Fast modular algorithms for squarefree factorization and Hermite integration

JÜRGEN GERHARD
Universität-GH Paderborn
Fachbereich Mathematik / Informatik
D-33095 Paderborn, Germany
jngerhar@uni-paderborn.de

April 20, 1999

Abstract

We present new modular algorithms for the squarefree factorization of a primitive polynomial in $\mathbb{Z}[x]$ and for computing the rational part of the integral of a rational function in $\mathbb{Q}[x]$. We analyze both algorithms with respect to classical and fast arithmetic and argue that the latter variants are—up to logarithmic factors—asymptotically optimal. Even for classical arithmetic, the integration algorithm is faster than previously known methods.

1 Introduction

The problem of rational function integration that we discuss in this paper is, given two nonzero polynomials $f, g \in \mathbb{Z}[x]$, to compute $\int(f/g)$. Most undergraduate calculus textbooks contain a solution, which Bronstein (1997) attributes to Johann Bernoulli (1703), by factoring the denominator g into linear polynomials over the complex numbers (or at most quadratic polynomials over the real numbers) and performing a partial fraction decomposition. For symbolic computation, this approach is inefficient since it involves polynomial factorization and computation with algebraic numbers, and the algorithms implemented in most computer algebra systems pursue a different approach due to Hermite (1872).

The idea is to extract the "largest possible rational part" from the integral $\int(f/g)$, i.e., to find polynomials $a, b, c, d \in \mathbb{Q}[x]$ such that

$$\int \frac{f}{g} = \frac{c}{d} + \int \frac{a}{b}, \tag{1}$$

where $\deg a < \deg b$ and the numerator b is of minimal degree. It turns out that this condition is equivalent to b being squarefree and coprime to a, and that the solution is unique if we stipulate that in addition b and d are monic and c and d are coprime. Hermite's algorithm only requires the squarefree decomposition of g, which is much easier to compute than the irreducible factorization, and the partial fraction decomposition along this squarefree factorization, and it involves only rational arithmetic. Algebraic numbers may solely occur in the subsequent computation of $\int(a/b)$, and the methods of Rothstein (1976, 1977) and Trager (1976) and of Lazard & Rioboo (1990) and Trager (unpublished) do this in an efficient way with as small an algebraic extension as possible. We do not discuss this last step here.

We present new modular algorithms for squarefree factorization in $\mathbb{Q}[x]$ and for computing the decomposition (1). The idea is to choose several single precision primes, execute a known algorithm modulo each prime, and then obtain the result in $\mathbb{Q}[x]$ by Chinese remaindering and rational number reconstruction. The main contributions of this paper are estimates of the number of "bad" primes, for which the modular computation might not work, and of the number of "good" primes required to reconstruct the result, and a cost analysis in terms of processor operations on machine words. More precisely, if $0 < \deg g = n \leq m$, $\deg f < m$, and the coefficients of f and g are of word size at most d, then we show that our algorithm for the squarefree factorization of g takes $O(n^3 + nd^2)$ word operations with classical arithmetic and $O^{\sim}(n^2 + nd)$ with fast arithmetic, where the $O^{\sim}$-notation ignores logarithmic factors and we neglect the cost for finding our prime moduli. Similarly, our algorithm for Hermite integration of f/g takes $O(m^3(n^2 + d^2))$ word operations with classical arithmetic and $O^{\sim}(m^2(n + d))$ with fast arithmetic. When using fast arithmetic, we argue that both algorithms are—up to logarithmic factors—asymptotically optimal. In a weak sense, namely when we assume that we have precomputed a list of suitable single precision primes, our algorithms are of Las Vegas type.

The direct way of implementing squarefree factorization and Hermite integration is to perform all computations in $\mathbb{Z}[x]$ or $\mathbb{Q}[x]$ and use modular algorithms only for the gcd computations involved. We believe that our algorithms are superior to this approach, already for medium size inputs, for several reasons. The all-modular approach avoids the well-known phenomenon of intermediate expression swell. In the case of Hermite integration, it delays arithmetic on rational numbers, which is expensive in practice, until the very last step. Also, it has the overhead for modular reduction and Chinese remaindering only once in the beginning and in the end of the algorithm, respectively.

We briefly comment on relations to previous work. Already Gauß (1863), article 368, contains an algorithm for computing the squarefree part. The first "modern" works on squarefree factorization are—among others—Tobey (1967), Horowitz (1969, 1971), Musser (1971), and Yun (1976, 1977). The latter papers contain the fastest currently known algorithm when counting arithmetic operations in the coefficient field, and also an estimate of the number of word operations that it uses. This bound is not so easy to compare with ours since it is expressed in terms of a different set of parameters; we briefly comment on this in Remark 12 at the end of Section 3.

Classical works on rational function integration are due to Ostrogradsky (1845) and Hermite (1872). Horowitz (1969, 1971), Mack (1975), and Yun (1977) have given and analyzed algorithms for computing the decomposition (1); Bronstein (1997) gives a nice overview and more references. Horowitz proves a time bound that is essentially proportional to n^5d^2 when $m = n$. Even for classical arithmetic, our estimate improves this by about two orders of magnitude when $n \approx d$. Rothstein (1976, 1977), Trager (1976), and Lazard & Rioboo (1990) and Trager (unpublished) gave algorithms for computing the logarithmic part $\int(a/b)$ of the integral of f/g. Variants of the latter algorithms and of Hermite integration form the basic ingredients of the symbolic integration routines of almost all computer algebra systems. Their software relies on Risch's (1969, 1970) famous algorithm for the integration of elementary functions, which are built from the rational functions and exp and log by the usual arithmetic operations and composition; Bronstein (1997) gives a comprehensive treatment.

The remainder of the paper is organized as follows. In Section 2, we recall and prove some technical ingredients about basic integer and polynomial arithmetic that are used in later sections. This section may be skipped at first reading and particular definitions or facts can be looked up later if necessary. Section 3 contains the modular algorithm for squarefree factorization and its analysis, and Section 4 discusses and analyzes the modular algorithm for Hermite integration.

2 Technical preliminaries

We denote by θ the word size of our processor, i.e., the number of bits that a machine word occupies; common values are 32 or 64. The *word length* of a nonzero integer $a \in \mathbb{Z}$ is the number of machine words that it occupies in memory, namely $\lfloor (\log_2 |a|)/\theta \rfloor + 1$. A nonnegative integer a is called *single precision* if it fits into one machine word, so that $0 \le a < 2^\theta$. In our cost estimates, we count *word operations*, which the reader may imagine as processor instructions, operating on single precision data. Moreover, we let M be a multiplication time for integers and polynomials, so that integers of word length at most n and polynomials of degree at most n can be multiplied using $O(\mathsf{M}(n))$ word operations or coefficient additions and multiplications, respectively. Using classical arithmetic corresponds to $\mathsf{M}(n) = n^2$, Karatsuba's (1962) algorithm has $\mathsf{M}(n) = n^{1.59}$, and the asymptotically fastest currently known algorithm of Schönhage & Strassen (1971), which is based on the Fast Fourier Transform, leads to $\mathsf{M}(n) = n \log n \operatorname{loglog} n$. We assume that M is *subadditive*, so that $\mathsf{M}(m) + \mathsf{M}(n) \le \mathsf{M}(m+n)$, and that $m\,\mathsf{M}(n) \le \mathsf{M}(mn) \le m^2\mathsf{M}(n)$, for all $m, n \in \mathbb{N}$.

We will freely use the following facts, whose proofs or references can be found in Aho, Hopcroft & Ullman (1974) or von zur Gathen & Gerhard (1999).

Fact 1 *(Basic arithmetic) For integers of word length at most n or polynomials of degree at most n, we have the following cost estimates in word operations or arithmetic operations on coefficients, respectively.*

(i) Division with remainder: $O(\mathsf{M}(n))$,

(ii) Extended Euclidean Algorithm: $O(\mathsf{M}(n) \log n)$,

(iii) Reduction modulo n single precision integers: $O(\mathsf{M}(n) \log n)$,

(iv) Chinese Remainder Algorithm for n pairwise coprime single precision integers: $O(\mathsf{M}(n) \log n)$.

In fact, when $\mathsf{M}(n) \in \Omega(n^{1+\varepsilon})$ for some positive ε, as for the classical and Karatsuba's (1962) algorithm, then the cost for (iii) and (iv) is $O(\mathsf{M}(n))$ as well. In particular, the cost for all four tasks is $O(n^2)$ when using classical arithmetic.

The following fact can be found in von zur Gathen & Gerhard (1999), Theorem 5.31 and Exercise 10.18.

Fact 2 *(Partial fraction decomposition) Let F be a field, $f, g \in F[x]$ nonzero and coprime with* $\deg f < \deg g = n$, *and* $g = g_1^{e_1} \cdots g_t^{e_t}$ *for nonconstant and pairwise coprime polynomials $g_1, \ldots, g_t \in F[x]$ and positive integers $e_1, \ldots, e_t$. Then we can compute the partial fraction decomposition*

$$\frac{f}{g} = \sum_{1 \leq i \leq t} \sum_{1 \leq j \leq e_i} \frac{\gamma_{ij}}{g_i^j},$$

i.e., the unique polynomials $\gamma_{ij} \in F[x]$ of degree less than $\deg g_i$ *for all i, j, using $O(n^2)$ arithmetic operations in F with classical arithmetic and $O(\mathsf{M}(n) \log n)$ with fast arithmetic.*

A *single precision prime* is a prime p that fits precisely into one machine word, so that $2^{\theta-1} < p < 2^{\theta}$. For a proof of the following fact and references, see von zur Gathen & Gerhard (1999), Section and Notes 5.10.

Fact 3 *(Rational number reconstruction) Given n pairwise coprime single precision primes $p_1, \ldots, p_s$, an integer c with $2|c| < p_1 \cdots p_s$, and a bound $C < p_1 \cdots p_s$, we can decide whether there exist coprime integers a, b such that $|a| < C$, $0 < b < p_1 \cdots p_s / 2C$, none of the primes divides b, and $ab^{-1} \equiv c \bmod p_i$ for $1 \leq i \leq s$, and if so, compute these unique a, b. This amounts to essentially one application of the Chinese Remainder Algorithm and one application of the Extended Euclidean Algorithm, so that it takes $O(s^2)$ word operations with classical arithmetic, and $O(\mathsf{M}(s) \log s)$ with fast arithmetic.*

For a polynomial $f = \sum_i f_i x^i \in \mathbb{Z}[x]$ or a vector $f = (f_1, \ldots, f_n) \in \mathbb{Z}^n$, we denote by $\|f\|_\infty = \max_i |f_i|$ its *max-norm*, by $\|f\|_2 = (\sum_i f_i^2)^{1/2}$ its *Euclidean norm*, and by $\|f\|_1 = \sum_i |f_i|$ its *one-norm*. The *formal derivative* of a polynomial f as above is $f' = \sum_i i f_i x^{i-1}$. Proofs and references for the following two facts can be found in Mignotte (1989), Theorème IV.4.4, and von zur Gathen & Gerhard (1999), Theorems 16.6 and 6.23 and Corollaries 6.20 and 14.25.

Fact 4 *Let $f, g \in \mathbb{Z}[x]$ be nonconstant of degrees at most n, and $A \in \mathbb{Z}^{n \times n}$.*

(i) *(Mignotte's bound) If $f = f_1 \cdots f_t$, where $f_1, \ldots, f_t \in \mathbb{Z}[x]$ are nonconstant polynomials, then*

$$\|f_1\|_\infty \cdots \|f_t\|_\infty \leq \|f_1\|_2 \cdots \|f_t\|_2 \leq \|f_1\|_1 \cdots \|f_t\|_1 \leq (n+1)^{1/2} 2^n \|f\|_\infty.$$

(ii) *(Hadamard's inequality) If $a_1, \ldots, a_n \in \mathbb{Z}^n$ are the rows (or columns) of A, then $|\det A| \leq \|a_1\|_2 \cdots \|a_n\|_2 \leq n^{n/2} \|a_1\|_\infty \cdots \|a_n\|_\infty$.*

(iii) *(Resultant bound) $|\mathrm{res}(f,g)| \leq (\|f\|_2 \|g\|_2)^n \leq ((n+1)\|f\|_\infty \|g\|_\infty)^n$.*

Fact 5 *Let R be a UFD and $f, g \in R[x]$ nonconstant polynomials.*

(i) *$\gcd(f,g)$ is nonconstant if and only if $\mathrm{res}(f,g) = 0$ in R.*

(ii) *g is squarefree (over the field of fractions of R) $\iff$ $\gcd(g, g')$ is constant $\iff$ $\mathrm{res}(g, g') \neq 0$.*

Fact 6 *For an integer $\theta \geq 7$, the number π_θ of primes between $2^{\theta-1}$ and 2^θ is at least*

$$\pi_\theta > \frac{2^\theta \log_2 e}{2\theta} \left(1 - \frac{1 + \log_2 e}{\theta - 1}\right).$$

For example, we have $\pi_{32} > 8.9 \cdot 10^8$ and $\pi_{64} > 1.99 \cdot 10^{18}$.

Proof. By the prime number theorem, the number $\pi(x)$ of primes less than $x \in \mathbb{R}_{>0}$ is about $x/\ln x$, and more precisely

$$\frac{x}{\ln x}\left(1 + \frac{1}{2\ln x}\right) < \pi(x) < \frac{x}{\ln x}\left(1 + \frac{3}{2\ln x}\right) \text{ if } x \geq 59$$

(Rosser & Schoenfeld 1962). Using this for $x = 2^\theta$ and $x = 2^{\theta-1}$, we obtain, after some calculation, the claim about $\pi_\theta = \pi(2^\theta) - \pi(2^{\theta-1})$. $\square$

A similar result is in Theorem 1.8 of Giesbrecht (1993).

3 Modular algorithm for squarefree factorization

Let $g \in \mathbb{Z}[x]$ be primitive of degree $n \in \mathbb{N}_{\geq 1}$, and assume that g is *normalized*, so that its leading coefficient is positive. Then g has a unique decomposition

$$g = \prod_{1 \leq i \leq n} g_i^i$$

into normalized primitive polynomials $g_1, \ldots, g_n \in \mathbb{Z}[x]$ such that each g_i is squarefree and $\gcd(g_i, g_j) = 1$ if $i \neq j$. We call this the *primitive squarefree decomposition* of g.

Let $p \in \mathbb{Z}$ be an odd prime not dividing the leading coefficient $b = \mathrm{lc}(g)$, and

$$g \equiv b \prod_{1 \leq i \leq n} h_i^i \bmod p$$

the *monic squarefree decomposition* of g modulo p, with unique monic polynomials $h_1, \ldots, h_n \in \mathbb{Z}[x]$ with coefficients in $\{0, \ldots, p-1\}$ that are squarefree and pairwise coprime modulo p. We say that p is a *good prime* if $g_i \equiv \mathrm{lc}(g_i) h_i \bmod p$ for $1 \leq i \leq n$, so that the squarefree decomposition of the modular image of g is the modular image of the squarefree decomposition, up to leading coefficients. The following lemma, which may be of independent interest, gives several conditions that are equivalent to being a good prime.

Lemma 7 *Let $g \in \mathbb{Z}[x]$ be primitive, normalized, and of degree $n \geq 1$, $g_1, \ldots, g_n \in \mathbb{Z}[x]$ its primitive squarefree decomposition, $g_* = g_1 \cdots g_n$ its (primitive) squarefree part, and $p \in \mathbb{Z}$ an odd prime greater than n that does not divide $\mathrm{lc}(g)$. Then the following are equivalent.*

(i) p is a good prime,

(ii) the $g_i \bmod p$ are squarefree and pairwise coprime,

(iii) $g_* \bmod p$ and $g_*' \bmod p$ are coprime,

(iv) p does not divide the discriminant $\mathrm{res}(g_*, g_*')$,

(v) the squarefree part of $g \bmod p$ is, up to a multiplicative constant, equal to $g_* \bmod p$.

Moreover, the squarefree part of $g \bmod p$ *divides* $g_* \bmod p$*, and* p *is a good prime if and only if the degrees of both polynomials coincide.*

Proof. "(i) $\iff$ (ii)" follows from the uniqueness of the monic squarefree deomposition modulo p. For "(ii) $\iff$ (iii)", we find $g_*' = \sum_{1 \le j \le n} g_j' g_*/g_j$, by the Leibniz rule. Since $p > n \ge \deg g_j$ and $p \nmid \mathrm{lc}(g_j)$, both g_j and g_j' are nonzero modulo p for $1 \le j \le n$. Thus g_i and g_*' are coprime modulo p if and only if g_i is coprime modulo p to g_i' and all g_j with $j \ne i$, for all i.

By Fact 5, (iii) is equivalent to $\mathrm{res}(g_* \bmod p, g_*' \bmod p) = 0$ in $\mathbb{F}_p$. Since $p > n$ and $p \nmid \mathrm{lc}(g_*)$, we have $\deg(g_* \bmod p) = n = \deg g_*$ and $\deg(g_*' \bmod p) = n - 1 = \deg g_*'$. Thus both the Sylvester matrix of g_* and g_*' and the Sylvester matrix of $g_* \bmod p$ and $g_*' \bmod p$ are of size $(2n-1) \times (2n-1)$, and the discriminant of $g_* \bmod p$ equals the discriminant of g_* taken modulo p. This proves the equivalence of (iii) and (iv).

Finally, since g^* contains all irreducible polynomials dividing g, its reduction $g_* \bmod p$ contains all irreducible polynomials dividing $g \bmod p$, and hence the squarefree parts of $g \bmod p$ and $g_* \bmod p$ coincide. In particular, the squarefree part of $g \bmod p$ divides $g_* \bmod p$, and they differ by a multiplicative constant if and only if $g_* \bmod p$ is squarefree, which together with Fact 5 proves "(iii) $\iff$ (v)" and the last assertion. $\square$

Here is the algorithm. We recall that θ is the word size of our processor. We will assume that our polynomials have degree at most $2^{\theta-1}$, which is not a serious restriction in practice for $\theta \ge 32$. The O-estimates that we will state for our algorithms are valid when θ is considered to be variable, and the implied constants do not depend on θ.

The *primitive part* $\mathrm{pp}(f)$ of a nonzero polynomial $f \in \mathbb{Z}[x]$ is the unique nonzero normalized primitive polynomial $h \in \mathbb{Z}[x]$ such that f/h is a nonzero constant in $\mathbb{Z}$. The cardinality of a finite set S is denoted by $\#S$.

Algorithm 8 *Modular squarefree factorization.*
Input: *A normalized primitive polynomial* $g \in \mathbb{Z}[x]$ *of degree* $1 \le n \le 2^{\theta-1}$ *and max-norm* $\|g\|_\infty < 2^{\theta d}$.
Output: *The primitive squarefree decomposition of* g*, or otherwise "FAIL".*

1. $b \leftarrow \mathrm{lc}(g), \quad B \leftarrow \lfloor (n+1)^{1/2} 2^{n+\theta d} \rfloor, \quad s \leftarrow \lceil (\log_2 2Bb)/(\theta-1) \rceil$

2. *choose a set* S_0 *of* $2s$ *distinct single precision primes*
   $S_1 \leftarrow \{p \in S_0 : p \nmid b\}$

3. *for all* $p \in S_1$ *do step 4*

4. *compute the monic squarefree decomposition*

$$g \equiv b \prod_{1 \le i \le n} h_{p,i}^i \bmod p$$

   of g *modulo* p, *with monic polynomials* $h_{p,1}, \ldots, h_{p,n} \in \mathbb{Z}[x]$ *of max-norm less than* $p/2$ *that are squarefree and pairwise coprime modulo* p

5. { *remove the bad primes from* S_1 }
   $d \leftarrow \max\{\deg h_{p,1} + \cdots + \deg h_{p,n} \colon p \in S_1\}$
   $S_2 \leftarrow \{p \in S_1 \colon \deg h_{p,1} + \cdots + \deg h_{p,n} = d\}$
   if $\#S_2 \ge s$ *then remove* $\#S_2 - s$ *primes from* S_2 *else return "FAIL"*

6. *for* $1 \le i \le n$ *do step 7*

7. *use the Chinese Remainder Algorithm to compute* $g_i^* \in \mathbb{Z}[x]$ *of max-norm less than* $(\prod_{p \in S_2} p)/2$ *such that* $g_i^* \equiv bh_{p,i} \bmod p$ *for all* $p \in S_2$

8. *if* $\prod_{1 \le i \le n} \|\mathrm{pp}(g_i^*)\|_1^i \le B$ *then return* $\mathrm{pp}(g_1^*), \ldots, \mathrm{pp}(g_n^*)$ *else return "FAIL"*

Theorem 9 *Algorithm 8 succeeds if and only if at most* s *of the initial primes divide* b *or are bad, and then it returns the correct primitive squarefree decomposition of* g.

Proof. Let $g_1, \ldots, g_n \in \mathbb{Z}[x]$ be the primitive squarefree decomposition of g, and $g_* = g_1 \cdots g_n$ its squarefree part. We first show that unless all $2s$ initial primes in S_0 divide b or are bad, a prime $p \in S_1$ is good if and only if $\deg(h_{p,1} \cdots h_{p,n}) = d$ in step 5. The last assertion of Lemma 7 implies that $\deg(h_{p,1} \cdots h_{p,n}) \le \deg g_*$, with equality if and only if p is good. Thus if at least one prime is good, we have $\deg g_* = d$, and the claim follows.

Now assume that the condition in step 5 is true, so that at least s of the initial primes in S_0 are good and do not divide b. Then S_2 contains s primes after step 5, all good. Let $m = \prod_{p \in S_2} p$ and $1 \le i \le n$. Since g_i divides g, also $\mathrm{lc}(g_i)$ divides b, and hence $bg_i/\mathrm{lc}(g_i) \equiv g_i^* \bmod m$. Mignotte's bound (Fact 4) implies that $\|g_i\|_\infty \le B$, and hence

$$\|bg_i/\mathrm{lc}(g_i)\|_\infty \le bB \le 2^{(\theta-1)s-1} < \frac{m}{2}.$$

Thus the coefficients of $bg_i/\mathrm{lc}(g_i)$ and of g_i^* are at most $m/2$ in absolute value, so that both polynomials are equal. Since g_i is primitive and normalized, it follows that $g_i = \mathrm{pp}(g_i^*)$, and again by Mignotte's bound, we find that the condition in step 8 is true and the algorithm returns the correct result.

Conversely, suppose that the condition in step 8 is true. Then

$$\left\| \prod_{1 \le i \le n} \mathrm{pp}(g_i^*)^i \right\|_\infty \le \left\| \prod_{1 \le i \le n} \mathrm{pp}(g_i^*)^i \right\|_1 \le \prod_{1 \le i \le n} \|\mathrm{pp}(g_i^*)\|_1^i \le B < \frac{m}{2}.$$

We have $\prod_{1 \le i \le n} \mathrm{pp}(g_i^*)^i \equiv g \bmod m$, and since the coefficients of both polynomials are absolutely bounded by $m/2$, they are equal. The $\mathrm{pp}(g_i^*)$ are squarefree and pairwise coprime in $\mathbb{Z}[x]$, since they are so modulo each prime in S_2 after step 5, and since they are primitive and normalized, we find that $g_i = \mathrm{pp}(g_i^*) \equiv \mathrm{lc}(g_i)h_{p,i} \bmod p$ for all $i \le n$ and $p \in S_2$, by the uniqueness of the primitive squarefree decomposition, and all s primes in S_2 are good after step 5. $\square$

We recall that π_θ is the number of single precision primes.

Theorem 10 *Let $g_* \in \mathbb{Z}[x]$ be the squarefree part of g. Then $r = \mathrm{res}(g_*, g_*')$ is a nonzero integer with $|r| \le D = (nB^2)^n$. If $\pi_\theta \ge 2\lceil(\log_2 D)/(\theta - 1)\rceil$ and the set S_0 in step 2 is chosen uniformly at random from among all subsets of cardinality $2s$ of the single precision primes, then Algorithm 8 returns "FAIL" with probability at most $1/2$.*

Proof. The polynomial g_* is squarefree, and Fact 5 implies that r is nonzero. We have $\|g_*\|_2 \le B$, by Mignotte's bound (Fact 4), $\|g_*'\|_2 \le n\|g_*\|_2 \le nB$, and the resultant bound of Fact 4 shows that $|r| \le D$.

We note next that $2Bb \le D$, so that under the assumptions of the theorem, it is possible to find $2s \le \pi_\theta$ single precision primes in step 2. A prime p divides $\mathrm{lc}(g)$ if and only if it divides $\mathrm{lc}(g_*)$, and the latter in turn divides r since it divides the first row of the Sylvester matrix of g_* and g_*'. By assumption, at least half of the single precision primes do not divide r, so that they do not divide $\mathrm{lc}(g)$ and are good, by Lemma 7. Therefore the probability that at least half of the primes of a randomly chosen set of $2s$ single precision primes are good is at least $1/2$ (see von zur Gathen & Gerhard 1999, Exercise 6.31), and the claim follows from Theorem 9. $\square$

Theorem 11 *If we neglect the cost for choosing primes in step 2, then Algorithm 8 takes $O(n^3+nd^2)$ word operations when using classical arithmetic, and $O((d\,\mathsf{M}(n)+n\,\mathsf{M}(n+d))\log(n+d))$ or $O^{\sim}(n^2+nd)$ word operations when using fast arithmetic, where the $O^{\sim}$-notation suppresses logarithmic factors.*

Proof. Reducing a coefficient of g, which has word length at most $d \in O(s)$, modulo at most $2s$ single precision primes takes $O(s^2)$ word operations with classical arithmetic and $O(\mathsf{M}(s)\log s)$ with fast arithmetic. There are at most $n+1$ coefficients, and hence the overall cost for this is $O(ns^2)$ and $O(n\,\mathsf{M}(s)\log s)$ word operations, respectively.

In step 4, we have $n < p$, so that taking derivatives does not conflict with the characteristic, and hence Yun's (1976) algorithm, designed for characteristic zero, can be used literally to compute the squarefree decomposition modulo p. This takes $O(n^2)$ arithmetic operations in $\mathbb{F}_p$ with classical arithmetic and $O(\mathsf{M}(n)\log n)$ with fast arithmetic (see Yun 1977 or von zur Gathen & Gerhard (1999), Theorem 14.23, for fast arithmetic, and a similar reasoning also yields the estimate for classical arithmetic). Since p is single precision, one arithmetic operation in $\mathbb{F}_p$ takes $O(1)$ word operations. There are at most $2s$ primes in S_1, and hence the overall cost of steps 3 and 4 is $O(sn^2)$ or $O(s\,\mathsf{M}(n)\log n)$ word operations when using classical or fast arithmetic, respectively.

Let $m = \prod_{p\in S_2} p < 2^{\theta s}$. Then the cost for computing one coefficient of some g_i^* in step 7 by Chinese remaindering is $O(s^2)$ word operations with classical arithmetic and $O(\mathsf{M}(s)\log s)$ with fast arithmetic. Since $n = \sum_{1\le i\le n} i \deg g_i^*$, the total number of non-leading coefficients of all nonconstant g_i^* is at most n, and the overall cost of steps 6 and 7 is $O(ns^2)$ and $O(n\,\mathsf{M}(s)\log s)$ word operations, respectively. (In the unlikely event that all primes in S_2 are bad, it may happen that not all degree sequences $\deg h_{p,1},\ldots,\deg h_{p,n}$ for $p \in S_2$ are equal, but we can detect this during the Chinese remaindering stage in step 7 and then return "FAIL".)

The cost for computing the primitive parts of all g_i^* in step 8 is $O(n)$ gcd's and divisions of integers of word length at most s, taking $O(ns^2)$ word operations with classical arithmetic and $O(n\,\mathsf{M}(s)\log s)$ with fast arithmetic.

The cost for all other steps is negligible, the claims follow by adding costs and using $s \in O(n+d)$, and taking $\mathsf{M}(n) = n\log n \operatorname{loglog} n$ gives the $O^{\sim}$-estimate. □

Remark 12 • When $\mathsf{M}(n) \in \Omega(n^{1+\varepsilon})$ for some positive ε, then we can drop the factor $\log(n+d)$ in the cost estimate for Algorithm 8.

- When using fast arithmetic, then Algorithm 8 is—up to logarithmic factors—asymptotically optimal in the diagonal case where $n \approx d$, since its running time is essentially linear in the input size, which is about n^2 words.

- Yun (1976) states an estimate of $O(k^4(\nu^2\delta + \nu\delta^2))$ word operations for his algorithm for univariate polynomials over $\mathbb{Z}$, employing modular gcd techniques. Here $k \le n$ is the largest index such that g_k is nonconstant, and δ and ν are upper bounds on the degrees and the bit size of the coefficients of all g_i, respectively. This result is a bit difficult to compare with ours, since it is expressed in terms of a different set of parameters. For example, in the case when k is small, we have the approximate correspondences $n \approx \delta$ and $d\theta \approx \nu$, and then his estimate aggrees with our estimate for classical arithmetic, and our result for fast arithmetic is—up to logarithmic factors—better by one order of magnitude.

  Yun (1976) also gives a modular algorithm for the squarefree factorization of multivariate polynomials based on Hensel lifting, but does not state a time estimate. For univariate integer polynomials, one would obtain about the same cost estimate as in Theorem 11, up to logarithmic factors, when using fast arithmetic, and the estimate for classical arithmetic is slower by one order of magnitude.

- The number $2s$ of initial primes in step 2 of Algorithm 8 is often much too large in practice, for two reasons. On the one hand, the estimate for the failure probability in Theorem 10 is too pessimistic when there are considerably more single precision primes than required by the theorem. On the other hand, the coefficients of the g_i are often significantly smaller than guaranteed by Mignotte's bound. One solution to overcome this is to work in an adaptive fashion: start with a reasonable number of primes (say d or even fewer), check whether $\prod_{1\le i\le n} \mathrm{pp}(g_i^*)^i = g$, and add more new primes in case of failure. In this way, the algorithm will never use more primes than needed, at the expense of an additional check after each new prime. Heuristically, most of the checks can even be avoided by first checking whether the equation holds for $x = 0$. This strategy has been successfully implemented by Shoup (1995) in his software package NTL (`http://www.shoup.net/ntl/`) for computing gcd's of integer polynomials.

- We did not say anything about how to find random single precision primes in step 2 of Algorithm 8 and what this costs. In principle, this can be done by repeatedly choosing random integers between $2^{\theta-1}$ and 2^{θ}, subjecting them to a probabilistic primality test (like the ones of Solovay & Strassen 1977 or Miller 1976 and Rabin 1976, 1980), and stop as soon as sufficiently many "probable" primes are found. Since we are looking for primes of a fixed precision, the cost for one primality test is $O(1)$ word operations, and the expected number of trials until a prime is found is also $O(1)$, provided that there are sufficiently many of them, say at least $4s$ (one has to be a bit careful about getting $2s$ *distinct* primes here). Thus the cost for randomly and independently choosing s single precision primes is $O(s)$ word operations. The prime number theorem (see Fact 6) guarantees that there are sufficiently many single precision primes for all practical purposes.

A second possibility is to use the well-known sieve of Eratosthenes to compute all primes below 2^{θ}, at a cost of $O(2^{\theta} \cdot \theta \log \theta)$ word operations (see, e.g., von zur Gathen & Gerhard (1999), Theorem 18.10). When θ is fixed, then this is $O(1)$, at least in theory, but in practice this is probably too expensive for $\theta \geq 32$.

In practice, however, it is useful to maintain a list of precomputed single precision primes and then to choose randomly from this list if there are sufficiently many of them (i.e., more than required by the resultant bound D from Theorem 10), or simply to take the first $2s$ primes from the list otherwise. In the last case, we no longer have the guaranteed bound on the error probability; in principle, it may happen that *all* primes from our list are bad.

Nevertheless, using such a precomputed list is very attractive, for two reasons. Firstly, the prime list can be reused for other kinds of modular algorithms as well, such as gcd computation and linear algebra. Secondly, we may choose all primes p to be *Fourier primes*, so that $p-1$ is divisible by a "large" power of two, say $2^{\theta/2}$. This allows for very efficient implementations of FFT-based polynomial arithmetic, as has been shown by Shoup (1995) with his software package NTL, at the expense of reducing the number of suitable primes.

Arne Storjohann asked whether choosing the primes from the list deterministically works in fact in *any* case within essentially the same time bound. This is an interesting question, which we leave for future research; the answer would be positive if it were possible to show that the resultant $\mathrm{res}(g_*, g_*')$ can not be divisible by "too many" single precision primes.

The above method can be adapted to bivariate polynomials over a field when we count coefficient operations. Both the algorithm and its analysis are then much easier due to the absence of carries, and in particular when the field is large enough such that all moduli may be chosen linear.

A similar approach yields also fast modular algorithms for *factor refinement*: given one (or several) nontrivial partial factorizations $g = h_1 \cdots h_t$ in $\mathbb{Z}[x]$, compute the finest partial factorization $g = \prod_i g_i^{e_i}$, with positive integers e_i and $g_i \in \mathbb{Z}[x]$ pairwise coprime, but not necessarily squarefree, that you can obtain from the given factorization by gcd computations.

4 Modular Hermite reduction

Let $f, g \in \mathbb{Z}[x]$ be nonzero polynomials with $0 < n = \deg g \leq m$ and $\deg f < m$. *Hermite reduction* finds polynomials $a, b, c, d, h \in \mathbb{Q}[x]$ with $\deg a < \deg b$, $\deg c < \deg d$, $\deg b + \deg d \leq n$, $\deg h \leq m - n$, $h(0) = 0$, and b squarefree, such that

$$\int \frac{f}{g} = h + \frac{c}{d} + \int \frac{a}{b},$$

or equivalently,

$$\frac{f}{g} = h' + \left(\frac{c}{d}\right)' + \frac{a}{b}. \tag{2}$$

The polynomial h is usually called the *polynomial part* of the integral $\int (f/g)$, c/d is the *rational part*, and $\int (a/b)$ is the *logarithmic part*. It is well-known (Ostrogradsky 1845, Horowitz 1971) that the denominators in the representation (2) may be chosen as $b = g_*$, the squarefree part of g, and $d = g/g_*$, and then the polynomials a, c, and h are uniquely determined.

We may assume that f and g are primitive and coprime and that g is normalized, and let $g_1, \ldots, g_n \in \mathbb{Z}[x]$ be the primitive squarefree decomposition of g, $b = g_* = g_1 \cdots g_n$ its (primitive) squarefree part, and $d = g/g_*$. Performing a partial fraction decomposition, we may rewrite (2) as

$$\frac{f}{g} = h' + \sum_{1 \leq j < i \leq n} \left(\frac{c_{ij}}{g_i^j}\right)' + \sum_{1 \leq i \leq n} \frac{a_i}{g_i}, \tag{3}$$

with unique polynomials $c_{ij}, a_i \in \mathbb{Q}[x]$ of degree less than $\deg g_i$ for all i, j.

For completeness, we briefly recall how Hermite reduction works (see von zur Gathen & Gerhard 1999, Section 22.2). After splitting off the derivative of the polynomial part by a division with remainder, the partial fraction decomposition

$$\frac{f}{g} = h' + \sum_{1 \le j \le i \le n} \frac{\gamma_{ij}}{g_i^j} \tag{4}$$

is computed, with $\gamma_{ij} \in \mathbb{Q}[x]$ of degree less than $\deg g_i$ for all i, j. Then for each $i \ge 2$ independently, the following reduction step is performed for $j = i, \ldots, 2$. First we compute $\sigma, \tau \in \mathbb{Q}[x]$ of degree less than $\deg g_i$ such that $\sigma g_i + \tau g_i' = \gamma_{ij}$, by means of the Extended Euclidean Algorithm. Then we set

$$c_{i,j-1} \leftarrow \frac{-\tau}{j-1}, \qquad \gamma_{i,j-1} \leftarrow \gamma_{i,j-1} + \sigma + \frac{\tau'}{j-1}. \tag{5}$$

The final γ_{i1} is then equal to a_i, for all i. Both the cost for computing the partial fraction decomposition and the total cost for all reduction steps is $O(n^2)$ arithmetic operations with classical arithmetic and $O(\mathsf{M}(n) \log n)$ with fast arithmetic (see Fact 2 for the partial fraction decomposition and, for example, von zur Gathen & Gerhard (1999), Theorem 22.7, for the reduction steps with fast arithmetic; the corresponding result with classical arithmetic follows by a similar reasoning).

Lemma 13 *Let F be a field of characteristic greater than $m \in \mathbb{N}$, $f, g \in F[x]$ nonzero and coprime polynomials with $0 < \deg g = n \le m$ and $\deg f < m$, and $g = \prod_{1 \le i \le n} g_i^i$ a squarefree decomposition of g, with nonconstant and pairwise coprime $g_1, \ldots, g_n \in F[x]$. Then there exist unique polynomials $h \in F[x]$ of degree at most $m - n$ and with $h(0) = 0$ and c_{ij}, a_i in $F[x]$ of degree less than $\deg g_i$ for all i, j, satisfying (2), and they can be computed by Hermite reduction as described above.*

Proof. For the existence, we note that Hermite reduction works literally as described above, since the degrees of all occurring polynomials are less than the characteristic, so that taking derivatives does not interfere with the characteristic, and also all integers occurring in denominators in (5) are units in F. The uniqueness follows from the uniqueness of the decomposition (2), the uniqueness of quotient and remainder in polynomial division, and the uniqueness of partial fraction decomposition (Fact 2). □

Back to equation (3), we have

$$\left(\frac{c_{ij}}{g_i^j}\right)' = \frac{c'_{ij}g_i - jc_{ij}g'_i}{g_i^{j+1}},$$

and after multplying both sides in (3) by g, we obtain the equivalent equation

$$f = gh' + \sum_{1\leq j<i\leq n} \left(\frac{g}{g_i^j}c'_{ij} - jg'_i\frac{g}{g_i^{j+1}}c_{ij}\right) + \sum_{1\leq i\leq n} \frac{g}{g_i}a_i. \tag{6}$$

Both taking derivatives and multiplying by a polynomial are linear operators on polynomials, and by comparing coefficients, we see that (6) is in turn equivalent to a linear system of m equations in the m coefficients of h (without the constant coeffificent $h(0)$), the c_{ij}, and the a_i.

The idea for a modular algorithm now is as follows. Using Cramer's rule and Hadamard's inequality, we obtain a bound on the numerators and the denominators of the coefficients of h, the c_{ij}, and the a_i. Then we choose sufficiently many single precision primes, so that their product exceeds this bound and we can uniquely reconstruct the coefficients from all the modular images, use Hermite's method modulo each of the primes, and obtain the result by rational number reconstruction. In practice, it may be advantageous to do this in an adaptive fashion, so that only as many primes as needed to recover the result are used; see Remark 19 below.

So let $A \in \mathbb{Z}^{m\times m}$ be the coefficient matrix of the linear system equivalent to (6) (a similar matrix for the linear system equivalent to (2) has been given explicitly by Horowitz 1971), and identify the polynomial $f \in \mathbb{Z}[x]$ with its coefficient vector in $\mathbb{Z}^m$ (this is the "right hand side" of the linear system). Thus if $y \in \mathbb{Q}^m$ contains the coefficients of h and the c_{ij} and a_i, then (6) is equivalent to $Ay = f$. Since (6) has a unique solution for a fixed g and arbitrary f of degree less than m, the matrix A, depending only on g but not on f, is nonsingular.

A variant of the following theorem for the linear system equivalent to (2) appears in Horowitz (1971).

Theorem 14 *Assume that $\|f\|_\infty, \|g\|_\infty < 2^{\theta d}$, and let $B = \lfloor (n+1)^{1/2}2^{n+\theta d}\rfloor$. Let A^* be the matrix resulting from A by replacing some column by f. Then $\det A$ and $\det A^*$ are absolutely less than $m^{1/2}(nB)^m$, and so are the (co-prime) numerators and denominators of the unique solutions h, c_{ij}, and a_i of (6).*

Proof. For $0 \le k \le m-n$, the entries of the column of the matrix A corresponding to the coefficient of x^{k+1} in h are coefficients of g multiplied by $k+1 \le m$. Now let $0 \le k < n_i = \deg g_i$. The entries of the column of A corresponding to the coefficient of x^k in a_i are coefficients of the polynomial g/g_i, for all i. For $1 \le j < i \le n$, the entries of the column corresponding to the coefficient of x^k in c_{ij} are of the form $(k+1)\alpha - j\beta$ if $0 \le k < n_i - 1$ and $-j\beta$ if $k = n_i - 1$, where α is some coefficient of g/g_i^j and β is some coefficient of $g_i' g/g_i^{j+1}$, by (6).

Now Mignotte's bound (Fact 4) implies that the one-norm, and hence also the Euclidean norm, of any g/g_i^j is at most B, and also that

$$\|g_i'g/g_i^{j+1}\|_2 \le \|g_i'g/g_i^{j+1}\|_1 \le \|g_i'\|_1 \|g/g_i^{j+1}\|_1 \le n_i \|g_i\|_1 \|g/g_i^{j+1}\|_1 \le n_i B.$$

Thus $\|(k+1)g/g_i^j\|_2 \le n_i B$, $\|jg_i'g/g_i^{j+1}\|_2 \le jn_iB$, and $\|f\|_2 \le m^{1/2}B$, so that the Euclidean norm of any column of A is at most $in_iB \le nB$, and this is also an upper bound on the Euclidean norm of all but one column of A^*. Now Hadamard's inequality (Fact 4) implies that $|\det A| \le (nB)^m$ and $|\det A^*| \le (nB)^m m^{1/2}/n$, and the last claim follows from Cramer's rule which implies that a coefficient of h, c_{ij}, or a_i is of the form $(\det A^*)/\det A$. $\square$

Here is the modular algorithm for Hermite reduction.

Algorithm 15 *Modular Hermite reduction.*
Input: *Primitive nonzero and coprime polynomials $f, g \in \mathbb{Z}[x]$ of max-norm less than $2^{\theta d}$ such that g is normalized, $0 < n = \deg g \le m$ and $\deg f < m < 2^{\theta-1}$, and the primitive squarefree decomposition $g_1, \ldots, g_n \in \mathbb{Z}[x]$ of g.*
Output: *The unique polynomials $h, c_{ij}, a_i \in \mathbb{Q}[x]$ solving (3) with $h(0) = 0$, $\deg h \le m - n$, and $\deg c_{ij}, \deg a_i < \deg g_i$ for all i, j.*

1. $B \leftarrow \lfloor (n+1)^{1/2} 2^{n+\theta d} \rfloor$, $\quad C \leftarrow m^{1/2}(nB)^m$,
   $t \leftarrow \left\lceil \dfrac{n(\log_2 nB^2)}{\theta - 1} \right\rceil$, $\quad s^* \leftarrow \left\lceil \dfrac{\log_2(2C^2)}{\theta - 1} \right\rceil$
2. *choose a set S_0 of $s = s^* + t$ distinct single precision primes*
   $S_1 \leftarrow \{p \in S\colon p \nmid \mathrm{lc}(g)$ *and* $\gcd(g_* \bmod p, g_*' \bmod p) = 1\}$, *where* $g_* = g_1 \cdots g_n$ *is the squarefree part of* g
3. *for each $p \in S_1$ do steps 4 and 5*
4. *compute the partial fraction decomposition (4) modulo p*

5. *use the Hermite reduction steps (5) to solve (3) modulo p*

6. *compute the coefficients of h and all c_{ij} and a_i from their images modulo all primes in S_1 by rational number reconstruction*

Theorem 16 *Algorithm 15 works correctly as specified.*

Proof. Let h and c_{ij}, a_i in $\mathbb{Q}[x]$ be the unique solution of (3). By Lemma 7, each prime $p \in S_1$ is a good prime in the sense of Section 3, so that $g_1 \bmod p, \ldots, g_n \bmod p$ are a squarefree decomposition of $g \bmod p$. Now $h \bmod p$ and the $c_{ij} \bmod p$ and $a_i \bmod p$ form a solution of (3) in $\mathbb{F}_p[x]$, and Lemma 13 implies that these are computed in steps 3–5 for all primes $p \in S_1$. In particular, the uniqueness statement of the lemma implies that none of the primes in S_1 divides the determinant of the coefficient matrix A of the linear system equivalent to (3), and by Cramer's rule, the denominators of h and the c_{ij} and a_i are not divisible by any prime in S_1.

As in the proof of Theorem 10, we find that all primes in $S_0 \setminus S_1$ are divisors of the (nonzero) discriminant $\mathrm{res}(g_*, g_*')$, by Lemma 7. We have

$$|\mathrm{res}(g_*, g_*')| \le (nB^2)^n \le 2^{(\theta-1)t},$$

also by Theorem 10, such that S_1 contains at least $s^* = s - t$ primes. Now $\prod_{p \in S_1} p > 2^{(\theta-1)s^*} \ge 2C^2$, the numerators and denominators of the coefficients of h and the c_{ij} and a_i are absolutely bounded by C, by Theorem 14, and hence we can apply Fact 3 to reconstruct them. $\square$

Theorem 17 *If $\pi_\theta \ge s$ and we neglect the cost for prime generation, then the cost for Algorithm 15 is $O(m^3(n^2 + d^2))$ word operations with classical arithmetic, and $O(m\,\mathsf{M}(m(n+d))\log(m(n+d)))$ or $O^\sim(m^2(n+d))$ with fast arithmetic.*

Proof. We can evaluate f, g and all g_i modulo all primes in S_1 and S_0, respectively, using $O(ms^2)$ word operations with classical arithmetic and $O(m\,\mathsf{M}(s)\log s)$ with fast arithmetic. The cost for computing $\gcd(g_* \bmod p, g_*' \bmod p)$ for all $p \in S_0$ is $O(n^2 s)$ or $O(\mathsf{M}(n)\log n \cdot s)$, respectively. The cost for computing the polynomial part modulo a prime $p \in S_1$ is essentially one division with remainder, taking $O(m^2)$ word operations, in total $O(m^2 s)$ word operations, with classical arithmetic, and $O(\mathsf{M}(m)s)$ with fast arithmetic. By the discussion preceding Lemma 13, the cost for steps 4 and 5 is $O(m^2)$

word operations per prime p, in total $O(m^2 s)$ word operations with classical arithmetic, and $O(\mathsf{M}(m)\log m \cdot s)$ with fast arithmetic. Finally, by Fact 3 the cost for rational number reconstruction in step 6 is $O(s^2)$ word operations per coefficient, or $O(ms^2)$ word operations in total, with classical arithmetic, and $O(m\,\mathsf{M}(s)\log s)$ with fast arithmetic. Now the claims follow from $s \in O(m(n+d))$ by adding up costs, and taking $\mathsf{M}(n) = n \log n \operatorname{loglog} n$ gives the $O^\sim$-estimate. $\square$

Since the cost for the modular Hermite reduction dominates the cost for computing the squarefree decomposition (Theorem 11), we have the following corollary.

Corollary 18 *Given two primitive nonzero and coprime polynomials $f, g \in \mathbb{Z}[x]$ of max-norm less than $2^{\theta d}$ such that g is normalized, $0 < n = \deg g \leq m$, $\deg f < m < 2^{\theta-1}$, and $\pi_\theta > s$, with s as in Algorithm 15, then we can compute the decomposition (2) using $O(m^3(n^2+d^2))$ word operations with classical arithmetic, and $O(m\,\mathsf{M}(m(n+d))\log(m(n+d)))$ or $O^\sim(m^2(n+d))$ with fast arithmetic.*

Remark 19

- When $\mathsf{M}(n) \in \Omega(n^{1+\varepsilon})$ for some positive ε, then we can drop the factor $\log(m(n+d))$ in the cost estimates of Theorem 17 and Corollary 18.

- For those inputs where the upper bound of $O(m^2(n+d))$ on the output size provided by Theorem 14 is achieved in the worst case, Algorithm 15 is—up to logarithmic factors—asymptotically optimal when using fast arithmetic.

- Horowitz (1971) has analyzed two variants of Hermite reduction for univariate polynomials over $\mathbb{Z}$ in the case $m = n$: a non-modular variant of Algorithm 15 and an algorithm solving (2) by linear algebra methods. In Theorems 4.1 and 4.3, he gives cost estimates of $O(k^3 n^5 \log^2(n\gamma))$ and $O(n^5 \log^2(n\gamma))$ word operations, respectively, where $k \leq n$ is the largest index such that g_k is nonconstant and γ corresponds essentially to our B, such that $\log(n\gamma) \in O(n+d)$. Even for classical arithmetic, the time bound of Theorem 17 is better by two orders of magnitude.

- In practice, the number s of initial primes in step 2 of Algorithm 15 may be too large. A reason for this is that the Mignotte bound B in

Theorem 14 is often too large. One solution to overcome this is to work in an adaptive fashion, as described in Remark 12.

- When using classical arithmetic, it may be advantageous to use a single prime and lifting instead of several primes and Chinese remaindering. For example, by suitably modifying the lifting algorithm given in Mulders & Storjohann (1999), we would obtain the same time estimate $O(m^3(n^2+d^2))$ as in Theorem 17.

Acknowledgements. The author would like to thank Joachim von zur Gathen and Arne Storjohann for many helpful remarks and suggestions.

References

ALFRED V. AHO, JOHN E. HOPCROFT, AND JEFFREY D. ULLMAN, *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading MA, 1974.

JOANNES BERNOULLUS, Problema exhibitum. *Acta eruditorum* (1703), 26–31.

MANUEL BRONSTEIN, *Symbolic Integration I—Transcendental Functions*. Algorithms and Computation in Mathematics **1**. Springer-Verlag, Berlin Heidelberg, 1997.

JOACHIM VON ZUR GATHEN AND JÜRGEN GERHARD, *Modern Computer Algebra*. Cambridge University Press, Cambridge, UK, 1999. To appear.

CARL FRIEDRICH GAUSS, Disquisitiones generales de congruentiis. Analysis residuorum caput octavum. In *Werke* **II**, Handschriftlicher Nachlass, ed. R. DEDEKIND, 212–242. Königliche Gesellschaft der Wissenschaften, Göttingen, 1863. Reprinted by Georg Olms Verlag, Hildesheim New York, 1973.

MARK WILLIAM GIESBRECHT, *Nearly Optimal Algorithms for Canonical Matrix Forms*. PhD thesis, Department of Computer Science, University of Toronto, 1993. Technical Report 268/93.

C. HERMITE, Sur l'intégration des fractions rationnelles. *Annales de Mathématiques, 2ème série* **11** (1872), 145–148.

ELLIS HOROWITZ, *Algorithms for Symbolic Integration of Rational Functions*. Ph.D. Dissertation, University of Wisconsin, Madison WI, 1969.

ELLIS HOROWITZ, Algorithms for partial fraction decomposition and rational function integration. In *Proceedings 2nd ACM Symposium on Symbolic and Algebraic Manipulation,* Los Angeles CA, ed. S. R. PETRICK. ACM Press, 1971, 441–457.

A. KARATSUBA AND YU. OFMAN, Умножение многозначных чисел на автоматах. Доклады Академий Наук СССР **145** (1962), 293–294. A. KARATSUBA and YU. OFMAN, Multiplication of multidigit numbers on automata, Soviet Physics–Doklady **7** (1963), 595–596.

D. LAZARD AND R. RIOBOO, Integration of rational functions: Rational computation of the logarithmic part. *Journal of Symbolic Computation* **9** (1990), 113–115.

D. MACK, On rational integration. Technical Report UCP-38, Department of Computer Science, University of Utah, 1975.

MAURICE MIGNOTTE, *Mathématiques pour le calcul formel.* Presses Universitaires de France, Paris, 1989. English translation: *Mathematics for Computer Algebra*, Springer-Verlag, New York, 1992.

GARY L. MILLER, Riemann's hypothesis and tests for primality. *Journal of Computer and System Sciences* **13** (1976), 300–317.

THOM MULDERS AND ARNE STORJOHANN, Diophantine linear system solving. In *Proceedings of the 1999 International Symposium on Symbolic and Algebraic Computation ISSAC '99,* Vancouver, Canada. ACM Press, 1999. To appear.

DAVID R. MUSSER, *Algorithms for Polynomial Factorization.* PhD thesis, Computer Science Department, University of Wisconsin, 1971. Technical Report #134, 174 pages.

M. OSTROGRADSKY, De l'intégration des fractions rationnelles. *Bulletin de la classe physico-mathématique de l'Académie Impériale des Sciences de Saint-Pétersbourg* **4**(82/83) (1845), 145–167.

MICHAEL O. RABIN, Probabilistic algorithms. In *Algorithms and Complexity*, ed. J. F. TRAUB, New York, 1976, Academic Press, 21–39.

MICHAEL O. RABIN, Probabilistic algorithms for testing primality. *Journal of Number Theory* **12** (1980), 128–138.

ROBERT H. RISCH, The problem of integration in finite terms. *Transactions of the American Mathematical Society* **139** (1969), 167–189.

Robert H. Risch, The solution of the problem of integration in finite terms. *Bulletin of the American Mathematical Society* **76**(3) (1970), 605–608.

J. Barkley Rosser and Lowell Schoenfeld, Approximate formulas for some functions of prime numbers. *Illinois Journal of Mathematics* **6** (1962), 64–94.

Michael Rothstein, *Aspects of symbolic integration and simplification of exponential and primitive functions.* PhD thesis, University of Wisconsin-Madison, 1976.

Michael Rothstein, A new algorithm for the integration of exponential and logarithmic functions. In *Proceedings of the 1977 MACSYMA Users Conference,* Berkeley CA, Washington DC, 1977, NASA, 263–274.

A. Schönhage and V. Strassen, Schnelle Multiplikation großer Zahlen. *Computing* **7** (1971), 281–292.

Victor Shoup, A new polynomial factorization algorithm and its implementation. *Journal of Symbolic Computation* **20** (1995), 363–397.

R. Solovay and V. Strassen, A fast Monte-Carlo test for primality. *SIAM Journal on Computing* **6**(1) (1977), 84–85. Erratum in **7** (1978), p. 118.

Robert George Tobey, *Algorithms for Antidifferentiation—Rational Functions.* PhD thesis, Harvard University, Cambridge MA, 1967.

Barry M. Trager, Algebraic factoring and rational function integration. In *Proceedings of the 1976 ACM Symposium on Symbolic and Algebraic Computation ISSAC '76,* Yorktown Heights NY, ed. R. D. Jenks. ACM Press, 1976, 219–226.

David Y. Y. Yun, On square-free decomposition algorithms. In *Proceedings of the 1976 ACM Symposium on Symbolic and Algebraic Computation ISSAC '76,* Yorktown Heights NY, ed. R. D. Jenks. ACM Press, 1976, 26–35.

David Y. Y. Yun, Fast algorithm for rational function integration. In *Information Processing 77—Proceedings of the IFIP Congress 77*, ed. B. Gilchrist, Amsterdam, 1977, North-Holland, 493–498.