Lecture II
The GCD

Next to the four arithmetic operations, the greatest common denominator (GCD) is perhaps the most basic operation in algebraic computing. The proper setting for discussing GCD's is in a unique factorization domain (UFD). For most common UFDs, that venerable algorithm of Euclid is available. In the domains $\mathbb{Z}$ and $F[X]$, an efficient method for implementing Euclid's algorithm is available. It is the so-called half-GCD approach, originating in ideas of Lehmer, Knuth and Schönhage. The presentation here is based on unpublished joint work with Klaus Thull, and gives a unified framework for the half-GCD approach for both integer and polynomial GCD. We also give the first proof for the correctness of the (corrected) polynomial half-GCD algorithm.

The student will not go far amiss if she interprets all references to rings as either the integers $\mathbb{Z}$ or a polynomial ring $F[X]$ over a field F (even taking $F = \mathbb{Q}$).

§1. Unique Factorization Domain

Let D be a commutative ring. All rings in this book contain unity 1 where $0 \neq 1$. For $a, b \in D$, we say b *divides* a, and write $b \,|\, a$, if there is a $c \in D$ such that $a = bc$. If b does not divide a, we write $b \nmid a$. We also call b a *divisor* of a, and a a *multiple* of b. Thus every element divides 0 but 0 does not divide any non-zero element. A *zero-divisor* is an element b such that $bc = 0$ for some non-zero c. We also call an element *regular* if it is not a zero-divisor. An *integral domain* (or *domain* for short) D is a commutative ring whose only zero-divisor is 0. A *unit* is an element that divides 1. (Alternatively, units are the invertible elements.) Thus the unity element 1 is always a unit and the zero element is never a unit. In a field, all non-zero elements are units. Two elements, a and b, are *associates* if $a = ub$ for some unit u. Clearly the relation of being associates is an equivalence relation. So the elements of D are partitioned into equivalence classes of associates.

Exercise 1.1:
(a) The set of units and the set of zero-divisors are disjoint.
(b) $a \,|\, b$ and $b \,|\, a$ iff a, b are associates. $\square$

Convention. For each equivalence class of associates, we assume that a *distinguished member* is chosen. The following convention captures most cases:
(i) The unity element 1 is always distinguished.
(ii) In $\mathbb{Z}$, the units are $+1$ and -1 and the equivalence classes are $\{-n, +n\}$ for each $n \in \mathbb{N}$. The non-negative elements will be distinguished in $\mathbb{Z}$.
(iii) In the polynomial ring $D[X]$ over a domain D, if we have specified distinguished elements in D then the distinguished elements of $D[X]$ will be those with distinguished leading coefficients. In case D is a field, this means the distinguished elements in $D[X]$ are the *monic polynomials*, *i.e.*, those with leading coefficient 1. Note that the product of distinguished elements are distinguished when $D = \mathbb{Z}$.

A *proper divisor* of b is any divisor that is neither a unit nor an associate of b. An element is *irreducible* if it has no proper divisors; otherwise it is *reducible*ring!reducible element. Since any divisor of a unit is a unit, it follows that units are irreducible. Furthermore, the zero element is irreducible if and only if D is a domain.

A *unique factorization domain* (abbreviated, UFD) is a domain D in which every non-unit b can be written as a product of irreducible non-units,

$$b = b_1 b_2 \cdots b_n \qquad (n \geq 1).$$

Moreover, these irreducible elements are unique up to reordering and associates. UFD's are also called *factorial domains.*

The importance of UFD's is that its elements are made up of "fundamental building blocks", namely the irreducible elements. Note that $\mathbb{Z}$ is a UFD, by the *Fundamental Theorem of Arithmetic.* In fact, a UFD can be said to be a domain that has an analogue to the Fundamental Theorem of arithmetic! The non-zero irreducible elements of $\mathbb{Z}$ are called primes. But in general, we define the *prime*ring!prime elements elements of a ring R to be those non-units $p \in R$ such that $p \neq 0$ and if p divides any product $a \cdot b$ then p divides either a or b.

One sees that prime elements are irreducible but the converse is not generally true. For example (see [29, page 173]), in $\mathbb{C}[X, Y, Z]/(Z^2 - XY)$, Z is irreducible but not prime because Z divides XY without dividing X or Y. It is easy to see that in a UFD, every irreducible element is also a prime. Hence this is an example of a non-UFD.

Theorem 1 *D is a UFD iff $D[X]$ is a UFD.*

It is clear that $D[X]$ is not a UFD if D is not a UFD. The proof of the other direction is due to Gauss and is deferred to the next lecture. Trivially, a field F is a UFD. Hence, by induction on $d \geq 1$, this theorem shows that $F[X_1, \ldots, X_d]$ is a UFD.

Greatest common divisor. Let D be a UFD and $S \subseteq D$ be a finite non-empty set. We write $a \,|\, S$ (read, a divides S) to mean $a \,|\, b$ for all $b \in S$. An element $d \in D$ is a *greatest common divisor* (abbreviated, GCD) of S if

1) $d \,|\, S$,

2) if $c \,|\, S$ then $c \,|\, d$.

Exercise 1.2: Prove that S has a greatest common divisor, and this is determined up to associates. □

We can therefore define the function $\mathtt{GCD}(S)$ by choosing the distinguished greatest common divisor of S. If $S = \{a_1, a_2, \ldots, a_m\}$, we write $\mathtt{GCD}(a_1, a_2, \ldots, a_m)$ for $\mathtt{GCD}(S)$. Unless otherwise noted, this lecture will assume that S has one or two elements: $S = \{a, b\}$. In this case, the GCD function may be regarded as a two argument function, $\mathtt{GCD}(a, b)$. It is called the *simple GCD function*, as opposed to the *multiple GCD function* for general sets. If S has $m \geq 2$ elements, we can compute $\mathtt{GCD}(S)$ using $m - 1$ simple GCD computations.

The following is easy.

$$\begin{array}{lcll}
\mathtt{GCD}(1, b) & = & 1 & \\
\mathtt{GCD}(0, b) & = & \widehat{b} & \text{where } \widehat{b} \text{ is the distinguished associate of } b \\
\mathtt{GCD}(a, b) & = & \mathtt{GCD}(b, a) & \\
\mathtt{GCD}(a + b, b) & = & \mathtt{GCD}(a, b) & \\
\mathtt{GCD}(ua, b) & = & \mathtt{GCD}(a, b) & \text{where } u \text{ is a unit}
\end{array}$$

Say a, b are *relatively prime* or *co-prime* if $\mathtt{GCD}(a, b) = 1$.

For instance, $\mathtt{GCD}(123, 234) = 3$ and $\mathtt{GCD}(3X^4 - 6X^3 + 13X^2 - 8X + 12, 6X^5 + 17X^3 - 3X^2 + 12X - 4) = 3X^2 + 4$.

GCD for ideals. Although we began with UFD's such as $\mathbb{Z}$ and $\mathbb{Q}[X]$, our Fundamental Problems force us to consider more general domains such as number rings (§VI.3). These rings need not be UFD's (exercise below). This led Kummer, Dedekind and Kronecker to develop ideal theory for algebraic numbers[1]. To regain the UFD property, we generalize numbers to ideals and introduce the concept of prime ideals. The ideal theoretic analogue of UFD's is this: a *Dedekind domain* is one in which every ideal is a product of prime ideals. It can be proved that such prime ideal factorizations are unique (e.g., [221, p. 273]). Number rings are Dedekind domains.

We do not define the concept of ideal divisibility via ideal products. Instead, if $I, J \subseteq D$ are ideals, we define I to be a *divisor* of J, and say I *divides* J, to mean $I \supseteq J$.

This definition is a stroke of inspiration from Dedekind (1871). Consider ideals in $\mathbb{Z}$: they have the form (n) where $n \in \mathbb{Z}$ since $\mathbb{Z}$ is a principal ideal domain (§3). Hence we can identify ideals of $\mathbb{Z}$ with numbers. Then $m, n \in \mathbb{Z}$ has the property that $m \mid n$ iff $(m) \supseteq (n)$, "agreeing" with our definition. In general, the relationship between ideal quotient and divisor property is only uni-directional: for ideals $I, J \subseteq D$, we have that $I \supseteq IJ$ and so I divides IJ.

The $\mathtt{GCD}$ of a set S of ideals is by definition the smallest ideal that divides each $I \in S$, and we easily verify that

$$\mathtt{GCD}(S) = \sum_{I \in S} I.$$

For $I = (a_1, \ldots, a_m)$ and $J = (b_1, \ldots, b_n)$, we have

$$\mathtt{GCD}(I, J) = I + J = (a_1, \ldots, a_m, b_1, \ldots, b_n). \tag{1}$$

So the GCD problem for ideals is trivial unless we require some other conditions on the ideal generators. For instance, for the ideals of $\mathbb{Z}$, the $\mathtt{GCD}$ of (a) and (b) is the ideal (a, b). But since $\mathbb{Z}$ is a principal ideal domain, we know that $(a, b) = (d)$ for some $d \in \mathbb{Z}$. We then interpret the $\mathtt{GCD}$ problem in $\mathbb{Z}$ to mean the computation of d from a, b. It is not hard to prove that d is what we have defined to be a greatest common divisor of a, b. Thus, the common notation '(a, b)' for $\mathtt{GCD}(a, b)$ is consistent with the ideal theoretic notation! In general, for a, b in a UFD, one should not expect $\mathtt{Ideal}(a, b)$ to be generated by the $\mathtt{GCD}(a, b)$. For instance, $\mathbb{Z}[X]$ is a UFD, $\mathtt{GCD}(2, X) = 1$ but $\mathtt{Ideal}(2, X) \neq \mathtt{Ideal}(1)$.

Exercises

Exercise 1.3:
(a) Is the set of ideals of a domain D under the ideal sum and ideal product operations a ring? The obvious candidates for the zero and unity elements are (0) and $(1) = D$.
(b) Verify equation (1).
(c) What is the least common multiple, $\mathtt{LCM}$, operation for ideal? □

Exercise 1.4: Say a domain D is *factorable* if every non-unit of D is a finite product of irreducible elements. Prove that a factorable domain D is a UFD iff irreducible elements are prime. □

[1]The other historical root for ideal theory is rational function fields in one variable.

Exercise 1.5: We will prove that the number ring $\mathbb{Z}[\sqrt{-5}] = \{x + y\sqrt{-5} : x, y \in \mathbb{Z}\}$ (cf. §VI.3) is not a UFD. The *norm* of an element $x + y\sqrt{-5}$ is $N(x + y\sqrt{-5}) := x^2 + 5y^2$. Show:
(a) $N(ab) = N(a)N(b)$.
(b) $N(a) = 1$ iff a is a unit. Determine the units of $\mathbb{Z}[\sqrt{-5}]$.
(c) If $N(a)$ is a prime integer then a is irreducible in $\mathbb{Z}[\sqrt{-5}]$.
(d) The numbers $2, 3, 1+\sqrt{-5}, 1-\sqrt{-5}$ are irreducible and not associates of each other. Since $6 = 2 \cdot 3 = (1+\sqrt{-5}) \cdot (1-\sqrt{-5})$, conclude that $\mathbb{Z}[\sqrt{-5}]$ is not a UFD. □

Exercise 1.6:
(a) In a principal ideal domain, the property "$I \supseteq J$" is equivalent to "there exists an ideal K such that $IK = J$".
(b) In $\mathbb{Z}[X, Y]$, there does not exist an ideal K such that $(X, Y) \cdot K = (X^2, Y^2)$. □

Exercise 1.7: (Lucas 1876) The GCD of two Fibonacci numbers is Fibonacci. □

Exercise 1.8: (Kaplansky) Define a *GCD-domain* to be a domain in which any two elements have a GCD.
a) Show that if D is such a domain, then so is $D[X]$.
b) Show that if for any two elements $u, v \in D$, either $u \,|\, v$ or $v \,|\, u$ (D is a *valuation domain*) then D is a GCD-domain. □

§2. Euclid's Algorithm

We describe Euclid's algorithm for computing the `GCD` of two positive integers

$$m_0 > m_1 > 0.$$

The algorithm amounts to constructing a sequence of remainders,

$$m_0, m_1, m_2, \ldots, m_k, \qquad (k \geq 1) \tag{2}$$

where

$$\begin{aligned} m_{i+1} &= m_{i-1} \,\mathbf{mod}\, m_i \qquad (i = 1, \ldots, k-1) \\ 0 &= m_{k-1} \,\mathbf{mod}\, m_k. \end{aligned}$$

Recall that $a \,\mathbf{mod}\, b$ is the remainder function that returns an integer in the range $[0, b)$. But this is not the only possibility (next section).

Let us prove that m_k equals $\mathtt{GCD}(m_0, m_1)$. We use the observation that if any number d divides m_i and m_{i+1}, then d divides m_{i-1} (provided $i \geq 1$) and d divides m_{i+2} (provided $i \leq k-2$). Note that m_k divides m_k and m_{k-1}. So by repeated application of the observation, m_k divides both m_0 and m_1. Next suppose d is any number that divides m_0 and m_1. Then repeated application of the observation implies d divides m_k. Thus we conclude that $m_k = \mathtt{GCD}(m_0, m_1)$.

Two pieces of data related to the $\mathtt{GCD}(m_0, m_1)$ are often important. Namely, there exist integers s, t such that

$$\mathtt{GCD}(m_0, m_1) = s m_0 + t m_1. \tag{3}$$

We call the pair (s, t) a *co-factor* of (m_0, m_1). By the *co-GCD problem*, we mean the problem of computing a co-factor for an input pair of numbers. It is easy to obtain the GCD from a co-factor.

However, most co-GCD algorithms also produces the GCD with no extra effort. By definition, an *extended GCD algorithm* solves both the GCD and co-GCD problems. The existence of co-factors will be proved by our construction of an extended GCD algorithm next.

We proceed as follows. Suppose q_i is the quotient of the ith remaindering step in (2):

$$m_{i+1} = m_{i-1} - q_i m_i \qquad (i = 2, \ldots, k-1) \tag{4}$$

We compute two auxiliary sequences

$$(s_0, s_1, \ldots, s_k) \quad \text{and} \quad (t_0, t_1, \ldots, t_k) \tag{5}$$

so that they satisfy the property

$$m_i = s_i m_0 + t_i m_1, \qquad (i = 0, \ldots, k). \tag{6}$$

Note that when $i = k$, this property is our desired equation (3). The auxiliary sequences are obtained by mirroring the remaindering step (4),

$$\left.\begin{array}{l} s_{i+1} = s_{i-1} - q_i s_i \\ t_{i+1} = t_{i-1} - q_i t_i \end{array}\right\} \qquad (i = 2, \ldots, k-1) \tag{7}$$

To initialize the values of s_0, s_1 and t_0, t_1, observe that

$$m_0 = 1 \cdot m_0 + 0 \cdot m_1$$

and

$$m_1 = 0 \cdot m_0 + 1 \cdot m_1.$$

Thus (6) is satisfied for $i = 0, 1$ if we set

$$(s_0, t_0) := (1, 0), \qquad (s_1, t_1) := (0, 1).$$

Inductively, (6) is satisfied because

$$\begin{array}{rcl} m_{i+1} & = & m_{i-1} - q_i m_i \\ & = & (s_{i-1} m_0 + t_{i-1} m_1) - q_i (s_i m_0 + t_i m_1) \\ & = & (s_{i-1} - q_i s_i) m_0 + (t_{i-1} - q_i t_i) m_1 \\ & = & s_{i+1} m_0 + t_{i+1} m_1. \end{array}$$

This completes the description and proof of correctness of the extended Euclidean algorithm.

Application. Suppose we want to compute multiplicative inverses modulo an integer m_0. An element m_1 has a multiplicative inverse modulo m_0 if and only if $\mathtt{GCD}(m_0, m_1) = 1$. Applying the extended algorithm to m_0, m_1, we obtain s, t such that

$$1 = \mathtt{GCD}(m_0, m_1) = s m_0 + t m_1.$$

But this implies

$$1 \equiv t m_1 (\bmod\, m_0),$$

i.e., t is the inverse of m_1 modulo m_0. Similarly s is the inverse of m_0 modulo m_1.

Exercises

Exercise 2.1: (i) Show that every two steps of the Euclidean algorithm reduce the (bit) size of the larger integer by at least one. Conclude that the bit complexity of the Euclidean algorithm is $O(nM_B(n))$ where $M_B(n)$ is the bit complexity of integer multiplication.
(ii) Improve this bound to $O(n^2)$. HINT: If the bit length of m_i in the remainder sequence is ℓ_i, then the bit length of q_i is at most $\ell_{i-1} - \ell_i + 1$. The ith step can be implemented in time $O(\ell_i(\ell_{i-1} - \ell_+ 1))$. □

Exercise 2.2: Consider the extended Euclidean algorithm.
(i) Show that for $i \geq 2$, we have $s_i t_i < 0$ and $s_i > 0$ iff i is even.
(ii) Show that the co-factor (s, t) computed by the algorithm satisfy $\max\{|s|, |t|\} < m_0$. □

Exercise 2.3: (Blankinship) The following is a simple basis for an extended multiple GCD algorithm. Let $N = (n_1, \ldots, n_k)^T$ be a k-column of integers and A the $k \times (k+1)$ matrix whose first column is N, and the remaining columns form an identity matrix. Now perform any sequence of row operations on A of the form "subtract an integer multiple of one row from another". It is clear that we can construct a finite sequence of such operations so that the first column eventually contains only one non-zero entry d where $d = \mathtt{GCD}(n_1, \ldots, n_k)$. If the row containing d is $(d, s_1, \ldots, s_k)$, prove that

$$d = \sum_{i=1}^{k} s_i n_i.$$

□

Exercise 2.4:
(i) Let $n_1 > n_2 > \cdots > n_k > 1$ $(k \geq 1)$ be integers. Let $S = (s_1, \ldots, s_k) \in \mathbb{Z}^k$ be called a *syzygy* of $N = (n_1, \ldots, n_k)$ if $\sum_{i=1}^k s_i n_i = 0$. Prove that the set of syzygies of N forms a $\mathbb{Z}$-module. For instance, let s_{ij} $(1 \leq i < j \leq n)$ be the k-vector $(0, \ldots, 0, n_j, 0, \ldots, 0, -n_i, 0, \ldots, 0)$ (where the only non-zero entries are at positions i and j as indicated). Clearly s_{ij} is a syzygy. This module has a finite basis (XI§1). Construct such a basis.
(ii) Two k-vectors S, S' are *equivalent* if $S - S'$ is a syzygy of N. Show that every S is equivalent to some S' where each component c of S' satisfies $|c| < n_1$. □

§3. Euclidean Ring

We define the abstract properties that make Euclid's algorithm work. A ring R is *Euclidean* if there is a function

$$\varphi : R \to \{-\infty\} \cup \mathbb{R}$$

such that

i) $b \neq 0$ and $a|b$ implies $\varphi(a) \leq \varphi(b)$;

ii) for all $r \in \mathbb{R}$, the set $\{\varphi(a) : a \in R, \varphi(a) < r\}$ is finite;

iii) for all $a, b \in R$ $(b \neq 0)$, there exists $q, r \in R$ such that

$$a = bq + r \quad \text{and} \quad \varphi(r) < \varphi(b).$$

We say that φ is an *Euclidean value function* for R, and call the q and r in iii) a *quotient* and *remainder* of a, b relative to φ. Property iii) is called the *division property* (relative to φ). We introduce the *remainder* $\mathtt{rem}(a,b)$ and *quotient* $\mathtt{quo}(a,b)$ functions that pick out some definite pair of remainder and quotient of a, b that simultaneously satisfy property iii). Note that these functions are only defined when $b \neq 0$. Often it is convenient to write these two functions using infix operators **mod** and **div**:

$$\mathtt{rem}(a,b) = a \,\mathbf{mod}\, b, \quad \mathtt{quo}(a,b) = a \,\mathbf{div}\, b. \tag{8}$$

A *Euclidean domain* is an Euclidean ring that is also a domain.

Exercise 3.1:

(a) $\mathtt{rem}(a,b) = 0$ if and only if $b|a$ (in particular, $\mathtt{rem}(0,b) = 0$).
(b) $\varphi(a) = \varphi(b)$ when a and b are associates.
(c) $\varphi(0) < \varphi(b)$ for all non-zero b. □

Our two standard domains, $\mathbb{Z}$ and $F[X]$, are Euclidean:
(A) $\mathbb{Z}$ is seen to be an Euclidean domain by letting $\varphi(n) = |n|$, the absolute value of n. There are two choices for $\mathtt{rem}(m,n)$ unless $n|m$, one positive and one negative. For instance, $\mathtt{rem}(8,5)$ can be taken to be 3 or -2. There are two standard ways to make $\mathtt{rem}(m,n)$ functional. In the present lecture, we choose the non-negative remainder. The corresponding function $\mathtt{rem}(m,n) \geq 0$ is called the *non-negative remainder function*. An alternative is to choose the remainder that minimizes the absolute value (choosing the positive one in case of a tie); this corresponds to the *symmetric remainder function*. The function $\mathtt{quo}(a,b)$ is uniquely determined once $\mathtt{rem}(a,b)$ is fixed. Again, we have the *non-negative quotient function* and the *symmetric quotient function*.
(B) If F is any field, the following *division property for polynomials* holds: for $A, B \in F[X]$ where $B \neq 0$, there exists $Q, R_0 \in F[X]$ such that

$$A = BQ + R_0, \qquad \deg(R_0) < \deg(B).$$

This can be proved by the synthetic division algorithm which one learns in high school. It follows that the polynomial ring $F[X]$ is an Euclidean domain, as witnessed by the choice $\varphi(P) = \deg P$, for $P \in F[X]$. In fact, the synthetic division algorithm shows that $\mathtt{rem}(P,Q)$ and $\mathtt{quo}(P,Q)$ are uniquely determined. Despite property ii) in the definition of φ, there may be infinitely many $a \in R$ with $\varphi(a) < r$. This is the case if $R = F[X]$ with F infinite.

Lemma 2 *If a is a proper divisor of b then $\varphi(a) < \varphi(b)$.*

Proof. By the division property, $a = bq + r$ where $\varphi(r) < \varphi(b)$. Now $r \neq 0$ since otherwise b divides a, which contradicts the assumption that a properly divides b. Since a properly divides b, let $b = ac$ for some c. Then $r = a - bq = a(1 - cq)$. Then property i) implies $\varphi(a) \leq \varphi(r) < \varphi(b)$. **Q.E.D.**

Theorem 3 *An Euclidean ring is a principal ideal ring. Indeed, if $b \in I \setminus \{0\}$ is such that $\varphi(b)$ is minimum then $I = \mathtt{Ideal}(b)$.*

Proof. Let I be any ideal. By property ii), there exists a $b \in I \setminus \{0\}$ such that $\varphi(b)$ is minimum. To show $I = \mathtt{Ideal}(b)$, it suffices to show that b divides any $c \in I \setminus \{0\}$. By the division property, $c = bq + r$ where $\varphi(r) < \varphi(b)$. If $r \neq 0$ then we have found an element $r = c - bq \in I \setminus \{0\}$ with $\varphi(r) < \varphi(b)$, contradicting our choice of b. If $r = 0$ then $b|c$. **Q.E.D.**

The converse is not true (see exercise).

Lemma 4 *In a principal ideal ring R, the non-zero irreducible elements are prime.*

Proof. Let $p \in R \setminus \{0\}$ be irreducible. If p divides the product bc, we must prove that p divides b or p divides c. Since R is a principal ideal ring, $\texttt{Ideal}(p, b) = \texttt{Ideal}(u)$ for some $u = \alpha p + \beta b$. So $u|p$. Since p is irreducible, u is a unit or an associate of p. If u is an associate, and since $u|b$, we have $p|b$, which proves the lemma. If u is a unit then $uc = \alpha pc + \beta bc$. Since $p|bc$, this implies $p|uc$, *i.e.*, $p|c$. **Q.E.D.**

Theorem 5 *In a principal ideal ring, the factorization of a non-unit into irreducible non-units is unique, up to reordering and associates.*

Proof. Suppose $b \in R$ is a non-unit with two factorizations into irreducible non-units:

$$b = p_1 p_2 \cdots p_m = q_1 q_2 \cdots q_n, \qquad 1 \le m \le n.$$

We use induction on m. If $m = 1$ then clearly $n = 1$ and $q_1 = p_1$. Assume $m > 1$. Since p_1 is a prime, it must divide some q_i, and we might as well assume $p_1|q_1$. But q_1 is also a prime and so it must be an associate of p_1. Dividing by p_1 on both sides of the expression, it follows that $p_2 \cdots p_m = q_2' q_3 \cdots q_n$ where q_2' is an associate of q_2. By induction, $m = n$ and the two factorizations are unique up to reordering and associates. This implies our theorem. **Q.E.D.**

Corollary 6 *An Euclidean domain is a UFD.*

Remainder sequences. Relative to the remainder and quotient functions, we define a *remainder sequence* for any pair $a, b \in R$ to be a sequence

$$a_0, a_1, \ldots, a_k \qquad (k \ge 1) \tag{9}$$

such that $a_0 = a, a_1 = b$ and for $i = 1, \ldots, k-1$, a_{i+1} is an associate of $\texttt{rem}(a_{i-1}, a_i)$, and $\texttt{rem}(a_{k-1}, a_k) = 0$. Note that termination of this sequence is guaranteed by property ii). The remainder sequence is *strict* if a_{i+1} is any remainder of a_{i-1}, a_i for all i; it is *Euclidean* if $a_{i+1} = \texttt{rem}(a_{i-1}, a_i)$.

Example: In $\mathbb{Z}$, $(13, 8, 5, 3, 2, 1)$, $(13, 8, -3, 2, \pm 1)$ and $(13, 8, 3, -1)$ are all strict remainder sequences for $(13, 8)$. A non-strict remainder sequence for $(13, 8)$ is $(13, 8, -5, 2, 1)$. ■

Associated to each remainder sequence (9) is another sequence

$$q_1, q_2, \ldots, q_k \tag{10}$$

where $a_{i-1} = a_i q_i + u_i a_{i+1}$ ($i = 1, \ldots, k-1$, u_i is a unit) and $a_{k-1} = a_k q_k$. We call (10) the *quotient sequence associated to*remainder sequence!2@
it see also quotient sequence the remainder sequence.

Norms. In some books, the function φ is restricted to the range $\mathbb{N}$. This restriction does not materially affect the concept of Euclidean domains, and has the advantage that property ii) is automatic. Our formulation makes it more convenient to formulate functions φ that have other desirable properties. For instance, we often find the properties:

$$\varphi(ab) = \varphi(a)\varphi(b)$$

and

$$\varphi(a+b) \le \varphi(a) + \varphi(b).$$

In this case, we call φ a *multiplicative norm* (or *valuation*), and might as well (why?) assume $\varphi(0) = 0$ and $\varphi(1) = 1$. Similarly, if $\varphi(ab) = \varphi(a) + \varphi(b)$ and $\varphi(a+b) = O(1) + \max\{\varphi(a), \varphi(b)\}$ then we call φ an *additive norm*, and might as well assume $\varphi(0) = -\infty$ and $\varphi(1) = 0$. Clearly φ is multiplicative implies $\log \varphi$ is additive.

Remarks. The number rings $\mathbb{O}_\alpha$ (§VI.3) have properties similar to the integers. In particular, they support the concepts of divisibility and factorization. Gauss pointed out that such rings may not be a UFD (class number 1). Even when $\mathbb{O}_\alpha$ is a UFD, it may not be Euclidean; the "simplest" example is $\mathbb{O}_{\sqrt{-19}}$ (see [174]). An obvious candidate for the Euclidean value function φ is the norm of algebraic numbers, but other functions are conceivable. Turning now to the quadratic number rings (*i.e.*, $\mathbb{O}_\alpha$ where $\alpha = \sqrt{d}$ and d is squarefree), Kurt Heegner [Diophantische Analysis und Modulfunktionen, *Mathematische Zeitschrift*, vol. 56, 1952, pp.227–253] was the first[2] to prove that there are exactly nine such UFD's in which $d < 0$, *viz.*, $d = -1, -2, -3, -7, -11, -19, -43, -67, -163$. In contrast, it is conjectured that there are infinitely many UFD's among the real (*i.e.*, $d > 0$) quadratic number fields. It is known that there are precisely 21 real quadratic domains that support the Euclidean algorithm. Currently, the most general GCD algorithms are from Kaltofen and Rolletschek [98] who presented polynomial-time GCD algorithms for each quadratic number ring $\mathbb{O}_{\sqrt{d}}$ that is a UFD, not necessarily Euclidean.

Exercises

Exercise 3.2: Justify the above remarks about multiplicative and additive norms. □

Exercise 3.3: Verify that the Euclidean algorithm computes the `GCD` in an Euclidean domain, relative to the function `rem`(a, b). □

Exercise 3.4: Show that the number ring $\mathbb{O}_{\mathbf{i}}$ $(= \mathbb{Z}[\mathbf{i}] = \{a + \mathbf{i}b \,:\, a, b \in \mathbb{Z}\})$ of Gaussian integers forms an Euclidean domain with respect to the multiplicative norm $\varphi(a + \mathbf{i}b) = a^2 + b^2$. Use the identity of Fibonacci,

$$\varphi(xy) = \varphi(x)\varphi(y), \qquad x, y \in \mathbb{Z}[\mathbf{i}].$$

What are the possible choices for defining the remainder and quotient functions here? □

Exercise 3.5: Consider the number ring $R = \mathbb{O}_{\sqrt{-19}}$. Note that $\mathbb{O}_{\sqrt{-19}} = \{m + n\omega : m, n \in \mathbb{Z}\}$ where $\omega = \frac{1+\sqrt{-19}}{2}$. The norm of $x + y\sqrt{-19} \in \mathbb{Q}(\sqrt{-19})$ is by $x^2 + 19y^2$.
(a) R is a principal ideal domain.
(b) R is not an Euclidean domain with respect to the standard norm function. HINT: What is the remainder of 5 divided by $\sqrt{-19}$? □

[2]This result is often attributed to Stark and Baker who independently proved this in 1966. See Buell [34].

§4. The Half-GCD Problem

An exercise in §2 shows that that Euclid's algorithm for integers has bit complexity $\Theta(n^2\mathcal{L}(n))$. Knuth [104] is the first to obtain a subquadratic complexity for this problem. In 1971, Schönhage [181] improved it to the current record of $O(\mathrm{M}_B(n)\log n) = n\mathcal{L}^2(n)$. Since $F[X]$ is an Euclidean domain, Euclid's algorithm can be applied to polynomials as well. Given $P_0, P_1 \in F[X]$ with $n = \deg P_0 > \deg P_1 \geq 0$, consider its Euclidean remainder sequence

$$P_0, P_1, P_2, \ldots, P_h \qquad (h \geq 1). \tag{11}$$

Call the sequence *normal* if $\deg P_{i-1} = 1 + \deg P_i$ for $i = 2, \ldots, h$. A random choice for P_0, P_1 gives rise to a normal sequence with high probability (this is because non-normal sequences arise from the vanishing of certain determinants involving the the coefficients of P_0, P_1, Lecture III). The algebraic complexity of this Euclidean algorithm is therefore

$$O(\mathrm{M}_A(n)n) = O(n^2 \log n) \tag{12}$$

where $\mathrm{M}_A(n) = O(n \log n)$ is the algebraic complexity of polynomial multiplication (Lecture 1). Moenck [141] improves (12) to $O(\mathrm{M}_A(n)\log n)$ in case the remainder sequence is normal. Aho-Hopcroft-Ullman [2] incorrectly claimed that the Moenck algorithm works in general. Brent-Gustavson-Yun [26] presented a corrected version without a proof. Independently, Thull and Yap [204] rectified the algorithm with a proof, reproduced below. This lecture follows the unified framework for both the polynomial and integer GCD algorithms, first presented in [204].

Let us motivate the approach of Schönhage and Moenck. These ideas are easiest seen in the case of the polynomials. If the sequence (11) is normal with $n = h$ and $\deg P_i = n - i$ $(i = 0, \ldots, n)$ then

$$\sum_{i=0}^{h} \deg P_i = n(n-1)/2.$$

So any algorithm that explicitly computes each member of the remainder sequence has at least quadratic complexity. On the other hand, if

$$Q_1, Q_2, \ldots, Q_h$$

is the quotient sequence associated to (11), then it is not hard to show that

$$\sum_{i=1}^{h} \deg Q_i = n. \tag{13}$$

Indeed, we can quickly (in $O(n \log^2 n)$ time, see Exercise below) obtain any member of the remainder sequence from the Q_i's. This suggests that we redirect attention to the quotient sequence.

Matrix Terminology. To facilitate description of our algorithms, we resort to the language of matrices and vectors. In this lecture, all matrices will be 2×2 matrices and all vectors will be column 2-vectors. The identity matrix is denoted by E. The Euclidean algorithm as embodied in (11) can be viewed as a sequence of transformations of 2-vectors:

$$\begin{bmatrix} P_0 \\ P_1 \end{bmatrix} \xrightarrow{M_1} \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} \xrightarrow{M_2} \cdots \xrightarrow{M_{h-1}} \begin{bmatrix} P_{h-1} \\ P_h \end{bmatrix} \xrightarrow{M_h} \begin{bmatrix} P_h \\ 0 \end{bmatrix}. \tag{14}$$

Precisely, if U, V are vectors and M a matrix, we write

$$U \xrightarrow{M} V$$

to mean that $U = MV$. Hence equation (14) can be correctly interpreted if we define

$$M_i = \begin{bmatrix} Q_i & 1 \\ 1 & 0 \end{bmatrix}.$$

In general, an *elementary matrix* refers to a matrix of the form $M = \begin{bmatrix} Q & 1 \\ 1 & 0 \end{bmatrix}$ where Q is a polynomial with positive degree. We call Q the *partial!quotient* in M. A *regular matrix* M is a product of zero or more elementary matrices,

$$M = M_1 M_2 \cdots M_k \qquad (k \geq 0). \tag{15}$$

When $k = 0$, M is interpreted to be E. The sequence $Q_1, \ldots, Q_k$ of partial quotients associated with the elementary matrices $M_1, \ldots, M_k$ in equation (15) is called the *sequence of partial quotients* of M. Also, Q_k is called its *last partial quotient* Note that regular matrices have determinant ± 1 and so are invertible. Regular matrices arise because

$$U \xrightarrow{M} V \text{ and } V \xrightarrow{M'} W \text{ implies } U \xrightarrow{MM'} W.$$

Our terminology here is motivated by the connection to continued fractions (for instance, regular matrices are related to regular continued fractions).

We are ready to define the *half-GCD* (or, HGCD) *problem* for a polynomial ring $F[X]$:

Given $P_0, P_1 \in F[X]$ *where* $n = \deg P_0 > \deg P_1$, *compute a regular matrix*

$$M := \mathtt{hGCD}(P_0, P_1)$$

such that if

$$\begin{bmatrix} P_0 \\ P_1 \end{bmatrix} \xrightarrow{M} \begin{bmatrix} P_2 \\ P_3 \end{bmatrix}$$

then

$$\deg P_2 \geq n/2 > \deg P_3. \tag{16}$$

In general, we say two numbers a, b *straddle* a third number c if $a \geq c > b$. Thus $\deg P_2, \deg P_3$ straddle $n/2$ in equation (16).

Now we show how to compute the GCD using the `hGCD`-subroutine. In fact the algorithm is really a "co-GCD" (§2) algorithm:

```
POLYNOMIAL CO-GCD ALGORITHM:
Input: A pair of polynomials with $\deg P_0 > \deg P_1$
Output: A regular matrix $M = \mathtt{co\text{-}GCD}(P_0, P_1)$ such that
        $\begin{bmatrix} P_0 \\ P_1 \end{bmatrix} \xrightarrow{M} \begin{bmatrix} \mathtt{GCD}(P_0,P_1) \\ 0 \end{bmatrix}.$
    [1] Compute $M_0 \leftarrow \mathtt{hGCD}(P_0, P_1)$.
    [2] Recover $P_2, P_3$ via
            $\begin{bmatrix} P_2 \\ P_3 \end{bmatrix} \leftarrow M_0^{-1} \cdot \begin{bmatrix} P_0 \\ P_1 \end{bmatrix}.$
    [3] if $P_3 = 0$ then return($M_0$).
        else, perform one step of the Euclidean algorithm,
            $\begin{bmatrix} P_2 \\ P_3 \end{bmatrix} \xrightarrow{M_1} \begin{bmatrix} P_3 \\ P_4 \end{bmatrix}$
        where $M_1$ is an elementary matrix.
    [4] if $P_4 = 0$ then return($M_0 M_1$).
        else, recursively compute $M_2 \leftarrow \mathtt{co\text{-}GCD}(P_3, P_4)$
        return($M_0 M_1 M_2$).
```

The correctness of this algorithm is clear. The reason for step [3] is to ensure that in our recursive call, the degree of the polynomials is less than $n/2$. The algebraic complexity $T(n)$ of this algorithm satisfies

$$T(n) = T'(n) + O(\mathrm{M}_A(n)) + T(n/2)$$

where $T'(n)$ is the complexity of the HGCD algorithm. Let us assume that

$$\mathrm{M}_A(n) = O(T'(n)), \qquad T'(\alpha n) \le \alpha T'(n).$$

For instance, the first relation holds if $T'(n) = \Omega(\mathrm{M}_A(n))$; the second relation holds if $T'(n)$ is bounded by a polynomial. In particular, they hold if $T'(n) = \Theta(M(n)\log n)$, which is what we will demonstrate below. Then

$$T(n) = O(T'(n) + T'(n/2) + T'(n/4) + \cdots) = O(T'(n)).$$

In conclusion, the complexity of the GCD problem is the same order of the complexity as the HGCD problem. Henceforth, we focus on the HGCD problem.

Remarks: The above complexity counts ring operations from F. If we count operations in $F[X]$, the complexity is $O(n \log n)$. This counting is more general because it applies also to the case of integer HGCD to be discussed. Strassen [196] has proved that this complexity is optimal: $O(n \log n)$ is both necessary and sufficient.

EXERCISES

Exercise 4.1: Recall the auxiliary sequences $(s_0, s_1, \ldots, s_k)$ and $(t_0, t_1, \ldots, t_k)$ computed in the Extended Euclidean algorithm (§2) for the GCD of a pair of integers $a_0 > a_1 > 1$. Show that the appropriate elementary matrices have the form $\begin{bmatrix} s_i & t_i \\ s_{i+1} & t_{i+1} \end{bmatrix}^{-1}$. □

Exercise 4.2:
(a) Show equation (13).

(b) Show that in $O(n \log^2 n)$ time, we can reconstruct the polynomials S, T from the quotient sequence $Q_1, \ldots, Q_k$ where $SP_0 + TP_1 = \mathtt{GCD}(P_0, P_1)$. HINT: note that $\begin{bmatrix} P_i \\ P_{i+1} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & -Q_i \end{bmatrix} \begin{bmatrix} P_{i-1} \\ P_i \end{bmatrix}$, and use a balanced binary tree scheme. □

§5. Properties of the Norm

For the rest of this Lecture, the domain D refers to either $\mathbb{Z}$ or $F[X]$. In this case, we define the *(additive) norm* $\|a\|$ of $a \in D$ thus:

$$\|a\| := \begin{cases} \log_2 |a| & \text{if } a \in \mathbb{Z}, \\ \deg(a) & \text{if } a \in F[X]. \end{cases}$$

The previous section describes the polynomial HGCD problem. A similar, albeit more complicated, development can be carried out for integers. We now describe a common framework for both the integer and polynomial HGCD algorithms.

The following properties are easy to check:

a) $\|a\| \in \{-\infty\} \cup \mathbb{R}^*$ where $\mathbb{R}^*$ is the set of non-negative real numbers.

b) $\|a\| = -\infty \iff a = 0$

c) $\|a\| = 0 \iff a$ is a unit.

d) $\|-a\| = \|a\|$

e) $\|ab\| = \|a\| + \|b\|$

f) $\|a + b\| \leq 1 + \max\{\|a\|, \|b\|\}$.

The last two properties imply that the norm is additive (§3). However, polynomials satisfy the stronger *non-Archimedean property* (cf. [111, p.283]):

$$\|a + b\| \leq \max\{\|a\|, \|b\|\}.$$

It is this non-Archimedean property that makes polynomials relatively easier than integers. This property implies

$$\|a + b\| = \max\{\|a\|, \|b\|\} \qquad \text{if } \|a\| \neq \|b\|. \tag{17}$$

Exercise 5.1: Prove this. □

This norm function serves as the Euclidean value function for D. In particular, the *division property* relative to the norm holds: for any $a, b \in D$, $b \neq 0$, there exists $q, r \in D$ such that

$$a = qb + r, \quad \|r\| < \|b\|.$$

The remainder and quotient functions, $\mathtt{rem}(a,b)$ and $\mathtt{quo}(a,b)$, can be defined as before. Recall that these functions are uniquely defined for polynomials, but for integers, we choose $\mathtt{rem}(a,b)$ to be the non-negative remainder function. Note that in the polynomial case,

$$\|a\,\mathbf{mod}\,X^m\| \quad\leq\quad \min\{\|a\|, m-1\} \tag{18}$$

$$\|a\,\mathbf{div}\,X^m\| \quad=\quad \begin{cases} \|a\|-m & \text{if } \|a\|\geq m \\ \\ -\infty & \text{else.}\end{cases} \tag{19}$$

Matrices and vectors. The previous section (§4) introduced the matrix concepts we needed. Those definitions extend in the obvious way to our present setting, except for one place, where we need special care: A matrix of the form $M=\begin{bmatrix} q & 1 \\ 1 & 0\end{bmatrix}$ (where $q\in D$) is denoted $\langle q\rangle$. A matrix is *elementary* if it has the form $\langle q\rangle$ where $\|q\|>0$ in the case of polynomials (as before), and $q>0$ in the case of integers. A finite product $\langle q_1\rangle\langle q_2\rangle\cdots\langle q_k\rangle$ $(k\geq 0)$ of elementary matrices is again called *regular* and may be denoted $\langle q_1,\ldots,q_k\rangle$. When $k=2$, the careful reader will note the clash with our notation for scalar products, but this ambiguity should never confuse.

A regular matrix $M=\begin{bmatrix} p & q \\ r & s\end{bmatrix}$ satisfies the following *ordering property*:

$$M\neq E \Rightarrow \quad \|p\|\geq\max\{\|q\|,\|r\|\}\geq\min\{\|q\|,\|r\|\}\geq\|s\|, \quad \|p\|>\|s\|. \tag{20}$$

Exercise 5.2:

a) Prove the ordering property.

b) If all the inequalities in the definition of the ordering property are in fact strict and $\|s\|\geq 0$, we say M satisfies the *strict ordering property*. Show that the product of three or more elementary matrices has the strict ordering property.

c) Bound the norms of the entries of the matrix $\langle q_1,\ldots,q_k\rangle$ in terms of the individual norms $\|q_i\|$. □

For vectors U,V and matrix M, we write

$$U\xrightarrow{M}V$$

(or simply, $U\longrightarrow V$) if $U=MV$. We say M *reduces*matrix!regular!reducing a vector U to V if M is a regular matrix. If, in addition, $U=\begin{bmatrix} a \\ b\end{bmatrix}, V=\begin{bmatrix} a' \\ b'\end{bmatrix}$ such that $\|a\|>\|b\|$ and $\|a'\|>\|b'\|$, then we say this is an *Euclidean reduction*.

A matrix is *unimodular* if[3] it has determinant ± 1. Clearly regular matrices are unimodular. Thus their inverses are easily obtained: if $M=\begin{bmatrix} p & q \\ r & s\end{bmatrix}$ is regular with determinant $\det M=\delta$ then

$$M^{-1}=\delta\begin{bmatrix} s & -q \\ -r & p\end{bmatrix}=\pm\begin{bmatrix} s & -q \\ -r & p\end{bmatrix}. \tag{21}$$

If $U=\begin{bmatrix} a \\ b\end{bmatrix}$ then we write $\mathtt{GCD}(U)$ for the GCD of a and b. We say U,V are *equivalent* if $U=MV$ for some unimodular matrix M.

[3]In some literature, "unimodular" refers to determinant $+1$.

Lemma 7 *U and V are equivalent if and only if* $\mathtt{GCD}(U) = \mathtt{GCD}(V)$.

Proof. It is easy to check that if the two vectors are equivalent then they must have the same GCD. Conversely, by Euclid's algorithm, they are both equivalent to the vector $\begin{bmatrix} g \\ 0 \end{bmatrix}$ where g is their common GCD. **Q.E.D.**

It follows that this binary relation between vectors is indeed a mathematical equivalence relation. The following is a key property of Euclidean remainder sequences (§3):

Lemma 8 *Given a, b, a', b' such that $\|a\| > \|b\| \geq 0$. The following are equivalent:*
(i) *a', b' are consecutive elements in the Euclidean remainder sequence of a, b.*
(ii) *There is a regular matrix M such that*

$$\begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{M} \begin{bmatrix} a' \\ b' \end{bmatrix} \tag{22}$$

and either $\|a'\| > \|b'\| \geq 0$ (polynomial case) or $a' > b' > 0$ (integer case).

Proof. If (i) holds then we can (by Euclid's algorithm) find some regular matrix M satisfying (ii). Conversely assume (ii). We show (i) by induction on the number of elementary matrices in the product M. The result is immediate if $M = E$. If M is elementary, then (i) follows from the division property for our particular choices for D. Otherwise let $M = M''M'$ where M' is elementary and M'' is regular. Then for some a'', b'',

$$\begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{M''} \begin{bmatrix} a'' \\ b'' \end{bmatrix} \xrightarrow{M'} \begin{bmatrix} a' \\ b' \end{bmatrix}.$$

But $a'' = a'q' + b'$ and $b'' = a'$ where q' is the partial quotient of M'. We verify that this means $\|a''\| > \|b''\|$. By induction, a'', b'' are consecutive elements in a strict remainder sequence of a, b. Then (i) follows. **Q.E.D.**

Exercises

Exercise 5.3: In Exercise 2.1, we upper bound the length of the integer Euclidean remainder sequence of $a > b > 0$ by $2\log_2 a$. We now give a slight improvement.
(a) Prove that for $k \geq 1$,

$$\underbrace{\langle 1, \ldots, 1 \rangle}_{k} = \langle 1 \rangle^k = \begin{bmatrix} F_{k+1} & F_k \\ F_k & F_{k-1} \end{bmatrix}$$

where $\{F_i\}_{i \geq 0}$ is the Fibonacci sequence defined by: $F_0 = 0, F_1 = 1$ and $F_{i+1} = F_i + F_{i-1}$ $(i \geq 1)$.
(b) Let ϕ be the positive root of the equation $X^2 - X - 1 = 0$ (so $\phi = (1 + \sqrt{5})/2 = 1.618...$). Prove inductively that

$$(1.5)^{k-1} \leq F_{k+1} \leq \phi^k.$$

(c) Say $(q_1, q_2, \ldots, q_k)$ is the quotient sequence associated to the remainder sequence of $a > b > 0$. If $\langle q_1, \ldots, q_k \rangle = \begin{bmatrix} p & q \\ r & s \end{bmatrix}$ prove that

$$\|p\| \leq \|a\| - \|b\|.$$

(d) Conclude that $k < 1 + \log_{1.5} a$.

(e) (Lamé) Give an exact worst case bound on k in terms of ϕ and its conjugate $\widehat{\phi}$. □

§6. Polynomial HGCD

We describe the polynomial HGCD algorithm and prove its correctness.

Parallel reduction. The idea we exploit in the HGCD algorithm might be called "parallel reduction". Suppose we want to compute the HGCD of the pair of polynomials $A, B \in F[X]$ where $\deg A = 2m$. First we truncate these polynomials to define

$$\begin{bmatrix} A_0 \\ B_0 \end{bmatrix} := \begin{bmatrix} A \,\mathbf{div}\, X^m \\ B \,\mathbf{div}\, X^m \end{bmatrix}. \tag{23}$$

Suppose that R is the matrix returned by $HGCD(A_0, B_0)$; so

$$\begin{bmatrix} A_0 \\ B_0 \end{bmatrix} \xrightarrow{R} \begin{bmatrix} A'_0 \\ B'_0 \end{bmatrix} \tag{24}$$

for some A'_0, B'_0. Then we can define A', B' via

$$\begin{bmatrix} A \\ B \end{bmatrix} \xrightarrow{R} \begin{bmatrix} A' \\ B' \end{bmatrix}. \tag{25}$$

Two reductions by the same matrix are said to be *parallel*. Thus (24) and (25) are parallel reductions. If A', B' turn out to be two consecutive terms in the remainder sequence of A, B, then we may have gained something! This is because we had computed R without looking at the lower order coefficients of A, B. But we need another property for R to be useful. We want the degrees of A', B' to straddle a sufficiently small value below $2m$. By definition of HGCD, the degrees of A'_0, B'_0 straddle $m/2$. A reasonable expectation is that the degrees of A', B' straddle $3m/2$. This would be the case if we could, for instance, prove that

$$\deg(A') = m + \deg(A'_0), \qquad \deg(B') = m + \deg(B'_0).$$

This is not quite correct, as we will see. But it will serve to motivate the following outline of the HGCD algorithm.

Outline. Given A, B with $\deg(A) = 2m > \deg(B) > 0$, we recursively compute $R \leftarrow HGCD(A_0, B_0)$ as above. Now use R to carry out the reduction of (A, B) to (A', B'). Note that although the degrees of A', B' straddle $3m/2$, we have no upper bound on the degree of A'. Hence we perform one step of the Euclidean reduction:

$$\begin{bmatrix} A' \\ B' \end{bmatrix} \xrightarrow{\langle Q \rangle} \begin{bmatrix} C \\ D \end{bmatrix}.$$

Now the degree of $C = B'$ is less than $3m/2$. We can again truncate the polynomials C, D via

$$\begin{bmatrix} C_0 \\ D_0 \end{bmatrix} := \begin{bmatrix} C \,\mathbf{div}\, X^k \\ D \,\mathbf{div}\, X^k \end{bmatrix}.$$

for a certain $k \geq 0$. Intuitively, we would like to pick $k = m/2$. Then we make our second recursive call to compute $S \leftarrow HGCD(C_0, D_0)$. We use S to reduce (C, D) to (C', D'). Hopefully, the degrees of C' and D' straddle m, which would imply that our output matrix is

$$R \cdot \langle Q \rangle \cdot S$$

The tricky part is that k cannot be simply taken to be $m/2$. This choice is correct only if the remainder sequence is normal, as Moenck assumed. Subject to a suitable choice of k, we have described the HGCD algorithm.

We are ready to present the actual algorithm. We now switch back to the norm notation, $\|A\|$ instead of $\deg(A)$, to conform to the general framework.

```
Algorithm Polynomial HGCD$(A, B)$:
    **Input:** $A, B$ are univariate polynomials with $\|A\| > \|B\| \geq 0$.
    **Output:** a regular matrix $M$ which reduces $(A, B)$ to $(C', D')$
            where $\|C'\|, \|D'\|$ straddle $\|A\|/2$.
    [1]  $m \leftarrow \left\lceil \frac{\|A\|}{2} \right\rceil$;      {This is the magic threshold}
         if $\|B\| < m$ then return$(E)$;
    [2]  $\begin{bmatrix} A_0 \\ B_0 \end{bmatrix} \leftarrow \begin{bmatrix} A \,\mathbf{div}\, X^m \\ B \,\mathbf{div}\, X^m \end{bmatrix}$.
             {now $\|A_0\| = m'$ where $m + m' = \|A\|$}
         $R \leftarrow \mathtt{hGCD}(A_0, B_0)$;
             {$\left\lceil \frac{m'}{2} \right\rceil$ is the magic threshold for this recursive call}
         $\begin{bmatrix} A' \\ B' \end{bmatrix} \leftarrow R^{-1} \begin{bmatrix} A \\ B \end{bmatrix}$;
    [3]  if $\|B'\| < m$ then return$(R)$;
    [4]  $Q \leftarrow A' \,\mathbf{div}\, B'$; $\begin{bmatrix} C \\ D \end{bmatrix} \leftarrow \begin{bmatrix} B' \\ A' \,\mathbf{mod}\, B' \end{bmatrix}$;
    [5]  $l \leftarrow \|C\|$; $k \leftarrow 2m - l$;      {now $l - m < \left\lceil \frac{m'}{2} \right\rceil$}
    [6]  $C_0 \leftarrow C \,\mathbf{div}\, X^k$; $D_0 \leftarrow D \,\mathbf{div}\, X^k$;      {now $\|C_0\| = 2(l - m)$}
         $S \leftarrow \mathtt{hGCD}(C_0, D_0)$;
             {$l - m$ is magic threshold for this recursive call.}
    [7]  $M \leftarrow R \cdot \langle Q \rangle \cdot S$; return$(M)$;
```

The programming variables in this algorithm are illustrated in the following figure.

To prove the correctness of this lemma, we must show that the output matrix M satisfies

$$\begin{bmatrix} A \\ B \end{bmatrix} \xrightarrow{M} \begin{bmatrix} C' \\ D' \end{bmatrix}, \qquad \|C'\| \geq \left\lceil \frac{\|A\|}{2} \right\rceil > \|D'\|. \tag{26}$$

The Basic Setup. Let $A, B \in F[X]$, $\|A\| > \|B\| \geq 0$ and $m \geq 1$ be given. Define A_0, B_0 as in equation (23). This determines A_1, B_1 via the equation

$$\begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} A_0 X^m + A_1 \\ B_0 X^m + B_1 \end{bmatrix} = \begin{bmatrix} A_0 & A_1 \\ B_0 & B_1 \end{bmatrix} \begin{bmatrix} X^m \\ 1 \end{bmatrix}. \tag{27}$$

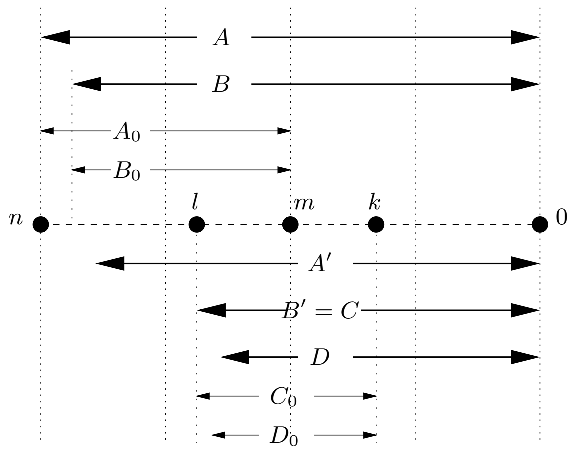

Figure 1: Variables in the polynomial HGCD algorithm.

Now let M be any given regular matrix. This determines A'_0, B'_0, A'_1, B'_1 via

$$\begin{bmatrix} A'_0 & A'_1 \\ B'_0 & B'_1 \end{bmatrix} := M^{-1} \begin{bmatrix} A_0 & A_1 \\ B_0 & B_1 \end{bmatrix}. \tag{28}$$

Finally, define A', B' via

$$\begin{bmatrix} A' \\ B' \end{bmatrix} := \begin{bmatrix} A'_0 & A'_1 \\ B'_0 & B'_1 \end{bmatrix} \begin{bmatrix} X^m \\ 1 \end{bmatrix}. \tag{29}$$

Hence we have the "parallel" reductions,

$$\begin{bmatrix} A_0 \\ B_0 \end{bmatrix} \xrightarrow{M} \begin{bmatrix} A'_0 \\ B'_0 \end{bmatrix}, \quad \begin{bmatrix} A \\ B \end{bmatrix} \xrightarrow{M} \begin{bmatrix} A' \\ B' \end{bmatrix}.$$

Lemma 9 (Correctness Criteria) *Let A, B, m, M be given, as in the Basic Setup, and define the remaining notations $A_i, B_i, A'_i, B'_i, A', B'$ $(i = 0, 1)$ as indicated. If*

$$\|A'_0\| > \|B'_0\|, \tag{30}$$
$$\|A_0\| \le 2\|A'_0\| \tag{31}$$

then

$$\|A'\| = m + \|A'_0\|, \qquad \|B'\| \le m + \max\{\|B'_0\|, \|A_0\| - \|A'_0\| - 1\}.$$

In particular,

$$\|A'\| > \|B'\|.$$

Proof. Let $M = \begin{bmatrix} P & Q \\ R & S \end{bmatrix}$. First observe that $\|A'_0\| > \|B'_0\|$ and $A_0 = A'_0 P + B'_0 Q$ implies $\|A_0\| = \|A'_0\| + \|P\|$. Hence (31) is equivalent to

$$\|P\| \le \|A'_0\|. \tag{32}$$

Since $M^{-1} = \pm \begin{bmatrix} S & -Q \\ -R & P \end{bmatrix}$ and $A'_1 = \pm(A_1 S - B_1 Q)$,

$$\|A'_1\| \le \max\{\|A_1 S\|, \|B_1 Q\|\} < m + \|P\| \le m + \|A'_0\|$$

Hence $A' = A'_0 X^m + A'_1$ implies $\|A'\| = m + \|A'_0\|$, as desired.

From $B'_1 = \pm(-A_1 R + B_1 P)$ we get $\|B'_1\| \le m - 1 + \|P\| = m - 1 + \|A_0\| - \|A'_0\|$. From $B' = B'_0 X^m + B'_1$ we get the desired inequality $\|B'\| \le m + \max\{\|B'_0\|, \|A_0\| - \|A'_0\| - 1\}$. **Q.E.D.**

We call the requirement (31) the (lower) "threshold" for $\|A'_0\|$. This threshold is the reason for the lower bound on $\|C'\|$ in the HGCD output specification (26).

Finally we prove the correctness of the HGCD algorithm.

Lemma 10 (HGCD Correctness) *Algorithm HGCD is correct: with input polynomials A, B where $\|A\| > \|B\| \ge 0$, it returns a regular matrix M satisfying (26).*

Proof. To keep track of the proof, the following sequence of reductions recalls the notations of the algorithm:

$$\begin{bmatrix} A \\ B \end{bmatrix} \xrightarrow{R} \begin{bmatrix} A' \\ B' \end{bmatrix} \xrightarrow{\langle Q \rangle} \begin{bmatrix} C \\ D \end{bmatrix} \xrightarrow{S} \begin{bmatrix} C' \\ D' \end{bmatrix}. \tag{33}$$

The algorithm returns a matrix in steps [1], [3] or [7]. Only when the algorithm reaches step [7] does the full sequence (33) take effect. It is clear that the returned matrix is always regular. So it remains to check the straddling condition of equation (26). In step [1], the result is clearly correct.

Consider the matrix R returned in step [3]: the notations m, A_0, B_0, A', B' in the algorithm conform to those in Correctness Criteria (lemma 9), after substituting R for M. By induction hypothesis, the matrix R returned by the first recursive call (step [2]) satisfies

$$\|A'_0\| \ge \lceil m'/2 \rceil > \|B'_0\|, \qquad (m' = \|A_0\|)$$

where $\begin{bmatrix} A_0 \\ B_0 \end{bmatrix} \xrightarrow{R} \begin{bmatrix} A'_0 \\ B'_0 \end{bmatrix}$. Then lemma 9 implies $\|A'\| = m + \|A'_0\| \ge m$. Since $m > \|B'\|$ is a condition for exit at step [3], it follows that the straddling condition (26) is satisfied at this exit.

Finally consider the matrix M returned in step [7]. Since we did not exit in step [3], we have $m \le \|B'\|$. In step [4] we form the quotient Q and remainder D of A' divided by B'. Also we renamed B' to C. Hence $m \le l$ where $l = \|C\|$. To see that C_0 is properly computed, let us verify

$$l \ge k \ge 0. \tag{34}$$

The first inequality in (34) follows from $l \ge m \ge m + (m - l) = k$. To see the second, $l = \|B'\| \le m + \max\{\|B'_0\|, \|A_0\| - \|A'_0\| + 1\}$ (Correctness Criteria) and so $l \le m + \max\{\lceil m'/2 \rceil - 1, \lfloor m'/2 \rfloor + 1\} \le m + \lfloor m'/2 \rfloor + 1$. Thus $l - m \le \lfloor m'/2 \rfloor + 1 \le m$. Hence $k = m - (l - m) \ge 0$, proving (34). In the second recursive call, HGCD(C_0, D_0) returns S. By induction,

$$\|C'_0\| \ge \lceil \|C_0\|/2 \rceil > \|D'_0\|, \qquad \text{where } \begin{bmatrix} C_0 \\ D_0 \end{bmatrix} \xrightarrow{S} \begin{bmatrix} C'_0 \\ D'_0 \end{bmatrix}. \tag{35}$$

But $\|C_0\| = l - k = 2(l - m)$ so (35) becomes

$$\|C'_0\| \ge l - m > \|D'_0\|.$$

Now let $\begin{bmatrix} C \\ D \end{bmatrix} \xrightarrow{S} \begin{bmatrix} C' \\ D' \end{bmatrix}$. Another application of lemma 9 shows that

$$\|C'\| = k + \|C'_0\| \ge k + l - m = m$$

and

$$\begin{aligned}\|D'\| &\le k+\max\{\|D_0'\|, \|C_0\|-\|C_0'\|-1\}\\ &\le k+\max\{l-m-1, l-m-1\}\\ &= k+l-m-1=m-1.\end{aligned}$$

This shows $\|C'\| \ge m > \|D'\|$ and hence (26). **Q.E.D.**

Remark: The proof shows we could have used $k \leftarrow 2m-l-1$ as well. Furthermore, we could modify our algorithm so that after step [4], we return $R\cdot\langle Q\rangle$ in case $\|D\| < m$. This may be slightly more efficient.

Complexity analysis. The HGCD algorithm makes two recursive calls to itself, $\texttt{hGCD}(A_0, B_0)$ and $\texttt{hGCD}(C_0, D_0)$. We check that $\|A_0\|$ and $\|C_0\|$ are both bounded by $n/2$. The work in each call to the algorithm, exclusive of recursion, is $O(M_B(n)) = O(n\log n)$. Hence the algebraic complexity $T(n)$ of this HGCD algorithm satisfies

$$T(n) = 2T(n/2) + O(n\log n).$$

This yields $T(n) = O(n\log^2 n)$.

Exercises

Exercise 6.1: Generalize the HGCD problem to the following: the function FGCD(A, B, f) whose arguments are polynomials A, B as in the HGCD problem, and f is a rational number between 0 and 1. FGCD(A, B, f) returns a matrix M that reduces the pair (A, B) to (A', B') such that $\|A'\|, \|B'\|$ straddle $f\|A\|$. Thus FGCD$(A, B, 1/2) = \texttt{hGCD}(A, B)$. Show that FGCD can be computed in the same complexity as hGCD by using hGCD as a subroutine. □

Exercise 6.2: Modify the polynomial HGCD algorithm so that in step [5], the variable k is set to $\lceil m/2\rceil$. This is essentially the algorithm of Moenck-Aho-Hopcroft-Ullman [2]. We want to construct inputs to make the algorithm return wrong answers. Note that since the modified algorithm works for inputs with normal remainder sequence (see §3), we are unlikely to find such inputs by generating random polynomials. Suppose the output M reduces the input (A, B) to (A', B'). The matrix M may be wrong for several reasons:
(i) $\|B'\| \ge \lceil \|A\|/2\rceil$.
(ii) $\|A'\| < \lceil \|A\|/2\rceil$.
(iii) A', B' are not consecutive entries of the Euclidean remainder sequence of A, B.
Construct inputs to induce each of these possibilities. (The possibilities (i) and (ii) are known to occur.) □

§A. APPENDIX: Integer HGCD

For the sake of completeness, we present an integer version of the HGCD algorithm. We initially use two simple tricks. The first is to recover the non-Archimedean property thus: for $a, b \in \mathbb{Z}$,

$$ab \leq 0 \qquad \Longrightarrow \qquad \|a+b\| \leq \max\{\|a\|, \|b\|\}.$$

One consequence of the non-Archimedean property we exploited was that if $\|a\| \neq \|b\|$ then $\|a+b\| = \max\{\|a\|, \|b\|\}$. Here is an integer analogue:

$$\|a\| - \|b\| \geq 1 \qquad \Longrightarrow \qquad \|a+b\| = \|a\| \pm \epsilon, \quad (0 \leq \epsilon \leq 1).$$

To carry out a parallel reduction, the integer analogue would perhaps be to call HGCD on $a\,\mathbf{div}\,2^m, b\,\mathbf{div}\,2^m$ for suitable m. Instead, the second trick will call HGCD on

$$a_0 := 1 + (a\,\mathbf{div}\,2^m), \quad b_0 := b\,\mathbf{div}\,2^m. \tag{36}$$

The Basic Setup. We begin by proving the analogue of the Correctness Criteria (lemma 9). The following notations will be fixed for the next two lemmas.

Assume that we are given $a > b > 0$ and $m \geq 1$ where $a \geq 2^m$. This determines the non-negative values a_0, a_1, b_0, b_1 via

$$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} a_0 & -a_1 \\ b_0 & b_1 \end{bmatrix} \begin{bmatrix} 2^m \\ 1 \end{bmatrix}, \quad 0 < a_1 \leq 2^m, \quad 0 \leq b_1 < 2^m. \tag{37}$$

Note that both tricks are incorporated in (37). Defining a_0, b_0 as in (36) is the same as choosing $a_1 := 2^m - (a\,\mathbf{mod}\,2^m)$ and $b_1 := b\,\mathbf{mod}\,2^m$. This choice ensures $a_0 > b_0$, as we assume in the recursive call to the algorithm on a_0, b_0.

We are also given a regular matrix M. This determines the values $a'_0, b'_0, a'_1, b'_1, a', b'$ via

$$\begin{bmatrix} a'_0 & a'_1 \\ b'_0 & b'_1 \end{bmatrix} := M^{-1} \begin{bmatrix} a_0 & -a_1 \\ b_0 & b_1 \end{bmatrix} \tag{38}$$

and

$$\begin{bmatrix} a' \\ b' \end{bmatrix} := \begin{bmatrix} a'_0 & a'_1 \\ b'_0 & b'_1 \end{bmatrix} \begin{bmatrix} 2^m \\ 1 \end{bmatrix}. \tag{39}$$

Hence we have the parallel reductions

$$\begin{bmatrix} a_0 \\ b_0 \end{bmatrix} \xrightarrow{M} \begin{bmatrix} a'_0 \\ b'_0 \end{bmatrix}, \quad \begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{M} \begin{bmatrix} a' \\ b' \end{bmatrix}.$$

Finally, we assume two key inequalities:

$$a'_0 > b'_0 \geq 0 \tag{40}$$

$$2\|a'_0\| - 2 \geq \|a_0\| \tag{41}$$

Now write M as

$$M = \begin{bmatrix} p & q \\ r & s \end{bmatrix}, \qquad M^{-1} = \delta \begin{bmatrix} s & -q \\ -r & p \end{bmatrix} \tag{42}$$

where $\delta = \det M = \pm 1$. From (38) we obtain

$$a_1' = -\delta(sa_1 + qb_1), \quad b_1' = \delta(ra_1 + pb_1). \tag{43}$$

The proof below uses (43) to predict the signs of a_1', b_1', assuming the sign of δ. This is possible thanks to the second trick.

The following is the integer analogue of lemma 9:

Lemma 11 (Partial Correctness Criteria)
With the Basic Setup:

(–) *Suppose* $\det M = -1$.
(–a) $\|a'\| = m + \|a_0'\| + \epsilon_1$, $(0 \le \epsilon_1 < 1)$.
(–b) $\|b'\| \le m + \max\{\|b_0'\|, \|a_0\| - \|a_0'\| + 1\}$.
Moreover, $\|a'\| > \|b'\|$.

(+) *Suppose* $\det M = +1$.
(+a) $\|a'\| = m + \|a_0'\| - \epsilon_2$, $(0 \le \epsilon_2 < 1)$.
(+b) $\|b'\| \le 1 + m + \max\{\|b_0'\|, \|a_0\| - \|a_0'\| + 1\}$.
Furthermore $b' \ge 0$.

In both cases (–) *and* (+), $a' > 0$.

Proof. Since $a_0 = pa_0' + qb_0'$, the ordering property (20) and (40) yields

$$\|a_0\| = \|p\| + \|a_0'\| + \epsilon_3 \quad (0 \le \epsilon_3 < 1).$$

Hence (41) is equivalent to

$$\|p\| + \epsilon_3 \le \|a_0'\| - 2.$$

We now prove cases (–) and (+) in parallel.

Part (a). From equation (43),

$$\begin{aligned} \|a_1'\| &\le \max\{\|sa_1\|, \|qb_1\|\} + 1 \\ &< \|p\| + m + 1 \qquad (\text{by } (20), \|a_1\| \le m, \|b_1\| < m) \\ &\le \|a_0'\| + m - 1. \end{aligned}$$

Hence $\|a_0' 2^m\| > 1 + \|a_1'\|$ and so $a' = a_0' 2^m + a_1' > 0$. This proves the desired $a' > 0$. If $\delta = -1$ then $a_1' \ge 0$ (by equation (43)) and hence $\|a'\| = m + \|a_0'\| + \epsilon_1$ as required by subcase (–a). On the other hand, if $\delta = +1$ then $a_1' \le 0$ and $a' = a_0' 2^m + a_1' > a_0' 2^{m-1}$ and hence subcase (+a) follows.

Part (b). Again from (43),

$$\begin{aligned} \|b_1'\| &\le \max\{\|ra_1\|, \|pb_1\|\} + 1 \\ &\le \|p\| + m + 1 \\ &\le \|a_0\| - \|a_0'\| + m + 1. \end{aligned}$$

In case $\delta = +1$, $b_1' \ge 0$ and hence $b' = b_0' 2^m + b_1' \ge 0$, as desired. Also subcase (+b) easily follows. In case $\delta = -1$, $b_1' \le 0$ and $b_0' b_1' \le 0$. This gives the non-Archimedean inequality:

$$\|b'\| \le \max\{\|b_0' 2^m\|, \|b_1'\|\},$$

which proves subcase (–b).

Finally, we must show that $\delta = -1$ implies $\|a'\| > \|b'\|$: this follows immediately from (40), (41) and subcases (–a) and (–b). **Q.E.D.**

To see inadequacies in the Partial Correctness Criteria, we state our precise algorithmic goal.

Integer HGCD Criteria: *On input $a > b \geq 0$, the integer HGCD algorithm outputs a regular matrix M such that*

$$a \leq 3 \Rightarrow \quad M = E \tag{44}$$

$$a \geq 4 \Rightarrow \|a'\| \geq 1 + \left\lceil \frac{\|a\|}{2} \right\rceil > \|b'\|, \quad a' > b' \geq 0 \tag{45}$$

where $\begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{M} \begin{bmatrix} a' \\ b' \end{bmatrix}$.

Note that if $a \geq 4$ then $\|a\| \geq 1 + \lceil \|a\|/2 \rceil$ and so the desired matrix M exists.

Discussion. We see that the pair a', b' obtained in the Partial Correctness Criteria lemma may fail two properties needed for our HGCD algorithm:
Case $\det M = -1$: b' can be negative.
Case $\det M = +1$: the inversion $b' \geq a'$ may occur.
Clearly these two failures are mutually exclusive. On deeper analysis, it turns out that we only have to modify M slightly to obtain some regular matrix M^* such that

$$\begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{M^*} \begin{bmatrix} a^* \\ b^* \end{bmatrix}$$

and a^*, b^* satisfy the correctness criteria, $\|a^*\| \geq m > \|b^*\|$, $a^* > b^* \geq 0$. The "Fixing Up lemma" below shows how to do this. The fixing up is based on three simple transformations of regular matrices: advancing, backing up and toggling.

In the following, let $a > b > 0$ and $M = \langle q_1, \ldots, q_k \rangle$ be a regular matrix such that

$$\begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{M} \begin{bmatrix} a' \\ b' \end{bmatrix}.$$

(I) Advancing: If $q' = a' \,\mathbf{div}\, b'$, then we say that M has *advanced by one step* to the matrix $\langle q_1, \ldots, q_k, q' \rangle$. Note that this operation defines a regular matrix iff $q \geq 1$, *i.e.*, $\|a'\| \geq \|b'\|$. In general, we may speak of advancing M by more than one step.

(II) Backing Up: We call the matrix $\langle q_1, \ldots, q_{k-i} \rangle$ the *backing up* of M by i steps ($0 \leq i \leq k$); in case $i = 1$, we simply call it the *backup* of M. To do backing up, we need to recover the last partial quotient x from a regular matrix $M = \begin{bmatrix} p & q \\ r & s \end{bmatrix}$. Note that $M = E$ if and only if $q = 0$, but in this case x is undefined. Hence assume $M \neq E$. Then M is elementary if and only if $s = 0$, and in this case $x = p$. So we next assume that M is not elementary. Write

$$M' = \begin{bmatrix} p' & q' \\ r' & s' \end{bmatrix}, \qquad M = M' \cdot \begin{bmatrix} x & 1 \\ 1 & 0 \end{bmatrix}$$

where $M' \neq E$ and $p = xp' + q', q = p'$. There are two cases. **Case of $q = 1$:** Clearly $p' = 1$. Since $p' \geq q' \geq 1$ (ordering property), we must have $q' = 1$. Hence x equals $p - 1$. **Case**

of $q > 1$: Then $p' > q'$ (there are two possibilities to check, depending on whether M' is elementary or not). This implies $x = p\,\mathbf{div}\,q$. In summary, the last partial quotient of M is given by

$$x = \begin{cases} \text{undefined} & \text{if } q = 0 \\ p & \text{if } s = 0 \\ p-1 & \text{if } q = 1 \\ p\,\mathbf{div}\,q & \text{otherwise} \end{cases}$$

(III) Toggling: We call $T = \begin{bmatrix} 1 & 1 \\ 0 & -1 \end{bmatrix}$ the *toggle matrix*, so-called because T is idempotent ($T^2 = E$). The matrix MT is the *toggle of M*. We observe that MT is equal to $\langle q_1, \ldots, q_{k-1}, q_k - 1, 1\rangle$ in case $q_k > 1$, and $MT = \langle q_1, \ldots, q_{k-2}, q_{k-1} + 1\rangle$ in case $q_k = 1$ and $k > 1$. However, if $q_k = 1$ and $k = 1$, MT is not a regular matrix. In any case, we have

$$\begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{MT} \begin{bmatrix} a' + b' \\ -b' \end{bmatrix}.$$

Exercise A.1: Verify the remarks on the toggling matrix T. □

Lemma 12 (Fixing Up)
With the notations from the Basic Setup, let t be any number (the "fixup threshold") such that

$$\|a'_0\| \ge t > \max\{\|b'_0\|, \|a_0\| - \|a'_0\| + 1\}. \tag{46}$$

Moreover, if we write M as $\langle q_1, \ldots, q_k\rangle$ and M^* is as specified below, then

$$\|a^*\| \ge m + t > \|b^*\| \tag{47}$$

and

$$b^* \ge 0, \tag{48}$$

where $\begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{M^*} \begin{bmatrix} a^* \\ b^* \end{bmatrix}$. *Here M^* is the regular matrix specified as follows:*

(−) *Suppose* $\det M = -1$.
  (−A) *If $b' \ge 0$ then $M^* := M$.*
  (−B) *Else if $\|a' + b'\| \ge m + t$ then M^* is the toggle of M.*
  (−C) *Else if $q_k \ge 2$ then $M^* := \langle q_1, \ldots, q_{k-1}, q_k - 1\rangle$ is the backup of the toggle of M.*
  (−D) *Else M^* is the backing up of M by two steps.*

(+) *Suppose* $\det M = +1$.
  (+A) *If $\|a'\| \le \|b'\|$ then M^* is the advancement of $\langle q_1, \ldots, q_{k-1}\rangle$ by at most two steps.*
  (+B) *Else if $\|a'\| < m + t$ then M^* is the backing up of M by one or two steps.*
  (+C) *Else M^* is the advancement of M by at most two steps.*

Proof. The Partial Correctness Criteria lemma will be repeatedly exploited. First assume $\det M = -1$.

Subcase (−A). In this subcase, (48) is automatic, and (47) follows from case (−) of the Partial Correctness Criteria lemma.

Subcase (–B). In this subcase, $M^* = MT$ reduces $\begin{bmatrix} a \\ b \end{bmatrix}$ to $\begin{bmatrix} a^* \\ b^* \end{bmatrix} = \begin{bmatrix} a'+b' \\ -b' \end{bmatrix}$, as noted earlier. Recall that MT is not regular in case $k = 1$ and $q_1 = 1$. But if this were the case then

$$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} a' \\ b' \end{bmatrix}.$$

This implies $a' + b' = a > b = a'$ and so $b' > 0$, contradicting the fact that subcase (–A) has been excluded. Again, (48) is immediate, and (47) follows from case (–) of the Partial Correctness Criteria Lemma ($\|a^*\| = \|a' + b'\| \geq m + t$ by assumption).

Subcase (–C). In this subcase, M^* can be written as $M \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}$, and so $\begin{bmatrix} a^* \\ b^* \end{bmatrix} = \begin{bmatrix} a' \\ a'+b' \end{bmatrix}$. We see that (48) holds by virtue of $a' + b' > 0$ (since $\|a'\| > \|b'\|, a' > 0$). Also (47) holds because the Partial Correctness Criteria lemma implies $\|a'\| \geq m + t$ and, since subcase (–B) fails, $\|a' + b'\| < m + t$.

Subcase (–D). Now $q_k = 1$ and M^* omits the last two partial quotients $(q_{k-1}, q_k) = (x, 1)$ where we write x for q_{k-1}. We ought to show $k \geq 2$, but this is the same argument as in subcase (–B). Hence $M^* = M \begin{bmatrix} 1 & -x \\ -1 & x+1 \end{bmatrix}$ and $\begin{bmatrix} a^* \\ b^* \end{bmatrix} = \begin{bmatrix} a'(x+1) + b'x \\ a'+b' \end{bmatrix}$. Hence $\|a^*\| = \|xb^* + a'\| > \|b^*\|$ and so a^*, b^* are consecutive elements of the remainder sequence of a, b. Then (48) holds because $a' + b' > 0$. To see (47), it is clear that $m + t > \|b^*\|$ (otherwise subcase (–B) applies) and it remains to show $\|a^*\| \geq m + t$. But this follows from $\|a^*\| = \|a' + x(a' + b')\| > \|a'\| \geq m + t$.

Now consider the case $\det M = +1$.

Subcase (+A). So there is inversion, $a' \leq b'$. Let us back up M to $\langle q_1, \ldots, q_{k-1}\rangle$:

$$\begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{\langle q_1,\ldots,q_{k-1}\rangle} \begin{bmatrix} a'' \\ a' \end{bmatrix} \xrightarrow{\langle q_k\rangle} \begin{bmatrix} a' \\ b' \end{bmatrix}.$$

Hence $a'' = a'q_k + b' > a'$. Thus a'', a' are consecutive members of the remainder sequence of a, b. Now $\|a''\| \geq \|2a'\| = \|a'\| + 1 > m + \|a'_0\| \geq m + t$. Also $\|a'\| \leq \|b'\| < 1 + m + t$ (by Partial Correctness Criteria). Therefore, if we advance $\langle q_1, \ldots, q_{k-1}\rangle$ by at most two steps, we would reduce a'', a' to a^*, b^* where $\|b^*\| < m + t$.

Subcase (+B). Now $a' > b' \geq 0$ and $\|a'\| < m + t$. So a', b' are consecutive members of the remainder sequence of a, b. Consider the entry $a'' = a'q_k + b'$ preceding a' in the remainder sequence of a, b. If $\|a''\| \geq m + t$, we are done since $\|a''\|, \|a'\|$ straddle $m + t$. Otherwise, consider the entry $a''' = a''q_{k-1} + a'$ preceding a''. We have

$$\|a'''\| = \|(a'q_k + b')q_{k-1} + a'\| \geq \|a'\| + 1 \geq m + t.$$

Thus $\|a'''\|, \|a''\|$ straddle $m + t$.

Subcase (+C). Again a', b' are consecutive members of the remainder sequence with $\|a'\| \geq m + t$. But $\|b'\| - 1 < m + t$ implies that if we advance M by at most two steps, the pair a', b' would be reduced to a^*, b^* where $\|b^*\| < m + t$.

Q.E.D.

It is interesting to note that in tests on randomly generated numbers of about 45 digits, subcase (–D) never arose.

The Fixup Procedure. The Fixing Up lemma and its proof provides a specific procedure to convert the tentative output matrix M of the HGCD algorithm into a valid one. To be specific, let

$$\texttt{Fixup}(M, a, b, m, t)$$

denote the subroutine that returns M^*, as specified in the Fixing Up lemma:

$$\begin{bmatrix} a \\ b \end{bmatrix} \xrightarrow{M^*} \begin{bmatrix} a^* \\ b^* \end{bmatrix}, \qquad \|a^*\| \geq m + t > \|b^*\|.$$

The correct behavior of the Fixup procedure depends on its input parameters fulfilling the conditions of the Fixing Up lemma. In particular, it must fulfil the conditions of the Basic Setup (mainly the inequalities (40) and (41)) and also the inequality (46).

In a typical invocation of $\texttt{Fixup}(M, a, b, m, t)$, the values M, a, b, m are available as in the Basic Setup. To pick a value of t, we use the fact that the following typically holds:

$$\|a_0'\| \geq 1 + \lceil \|a_0\|/2 \rceil > \|b_0'\| \tag{49}$$

(cf. (45)). In this case, it is easily verified that the choice $t = 1 + \lceil \|a_0\|/2 \rceil$ will satisfy (46). Of course inequality (49) also implies inequality (41).

However, our Fixup procedure may also be called in a situation when the Fixing Up lemma does not hold, namely, when $a_0 = 1 + (a \,\mathbf{div}\, 2^m) \leq 3$. In this case, no choice of t satisfying inequality (46) is possible. Note that $b_0 < a_0 \leq 3$ implies $b = b_0 2^m + b_1 < 3 \cdot 2^m$. It is easy to check that if we take at most three of the usual Euclidean remaindering steps, we reduce a, b to a^*, b^* where $\|a^*\| \geq m > \|b^*\|$. In such a situation, if we assume that the Fixup procedure is called with $M = E$ and $t = 0$, the returned matrix M^* is the advancement of E by at most three steps. More generally, if $\|a\|, \|b\|$ straddle $m + i$ where $i = 0, 1, 2$, and we call Fixup with the arguments

$$\texttt{Fixup}(E, a, b, m, 0),$$

we say this is the "easy fixup" case, because M^* is the advancement of E by at most 4 steps.

We present the integer HGCD algorithm.

```
Algorithm Integer HGCD$(a, b)$:
    Input: integers $a, b$ with $a > b \geq 0$.
    Output: a regular matrix $M$ satisfying the integer HGCD criteria (44) or (45).
    [1]  $m \leftarrow 1 + \left\lceil \frac{\|a\|}{2} \right\rceil$;        {this is the magic threshold}
         if $a \leq 3$ or $\|b\| < m$ then return$(E)$;
    [2]  $a_0 \leftarrow 1 + (a \,\mathbf{div}\, 2^m)$;   $b_0 \leftarrow b \,\mathbf{div}\, 2^m$;
             if $a_0 \leq 3$ then $t \leftarrow 0$ else $t \leftarrow 1 + \lceil \frac{\|a_0\|}{2} \rceil$;
             $R \leftarrow$ Fixup(hGCD$(a_0, b_0), a, b, m, t)$;
             $\begin{bmatrix} a' \\ b' \end{bmatrix} \leftarrow R^{-1} \begin{bmatrix} a \\ b \end{bmatrix}$;
    [3]  if $\|b'\| < m$ then return$(R)$;
    [4]  $q \leftarrow a' \,\mathbf{div}\, b'$;  $\begin{bmatrix} c \\ d \end{bmatrix} \leftarrow \begin{bmatrix} b' \\ a' \,\mathbf{mod}\, b' \end{bmatrix}$;
             if $1 + (c \,\mathbf{div}\, 2^m) \leq 3$ then return$(R \cdot$ Fixup$(E, c, d, m, 0))$;
    [5]  $l \leftarrow \lceil \|c\| \rceil$;   $k \leftarrow 2m - l - 1$;
             {Now $\|c\| \geq m + 1 \geq 4$. We claim $\|c\| - 1 \geq k \geq 0$}
    [6]  $c_0 \leftarrow 1 + (c \,\mathbf{div}\, 2^k)$;   $d_0 \leftarrow d \,\mathbf{div}\, 2^k$;
             if $c_0 \leq 3$ then $t' \leftarrow 0$ else $t' \leftarrow 1 + \left\lceil \frac{\|c_0\|}{2} \right\rceil$;
             $S \leftarrow$ Fixup(hGCD$(c_0, d_0), c, d, k, t')$;        {We claim $k + t' = m + 1$.}
    [7]  $\begin{bmatrix} c' \\ d' \end{bmatrix} \leftarrow S^{-1} \begin{bmatrix} c \\ d \end{bmatrix}$;        {So $\|c'\|, \|d'\|$ straddle $k + t'$}
             $T \leftarrow$ Fixup$(E, c', d', m, 0)$;
             $M \leftarrow R \cdot \langle q \rangle \cdot S \cdot T$;   return$(M)$;
```

Correctness. Procedure hGCD returns in four places (steps [1], [3], [4] and [7]) in the algorithm. We show that the matrix returned at each of these places is correct. Since these matrices are regular, we basically have to check the straddling property (45) when $a \geq 4$. We will also need to check that each call to the Fixup procedure is proper.

a) In case the algorithm returns the identity matrix E in step [1], the correctness is trivial.

b) In step [2], we must check that the proper conditions are fulfilled for calling Fixup. When $a_0 \leq 3$ we have the "easy fixup" case. Otherwise $a_0 \geq 4$ and the first recursive call in hGCD returns some regular matrix R' which is fixed up as R by Fixup. The conditions of the Basic Setup are fulfilled with a, b, m as usual and $M = R'$. If $\begin{bmatrix} a_0 \\ b_0 \end{bmatrix} \xrightarrow{R'} \begin{bmatrix} a'_0 \\ b'_0 \end{bmatrix}$, then inductively, the correctness of the HGCD procedure implies equation (49) (and hence (41)) holds. As discussed following equation (49), the choice $t = 1 + \lceil \|a_0\|/2 \rceil$ then satisfies (46),

c) Suppose the matrix R is returned at step [3]. This is correct since $\|a'\|, \|b'\|$ straddle $m + t$ (by correctness of the Fixup procedure) and the condition for exit is $\|b'\| < m$.

d) In step [4], the call to Fixup is the "easy fixup" case since $\|c\| \geq m$ and $\|c\| \leq m + 2$. The returned matrix is clearly correct.

e) Suppose we reach step [5]. We show the claim $\|c\| - 1 \geq k \geq 0$. We have $\|c\| - 1 \geq m - 1 \geq (m - 1) + (m - l) = k$. Next, $k \geq 0$ is equivalent to $2m - 1 \geq l$. This follows from:

$$l = \lceil \|b'\| \rceil \quad \leq \quad m + t \qquad \text{(from the first FIXUP)}$$

$$\begin{aligned}
&= m+1+\left\lceil \frac{\|1+(a\,\mathbf{div}\,2^m)\|}{2}\right\rceil \\
&\le m+1+\left\lceil \frac{1+\lfloor \|a\,\mathbf{div}\,2^m\|\rfloor}{2}\right\rceil \quad (\text{since } \|1+x\|\le 1+\lfloor\|x\|\rfloor) \\
&= m+1+\left\lceil \frac{1+\lfloor \|a/2^m\|\rfloor}{2}\right\rceil \quad (\text{since } \lfloor\|x\,\mathbf{div}\,y\|\rfloor = \lfloor\|x/y\|\rfloor) \\
&= m+1+\left\lceil \frac{1+\lfloor \|a\|\rfloor - m}{2}\right\rceil \\
&\le m+\lceil (m-1)/2\rceil \quad (\text{since } m = 1+\lceil\|a\|/2\rceil) \\
&\le 2m-1 \qquad (m\ge 2)
\end{aligned}$$

f) The call to `Fixup` in step [6] fulfills the appropriate conditions. [Reasoning as before: note that $\|c\|-1\ge k$ implies that $c_0\ge 3$. Hence, the "easy fixup" case occurs iff $c_0=3$. Otherwise, the Basic Setup conditions prevail with a,b,m,M in the Basic Setup replaced by $c,d,k,\mathtt{hGCD}(c_0,d_0)$.] Next we prove the claim $k+t'=m+1$:

$$\begin{aligned}
t' &= 1+\lceil\|c_0\|/2\rceil \\
&= 1+\left\lceil\frac{\lceil\|c_0\|\rceil}{2}\right\rceil \\
&= 1+\left\lceil\frac{\lceil\|\epsilon+(c/2^k)\|\rceil}{2}\right\rceil \quad (0<\epsilon\le 1, \quad c_0 = 1+\lfloor c/2^k\rfloor) \\
&= 1+\left\lceil\frac{l-k+\delta}{2}\right\rceil \quad (\delta = 0\text{or } \ 1, \quad l=\lceil\|c\|\rceil) \\
&= 1+\left\lceil\frac{2l-2m+1+\delta}{2}\right\rceil \quad (k = 2m-l-1) \\
&= 2+l-m.
\end{aligned}$$

Thus $k+t' = k+(2+l-m) = m+1$, as desired.

g) We reach step [7]. By the correctness of the Fixup procedure, $\|c'\|,\|d'\|$ straddle $k+t'=m+1$. Hence we have the right conditions for the "easy fixup" case. The final output is clearly correct.

This concludes our correctness proof.

Computational details and analysis. The algorithm has to perform comparisons of the kind

$$\|a\| : m,$$

and compute ceiling functions in the special forms

$$\lceil\|a\|\rceil, \qquad \lceil\|a\|/2\rceil,$$

where a,m are positive integers. (In the algorithm a may be zero, but we can treat those as special cases.) Since $\|a\|$ is generally not rational, we do not want to explicitly compute it. Instead we reduce these operations to integer comparisons, checking if a is a power of two, and to computing the function $\mathtt{bit}(a)$, which is defined to be the number of bits in the binary representation of a positive integer a. So $\mathtt{bit}(a) = 1+\lfloor\log_2 a\rfloor$ and clearly this function is easily computed in linear time in the usual Turing machine model. Then we have

$$\|a\|\ge m \Leftrightarrow \mathtt{bit}(a)-1\ge m$$

and

$$\|a\| > m \Leftrightarrow \begin{cases} \texttt{bit}(a) - 1 & > & m & \text{if } a \text{ is a power of 2} \\ \texttt{bit}(a) & > & m & \text{else.} \end{cases}$$

and finally,

$$\left\lceil \frac{\|a\|}{2} \right\rceil = \begin{cases} \left\lceil \frac{\texttt{bit}(a)-1}{2} \right\rceil & \text{if } a \text{ is a power of 2} \\ \\ \left\lceil \frac{\texttt{bit}(a)}{2} \right\rceil & \text{else} \end{cases}$$

The global structure of the complexity analysis is similar to the polynomial HGCD case: with $M_B(n) = n\mathcal{L}(n)$ denoting as usual the bit complexity of integer multiplication, it is not hard to see that `Fixup` takes time $O(M_B(n))$, under the conditions stipulated for its invocation. In the two recursive calls to `hGCD`, it is easy to check that the integers have bit size $\frac{n}{2} + O(1)$. Hence, if $T(n)$ is the bit complexity of our HGCD algorithm on inputs of size at most n, then

$$T(n) = O(M_B(n)) + 2T(\frac{n}{2} + O(1)).$$

This has solution $T(n) = O(M_B(n) \log n) = n\mathcal{L}^2(n)$.

Exercises

Exercise A.2:
(a) Verify the remarks on reducing operations involving $\|a\|$ to integer operations, the function $\texttt{bit}(a)$ and testing if a is a power of 2.
(b) Derive the time bound $T(n) = n\mathcal{L}^2(n)$ for the HGCD algorithm. □

Exercise A.3: Try to simplify the integer HGCD algorithm by separating the truncation value t (as in $a_0 := a \, \mathbf{div} \, 2^t$) from the straddling value s (as in $\|a'\| \geq s > \|b'\|$). Currently $t = s = 1 + \lceil \|a\|/2 \rceil$. □

References

[1] W. W. Adams and P. Loustaunau. *An Introduction to Gröbner Bases*. Graduate Studies in Mathematics, Vol. 3. American Mathematical Society, Providence, R.I., 1994.

[2] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading, Massachusetts, 1974.

[3] S. Akbulut and H. King. *Topology of Real Algebraic Sets*. Mathematical Sciences Research Institute Publications. Springer-Verlag, Berlin, 1992.

[4] E. Artin. *Modern Higher Algebra (Galois Theory)*. Courant Institute of Mathematical Sciences, New York University, New York, 1947. (Notes by Albert A. Blank).

[5] E. Artin. *Elements of algebraic geometry*. Courant Institute of Mathematical Sciences, New York University, New York, 1955. (Lectures. Notes by G. Bachman).

[6] M. Artin. *Algebra*. Prentice Hall, Englewood Cliffs, NJ, 1991.

[7] A. Bachem and R. Kannan. Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. *SIAM J. Computing*, 8:499–507, 1979.

[8] C. Bajaj. Algorithmic implicitization of algebraic curves and surfaces. Technical Report CSD-TR-681, Computer Science Department, Purdue University, November, 1988.

[9] C. Bajaj, T. Garrity, and J. Warren. On the applications of the multi-equational resultants. Technical Report CSD-TR-826, Computer Science Department, Purdue University, November, 1988.

[10] E. F. Bareiss. Sylvester's identity and multistep integer-preserving Gaussian elimination. *Math. Comp.*, 103:565–578, 1968.

[11] E. F. Bareiss. Computational solutions of matrix problems over an integral domain. *J. Inst. Math. Appl.*, 10:68–104, 1972.

[12] D. Bayer and M. Stillman. A theorem on refining division orders by the reverse lexicographic order. *Duke Math. J.*, 55(2):321–328, 1987.

[13] D. Bayer and M. Stillman. On the complexity of computing syzygies. *J. of Symbolic Computation*, 6:135–147, 1988.

[14] D. Bayer and M. Stillman. Computation of Hilbert functions. *J. of Symbolic Computation*, 14(1):31–50, 1992.

[15] A. F. Beardon. *The Geometry of Discrete Groups*. Springer-Verlag, New York, 1983.

[16] B. Beauzamy. Products of polynomials and a priori estimates for coefficients in polynomial decompositions: a sharp result. *J. of Symbolic Computation*, 13:463–472, 1992.

[17] T. Becker and V. Weispfenning. *Gröbner bases : a Computational Approach to Commutative Algebra*. Springer-Verlag, New York, 1993. (written in cooperation with Heinz Kredel).

[18] M. Beeler, R. W. Gosper, and R. Schroepppel. HAKMEM. A. I. Memo 239, M.I.T., February 1972.

[19] M. Ben-Or, D. Kozen, and J. Reif. The complexity of elementary algebra and geometry. *J. of Computer and System Sciences*, 32:251–264, 1986.

[20] R. Benedetti and J.-J. Risler. *Real Algebraic and Semi-Algebraic Sets*. Actualités Mathématiques. Hermann, Paris, 1990.

[21] S. J. Berkowitz. On computing the determinant in small parallel time using a small number of processors. *Info. Processing Letters*, 18:147–150, 1984.

[22] E. R. Berlekamp. *Algebraic Coding Theory.* McGraw-Hill Book Company, New York, 1968.

[23] J. Bochnak, M. Coste, and M.-F. Roy. *Geometrie algebrique reelle.* Springer-Verlag, Berlin, 1987.

[24] A. Borodin and I. Munro. *The Computational Complexity of Algebraic and Numeric Problems.* American Elsevier Publishing Company, Inc., New York, 1975.

[25] D. W. Boyd. Two sharp inequalities for the norm of a factor of a polynomial. *Mathematika*, 39:341–349, 1992.

[26] R. P. Brent, F. G. Gustavson, and D. Y. Y. Yun. Fast solution of Toeplitz systems of equations and computation of Padé approximants. *J. Algorithms*, 1:259–295, 1980.

[27] J. W. Brewer and M. K. Smith, editors. *Emmy Noether: a Tribute to Her Life and Work.* Marcel Dekker, Inc, New York and Basel, 1981.

[28] C. Brezinski. *History of Continued Fractions and Padé Approximants.* Springer Series in Computational Mathematics, vol.12. Springer-Verlag, 1991.

[29] E. Brieskorn and H. Knörrer. *Plane Algebraic Curves.* Birkhäuser Verlag, Berlin, 1986.

[30] W. S. Brown. The subresultant PRS algorithm. *ACM Trans. on Math. Software*, 4:237–249, 1978.

[31] W. D. Brownawell. Bounds for the degrees in Nullstellensatz. *Ann. of Math.*, 126:577–592, 1987.

[32] B. Buchberger. Gröbner bases: An algorithmic method in polynomial ideal theory. In N. K. Bose, editor, *Multidimensional Systems Theory*, Mathematics and its Applications, chapter 6, pages 184–229. D. Reidel Pub. Co., Boston, 1985.

[33] B. Buchberger, G. E. Collins, and R. L. (eds.). *Computer Algebra.* Springer-Verlag, Berlin, 2nd edition, 1983.

[34] D. A. Buell. *Binary Quadratic Forms: classical theory and modern computations.* Springer-Verlag, 1989.

[35] W. S. Burnside and A. W. Panton. *The Theory of Equations*, volume 1. Dover Publications, New York, 1912.

[36] J. F. Canny. *The complexity of robot motion planning.* ACM Doctoral Dissertion Award Series. The MIT Press, Cambridge, MA, 1988. PhD thesis, M.I.T.

[37] J. F. Canny. Generalized characteristic polynomials. *J. of Symbolic Computation*, 9:241–250, 1990.

[38] D. G. Cantor, P. H. Galyean, and H. G. Zimmer. A continued fraction algorithm for real algebraic numbers. *Math. of Computation*, 26(119):785–791, 1972.

[39] J. W. S. Cassels. *An Introduction to Diophantine Approximation.* Cambridge University Press, Cambridge, 1957.

[40] J. W. S. Cassels. *An Introduction to the Geometry of Numbers.* Springer-Verlag, Berlin, 1971.

[41] J. W. S. Cassels. *Rational Quadratic Forms.* Academic Press, New York, 1978.

[42] T. J. Chou and G. E. Collins. Algorithms for the solution of linear Diophantine equations. *SIAM J. Computing*, 11:687–708, 1982.

[43] H. Cohen. *A Course in Computational Algebraic Number Theory.* Springer-Verlag, 1993.

[44] G. E. Collins. Subresultants and reduced polynomial remainder sequences. *J. of the ACM*, 14:128–142, 1967.

[45] G. E. Collins. Computer algebra of polynomials and rational functions. *Amer. Math. Monthly*, 80:725–755, 1975.

[46] G. E. Collins. Infallible calculation of polynomial zeros to specified precision. In J. R. Rice, editor, *Mathematical Software III*, pages 35–68. Academic Press, New York, 1977.

[47] J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Math. Comp.*, 19:297–301, 1965.

[48] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *J. of Symbolic Computation*, 9:251–280, 1990. Extended Abstract: ACM Symp. on Theory of Computing, Vol.19, 1987, pp.1-6.

[49] M. Coste and M. F. Roy. Thom's lemma, the coding of real algebraic numbers and the computation of the topology of semi-algebraic sets. *J. of Symbolic Computation*, 5:121–130, 1988.

[50] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra.* Springer-Verlag, New York, 1992.

[51] J. H. Davenport, Y. Siret, and E. Tournier. *Computer Algebra: Systems and Algorithms for Algebraic Computation.* Academic Press, New York, 1988.

[52] M. Davis. *Computability and Unsolvability.* Dover Publications, Inc., New York, 1982.

[53] M. Davis, H. Putnam, and J. Robinson. The decision problem for exponential Diophantine equations. *Annals of Mathematics, 2nd Series*, 74(3):425–436, 1962.

[54] J. Dieudonné. *History of Algebraic Geometry.* Wadsworth Advanced Books & Software, Monterey, CA, 1985. Trans. from French by Judith D. Sally.

[55] L. E. Dixon. Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors. *Amer. J. of Math.*, 35:413–426, 1913.

[56] T. Dubé, B. Mishra, and C. K. Yap. Admissible orderings and bounds for Gröbner bases normal form algorithm. Report 88, Courant Institute of Mathematical Sciences, Robotics Laboratory, New York University, 1986.

[57] T. Dubé and C. K. Yap. A basis for implementing exact geometric algorithms (extended abstract), September, 1993. Paper from URL `http://cs.nyu.edu/cs/faculty/yap`.

[58] T. W. Dubé. *Quantitative analysis of problems in computer algebra: Gröbner bases and the Nullstellensatz.* PhD thesis, Courant Institute, N.Y.U., 1989.

[59] T. W. Dubé. The structure of polynomial ideals and Gröbner bases. *SIAM J. Computing*, 19(4):750–773, 1990.

[60] T. W. Dubé. A combinatorial proof of the effective Nullstellensatz. *J. of Symbolic Computation*, 15:277–296, 1993.

[61] R. L. Duncan. Some inequalities for polynomials. *Amer. Math. Monthly*, 73:58–59, 1966.

[62] J. Edmonds. Systems of distinct representatives and linear algebra. *J. Res. National Bureau of Standards*, 71B:241–245, 1967.

[63] H. M. Edwards. *Divisor Theory.* Birkhauser, Boston, 1990.

[64] I. Z. Emiris. *Sparse Elimination and Applications in Kinematics.* PhD thesis, Department of Computer Science, University of California, Berkeley, 1989.

[65] W. Ewald. *From Kant to Hilbert: a Source Book in the Foundations of Mathematics.* Clarendon Press, Oxford, 1996. In 3 Volumes.

[66] B. J. Fino and V. R. Algazi. A unified treatment of discrete fast unitary transforms. *SIAM J. Computing*, 6(4):700–717, 1977.

[67] E. Frank. Continued fractions, lectures by Dr. E. Frank. Technical report, Numerical Analysis Research, University of California, Los Angeles, August 23, 1957.

[68] J. Friedman. On the convergence of Newton's method. *Journal of Complexity*, 5:12–33, 1989.

[69] F. R. Gantmacher. *The Theory of Matrices, volume 1.* Chelsea Publishing Co., New York, 1959.

[70] I. M. Gelfand, M. M. Kapranov, and A. V. Zelevinsky. *Discriminants, Resultants and Multi-dimensional Determinants.* Birkhäuser, Boston, 1994.

[71] M. Giusti. Some effectivity problems in polynomial ideal theory. In *Lecture Notes in Computer Science*, volume 174, pages 159–171, Berlin, 1984. Springer-Verlag.

[72] A. J. Goldstein and R. L. Graham. A Hadamard-type bound on the coefficients of a determinant of polynomials. *SIAM Review*, 16:394–395, 1974.

[73] H. H. Goldstine. *A History of Numerical Analysis from the 16th through the 19th Century.* Springer-Verlag, New York, 1977.

[74] W. Gröbner. *Moderne Algebraische Geometrie.* Springer-Verlag, Vienna, 1949.

[75] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer-Verlag, Berlin, 1988.

[76] W. Habicht. Eine Verallgemeinerung des Sturmschen Wurzelzählverfahrens. *Comm. Math. Helvetici*, 21:99–116, 1948.

[77] J. L. Hafner and K. S. McCurley. Asymptotically fast triangularization of matrices over rings. *SIAM J. Computing*, 20:1068–1083, 1991.

[78] G. H. Hardy and E. M. Wright. *An Introduction to the Theory of Numbers.* Oxford University Press, New York, 1959. 4th Edition.

[79] P. Henrici. *Elements of Numerical Analysis.* John Wiley, New York, 1964.

[80] G. Hermann. Die Frage der endlich vielen Schritte in der Theorie der Polynomideale. *Math. Ann.*, 95:736–788, 1926.

[81] N. J. Higham. *Accuracy and stability of numerical algorithms.* Society for Industrial and Applied Mathematics, Philadelphia, 1996.

[82] C. Ho. Fast parallel gcd algorithms for several polynomials over integral domain. Technical Report 142, Courant Institute of Mathematical Sciences, Robotics Laboratory, New York University, 1988.

[83] C. Ho. *Topics in algebraic computing: subresultants, GCD, factoring and primary ideal decomposition.* PhD thesis, Courant Institute, New York University, June 1989.

[84] C. Ho and C. K. Yap. The Habicht approach to subresultants. *J. of Symbolic Computation*, 21:1–14, 1996.

[85] A. S. Householder. *Principles of Numerical Analysis.* McGraw-Hill, New York, 1953.

[86] L. K. Hua. *Introduction to Number Theory.* Springer-Verlag, Berlin, 1982.

[87] A. Hurwitz. Über die Trägheitsformem eines algebraischen Moduls. *Ann. Mat. Pura Appl.*, 3(20):113–151, 1913.

[88] D. T. Huynh. A superexponential lower bound for Gröbner bases and Church-Rosser commutative Thue systems. *Info. and Computation*, 68:196–206, 1986.

[89] C. S. Iliopoulous. Worst-case complexity bounds on algorithms for computing the canonical structure of finite Abelian groups and Hermite and Smith normal form of an integer matrix. *SIAM J. Computing*, 18:658–669, 1989.

[90] N. Jacobson. *Lectures in Abstract Algebra, Volume 3.* Van Nostrand, New York, 1951.

[91] N. Jacobson. *Basic Algebra 1.* W. H. Freeman, San Francisco, 1974.

[92] T. Jebelean. An algorithm for exact division. *J. of Symbolic Computation*, 15(2):169–180, 1993.

[93] M. A. Jenkins and J. F. Traub. Principles for testing polynomial zerofinding programs. *ACM Trans. on Math. Software*, 1:26–34, 1975.

[94] W. B. Jones and W. J. Thron. *Continued Fractions: Analytic Theory and Applications.* vol. 11, Encyclopedia of Mathematics and its Applications. Addison-Wesley, 1981.

[95] E. Kaltofen. Effective Hilbert irreducibility. *Information and Control*, 66(3):123–137, 1985.

[96] E. Kaltofen. Polynomial-time reductions from multivariate to bi- and univariate integral polynomial factorization. *SIAM J. Computing*, 12:469–489, 1985.

[97] E. Kaltofen. Polynomial factorization 1982-1986. Dept. of Comp. Sci. Report 86-19, Rensselaer Polytechnic Institute, Troy, NY, September 1986.

[98] E. Kaltofen and H. Rolletschek. Computing greatest common divisors and factorizations in quadratic number fields. *Math. Comp.*, 52:697–720, 1989.

[99] R. Kannan, A. K. Lenstra, and L. Lovász. Polynomial factorization and nonrandomness of bits of algebraic and some transcendental numbers. *Math. Comp.*, 50:235–250, 1988.

[100] H. Kapferer. Über Resultanten und Resultanten-Systeme. *Sitzungsber. Bayer. Akad. München*, pages 179–200, 1929.

[101] A. N. Khovanskii. *The Application of Continued Fractions and their Generalizations to Problems in Approximation Theory.* P. Noordhoff N. V., Groningen, the Netherlands, 1963.

[102] A. G. Khovanskiĭ. *Fewnomials*, volume 88 of *Translations of Mathematical Monographs.* American Mathematical Society, Providence, RI, 1991. tr. from Russian by Smilka Zdravkovska.

[103] M. Kline. *Mathematical Thought from Ancient to Modern Times*, volume 3. Oxford University Press, New York and Oxford, 1972.

[104] D. E. Knuth. The analysis of algorithms. In *Actes du Congrés International des Mathématiciens*, pages 269–274, Nice, France, 1970. Gauthier-Villars.

[105] D. E. Knuth. *The Art of Computer Programming: Seminumerical Algorithms*, volume 2. Addison-Wesley, Boston, 2nd edition edition, 1981.

[106] J. Kollár. Sharp effective Nullstellensatz. *J. American Math. Soc.*, 1(4):963–975, 1988.

[107] E. Kunz. *Introduction to Commutative Algebra and Algebraic Geometry.* Birkhäuser, Boston, 1985.

[108] J. C. Lagarias. Worst-case complexity bounds for algorithms in the theory of integral quadratic forms. *J. of Algorithms*, 1:184–186, 1980.

[109] S. Landau. Factoring polynomials over algebraic number fields. *SIAM J. Computing*, 14:184–195, 1985.

[110] S. Landau and G. L. Miller. Solvability by radicals in polynomial time. *J. of Computer and System Sciences*, 30:179–208, 1985.

[111] S. Lang. *Algebra.* Addison-Wesley, Boston, 3rd edition, 1971.

[112] L. Langemyr. *Computing the GCD of two polynomials over an algebraic number field.* PhD thesis, The Royal Institute of Technology, Stockholm, Sweden, January 1989. Technical Report TRITA-NA-8804.

[113] D. Lazard. Résolution des systémes d'équations algébriques. *Theor. Computer Science*, 15:146–156, 1981.

[114] D. Lazard. A note on upper bounds for ideal theoretic problems. *J. of Symbolic Computation*, 13:231–233, 1992.

[115] A. K. Lenstra. Factoring multivariate integral polynomials. *Theor. Computer Science*, 34:207–213, 1984.

[116] A. K. Lenstra. Factoring multivariate polynomials over algebraic number fields. *SIAM J. Computing*, 16:591–598, 1987.

[117] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261:515–534, 1982.

[118] W. Li. Degree bounds of Gröbner bases. In C. L. Bajaj, editor, *Algebraic Geometry and its Applications*, chapter 30, pages 477–490. Springer-Verlag, Berlin, 1994.

[119] R. Loos. Generalized polynomial remainder sequences. In B. Buchberger, G. E. Collins, and R. Loos, editors, *Computer Algebra*, pages 115–138. Springer-Verlag, Berlin, 2nd edition, 1983.

[120] L. Lorentzen and H. Waadeland. *Continued Fractions with Applications.* Studies in Computational Mathematics 3. North-Holland, Amsterdam, 1992.

[121] H. Lüneburg. On the computation of the Smith Normal Form. Preprint 117, Universität Kaiserslautern, Fachbereich Mathematik, Erwin-Schrödinger-Straße, D-67653 Kaiserslautern, Germany, March 1987.

[122] F. S. Macaulay. Some formulae in elimination. *Proc. London Math. Soc.*, 35(1):3–27, 1903.

[123] F. S. Macaulay. *The Algebraic Theory of Modular Systems.* Cambridge University Press, Cambridge, 1916.

[124] F. S. Macaulay. Note on the resultant of a number of polynomials of the same degree. *Proc. London Math. Soc*, pages 14–21, 1921.

[125] K. Mahler. An application of Jensen's formula to polynomials. *Mathematika*, 7:98–100, 1960.

[126] K. Mahler. On some inequalities for polynomials in several variables. *J. London Math. Soc.*, 37:341–344, 1962.

[127] M. Marden. *The Geometry of Zeros of a Polynomial in a Complex Variable.* Math. Surveys. American Math. Soc., New York, 1949.

[128] Y. V. Matiyasevich. *Hilbert's Tenth Problem.* The MIT Press, Cambridge, Massachusetts, 1994.

[129] E. W. Mayr and A. R. Meyer. The complexity of the word problems for commutative semigroups and polynomial ideals. *Adv. Math.*, 46:305–329, 1982.

[130] F. Mertens. Zur Eliminationstheorie. *Sitzungsber. K. Akad. Wiss. Wien, Math. Naturw. Kl. 108*, pages 1178–1228, 1244–1386, 1899.

[131] M. Mignotte. *Mathematics for Computer Algebra.* Springer-Verlag, Berlin, 1992.

[132] M. Mignotte. On the product of the largest roots of a polynomial. *J. of Symbolic Computation*, 13:605–611, 1992.

[133] W. Miller. Computational complexity and numerical stability. *SIAM J. Computing*, 4(2):97–107, 1975.

[134] P. S. Milne. On the solutions of a set of polynomial equations. In B. R. Donald, D. Kapur, and J. L. Mundy, editors, *Symbolic and Numerical Computation for Artificial Intelligence*, pages 89–102. Academic Press, London, 1992.

[135] G. V. Milovanović, D. S. Mitrinović, and T. M. Rassias. *Topics in Polynomials: Extremal Problems, Inequalities, Zeros.* World Scientific, Singapore, 1994.

[136] B. Mishra. Lecture Notes on Lattices, Bases and the Reduction Problem. Technical Report 300, Courant Institute of Mathematical Sciences, Robotics Laboratory, New York University, June 1987.

[137] B. Mishra. *Algorithmic Algebra.* Springer-Verlag, New York, 1993. Texts and Monographs in Computer Science Series.

[138] B. Mishra. Computational real algebraic geometry. In J. O'Rourke and J. Goodman, editors, *CRC Handbook of Discrete and Comp. Geom.* CRC Press, Boca Raton, FL, 1997.

[139] B. Mishra and P. Pedersen. Arithmetic of real algebraic numbers is in *NC.* Technical Report 220, Courant Institute of Mathematical Sciences, Robotics Laboratory, New York University, Jan 1990.

[140] B. Mishra and C. K. Yap. Notes on Gröbner bases. *Information Sciences*, 48:219–252, 1989.

[141] R. Moenck. Fast computations of GCD's. *Proc. ACM Symp. on Theory of Computation*, 5:142–171, 1973.

[142] H. M. Möller and F. Mora. Upper and lower bounds for the degree of Gröbner bases. In *Lecture Notes in Computer Science*, volume 174, pages 172–183, 1984. (Eurosam 84).

[143] D. Mumford. *Algebraic Geometry, I. Complex Projective Varieties.* Springer-Verlag, Berlin, 1976.

[144] C. A. Neff. Specified precision polynomial root isolation is in *NC. J. of Computer and System Sciences*, 48(3):429–463, 1994.

[145] M. Newman. *Integral Matrices.* Pure and Applied Mathematics Series, vol. 45. Academic Press, New York, 1972.

[146] L. Nový. *Origins of modern algebra.* Academia, Prague, 1973. Czech to English Transl., Jaroslav Tauer.

[147] N. Obreschkoff. *Verteilung and Berechnung der Nullstellen reeller Polynome.* VEB Deutscher Verlag der Wissenschaften, Berlin, German Democratic Republic, 1963.

[148] C. Ó'Dúnlaing and C. Yap. Generic transformation of data structures. *IEEE Foundations of Computer Science*, 23:186–195, 1982.

[149] C. Ó'Dúnlaing and C. Yap. Counting digraphs and hypergraphs. *Bulletin of EATCS*, 24, October 1984.

[150] C. D. Olds. *Continued Fractions.* Random House, New York, NY, 1963.

[151] A. M. Ostrowski. *Solution of Equations and Systems of Equations.* Academic Press, New York, 1960.

[152] V. Y. Pan. Algebraic complexity of computing polynomial zeros. *Comput. Math. Applic.*, 14:285–304, 1987.

[153] V. Y. Pan. Solving a polynomial equation: some history and recent progress. *SIAM Review*, 39(2):187–220, 1997.

[154] P. Pedersen. Counting real zeroes. Technical Report 243, Courant Institute of Mathematical Sciences, Robotics Laboratory, New York University, 1990. PhD Thesis, Courant Institute, New York University.

[155] O. Perron. *Die Lehre von den Kettenbrüchen.* Teubner, Leipzig, 2nd edition, 1929.

[156] O. Perron. *Algebra*, volume 1. de Gruyter, Berlin, 3rd edition, 1951.

[157] O. Perron. *Die Lehre von den Kettenbrüchen.* Teubner, Stuttgart, 1954. Volumes 1 & 2.

[158] J. R. Pinkert. An exact method for finding the roots of a complex polynomial. *ACM Trans. on Math. Software*, 2:351–363, 1976.

[159] D. A. Plaisted. New *NP*-hard and *NP*-complete polynomial and integer divisibility problems. *Theor. Computer Science*, 31:125–138, 1984.

[160] D. A. Plaisted. Complete divisibility problems for slowly utilized oracles. *Theor. Computer Science*, 35:245–260, 1985.

[161] E. L. Post. Recursive unsolvability of a problem of Thue. *J. of Symbolic Logic*, 12:1–11, 1947.

[162] A. Pringsheim. Irrationalzahlen und Konvergenz unendlicher Prozesse. In *Enzyklopädie der Mathematischen Wissenschaften, Vol. I*, pages 47–146, 1899.

[163] M. O. Rabin. Probabilistic algorithms for finite fields. *SIAM J. Computing*, 9(2):273–280, 1980.

[164] A. R. Rajwade. *Squares.* London Math. Society, Lecture Note Series 171. Cambridge University Press, Cambridge, 1993.

[165] C. Reid. *Hilbert.* Springer-Verlag, Berlin, 1970.

[166] J. Renegar. On the worst-case arithmetic complexity of approximating zeros of polynomials. *Journal of Complexity*, 3:90–113, 1987.

[167] J. Renegar. On the Computational Complexity and Geometry of the First-Order Theory of the Reals, Part I: Introduction. Preliminaries. The Geometry of Semi-Algebraic Sets. The Decision Problem for the Existential Theory of the Reals. *J. of Symbolic Computation*, 13(3):255–300, March 1992.

[168] L. Robbiano. Term orderings on the polynomial ring. In *Lecture Notes in Computer Science*, volume 204, pages 513–517. Springer-Verlag, 1985. Proceed. EUROCAL '85.

[169] L. Robbiano. On the theory of graded structures. *J. of Symbolic Computation*, 2:139–170, 1986.

[170] L. Robbiano, editor. *Computational Aspects of Commutative Algebra*. Academic Press, London, 1989.

[171] J. B. Rosser and L. Schoenfeld. Approximate formulas for some functions of prime numbers. *Illinois J. Math.*, 6:64–94, 1962.

[172] S. Rump. On the sign of a real algebraic number. *Proceedings of 1976 ACM Symp. on Symbolic and Algebraic Computation (SYMSAC 76)*, pages 238–241, 1976. Yorktown Heights, New York.

[173] S. M. Rump. Polynomial minimum root separation. *Math. Comp.*, 33:327–336, 1979.

[174] P. Samuel. About Euclidean rings. *J. Algebra*, 19:282–301, 1971.

[175] T. Sasaki and H. Murao. Efficient Gaussian elimination method for symbolic determinants and linear systems. *ACM Trans. on Math. Software*, 8:277–289, 1982.

[176] W. Scharlau. *Quadratic and Hermitian Forms*. Grundlehren der mathematischen Wissenschaften. Springer-Verlag, Berlin, 1985.

[177] W. Scharlau and H. Opolka. *From Fermat to Minkowski: Lectures on the Theory of Numbers and its Historical Development*. Undergraduate Texts in Mathematics. Springer-Verlag, New York, 1985.

[178] A. Schinzel. *Selected Topics on Polynomials*. The University of Michigan Press, Ann Arbor, 1982.

[179] W. M. Schmidt. *Diophantine Approximations and Diophantine Equations*. Lecture Notes in Mathematics, No. 1467. Springer-Verlag, Berlin, 1991.

[180] C. P. Schnorr. A more efficient algorithm for lattice basis reduction. *J. of Algorithms*, 9:47–62, 1988.

[181] A. Schönhage. Schnelle Berechnung von Kettenbruchentwicklungen. *Acta Informatica*, 1:139–144, 1971.

[182] A. Schönhage. Storage modification machines. *SIAM J. Computing*, 9:490–508, 1980.

[183] A. Schönhage. Factorization of univariate integer polynomials by Diophantine approximation and an improved basis reduction algorithm. In *Lecture Notes in Computer Science*, volume 172, pages 436–447. Springer-Verlag, 1984. Proc. 11th ICALP.

[184] A. Schönhage. The fundamental theorem of algebra in terms of computational complexity, 1985. Manuscript, Department of Mathematics, University of Tübingen.

[185] A. Schönhage and V. Strassen. Schnelle Multiplikation großer Zahlen. *Computing*, 7:281–292, 1971.

[186] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. of the ACM*, 27:701–717, 1980.

[187] J. T. Schwartz. Polynomial minimum root separation (Note to a paper of S. M. Rump). Technical Report 39, Courant Institute of Mathematical Sciences, Robotics Laboratory, New York University, February 1985.

[188] J. T. Schwartz and M. Sharir. On the piano movers' problem: II. General techniques for computing topological properties of real algebraic manifolds. *Advances in Appl. Math.*, 4:298–351, 1983.

[189] A. Seidenberg. Constructions in algebra. *Trans. Amer. Math. Soc.*, 197:273–313, 1974.

[190] B. Shiffman. Degree bounds for the division problem in polynomial ideals. *Mich. Math. J.*, 36:162–171, 1988.

[191] C. L. Siegel. *Lectures on the Geometry of Numbers*. Springer-Verlag, Berlin, 1988. Notes by B. Friedman, rewritten by K. Chandrasekharan, with assistance of R. Suter.

[192] S. Smale. The fundamental theorem of algebra and complexity theory. *Bulletin (N.S.) of the AMS*, 4(1):1–36, 1981.

[193] S. Smale. On the efficiency of algorithms of analysis. *Bulletin (N.S.) of the AMS*, 13(2):87–121, 1985.

[194] D. E. Smith. *A Source Book in Mathematics*. Dover Publications, New York, 1959. (Volumes 1 and 2. Originally in one volume, published 1929).

[195] V. Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 14:354–356, 1969.

[196] V. Strassen. The computational complexity of continued fractions. *SIAM J. Computing*, 12:1–27, 1983.

[197] D. J. Struik, editor. *A Source Book in Mathematics, 1200-1800*. Princeton University Press, Princeton, NJ, 1986.

[198] B. Sturmfels. *Algorithms in Invariant Theory*. Springer-Verlag, Vienna, 1993.

[199] B. Sturmfels. Sparse elimination theory. In D. Eisenbud and L. Robbiano, editors, *Proc. Computational Algebraic Geometry and Commutative Algebra 1991*, pages 377–397. Cambridge Univ. Press, Cambridge, 1993.

[200] J. J. Sylvester. On a remarkable modification of Sturm's theorem. *Philosophical Magazine*, pages 446–456, 1853.

[201] J. J. Sylvester. On a theory of the syzegetic relations of two rational integral functions, comprising an application to the theory of Sturm's functions, and that of the greatest algebraical common measure. *Philosophical Trans.*, 143:407–584, 1853.

[202] J. J. Sylvester. *The Collected Mathematical Papers of James Joseph Sylvester*, volume 1. Cambridge University Press, Cambridge, 1904.

[203] K. Thull. Approximation by continued fraction of a polynomial real root. *Proc. EUROSAM '84*, pages 367–377, 1984. Lecture Notes in Computer Science, No. 174.

[204] K. Thull and C. K. Yap. A unified approach to fast GCD algorithms for polynomials and integers. Technical report, Courant Institute of Mathematical Sciences, Robotics Laboratory, New York University, 1992.

[205] J. V. Uspensky. *Theory of Equations*. McGraw-Hill, New York, 1948.

[206] B. Vallée. Gauss' algorithm revisited. *J. of Algorithms*, 12:556–572, 1991.

[207] B. Vallée and P. Flajolet. The lattice reduction algorithm of Gauss: an average case analysis. *IEEE Foundations of Computer Science*, 31:830–839, 1990.

[208] B. L. van der Waerden. *Modern Algebra*, volume 2. Frederick Ungar Publishing Co., New York, 1950. (Translated by T. J. Benac, from the second revised German edition).

[209] B. L. van der Waerden. *Algebra*. Frederick Ungar Publishing Co., New York, 1970. Volumes 1 & 2.

[210] J. van Hulzen and J. Calmet. Computer algebra systems. In B. Buchberger, G. E. Collins, and R. Loos, editors, *Computer Algebra*, pages 221–244. Springer-Verlag, Berlin, 2nd edition, 1983.

[211] F. Viète. *The Analytic Art*. The Kent State University Press, 1983. Translated by T. Richard Witmer.

[212] N. Vikas. An $O(n)$ algorithm for Abelian p-group isomorphism and an $O(n \log n)$ algorithm for Abelian group isomorphism. *J. of Computer and System Sciences*, 53:1–9, 1996.

[213] J. Vuillemin. Exact real computer arithmetic with continued fractions. *IEEE Trans. on Computers*, 39(5):605–614, 1990. Also, 1988 ACM Conf. on LISP & Functional Programming, Salt Lake City.

[214] H. S. Wall. *Analytic Theory of Continued Fractions*. Chelsea, New York, 1973.

[215] I. Wegener. *The Complexity of Boolean Functions*. B. G. Teubner, Stuttgart, and John Wiley, Chichester, 1987.

[216] W. T. Wu. *Mechanical Theorem Proving in Geometries: Basic Principles*. Springer-Verlag, Berlin, 1994. (Trans. from Chinese by X. Jin and D. Wang).

[217] C. K. Yap. A new lower bound construction for commutative Thue systems with applications. *J. of Symbolic Computation*, 12:1–28, 1991.

[218] C. K. Yap. Fast unimodular reductions: planar integer lattices. *IEEE Foundations of Computer Science*, 33:437–446, 1992.

[219] C. K. Yap. A double exponential lower bound for degree-compatible Gröbner bases. Technical Report B-88-07, Fachbereich Mathematik, Institut für Informatik, Freie Universität Berlin, October 1988.

[220] K. Yokoyama, M. Noro, and T. Takeshima. On determining the solvability of polynomials. In *Proc. ISSAC'90*, pages 127–134. ACM Press, 1990.

[221] O. Zariski and P. Samuel. *Commutative Algebra*, volume 1. Springer-Verlag, New York, 1975.

[222] O. Zariski and P. Samuel. *Commutative Algebra*, volume 2. Springer-Verlag, New York, 1975.

[223] H. G. Zimmer. *Computational Problems, Methods, and Results in Algebraic Number Theory*. Lecture Notes in Mathematics, Volume 262. Springer-Verlag, Berlin, 1972.

[224] R. Zippel. *Effective Polynomial Computation*. Kluwer Academic Publishers, Boston, 1993.

Contents