have $\deg(g_1 g_2) = r + s < r + ps \leq \deg(f)$, hence the recursive call in line (10) yields

$$g_3 = \text{sep}(g_1 g_2) = \text{sep}(f).$$

□

2 Zero-dimensional radicals

The goal of this section is to give an algorithm for the computation of the radical of a zero-dimensional ideal in a polynomial ring. The following proposition is Lemma 92 in Seidenberg [24] (see also Becker and Weispfenning [2, Lemma 8.13]).

Proposition 4. *Let $I \trianglelefteq K[x_1, \dots, x_n]$ be an ideal in a polynomial ring over a field K . If $I \cap K[x_i]$ contains a separable polynomial for each $i = 1, \dots, n$, then $I = \sqrt{I}$.*

Remark 5. Krick and Logar [20] state Proposition 4 with “separable” replaced by “squarefree”, but this does not hold in positive characteristic. A counter example is given by the ideal $I = (x_1^p - t, x_2^p - t) \trianglelefteq \mathbb{F}_p(t)[x_1, x_2]$, since $x_1 - x_2 \in \sqrt{I} \setminus I$ (see Becker and Weispfenning [2, Example 8.16]).

◁

Suppose that $I \trianglelefteq K[x_1, \dots, x_n]$ is a zero-dimensional ideal (i.e., $K[\underline{x}]/I$ has Krull dimension 0). Then $I \cap K[x_i] \neq 0$ for all i (see, for example, Eisenbud [9, Corollaries 2.15 and 9.1]). One can compute the elimination ideals $I \cap K[x_i]$ by using Gröbner bases with respect to appropriate

monomial orderings (see Becker and Weispfenning [2, Algorithm 6.1]). A more efficient algorithm for the computation of a non-zero $f_i \in I \cap K[x_i]$, which requires only one Gröbner basis computation with respect to an arbitrary monomial ordering and linear algebra, was given by Faugère et al. [11]. The following algorithm computes the radical of a zero-dimensional ideal over $k(t_1, \dots, t_m)$ with k a perfect field of positive characteristic. This algorithm extends Algorithm 8.3 in Becker and Weispfenning [2] and Corollary 3.7.16 in Kreuzer and Robbiano [19].

Algorithm 6 (Zero-dimensional radical).

Input: A zero-dimensional ideal $I \subseteq K[x_1, \dots, x_n]$ in a polynomial ring over the rational function field $K = k(t_1, \dots, t_m)$ with k a perfect field of positive characteristic p .

Output: \sqrt{I} .

- (1) For $i \in \{1, \dots, n\}$, find a non-zero $f_i \in I \cap K[x_i]$.
- (2) For each i , compute $\text{sep}(f_i) \in k(\sqrt[p^{r_i}]{t_1}, \dots, \sqrt[p^{r_i}]{t_m})[x_i]$ by using Algorithm 1.
- (3) For each i , write $\text{sep}(f_i) = g_i(\sqrt[q]{t_1}, \dots, \sqrt[q]{t_m}, x_i)$, where $q := p^r$, $r := \max\{r_1, \dots, r_n\}$, and $g_i \in K[y_1, \dots, y_m, x_i]$ with new indeterminates y_1, \dots, y_m .
- (4) Form the ideal

$$J := IK[y_1, \dots, y_m, x_1, \dots, x_n] + (g_1, \dots, g_n) + (y_1^q - t_1, \dots, y_m^q - t_m) \\ \subseteq K[y_1, \dots, y_m, x_1, \dots, x_n].$$

- (5) Calculate the elimination ideal

$$\tilde{J} := J \cap K[x_1, \dots, x_n]$$

(by using Algorithm 6.1 in Becker and Weispfenning [2]) and return \tilde{J} .

Theorem 7. *Algorithm 6 is correct.*

Proof. Set

$$L := K[y_1, \dots, y_m]/(y_1^q - t_1, \dots, y_m^q - t_m) \cong k(\sqrt[q]{t_1}, \dots, \sqrt[q]{t_m}).$$

The canonical projection $K[y_1, \dots, y_m] \rightarrow L$ induces a map

$$\varphi: K[y_1, \dots, y_m, x_1, \dots, x_n] \rightarrow L[x_1, \dots, x_n],$$

and $\varphi(g_i)$ maps to $\text{sep}(f_i)$ under the isomorphism $L \cong k(\sqrt[q]{t_1}, \dots, \sqrt[q]{t_m})$. The restriction of φ to $K[x_1, \dots, x_n]$ is injective, so we may view $K[x_1, \dots, x_n]$ as a subring of $L[x_1, \dots, x_n]$. Consider the ideal

$$\bar{J} := IL[x_1, \dots, x_n] + (\varphi(g_1), \dots, \varphi(g_n)) \subseteq L[x_1, \dots, x_n],$$

where $IL[x_1, \dots, x_n]$ denotes the ideal in $L[x_1, \dots, x_n]$ generated by I . It follows from Proposition 4 that $\bar{J} = \sqrt{IL[x_1, \dots, x_n]}$. Consider the composition

$$K[y_1, \dots, y_m, x_1, \dots, x_n] \xrightarrow{\varphi} L[x_1, \dots, x_n] \rightarrow L[x_1, \dots, x_n]/\bar{J}.$$

The kernel of this composition is clearly J (as defined in step (4) of Algorithm 6), and hence for

$$\psi: K[x_1, \dots, x_n] \rightarrow K[y_1, \dots, y_m, x_1, \dots, x_n] \xrightarrow{\varphi} L[x_1, \dots, x_n] \rightarrow L[x_1, \dots, x_n]/\bar{J},$$

where the first map is the inclusion, we have $\ker(\psi) = J \cap K[x_1, \dots, x_n] = \tilde{J}$. Since ψ is the same as the inclusion $K[x_1, \dots, x_n] \subseteq L[x_1, \dots, x_n]$ followed by reduction modulo \bar{J} , we obtain $\tilde{J} = \bar{J} \cap K[x_1, \dots, x_n]$. Therefore

$$\tilde{J} = K[x_1, \dots, x_n] \cap \sqrt{IL[x_1, \dots, x_n]} = \sqrt{K[x_1, \dots, x_n] \cap IL[x_1, \dots, x_n]} = \sqrt{I},$$

where Proposition 2.6.12 in Kreuzer and Robbiano [19] was used for the last equality. \square

Remark 8. (a) Suppose that K is a field of positive characteristic which is finitely generated over a perfect field k . Then K can be written as

$$K = k(t_1, \dots, t_m)[z_1, \dots, z_r]/\mathfrak{m}$$

with $\mathfrak{m} \trianglelefteq k(t_1, \dots, t_m)[z_1, \dots, z_r]$ a maximal ideal. If K is given as the field of fractions of a finitely presented algebra over k , it is not hard to get K into the above form. Let x_1, \dots, x_n be further indeterminates and consider the canonical epimorphism

$$\varphi: k(t_1, \dots, t_m)[z_1, \dots, z_r, x_1, \dots, x_n] \rightarrow K[x_1, \dots, x_n].$$

If an ideal $I \trianglelefteq K[x_1, \dots, x_n]$ is given by $I = (\varphi(g_1), \dots, \varphi(g_l))$ with $g_i \in k(t_1, \dots, t_m)[z_1, \dots, z_r, x_1, \dots, x_n]$, then it is easy to see that

$$\sqrt{I} = \varphi\left(\sqrt{\mathfrak{m} + (g_1, \dots, g_l)}\right).$$

If moreover I is zero-dimensional, then the same is true for $\mathfrak{m} + (g_1, \dots, g_l)$, hence we can calculate the radical ideal on the right hand side by Algorithm 6.

- (b) If K is any field of characteristic 0, then the radical of a zero-dimensional ideal $I \trianglelefteq K[x_1, \dots, x_n]$ can be computed by Algorithm 8.3 in Becker and Weispfenning [2]. \triangleleft

Example 9. (a) Let us examine the example $I = (x_1^p - t, x_2^p - t) \trianglelefteq \mathbb{F}_p(t)[x_1, x_2]$ (see Remark 5). We have

$$\text{sep}(x_i^p - t) = x_i - \sqrt[p]{t},$$

so in step (4) of Algorithm 6 we obtain the ideal

$$J = (x_1 - y, x_2 - y, y^p - t) \trianglelefteq \mathbb{F}_p(t)[x_1, x_2, y].$$

We choose the lexicographical monomial ordering with $y > x_1 > x_2$ on $\mathbb{F}_p(t)[x_1, x_2, y]$. By replacing $x_1 - y$ and $y^p - t$ by their normal forms with respect to $x_2 - y$, we obtain the new basis

$$G = \{x_1 - x_2, x_2 - y, x_2^p - t\}.$$

G is a Gröbner basis since the polynomials in G have pairwise coprime leading monomials. Hence step (5) of Algorithm 6 yields

$$\sqrt{I} = J \cap \mathbb{F}_p(t)[x_1, x_2] = (x_1 - x_2, x_2^p - t),$$

which is the correct result.

- (b) We can also use Algorithm 6 to compute the squarefree part $\text{sqrfree}(f)$ of a polynomial f , since $\sqrt{(f)} = (\text{sqrfree}(f))$. Consider the example

$$f = x^{2p} - (t^p + t)x^p + t^{p+1} = (x^p - t)(x^p - t^p) \in \mathbb{F}_p(t)[x].$$

In step (4) of Algorithm 6 we obtain the ideal

$$J = ((x - y)(x - y^p), y^p - t).$$

It is not hard to see that a Gröbner basis with respect to the lexicographical monomial ordering with $y > x$ is

$$\{y^p - t, (y - x)(x - t), (x^p - t)(x - t)\}.$$

Thus $\sqrt{(f)} = ((x^p - t)(x - t))$, and we obtain $\text{sqrfree}(f) = (x^p - t)(x - t)$. Of course the algorithm runs without “knowing” the factorization of f . \triangleleft

3 Matsumoto’s algorithm and some running times

As mentioned at the end of the Introduction, an alternative algorithm for computing radical ideals in positive characteristic appeared while this paper was in the refereeing process. This algorithm was given by Matsumoto [23]. The central idea in Matsumoto’s approach is to consider the endomorphism φ on $K[X_1, \dots, X_n]$ given by $f \mapsto f^p$ where $p = \text{char}(K)$. Matsumoto’s algorithm iteratively replaces I by its preimage $\varphi^{-1}(I)$ until $I = \varphi^{-1}(I)$. The main part of the algorithm is the computation of $\varphi^{-1}(I)$, which for K finitely generated over a perfect field can be done by calculating a certain elimination ideal. The most remarkable feature of Matsumoto’s algorithm is that it is not restricted to the zero-dimensional case. Thus for higher-dimensional ideals the standard reduction to dimension zero (see in the Introduction) is not necessary, which certainly makes Matsumoto’s algorithm more elegant. On the other hand, the computation of $\varphi^{-1}(I)$ becomes increasingly expensive for large p .

Matsumoto [23] gave a number of benchmark tests for his algorithm, which were taken from Caboara et al. [4]. Of course the examples from [4] are all in characteristic zero, and Matsumoto reduced them modulo various primes. For getting a fair comparison of the performance of Matsumoto’s algorithm and the algorithm from this paper, we used exactly the same benchmark tests, and added one more. This additional test is given by the principal ideal generated by

$$\mathbf{D}: x_4^2 + 4x_1^3x_3 - x_1^2x_2^2 - 18x_1x_2x_3 + 4x_2^3 + 27x_3^2,$$

which is the relation satisfied by the square root of the discriminant of a general polynomial of degree 3. Of course this ideal is already radical except in characteristic 2. The other test ideals are labeled **E2**, **E3**, **L**, **M**, **8₃**, and **C**. We do not reprint them here, since they can be found in Matsumoto [23] or Caboara et al. [4]. In order to obtain meaningful running times, the author implemented Matsumoto’s algorithm in MAGMA, and compared the timings with the MAGMA implementation of the algorithm from this paper. It should be noted that Algorithm 6 of this paper was only used in its pure form for the examples in dimension zero (e.i., **E2** and **E3**), whereas in positive dimension the somewhat cumbersome reduction technique from Becker and Weispfenning [2, Section 8.7] (with Algorithm 6 as its core) was used. The computation times (in seconds) are given in Table 1. All computations were done on a SUN workstation with a 440MHz Ultrasparc processor and 512 MB of memory. The entry “infeasible” in Table 1 signifies that after about half an hour of computing the algorithm was not even near completion. Every computation was attempted in the characteristics 2, 3, 5, 7, 11, 53, and 251, following the standard of [23]. If a characteristic does not appear in Table 1, this means that none of the algorithms was feasible.

We finish with three observations on the timings:

- Our timings for Matsumoto’s algorithm are generally somewhat smaller than the ones given in [23], except for **L** in characteristic 2. But the ratios between the timings for different examples roughly correspond to those obtained in [23].
- For ideals of positive dimension, Matsumoto’s algorithm is usually better, except for large characteristics.
- For zero-dimensional ideals, the algorithm of this paper performs better. The difference in performance becomes larger as the characteristic grows.

References

- [1] Maria Emilia Alonso, Teo Mora, Mario Raimondo, *Local Decomposition Algorithms*, in: Shojiro Sakata, ed., *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (AAECC-8)*, Lect. Notes Comput. Sci. **508**, pp. 208–221, Springer-Verlag, Berlin, Heidelberg, New York 1991.

- [2] Thomas Becker, Volker Weispfenning, *Gröbner Bases*, Springer-Verlag, Berlin, Heidelberg, New York 1993.
- [3] Wieb Bosma, John J. Cannon, Catherine Playoust, *The Magma Algebra System I: The User Language*, J. Symbolic Comput. **24** (1997), 235–265.
- [4] Massimo Caboara, Pasqualina Conti, Carlo Traverso, *Yet Another Algorithm for Ideal Decomposition*, in: Harold F. Mattson, Teo Mora, eds., *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (AAECC-12)*, Lect. Notes Comput. Sci. **1255**, pp. 39–54, Springer-Verlag, Berlin, Heidelberg, New York 1997.
- [5] A. Capani, G. Niesi, L. Robbiano, *CoCoA: A System for Doing Computations in Commutative Algebra*, available via anonymous ftp from cocoa.dima.unige.it, 2000.
- [6] James H. Davenport, Barry M. Trager, *Factorization over Finitely Generated Fields*, in: *Symbolic and Algebraic Computation, Snowbird/Utah 1981*, pp. 200–205, ACM, 1981.
- [7] James Harold Davenport, *On the Integration of Algebraic Functions*, Lect. Notes Comput. Sci. **102**, Springer-Verlag, Berlin, Heidelberg, New York 1981.
- [8] Wolfram Decker, Theo de Jong, Gert-Martin Greuel, Gerhard Pfister, *The Normalization: a New Algorithm, Implementation and Comparisons*, in: P. Dräxler, G.O. Michler, C.M. Ringel, eds., *Computational Methods for Representations of Groups and Algebras, Euroconference in Essen, Germany, April 1-5, 1997*, Prog. Math. **173**, pp. 177–185, Birkhäuser, Basel 1999.
- [9] David Eisenbud, *Commutative Algebra with a View Toward Algebraic Geometry*, Springer-Verlag, New York 1995.
- [10] David Eisenbud, Craig Huneke, Wolmer V. Vasconcelos, *Direct Methods for Primary Decomposition*, Invent. Math. **110** (1992), 207–235.
- [11] J. C. Faugère, P. Gianni, D. Lazard, T. Mora, *Efficient Computation of Zero-dimensional Gröbner Bases by Change of Ordering*, J. Symbolic Comput. **16** (1993), 329–344.
- [12] K. O. Geddes, S. R. Czapor, G. Labahn, *Algorithms for Computer Algebra*, Kluwer, Boston, Dordrecht, London 1992.
- [13] Patrizia Gianni, Teo Mora, *Algebraic Solution of Systems of Polynomial Equations Using Gröbner Bases*, in: L. Huguët, A. Poli, eds., *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (AAECC-5)*, Lect. Notes Comput. Sci. **356**, pp. 247–257, Springer-Verlag, Berlin, Heidelberg, New York 1989.
- [14] Patrizia Gianni, Barry Trager, Gail Zacharias, *Gröbner Bases and Primary Decomposition of Polynomial Ideals*, J. Symbolic Comput. **6** (1988), 149–267.
- [15] Daniel R. Grayson, Michael E. Stillman, *Macaulay 2, a Software System for Research in Algebraic Geometry*, available at <http://www.math.uiuc.edu/Macaulay2>, 1996.
- [16] Gert-Martin Greuel, Gerhard Pfister, Hannes Schönemann, *Singular Version 1.2 User Manual*, Reports On Computer Algebra **21**, Centre for Computer Algebra, University of Kaiserslautern, 1998, available at <http://www.mathematik.uni-kl.de/~zca/Singular>.
- [17] Grete Hermann, *Die Frage der endlich vielen Schritte in der Theorie der Polynomideale*, Math. Ann. **95** (1926), 736–788.
- [18] Theo de Jong, *An Algorithm for Computing the Integral Closure*, J. Symbolic Comput. **26** (1998), 273–277.

- [19] Martin Kreuzer, Lorenzo Robbiano, *Computational Commutative Algebra 1*, Springer-Verlag, Berlin 2000.
- [20] Teresa Krick, Alessandro Logar, *An Algorithm for the Computation of the Radical of an Ideal in the Ring of Polynomials*, in: Harold F. Mattson, Teo Mora, T. R. N. Rao, eds., *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (AAECC-9)*, Lect. Notes Comput. Sci. **539**, pp. 195–205, Springer-Verlag, Berlin, Heidelberg, New York 1991.
- [21] Serge Lang, *Algebra*, second edn., Addison-Wesley, Redwood City, California 1984.
- [22] Ryutaroh Matsumoto, *On Computing the Integral Closure*, Comm. in Algebra **28** (2000), 401–405.
- [23] Ryutaroh Matsumoto, *Computing the Radical of an Ideal in Positive Characteristic*, J. Symbolic Comput. **32** (2001), 263–271.
- [24] A. Seidenberg, *Constructions in Algebra*, Trans. Amer. Math. Soc. **197** (1974), 273–313.

Example	characteristic	dimension	Matsumoto	this paper
E2	2	0	0.23	0.06
E2	3	0	1.18	0.09
E2	5	0	3.79	0.19
E2	7	0	3.63	0.22
E2	11	0	2.71	0.21
E2	53	0	10.25	0.22
E2	251	0	infeasible	0.23
E3	2	0	0.33	0.09
E3	3	0	1.59	0.16
E3	5	0	3.08	0.20
E3	7	0	3.19	0.23
E3	11	0	2.99	0.24
E3	53	0	25.0	0.27
E3	251	0	infeasible	0.26
L	2	7	477.53	infeasible
M	2	1	0.16	0.26
M	3	1	0.18	0.33
M	5	1	0.18	0.33
M	7	1	0.21	0.32
M	11	1	0.20	0.33
M	53	1	0.21	0.30
M	251	1	1.44	0.30
8₃	2	5	6.87	infeasible
8₃	3	5	15.800	infeasible
C	2	5	4.89	infeasible
C	3	5	49.83	infeasible
D	2	3	0.01	0.04
D	3	3	0.00	0.03
D	5	3	0.16	0.03
D	7	3	1.31	0.03
D	11	3	33.37	0.03
D	53	3	infeasible	0.03
D	251	3	infeasible	0.03

Table 1: Timings for Matsumoto's algorithm [23] and the algorithm in this paper.