Universität-GH Paderborn
FB 17, AG Algorithmische Mathematik

Skript Computeralgebra II

Sommersemester 1995/96

Jürgen Gerhard

August 1, 1996

Contents

Universität-GH Paderborn
FB 17, AG Algorithmische Mathematik
Skript Computeralgebra II, Teil 1
Sommersemester 1996

17. Factoring in $\mathbb{Q}[x]$: the basic idea

The goal in the first sections of this course is to provide (random) polynomial time factorization in $\mathbb{Q}[x]$ (and $\mathbb{F}_q[y,x]$). Our first goal is to understand the difference between "factoring in $\mathbb{Z}[x]$" and "factoring a polynomial with integer coefficients in $\mathbb{Q}[x]$". We start with the following celebrated theorem of Gauß. Throughout this section, R denotes a Unique Factorization Domain(UFD), i.e., an integral domain where every nonzero nonunit can be written as a product of irreducibles which is unique up to reordering and multiplication by units.

THEOREM 17.1. *(GAUSS) If R is a UFD, then $R[x]$ is a UFD.*

Since any two elements of a UFD have a gcd (see section 9), we conclude that it makes sense to talk of gcd's of polynomials over unique factorization domains. We write $a \sim b$ for two elements $a, b \in R$ if there is a unit $u \in R$ with $a = ub$, and say that a and b are *associates*. In general, the gcd is defined only up to associates. This poses a slightly annoying notational problem, in that gcd is not a function, but only a (ternary) relation. We will nevertheless (ab)use the functional notation $c = \gcd(a,b)$. In particular, we suppose that "$\gcd(a,b) = 1$" whenever this gcd is a unit. In the following we assume that we have a (unique) function "gcd". This is certainly justified in the two cases that we are most interested in: $R = \mathbb{Z}$, or $R = F[y]$ with a field F. In these two environments the gcd can be made unique by choosing the unique positive or monic associate, respectively.

DEFINITION 17.2. *The* content $\mathrm{cont}(f)$ *of* $f = f_n x^n + \cdots + f_1 x + f_0 \in R[x]$ *with* $f_0, \ldots, f_n \in R$ *is defined as* $\mathrm{cont}(f) = \gcd(f_0, f_1, \ldots, f_n) \in R$. *The polynomial f is* primitive *if* $\mathrm{cont}(f) = 1$. *We define the* primitive part $\mathrm{pp}(f)$ *of f by* $f = \mathrm{cont}(f) \cdot \mathrm{pp}(f)$. *The content and primitive part of f are unique since we have a unique* gcd.

These notions are illustrated in Examples 17.7 and 17.8 below.

EXERCISE 17.3. *Show that* $f = \mathrm{pp}(f) \iff f$ *is primitive.*

LEMMA 17.4. *For* $f \in R[x]$ *and* $c \in R$, $\mathrm{cont}(cf) = uc \cdot \mathrm{cont}(f)$ *and* $\mathrm{pp}(cf) = u^{-1} pp(f)$ *for some unit* $u \in R$.

PROOF. Exercise. □

The following lemma, due to Gauß, is a cornerstone for unique factorization of polynomials over UFDs.

LEMMA 17.5. *(Gauß' Lemma) For a Unique Factorization Domain R, the product of two primitive polynomials in $R[x]$ is primitive.*

PROOF. Let $f = \sum f_i x^i$ and $g = \sum g_i x^i \in R[x]$ be primitive, and assume that fg is not primitive. Then there is some prime $p \in R$ dividing every coefficient of fg. Let j be the smallest integer for which p does not divide f_j. Such an integer exists since f is primitive. Similarly, let k be the smallest integer for which g_k is not a multiple of p.

Consider the coefficient of x^{j+k} in fg. This coefficient is divisible by p and equals

$$\begin{aligned}\sum_{l+m=j+k} f_l g_m &= f_0 g_{j+k} + f_1 g_{j+k-1} + \cdots + f_{j-1} g_{k+1} \\ &+ f_j g_k \\ &+ f_{j+1} g_{k-1} + \cdots + f_{j+k} g_0.\end{aligned}$$

Since p divides f_l for $0 \le l < j$, it divides every term in the first row of the above equation. Similarly p divides every term in the third row of the equation. Thus p divides $f_j g_k$, and since it is prime, it divides f_j or g_k. This contradicts our definition of f_j and g_k. □

COROLLARY 17.6. *For $f, g \in R[x]$, $\mathrm{cont}(fg) = u \cdot \mathrm{cont}(f)\mathrm{cont}(g)$ and $\mathrm{pp}(fg) = u^{-1}\mathrm{pp}(f)\mathrm{pp}(g)$, for some unit $u \in R$.*

PROOF. Let $h = \mathrm{pp}(fg)$. By Gauß' lemma, $h' = \mathrm{pp}(f)\mathrm{pp}(g)$ is primitive. We have

$$fg = (\mathrm{cont}(f) \cdot \mathrm{pp}(f))(\mathrm{cont}(g) \cdot \mathrm{pp}(g)) = \mathrm{cont}(f)\mathrm{cont}(g) \cdot h'.$$

Hence

$$\mathrm{cont}(fg) = u \cdot \mathrm{cont}(f)\mathrm{cont}(g) \cdot \mathrm{cont}(h') = u \cdot \mathrm{cont}(f)\mathrm{cont}(g)$$

for some unit $u \in R$, by Lemma 17.4, and the claim follows. □

PROOF of Theorem 17.1. Since R is an integral domain , $\deg(fg) = \deg f + \deg g$ for any nonzero polynomials $f, g \in R[x]$. This implies that the units of $R[x]$ are precisely the units of R, and that a prime $p \in R$ is also irreducible in $R[x]$.

Let $f \in R[x]$ be a nonzero nonunit. Since R is a UFD, $\mathrm{cont}(f)$ can be written as a product of irreducibles of $R[x]$, by the above. Let K denote the field of fractions of R. Then $K[x]$ is a Euclidean domain and therefore a UFD, and $\mathrm{pp}(f) = f_1 f_2 \cdots f_r$ in $K[x]$ with (over K) irreducible nonconstant polynomials $f_1, \ldots, f_r$. We can choose constants $a_1, \ldots, a_r \in R$ such that $a_1 f_1, \ldots, a_r f_r$ are in $R[x]$ (e.g., for a_i we may take the lcm of the reduced denominators of the coefficients of f_i), and get a factorization

$$\mathrm{pp}(f) \sim \mathrm{pp}(a_1 \cdots a_r \cdot \mathrm{pp}(f)) \sim \mathrm{pp}(a_1 f_1) \cdots \mathrm{pp}(a_r f_r)$$

in $R[x]$, by Corollary 17.6. Now each $\mathrm{pp}(a_i f_i)$ is primitive in $R[x]$ and irreducible in $K[x]$ and hence irreducible in $R[x]$. This proves the existence of a factorization into irreducibles in $R[x]$.

For the uniqueness, let

$$p_1 \cdots p_k \cdot f_1 \cdots f_r = f = q_1 \cdots q_l \cdot g_1 \cdots g_s$$

be two factorizations of f into irreducibles, with $p_1, \ldots, p_k, q_1, \ldots, q_l \in R$ and $f_1, \ldots, f_r, g_1, \ldots, g_s \in R[x]$ nonconstant and primitive. By Gauß' Lemma, $p_1 \cdots p_k \sim \mathrm{cont}(f) \sim q_1 \cdots q_s$. Thus $k = l$ and $p_1 \sim q_1, \ldots p_k \sim q_k$ after reordering, since R is a UFD. Furthermore,

$$f_1 \cdots f_r \sim \mathrm{pp}(f) \sim g_1 \cdots g_s \tag{17.1}$$

in $R[x]$. If $h \in R[x]$ is nonconstant and primitive, then any factorization $h = h_1 h_2$ with nonconstant polynomials $h_1, h_2 \in K[x]$ leads to a factorization $h \sim \mathrm{pp}(c_1 c_2 h) \sim \mathrm{pp}(c_1 h_1)\mathrm{pp}(c_2 h_2)$ into nonconstant polynomials in $R[x]$, with appropriately chosen $c_1, c_2 \in R$. Thus nonconstant irreducible polynomials in $R[x]$ remain irreducible in $K[x]$, and in (17.1), we also have two factorizations of $\mathrm{pp}(f)$ in $K[x]$ into irreducibles, and hence $r = s$ and $a_i f_i = b_i g_i$, with $a_i, b_i \in R$, for $1 \le i \le r$. Since f_i and g_i are primitive, we have

$$f_i \sim \mathrm{pp}(a_i f_i) = \mathrm{pp}(b_i g_i) \sim g_i$$

for $1 \le i \le r$, which concludes the proof. □

We see from the proof that the primes of $R[x]$ are the primes of R plus the primitive polynomials in $R[x]$ that are irreducible over K.

The following two examples illustrate the difference between taking gcd's in $R[x]$ and $K[x]$.

Example 17.7. *Let* $R = \mathbb{Z}$, $K = \mathbb{Q}$,

$$f = 12x^3 + 28x^2 + 20x + 4,\ g = 12x^2 + 10x + 2 \in \mathbb{Z}[x].$$

Then

$$\begin{aligned} \mathrm{cont}(f) &= \gcd(12, 28, 20, 4) = 4,\ \mathrm{cont}(g) = \gcd(12, 10, 2) = 2, \\ \mathrm{pp}(f) &= 3x^3 + 7x^2 + 5x + 1,\ \mathrm{pp}(g) = 6x^2 + 5x + 1. \end{aligned}$$

With the Euclidean Algorithm in $\mathbb{Q}[x]$ *we find that* $\gcd(f, g) = \gcd(\mathrm{pp}(f), \mathrm{pp}(g)) = x + \frac{1}{3}$ *in* $\mathbb{Q}[x]$. *Hence*

$$\begin{aligned} \mathrm{pp}(\gcd(f, g)) &= \gcd(\mathrm{pp}(f), \mathrm{pp}(g)) = 3x + 1 \text{ in } \mathbb{Z}[x], \\ \mathrm{cont}(\gcd(f, g)) &= \gcd(\mathrm{cont}(f), \mathrm{cont}(g)) = \gcd(4, 2) = 2, \\ \gcd(f, g) &= \mathrm{cont}(\gcd(f, g)) \cdot \mathrm{pp}(\gcd(f, g)) = 6x + 2 \text{ in } \mathbb{Z}[x]. \end{aligned}$$

Example 17.8. *Let*

$$\begin{aligned} f &= (y^3 + 3y^2 + 2y)x^3 + (y^2 + 3y + 2)x^2 + (y^3 + 3y^2 + 2y)x + (y^2 + 3y + 2), \\ g &= (2y^3 + 3y^2 + y)x^2 + (3y^2 + 4y + 1)x + (y + 1) \end{aligned}$$

be polynomials in $\mathbb{Z}_5[y,x]$. Since $R=\mathbb{Z}_5[y]$ is a Euclidean domain and we can regard f,g as polynomials in $R[x]$, we can apply the above results to compute the greatest common divisor of f and g.

Using the Euclidean algorithm in R we compute

$$\begin{aligned}
\mathrm{cont}(f) &= \gcd(y^3+3y^2+2y, y^2+3y+2) = y^2+3y+2,\\
\mathrm{cont}(g) &= \gcd(2y^3+3y^2+y, 3y^2+4y+1, y+1) = y+1,\\
\gcd(\mathrm{cont}(f),\mathrm{cont}(g)) &= y+1.
\end{aligned}$$

Then

$$\begin{aligned}
\mathrm{pp}(f) &= yx^3+x^2+yx+1,\\
\mathrm{pp}(g) &= (2y^2+y)x^2+(3y+1)x+1,\\
\gcd(f,g) &= \gcd(\mathrm{pp}(f),\mathrm{pp}(g)) = x+\frac{1}{y} \text{ in } \mathbb{Z}_5(y)[x],\\
\gcd(\mathrm{pp}(f),\mathrm{pp}(g)) &= yx+1 \text{ in } \mathbb{Z}_5[y][x],\\
\gcd(f,g) &= \gcd(\mathrm{cont}(f),\mathrm{cont}(g))\cdot\gcd(\mathrm{pp}(f),\mathrm{pp}(g))\\
&= (y^2+y)x+(y+1) \text{ in } \mathbb{Z}_5[y][x].
\end{aligned}$$

We use these results to show the relationship between factoring polynomials over $\mathbb{Q}$ and factoring polynomials over $\mathbb{Z}$. If $f\in\mathbb{Z}[x]$ is primitive, then to a factorization of f in $\mathbb{Q}[x]$ corresponds a factorization of f in $\mathbb{Z}[x]$, as in the proof of Gauß' Theorem. If f is not primitive, then the above holds for $\mathrm{pp}(f)$. The factorization of f is the factorization of the content of f together with the factorization of the primitive part of f. Thus

$$\text{factoring in } \mathbb{Z}[x] \longleftrightarrow \text{factoring in } \mathbb{Q}[x] \text{ plus factoring in } \mathbb{Z}.$$

The best known algorithms for factoring in $\mathbb{Z}$are much less efficient than the one for $\mathbb{Q}[x]$ that we present in the next sections.

We now present a sketch of an algorithm that factors primitive polynomials over $\mathbb{Z}$ and that can be adapted with minor changes to the problem of factoring polynomials in $F[y,x]$ for a finite field F. Sections 18–20 will then fill in the details of this sketch. We use an algorithm for factoring polynomials over finite fields as a subroutine that runs in random polynomial time (see section 16).

Algorithm 17.9. *Factorization of a primitive polynomial in $\mathbb{Z}[x]$.*
Input: *$f\in\mathbb{Z}[x]$ primitive.*
Output: *"f is irreducible" or a nontrivial factor of f.*

0. *If f is not squarefree, replace f by $f/\gcd(f,f')$.*
1. (Modular factorization) *Choose a prime $p\in\mathbb{Z}$ such that $f \bmod p$ has the same degree as f and is squarefree. Factor $f \bmod p \in \mathbb{F}_p[x]$.*

2. (Hensel lifting) *Choose $N \in \mathbb{N}$ sufficiently large and "lift" the factorization from step 1 to a factorization of f modulo p^N.*

3. (Short vectors) *Derive a "short vector" problem from the factorization in step 2 and compute a solution.*

4. (Factor extraction) *Derive a nontrival factor of f over $\mathbb{Z}$ from the solution in step 3, if f is not irreducible.*

If we denote by $^{-}$ the reduction modulo the prime p in step 1, we have that

$$\begin{aligned} \bar{f} \text{ squarefree} \quad &\Longleftrightarrow \quad \gcd(\bar{f}, \bar{f}') = 1 \\ &\Longleftrightarrow \quad \operatorname{res}(\bar{f}, \bar{f}') \neq 0 \text{ in } \mathbb{F}_p \\ &\Longleftrightarrow \quad \operatorname{res}(f, f') \not\equiv 0 \bmod p \\ &\Longleftrightarrow \quad p \nmid \operatorname{res}(f, f'). \end{aligned}$$

Hence p should be chosen such that it does not divide $s = \operatorname{lc}(f) \cdot \operatorname{res}(f, f') \in \mathbb{Z} \setminus \{0\}$, where $\operatorname{lc}(f)$ is the leading coefficient of f. Since s is a "small" number, it has only "few" prime factors, and we now show that a "randomly" chosen prime p does the job with high probability.

LEMMA 17.10. *Let $f = \sum_{0 \leq i \leq n} f_i x^i \in \mathbb{Z}[x]$ with $f_i \in \mathbb{Z}$ and $B \in \mathbb{N}$ with $|f_i| \leq B$ for all i, s as above, $\epsilon \in \mathbb{R}_{>0}$, and $P \in \mathbb{N}$ such that*

$$\frac{P}{\log_e P} \geq \frac{n}{\epsilon} \log_2(2n^2B^2).$$

Then

(i) $|s| \leq (2n^2B^2)^n$.

(ii) If p is chosen randomly (and uniformly) among the prime numbers $\leq P$, then the probability that p divides s is at most ϵ.

We note that the size of P is polynomially large in $1/\epsilon$ and the input size $n \log_2 B$.

PROOF.

(i) From Hadamard's inequality (12.4), we find that

$$|\operatorname{res}(f, f')| = |\det \operatorname{Syl}(f, f')| \leq (2n-1)^{(2n-1)/2} B^n (nB)^{n-1} \leq n^{2n} 2^n B^{2n-1},$$

since the coefficients of f' are absolutely bounded from above by nB, and the claim follows, using that $|\operatorname{lc}(f)| \leq B$.

(ii) s has at most $\log_2 |s| \leq n \log_2(2n^2B^2)$ different prime divisors, and there are at least

$$\frac{P}{\log_e P}\left(1 + \frac{1}{2\log_e P}\right) \geq \frac{n}{\epsilon}(2n^2B^2)$$

prime numbers $\leq P$, by the Prime Number Theorem (see Fact 12.5). □

18. Hensel lifting

In sections 7 and 8, we employed Newton iteration for computing inverses and solving polynomial equations modulo p^N over a Euclidean domain R, where $p \in R$ is irreducible, $N \in \mathbb{N}$, and an initial solution modulo p is given. In this section, we will use a similar method to compute factorizations of polynomials modulo p^N. For simplicity, we will present the results only for the case $R = \mathbb{Z}$, but the generalization to arbitrary Euclidean domains is immediate.

Let $f \in \mathbb{Z}[x]$ be squarefree and primitive and $p \in \mathbb{Z}$ prime such that $f \bmod p$ has the same degree as f and is squarefree. Suppose that we have already computed the factorization of $f \bmod p$ into irreducibles, i.e., $f_1, \ldots, f_r \in \mathbb{Z}[x]$ such that $f \equiv f_1 \cdots f_r \bmod p$ and $f_1 \bmod p, \ldots, f_r \bmod p$ are irreducible and pairwise coprime. We now let $h_1 = f_1$, $g_1 = f_2 \cdots f_r$, and want to lift the factorization $f \equiv g_1 h_1 \bmod p$ to a factorization modulo p^N for some $N \in \mathbb{N}$. We may assume that h_1 is monic.

Since $f \bmod p$ is squarefree, $\gcd(g_1 \bmod p, h_1 \bmod p) = 1$ in $\mathbb{F}_p[x]$, and we can compute $s, t \in \mathbb{Z}[x]$ with $sg_1 + th_1 \equiv 1 \bmod p$, using the Extended Euclidean Algorithm in $\mathbb{F}_p[x]$.

Now suppose that we have already computed $g, h \in \mathbb{Z}[x]$ with $g \equiv g_1 \bmod p$, $h \equiv h_1 \bmod p$, and $f \equiv gh \bmod p^k$. We then calculate

$$e = f - gh, \quad \hat{g} = g + te, \quad \hat{h} = h + se, \tag{18.1}$$

and get

$$\begin{aligned} f - \hat{g}\hat{h} &= f - gh - gse - hte - ste^2 \\ &= f - gh - (sg + th)e - ste^2 \\ &= (1 - sg - th)e - ste^2 \equiv 0 \bmod p^{k+1}, \end{aligned}$$

since $e \equiv 0 \bmod p^k$ and $1 - sg - th \equiv 1 - sg_1 - th_1 \equiv 0 \bmod p$. Hence $f \equiv \hat{g}\hat{h} \bmod p^{k+1}$, i.e., $\hat{g}\hat{h}$ is a factorization of f modulo p^{k+1}. Proceeding inductively, we can lift the factorization modulo arbitrary powers of p.

Example 18.1. *In Example 8.9, we have computed a nontrivial solution to* $x^4 - 1 \equiv 0 \bmod 625$ *using Newton iteration with the starting value* $x = 2$ *modulo* 5. *This can also be regarded as lifting the factorization*

$$x^4 - 1 \equiv (x-2)(x^3 + 2x^2 - x - 2) \bmod 5$$

to a factorization modulo 625. *In the above setting, we then have* $f = x^4 - 1$, $p = 5$, $g_1 = x^3 + 2x^2 - x - 2$ *and* $h_1 = x - 2$. *The polynomials* g_1 *and* h_1 *are relatively prime modulo* 5, *and with the Extended Euclidean Algorithmin* $\mathbb{F}_5[x]$, *we get* $s = -2$ *and* $t = 2x^2 - 2x + 1$ *such that* $sg_1 + th_1 \equiv 1 \bmod p$. *With* $k = 1$, $g = g_1$, *and* $h = h_1$, *we obtain*

$$\begin{aligned} e &= f - gh = 5x^2 - 5, \\ \hat{g} &= g + te = 10x^4 - 9x^3 - 13x^2 + 9x + 3, \\ \hat{h} &= h + se = -10x^2 + x + 8, \end{aligned}$$

and in fact

$$f - \hat{g}\hat{h} = 25 \cdot (4\,x^6 - 4\,x^5 - 8\,x^4 + 7\,x^3 + 5\,x^2 - 3\,x - 1) \equiv 0 \bmod 25,$$

i.e., $f \equiv \hat{g}\hat{h} \bmod 25$.

The above example shows one drawback in our first approach: the degrees of $\hat{g}$ and $\hat{h}$ are higher than those of g and h, in particular their sum exceeds the degree of f. This may happen because the multiples of p are zero divisors modulo p^k if $k \geq 2$, and hence the product of the leading coefficients of two polynomials may vanish modulo p^k.

To overcome this problem, we use division with remainder in $\mathbb{Z}[x]$. Since $\mathbb{Z}$ is not a field, this is not always possible (try dividing x by 2 with remainder). The following lemma states that division with remainder by *monic* polynomials always works. Its proof is left as an exercise.

LEMMA 18.2. *(i) Let $f, g \in \mathbb{Z}[x]$, g nonzero and monic. Then there exist unique polynomials $q, r \in \mathbb{Z}[x]$ with $f = qg + r$ and $\deg r < \deg g$.*

(ii) If f, g, q, r are as in (i) and $f \equiv 0 \bmod m$ for some $m \in \mathbb{Z}$, then $q \equiv r \equiv 0 \bmod m$.

With a factorization g, h of f modulo p^k as before and the additional requirement that h be monic, we define the new factorization by

$$\begin{array}{rcl} e &=& f - gh \text{ rem } p^{k+1}, \\ se &=& qh + r \text{ rem } p^{k+1} \text{ with } \deg r < \deg h, \\ g^* &=& g + te + qg \text{ rem } p^{k+1}, \\ h^* &=& h + r \text{ rem } p^{k+1} = h + se - qh \text{ rem } p^{k+1}. \end{array} \tag{18.2}$$

Recall that "rem p^{k+1}" is the remainder on division by p^{k+1}. In the second row of (18.2), we mean that the coefficients of q and r are reduced modulo p^{k+1}. We have two standard choices for such a remainder: either all coefficients are nonnegative and less than p^{k+1}, or the more symmetric variant where all coefficients are in the range between $-\frac{p^{k+1}}{2}$ and $\frac{p^{k+1}}{2}$. In our examples, we will use the latter. This reduction modulo p^{k+1} is justified by the fact that we want to use this inductively and are therefore only interested in a factorization modulo p^{k+1}. This also keeps the size of the coefficents as small as possible.

EXAMPLE 18.3. *(Example 18.1 continued) Let $p = 5$ and $f, g = g_1, h = g_1, s, t, e \in \mathbb{Z}[x]$ as in Example 18.1. Division of se by h with remainder yields $se \equiv qh + r \bmod 25$, where*

$$q = -10\,x + 5, \quad r = -5.$$

Furthermore,

$$\begin{array}{rcl} g^* &\equiv& g + te + qg \equiv x^3 + 7x^2 - x - 7 \bmod 25, \\ h^* &\equiv& h + r \equiv x - 7 \bmod 25, \end{array}$$

$f \equiv g^* h^* \bmod 25$, and the degrees of g^*, h^* are the same as that of g and h. As in Example 8.9, we obtain that 7 is a solution to $x^4 - 1 \equiv 0 \bmod 25$ that is congruent to the starting solution 2 modulo 5.

Here comes the central lemma about Hensel lifting.

LEMMA 18.4. *Let $p \in \mathbb{Z}$ be prime, $k \in \mathbb{N}$, $f, g, h, s, t \in \mathbb{Z}[x]$, h be monic, $\deg f = n = \deg g + \deg h$, and*

$$f \equiv gh \bmod p^k, \quad sg + th \equiv 1 \bmod p.$$

If we define $g^*, h^* \in \mathbb{Z}[x]$ by (18.2), then h^* is monic, and

$$\begin{aligned} f &\equiv g^* h^* \bmod p^{k+1}, \\ g^* &\equiv g \bmod p^k, \quad h^* \equiv h \bmod p^k, \\ sg^* + th^* &\equiv 1 \bmod p^{k+1}, \\ \deg g^* &= \deg g, \quad \deg h^* = \deg h. \end{aligned}$$

PROOF. We calculate

$$\begin{aligned} f - g^* h^* &\equiv f - (g + te + qg)(h + se - qh) \\ &= f - gh - (sg + th)e - ste^2 - (sg - th)qe + ghq^2 \\ &\equiv (1 - sg - th)e - ste^2 - (sg - th)qe + ghq^2 \equiv 0 \bmod p^{k+1}, \end{aligned}$$

since $1 - sg - th \equiv 0 \bmod p$, $e \equiv 0 \bmod p^k$, and $q \equiv 0 \bmod p^k$, by Lemma 18.2. These also imply that $g^* \equiv g \bmod p^k$ and $h^* \equiv h \bmod p^k$, and hence $sg^* + th^* \equiv sg + th \equiv 1 \bmod p$. Now $\deg r < \deg h$ by construction, and thus h^* is also monic and of the same degree as h. Finally, $f \equiv g^* h^* \bmod p^{k+1}$ and h^* monic imply that $\deg g^* = \deg f - \deg h^* = \deg f - \deg h = \deg g$. □

If one starts with the stronger assumption $sg + th \equiv 1 \bmod p^k$, one obtains $f \equiv g^* h^* \bmod p^{2k}$. This leads to a Hensel lifting with "quadratic convergence" like Newton iteration (see section 8). Asymptotically, however, the running times of both variants are the same, since the smaller number of steps is weighed out by a greater amount of work per step.

THEOREM 18.5. *(Hensel's Lemma, finite version) With the assumptions of Lemma 18.4 for $k = 1$, one can for any $N \in \mathbb{N}$ compute $\bar{g}, \bar{h} \in \mathbb{Z}[x]$, $\bar{h}$ monic, such that*

$$\begin{aligned} f &\equiv \bar{g}\bar{h} \bmod p^N, \\ \bar{g} &\equiv g \bmod p, \quad \bar{h} \equiv h \bmod p, \\ s\bar{g} + t\bar{h} &\equiv 1 \bmod p^N, \\ \deg \bar{g} &= \deg g, \quad \deg \bar{h} = \deg h. \end{aligned}$$

Proof. Apply Lemma 18.4 inductively. □

There is also an infinite version of Hensel's Lemma. The ring $\mathbb{Z}_{(p)}$ of *p-adic integers*, whose elements can be represented by "power series in p" of the form $\sum_{i\geq 0} a_i p^i$ with $0 \leq a_i < p$ for all $i \in \mathbb{N}$, is the *completion* of $\mathbb{Z}$ with respect to the prime p.

Theorem 18.6. *(Hensel's Lemma, infinite version) With the assumptions of Lemma 18.4 for $k = 1$, there are $\check{g}, \check{h} \in \mathbb{Z}_{(p)}[x]$, $\check{h}$ monic, such that*

$$\begin{aligned} f &= \check{g}\check{h}, \\ \check{g} &\equiv g \bmod p, \quad \check{h} \equiv h \bmod p, \\ s\check{g} + t\check{h} &= 1, \\ \deg \check{g} &= \deg g, \quad \deg \check{h} = \deg h. \end{aligned}$$

For completeness, we now state the algorithm.

Algorithm 18.7. *Hensel lifting*
Input: *A prime $p \in \mathbb{N}$, polynomials $f, g, h, s, t \in \mathbb{Z}[x]$, such that h is monic, $\deg f = n = \deg g + \deg h$, $f \equiv gh \bmod p$, $sg + th \equiv 1 \bmod p$, and $N \in \mathbb{N}$.*
Output: *$\bar{g}, \bar{h} \in \mathbb{Z}[x]$ such that $\bar{h}$ is monic, $f \equiv \bar{g}\bar{h} \bmod p^N$, $\bar{g} \equiv g \bmod p$, $\bar{h} \equiv h \bmod p$, $\deg \bar{h} = \deg h$, and $\deg \bar{g} = \deg g$.*

1. *Set $g_1 = g$ and $h_1 = h$.*
2. *Repeat step 3 for $i = 1, \ldots, N-1$.*
3. *Set $e_i = f - g_i h_i \operatorname{rem} p^{i+1}$ and compute $q_i, r_i \in \mathbb{Z}[x]$ with $se_i = q_i h_i + r_i \operatorname{rem} p^{i+1}$ and $\deg r_i < \deg h_i$. Compute $g_{i+1}, h_{i+1} \in R[x]$ such that $g_{i+1} = g_i + te_i + q_i g_i \operatorname{rem} p^{i+1}$ and $h_{i+1} = h_i + r_i \operatorname{rem} p^{i+1}$.*
4. *Return g_N, h_N.*

Example 18.8. *(Example 18.3 continued) Let $p = 5$, $f, s, t \in \mathbb{Z}[x]$ as in Example 18.3, and*

$$g_2 = x^3 + 7\,x^2 - x - 7, \quad h_2 = x - 7.$$

Then $f \equiv g_2 h_2 \bmod 25$, and the next two steps of Algorithm 18.7 yield

$$\begin{aligned} e_2 &\equiv 50\,x^2 - 50 \bmod 125, \\ q_2 &\equiv 25x + 50 \bmod 125, \quad r_2 \equiv -50 \bmod 125, \\ g_3 &\equiv x^3 + 57\,x^2 - x - 57 \bmod 125, \\ h_3 &\equiv x - 57 \bmod 125, \\ e_3 &\equiv 125\,x^2 - 125 \bmod 625, \\ q_3 &\equiv -250x + 125 \bmod 625, \quad r_3 \equiv -125 \bmod 625, \\ g_4 &\equiv x^3 + 182\,x^2 - x - 182 \bmod 625, \\ h_4 &\equiv x - 182 \bmod 625, \end{aligned}$$

$f \equiv g_4h_4 \bmod 625$, and as in Example 8.9, we see that 182 is a fourth root of 1 modulo 625 that is congruent to the starting solution 2 modulo 5.

EXAMPLE 18.9. *Let $p = 3$, $f = x^4 - 2x^3 - 11x^2 + 4x + 3 \in \mathbb{Z}[x]$. Then $f \equiv x(x+1)(x^2+1) \bmod 3$, and with $g_1 = x^2 + x$ and $h_1 = x^2 + 1$, we compute $s = x + 1$ and $t = -x + 1$ such that $sg_1 + th_1 \equiv 1 \bmod 3$. The next two steps of Hensel lifting yield*

$$\begin{aligned}
e_2 &\equiv -3x^3 - 3x^2 + 3x + 3 \bmod 9, \\
q_2 &\equiv -3x^2 + 3x + 3 \bmod 9, \quad r_2 \equiv 3x \bmod 9, \\
g_3 &\equiv x^2 + 4x + 3 \bmod 9, \\
h_3 &\equiv x^2 + 3x + 1 \bmod 9, \\
e_3 &\equiv -9x^3 - 9x \bmod 27, \\
q_3 &\equiv -9x^2 - 9x \bmod 27, \quad r_3 \equiv 0 \bmod 27, \\
g_3 &\equiv x^2 - 5x + 3 \bmod 27, \\
h_3 &\equiv x^2 + 3x + 1 \bmod 27, \\
e_4 &\equiv 0 \bmod 81,
\end{aligned}$$

and in fact $f = g_3h_3$ is the irreducible factorization of f in $\mathbb{Z}[x]$.

We denote by $\mathsf{M}(n)$ the cost for multiplying n-bit integers, in bit operations. By Schönhage & Strassen (1971), we may assume $\mathsf{M}(n) = n \log n \operatorname{loglog} n$.

THEOREM 18.10. *Algorithm 18.7 uses no more than $O(N \cdot \mathsf{M}(n)\mathsf{M}(N \log p))$ bit operations.*

PROOF. We note that for any arithmetic operation (addition, multiplication, division with remainder) of polynomials in $\mathbb{Z}[x]$ occuring in the algorithm, their degrees and the bit length of their coefficients are bounded from above by $2n$ and $2N\lceil \log_2 p \rceil$, respectively. The cost for one arithmetical operation (addition, multiplication, division with remainder) in $\mathbb{Z}$ is $O(\mathsf{M}(N \log p))$ bit operations. The number of arithmetic operations in $\mathbb{Z}$ for one polynomial addition, multiplication, or division with remainder is $O(\mathsf{M}(n))$. Per iteration of the algorithm, a constant number of operations on polynomials is performed, and the cost for one iteration is $O(\mathsf{M}(n)\mathsf{M}(N \log p))$ bit operations. Hence we get a total cost of $O(N \cdot \mathsf{M}(n)\mathsf{M}(N \log p))$ for $N - 1$ iterations. □

THEOREM 18.11. *(Uniqueness of Hensel lifting) Let $p \in \mathbb{Z}$ be prime, $N \in \mathbb{N}$, and $f, g_1, h_1, g_2, h_2 \in \mathbb{Z}[x]$ with h_1, h_2 monic, $\deg h_1 = \deg h_2$, $g_1 \equiv g_2 \bmod p$, $h_1 \equiv h_2 \bmod p$, $\gcd(g_1, h_1) = 1$ modulo p, and $f \equiv g_1h_1 \equiv g_2h_2 \bmod p^N$. Then $g_1 \equiv g_2 \bmod p^N$ and $h_1 \equiv h_2 \bmod p^N$.*

PROOF. We prove by induction that $g_1 \equiv g_2 \bmod p^l$ and $h_1 \equiv h_2 \bmod p^l$ hold for $1 \le l \le N$. There is nothing to prove for $l = 1$, so we assume that $l \ge 2$. Now

$$\begin{aligned}(h_1 - h_2)g_1 + (g_1 - g_2)h_1 &\equiv (f - g_2h_2) - (f - g_1h_1) + (g_1 - g_2)(h_1 - h_2)\\ &\equiv 0 \bmod p^l, \qquad (18.3)\end{aligned}$$

since $(g_1 - g_2)(h_1 - h_2) \equiv 0 \bmod p^{2l-2}$, by the induction hypothesis, and $2l - 2 \ge l$ if $l \ge 2$. If we let $\bar{s} = \frac{h_1 - h_2}{p^{l-1}}$ and $\bar{t} = \frac{g_1 - g_2}{p^{l-1}}$, and divide 18.3 by p^{l-1}, we obtain

$$\bar{s}g_1 + \bar{t}h_1 \equiv 0 \bmod p. \qquad (18.4)$$

This together with the relative primality of g_1 and h_1 modulo p implies that $h_1 \bmod p$ divides $\bar{s} \bmod p$. But $\deg(\bar{s}) < \deg h_1$ since h_1, h_2 are monic polynomials of the same degree, and hence $\bar{s} \equiv 0 \bmod p$. Again by 18.4, we find that also $\bar{t} \equiv 0 \bmod p$, and the claim follows. □

REMARK 18.12. *Hensel lifting can be regarded as a two-dimensional Newton iteration. Consider $\Phi = f - uv \in \mathbb{Z}[x][u,v]$ as a bivariate polynomial in the indeterminates u and v with coefficients in $\mathbb{Z}[x]$. One step in the iteration is then, given $k \in \mathbb{N}$ and a zero $(g,h) \in \mathbb{Z}[x]^2$ of Φ modulo p^k, to compute a zero $(g^*, h^*) \in \mathbb{Z}[x]^2$ of Φ modulo p^{k+1}. Then similar formulas as (18.2) for g^*, h^* can be derived from the first-degree Taylor expansion of Φ, which equals*

$$\Phi(u + \bar{t}, v + \bar{s}) = \Phi + \frac{\partial \Phi}{\partial u}\bar{t} + \frac{\partial \Phi}{\partial v}\bar{s} - \bar{t}\bar{s}$$

for arbitrary $\bar{t}, \bar{s} \in \mathbb{Z}[x]$, where $\frac{\partial \Phi}{\partial u} = v$ and $\frac{\partial \Phi}{\partial v} = u$. By substituting $u = g$, $v = h$, letting $\bar{t} = g^* - g$, $\bar{s} = h^* - h$, dropping the quadratic error term $\bar{t}\bar{s}$ (which is supposed to vanish modulo p^{k+1}), and solving $\Phi(g^*, h^*) \equiv 0 \bmod p^{k+1}$, one arrives at the linear Diophantine equation

$$e = \Phi(g,h) \equiv h \cdot \bar{t} + g \cdot \bar{s} \bmod p^{k+1},$$

which can be shown to have a solution $(\bar{t}, \bar{s}) \in \mathbb{Z}[x]^2$ such that $g^* = g + \bar{t}$ and $h^* = h + \bar{s}$ satisfy the conclusions of Lemma 18.4, provided that g and h are relatively prime modulo p.

REMARK 18.13. *The general multivariate Newton iteration works as follows. One has n functions $f = (f_1, \ldots, f_n)$ in n variables $x_1, \ldots, x_n$, and from an approximation $a \in \mathbb{R}^n$ to a common root of $f_1, \ldots, f_n$ obtains a better one a^* as*

$$a^* = a - J^{-1}(a)f(a),$$

where $J = (\partial f_i / \partial x_j)_{1 \le i,j \le n} \in \mathbb{R}[x_1, \ldots, x_n]^{n \times n}$ is the Jacobian *of f.*

Hensel lifting can also be considered as a multivariate Newton iteration in the above sense, as follows. We regard the coefficients of g and h in a factorization $f = gh$ as indeterminates. Comparing the coefficients of $1, x, \ldots, x^{n-1}$ on both sides, we obtain n equations for the n unknown coefficients, and a common solution to those equations corresponds to a factorization of f.

19. Short vectors in lattices

The methods we discuss in this section deal with computational aspects of the "geometry of numbers", a mathematical theory introduced by Minkowski around the turn of the century. This theory produces many results about Diophantine approximation, convex bodies, embeddings of algebraic number fields in $\mathbb{C}$, and the "ellipsoid method" for rational linear programming (see Kannan (1987) for an overview).

Let $f = (f_1, \ldots, f_n) \in \mathbb{R}^n$. The *norm* of f is

$$|f| = \sqrt{\sum_{1 \le i \le n} f_i^2} = \sqrt{\langle f, f \rangle} \in \mathbb{R},$$

where $\langle f, g \rangle = \sum_{1 \le i \le n} f_i g_i = f \cdot g^T \in \mathbb{R}$ is the usual scalar product of two vectors f and $g = (g_1, \ldots, g_n)$ in $\mathbb{R}^n$. The vectors f and g are *orthogonal* if $\langle f, g \rangle = 0$.

Definition 19.1. *Let $n \in \mathbb{N}$ and $f_1, \ldots, f_n \in \mathbb{R}^n$ with $f_i = (f_{i1}, \ldots, f_{in})$. Then*

$$L = \sum_{1 \le i \le n} \mathbb{Z} f_i = \{ \sum_{1 \le i \le n} r_i f_i \colon r_1, \ldots, r_n \in \mathbb{Z} \}$$

is called the lattice *or* $\mathbb{Z}$-module *generated by $f_1, \ldots, f_n$. The vectors $f_1, \ldots, f_n$ are a* basis *of L. The* norm of L *is $|L| = |\det(f_{ij})_{1 \le i,j \le n}| \in \mathbb{R}$. The lemma below implies that it is well defined, i.e., the norm is independent of the choice of the generators.*

Lemma 19.2. *Let $N \subseteq M \subseteq \mathbb{R}^n$ be lattices, generated by $g_1, \ldots, g_n$ and $f_1, \ldots, f_n$, respectively, where $f_i = (f_{i1}, \ldots, f_{in})$ and $g_i = (g_{i1}, \ldots, g_{in})$. Then*

$$|\det(g_{ij})_{1 \le i,j \le n}| = 0 \text{ or } |\det(g_{ij})_{1 \le i,j \le n}| \ge |\det(f_{ij})_{1 \le i,j \le n}|.$$

Proof. For $1 \le i, j \le n$ there exist $a_{ij} \in \mathbb{Z}$ such that $g_i = \sum_{1 \le i \le n} a_{ij} f_j$. Hence

$$|\det(g_{ij})| = |\det(a_{ij})| \cdot |\det(f_{ij})|,$$

and the claim follows. □

If we let $N = M$ in the above lemma, i.e., $f_1, \ldots, f_n$ and $g_1, \ldots, g_n$ both generate the same module, we see that $|\det(f_{ij})| = |\det(g_{ij})|$. Hence the norm is indeed independent of the choice of basis of L. Geometrically, $|L|$ is the volume of the parallelepiped spanned by $f_1, \ldots, f_n$, and Hadamard's inequality (12.4) says that $|L| \le |f_1| \cdots |f_n|$ holds.

Example 19.3. *$n = 2$, $f_1 = (-2, 1)$, $f_2 = (2, 0)$, $L = \mathbb{Z} f_1 + \mathbb{Z} f_2$. Figure 19.1 shows the lattice points of L around the origin of the plane $\mathbb{R}^2$. The norm of L is*

$$|L| = \left| \det \begin{pmatrix} -2 & 1 \\ 2 & 0 \end{pmatrix} \right| = 2$$

and equals the area of the parallelogram shaded in Figure 19.1. Another basis of L is $g_1 = (0, 1)$ and $g_2 = f_2$. Both $(0, 1)$ and $(0, -1)$ are shortest vectors in L w.r.t. the Euclidean norm $|\cdot|$.

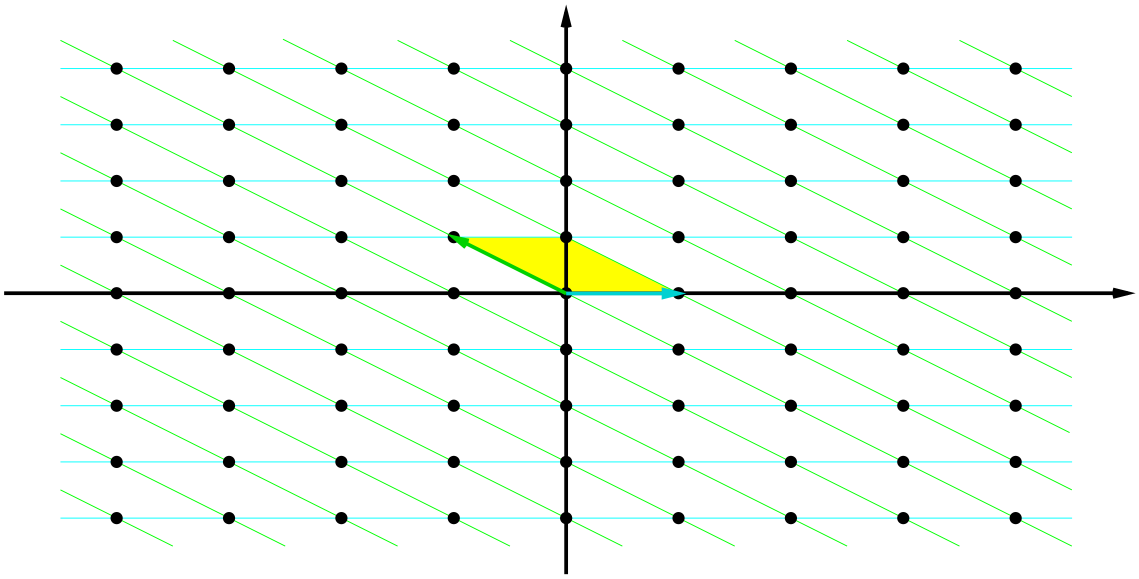


Figure 19.1: The lattice in $\mathbb{R}^2$ generated by $(-2,1)$ and $(2,0)$.

Our goal is to compute "short" vectors in lattices. No polynomial time algorithm for computing a *shortest* vector in a lattice is known, and it is conjectured that the latter task is $\mathcal{NP}$-hard. But for our main application, the factorization of polynomials with integer coefficients, it will be sufficient to compute a "relatively short" vector, a problem for which Lenstra *et al.* (1982) first gave a polynomial time algorithm.

Recall the Gram-Schmidt orthogonalization procedure from linear algebra. Given an arbitrary basis $(f_1,\ldots,f_n)$ of $\mathbb{R}^n$, it computes an orthogonal basis $(f_1^*,\ldots,f_n^*)$ of $\mathbb{R}^n$. The f_i^* are defined inductively as follows.

$$f_i^* = f_i - \sum_{1\le j<i} \mu_{ij} f_j^*, \text{ where } \mu_{ij} = \frac{\langle f_i, f_j^*\rangle}{\langle f_j^*, f_j^*\rangle} = \frac{f_i (f_j^*)^T}{|f_j^*|^2} \text{ for } 1 \le j < i. \tag{19.1}$$

In particular, $f_1^* = f_1$. We will call $(f_1^*,\ldots,f_n^*)$ the *Gram-Schmidt orthogonal basis* (or GSO for short) of $(f_1,\ldots,f_n)$. It is clear that $f_1^*,\ldots,f_n^*$ have rational coefficients if $f_1,\ldots,f_n$ have. The μ_{ij} are called the *basis change coefficients* in the sequel.

Example 19.4. $n=3$, $f_1=(1,1,0)$, $f_2=(1,0,1)$, $f_3=(0,1,1)$. *We calculate* $f_1^* = f_1 = (1,1,0)$,

$$\mu_{21} = \frac{\langle f_2, f_1^*\rangle}{\langle f_1^*, f_1^*\rangle} = \frac{1}{2}, \quad f_2^* = f_2 - \mu_{21} f_1^* = \left(\frac{1}{2}, -\frac{1}{2}, 1\right),$$

$$\mu_{31} = \frac{\langle f_3, f_1^*\rangle}{\langle f_1^*, f_1^*\rangle} = \frac{1}{2}, \quad \mu_{32} = \frac{\langle f_3, f_2^*\rangle}{\langle f_2^*, f_2^*\rangle} = \frac{1}{3}, \quad f_3^* = f_3 - \mu_{31} f_1^* - \mu_{32} f_2^* = \left(-\frac{2}{3}, \frac{2}{3}, \frac{2}{3}\right).$$

The following theorem collects the properties of the Gram-Schmidt orthogonal basis we will need. The proof is left to the reader; see, e.g., Brieskorn (1985).

THEOREM 19.5. *Let $f_1, \ldots, f_n \in \mathbb{R}^n$ be linearly independent, and f_i^*, μ_{ij} as in (19.1). Define $U_k = \operatorname{span}\{f_1, \ldots, f_k\} \subseteq \mathbb{R}^n$ for $0 \le k \le n$.*

(i) $\operatorname{span}\{f_1^*, \ldots, f_k^*\} = U_k$ *for* $0 \le k \le n$.

(ii) For $1 \le k \le n$, f_k^* is the projection of f_k onto the orthogonal complement $U_{k-1}^{\perp} = \{f \in \mathbb{R}^n : \langle f, u\rangle = 0$ for all $u \in U_{k-1}\}$ of U_{k-1}, and hence in particular $|f_k^*| \le |f_k|$.

(iii) Define $\mu_{ii} = 1$ and $\mu_{ij} = 0$ if $j > i$. Then the matrix $M_k = (\mu_{ij})_{1 \le i,j \le k} \in \mathbb{R}^{k \times k}$ is lower triangular with ones on the diagonal, and it is the matrix for the change of basis from $f_1^*, \ldots, f_k^*$ to $f_1, \ldots, f_k$, i.e.,

$$\begin{pmatrix} f_1 \\ \vdots \\ f_k \end{pmatrix} = M_k \begin{pmatrix} f_1^* \\ \vdots \\ f_k^* \end{pmatrix}$$

for $1 \le k \le n$.

(iv) $\prod_{1 \le j \le k} |g_j^*|^2 = \det(\langle g_i, g_j \rangle)_{1 \le i,j \le k}$ *for $1 \le k \le n$. The matrix $(\langle g_i, g_j \rangle)_{1 \le i,j \le k} \in \mathbb{R}^{k \times k}$ is called the* Gramian matrix *of $g_1, \ldots, g_k$, its determinant is the* Gramian determinant *of $g_1, \ldots, g_k$.*

(v) $f_1^*, \ldots, f_n^*$ are pairwise orthogonal, i.e., $\langle f_i^*, f_j^* \rangle = 0$ if $i \ne j$.

(vi) $\det \begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix} = \det \begin{pmatrix} f_1^* \\ \vdots \\ f_n^* \end{pmatrix}$.

The next lemma shows the connection between the Gram-Schmidt orthogonal bases and short vectors.

LEMMA 19.6. *Let $L \subseteq \mathbb{R}^n$ be a lattice with basis $(f_1, \ldots, f_n)$, and let $(f_1^*, \ldots, f_n^*)$ its GSO. Then for any $f \in L \setminus \{0\}$*

$$|f| \ge \min\{|f_1^*|, \ldots, |f_n^*|\}.$$

PROOF. Let $f = \sum_{1 \le i \le n} \lambda_i f_i \in L \setminus \{0\}$ be arbitrary, with $\lambda_i \in \mathbb{Z}$ for $1 \le i \le n$, and let k be the highest index such that $\lambda_k \ne 0$. Substituting $f_i^* + \sum_{1 \le j < i} \mu_{ij} f_j^*$ for f_i, where the μ_{ij} are as in (19.1), yields

$$f = \sum_{1 \le i \le k} \lambda_i (f_i^* + \sum_{1 \le j < i} \mu_{ij} f_j^*) = \lambda_k f_k^* + \sum_{1 \le i < k} \nu_i f_i^*$$

for some appropriate $\nu_i \in \mathbb{R}$. Then

$$\begin{aligned}
|f|^2 &= \langle f, f\rangle = \left\langle \lambda_k f_k^* + \sum_{1\le i<k} \nu_i f_i^*,\ \lambda_k f_k^* + \sum_{1\le i<k} \nu_i f_i^* \right\rangle \\
&= \lambda_k^2 \langle f_k^*, f_k^*\rangle + \sum_{1\le i<k} \nu_i^2 \langle f_i^*, f_i^*\rangle \ge \lambda_k^2 \cdot |f_k^*|^2 \\
&\ge |f_k^*|^2 \ge \min\{|f_1^*|^2, \ldots, |f_n^*|^2\},
\end{aligned}$$

where we used the pairwise orthogonality of the f_i^* and the fact that $\lambda_k \in \mathbb{Z} \setminus \{0\}$.
□

Our goal is to compute a short vector in L. If the GSO of $(f_1, \ldots, f_n)$ is a basis for the lattice L generated by $f_1, \ldots, f_n$, then the lemma says that one of the f_i^* is a shortest vector. But usually the f_i^* are not even in L (see Example 19.4). This problem motivates the following definition of a reduced basis as an "almost orthogonal" basis of L.

Definition 19.7. *Let $f_1, \ldots, f_n \in \mathbb{R}^n$ be linearly independent and f_i^*, μ_{ij} as in (19.1). Then $(f_1, \ldots, f_n)$ is* reduced *if*

(L_1) $|\mu_{ij}| \le \frac{1}{2}$ *for* $1 \le j < i \le n$,

(L_2) $|f_i^* + \mu_{i,i-1} f_{i-1}^*|^2 \ge \frac{3}{4}|f_{i-1}^*|^2$ *for* $1 < i \le n$.

Taking a closer look, we see that the basis from Example 19.4 is reduced.

Theorem 19.8. *Let $(f_1, \ldots, f_n)$ be a reduced basis of the lattice $L \subseteq \mathbb{R}^n$. Then the following hold.*

(i) $|f_1| \cdots |f_n| \le 2^{\frac{n(n-1)}{4}} \cdot |L|$,

(ii) $|f_1| \le 2^{\frac{n-1}{4}} \cdot \sqrt[n]{|L|}$,

(iii) $|f_1| \le 2^{\frac{n-1}{2}} \cdot \min\{|f| \colon 0 \ne f \in L\}$.

Proof. Let f_i^*, μ_{ij} be as in (19.1), and $1 < i \le n$. Then

$$\begin{aligned}
|f_i^* + \mu_{i,i-1} f_{i-1}^*|^2 &= \left\langle f_i^* + \mu_{i,i-1} f_{i-1}^*,\ f_i^* + \mu_{i,i-1} f_{i-1}^* \right\rangle \\
&= \langle f_i^*, f_i^*\rangle + 2\mu_{i,i-1} \left\langle f_i^*, f_{i-1}^*\right\rangle + \mu_{i,i-1}^2 \left\langle f_{i-1}^*, f_{i-1}^*\right\rangle \\
&= |f_i^*|^2 + \mu_{i,i-1}^2 |f_{i-1}^*|^2 \ge \frac{3}{4}|f_{i-1}^*|^2,
\end{aligned}$$

since f_i^* and f_{i-1}^* are orthogonal and by (L_2). Hence

$$|f_i^*|^2 \ge \left(\frac{3}{4} - \mu_{i,i-1}^2\right) |f_{i-1}^*|^2 \ge \frac{1}{2}|f_{i-1}^*|^2,$$

by (L_1). Using this repeatedly, we get

$$|f_i^*|^2 \geq 2^{j-i} \cdot |f_j^*|^2 \text{ for } 1 \leq j \leq i. \tag{19.2}$$

(i) Fix i with $1 \leq i \leq n$. Since $f_i = f_i^* + \sum_{1 \leq j < i} \mu_{ij} f_j^*$, we have

$$\begin{aligned}
|f_i|^2 &= \langle f_i, f_i \rangle = \left\langle f_i^* + \sum_{1 \leq j < i} \mu_{ij} f_j^*,\ f_i^* + \sum_{1 \leq j < i} \mu_{ij} f_j^* \right\rangle \\
&= \langle f_i^*, f_i^* \rangle + \sum_{1 \leq j < i} \mu_{ij}^2 \langle f_j^*, f_j^* \rangle \leq |f_i^*|^2 + \sum_{1 \leq j < i} \frac{1}{4} |f_j^*|^2 \\
&\leq |f_i^*|^2 \left(1 + \frac{1}{4} \sum_{1 \leq j < i} 2^{i-j} \right) = |f_i^*|^2 \left(1 + \frac{1}{4}(2^i - 2) \right) \\
&\leq 2^{i-1} |f_i^*|^2,
\end{aligned}$$

by (L_1) and (19.2). Multiplying the inequalities for $1 \leq i \leq n$ yields

$$\begin{aligned}
|f_1|^2 \cdots |f_n|^2 &\leq 2^{n(n-1)/2} |f_1^*|^2 \cdots |f_n^*|^2 \\
&= 2^{n(n-1)/2} \cdot \det \left[\begin{pmatrix} f_1^* \\ \vdots \\ f_n^* \end{pmatrix} \left((f_1^*)^T \cdots (f_n^*)^T \right) \right] \\
&= 2^{n(n-1)/2} \cdot \det \begin{pmatrix} f_1^* \\ \vdots \\ f_n^* \end{pmatrix}^2 = 2^{n(n-1)/2} \cdot \det \begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix}^2 \\
&= 2^{n(n-1)/2} \cdot |L|^2,
\end{aligned}$$

by Theorem 19.5.

(ii) Inequality (19.2) implies that $|f_1|^2 = |f_1^*|^2 \leq 2^{i-1} \cdot |f_i^*|^2$. By multiplying the inequalities for $1 \leq i \leq n$, we get

$$|f_1|^{2n} \leq 2^{n(n-1)/2} |f_1^*|^2 \cdots |f_n^*|^2 = 2^{n(n-1)/2} \cdot |L|^2,$$

as above.

(iii) Again by (19.2),

$$\min_{1 \leq i \leq n} |f_i^*|^2 \geq 2^{-(n-1)} \cdot |f_1^*|^2 = 2^{-(n-1)} \cdot |f_1|^2,$$

and the claim follows with Lemma 19.6. □

We now present an algorithm that computes a reduced basis of a lattice $L \subseteq \mathbb{Q}^n$ from an arbitrary basis. We restrict our attention to lattices in $\mathbb{Z}^n$, since the general case can be reduced to this by multiplying with a common denominator of the basis vectors.

ALGORITHM 19.9. *Basis reduction over* $\mathbb{Z}$.
Input: *Linearly independent row vectors* $f_1, \ldots, f_n \in \mathbb{Z}^n$.
Output: *A reduced basis* $(g_1, \ldots, g_n)$ *of the lattice* $L = \sum_{1 \le i \le n} \mathbb{Z} f_i \subseteq \mathbb{Z}^n$.

1. *Set* $g_i = f_i$ *for* $1 \le i \le n$, *and compute the GSO* $(g_1^*, \ldots, g_n^*)$ *of* $(g_1, \ldots, g_n)$ *and the* μ_{kl} *for* $1 \le l < k \le n$ *as in (19.1).*
2. *Repeat step 3 for* $i = 2, \ldots, n$, *and, given* i, *for* $j = i-1, \ldots, 1$.
3. *Replace* g_i *by* $g_i - \lceil \mu_{ij} \rfloor g_j$, *where* $\lceil \mu_{ij} \rfloor$ *is the nearest integer to* μ_{ij}. *For* $1 \le l < j$, *replace* μ_{il} *by* $\mu_{il} - \lceil \mu_{ij} \rfloor \mu_{jl}$, *and replace* μ_{ij} *by* $\mu_{ij} - \lceil \mu_{ij} \rfloor$.
4. *If there is an index* i *with* $1 < i \le n$ *violating condition* (L_2) *of a reduced sequence, exchange* g_i *and* g_{i-1} *(for the smallest such* i*). Compute the GSO* $(g_1^*, \ldots, g_n^*)$ *of* $(g_1, \ldots, g_n)$ *and the* μ_{kl} *for* $1 \le l < k \le n$, *and go back to step 2.*
5. *Return* $g_1, \ldots, g_n$.

Before analyzing the algorithm, we will run it on the lattice of Example 19.3 to see how it works.

step	$\binom{g_1}{g_2}$	μ_{21}	$\binom{g_1^*}{g_2^*}$	action
3	$\begin{pmatrix} -2 & 1 \\ 2 & 0 \end{pmatrix}$	$\frac{-4}{5}$	$\begin{pmatrix} -2 & 1 \\ \frac{2}{5} & \frac{4}{5} \end{pmatrix}$	row 2 = row 2 + row 1
4	$\begin{pmatrix} -2 & 1 \\ 0 & 1 \end{pmatrix}$	$\frac{1}{5}$	$\begin{pmatrix} -2 & 1 \\ \frac{2}{5} & \frac{4}{5} \end{pmatrix}$	exchange rows 1 and 2
3	$\begin{pmatrix} 0 & 1 \\ -2 & 1 \end{pmatrix}$	1	$\begin{pmatrix} 0 & 1 \\ -2 & 0 \end{pmatrix}$	row 2 = row 2 − row 1
5	$\begin{pmatrix} 0 & 1 \\ -2 & 0 \end{pmatrix}$	0	$\begin{pmatrix} 0 & 1 \\ -2 & 0 \end{pmatrix}$	

In this situation, the algorithm actually computes an *orthogonal* basis of the lattice, while in general a lattice need not even have an orthogonal basis. Furthermore, the final g_1 is actually a *shortest* vector. This turns out to be the case for in many practical applications, too, but Theorem 19.8 only says that the norm of the first vector in the computed basis is at most by a factor of $2^{(n-1)/2}$ bigger than the norm of a shortest vector, where n is the dimension of the lattice.

THEOREM 19.10. *(i) If the algorithm terminates, it correctly computes a reduced basis of* L.

(ii) The algorithm runs in polynomial time. It uses $O(n^5 \log B)$ *arithmetic operations on integers whose binary length is* $O(n^3 \log B)$, *where* $B = \max_{1 \le i \le n} |f_i|$.

To structure the somewhat lengthy proof, we first investigate in the following two lemmas how the GSO of $(g_1, \ldots, g_n)$ changes in steps 3 and 4, respectively.

LEMMA 19.11. *Let $g_1, \ldots, g_n \in \mathbb{R}^n$ be linearly independent with GSO $(g_1^*, \ldots, g_n^*)$ and μ_{kl} as in (19.1) for $1 \le l < k \le n$. Furthermore, let $1 \le j < i \le n$ and $\lambda \in \mathbb{R}$, and define $h_1, \ldots, h_n \in \mathbb{R}^n$ by $h_i = g_i - \lambda g_j$ and $h_k = g_k$ if $k \ne i$. Let $(h_1^*, \ldots, h_n^*)$ be the GSO of $(h_1, \ldots, h_n)$, and denote the corresponding basis change coefficients by ν_{kl} for $1 \le l < k \le n$. Then the following hold.*

(i) $\sum_{1 \le k \le n} \mathbb{Z} g_k = \sum_{1 \le k \le n} \mathbb{Z} h_k$, *and* $h_k^* = g_k^*$ *for* $1 \le k \le n$.

(ii) $\nu_{kl} = \mu_{kl}$ *for* $1 \le l < k \le n$ *if* $k \ne i$, *and*

$$\nu_{il} = \begin{cases} \mu_{il} - \lambda \mu_{jl} & \text{if } l < j, \\ \mu_{ij} - \lambda & \text{if } l = j, \\ \mu_{il} & \text{if } l > j. \end{cases}$$

PROOF. (i) The first statement is clear. Obviously $h_k^* = g_k^*$ for $1 \le k < i$, and $h_i^* = g_i^*$ since h_i^* is the projection of $h_i = g_i - \lambda g_j$ onto $\operatorname{span}\{h_1, \ldots, h_{i-1}\}^\perp = \operatorname{span}\{g_1, \ldots, g_{i-1}\}^\perp$. By a similar argument, we see that $h_k^* = g_k^*$ for $i < k \le n$, since $\operatorname{span}\{h_1, \ldots, h_{k-1}\} = \operatorname{span}\{g_1, \ldots, g_{k-1}\}$.

(ii) By (i), we see that $\nu_{kl} = \mu_{kl}$ if $k \ne i$. For $1 \le l \le n$, we have

$$\nu_{il} = \frac{\langle h_i, h_l^* \rangle}{\langle h_l^*, h_l^* \rangle} = \frac{\langle g_i - \lambda g_j, g_l^* \rangle}{\langle g_l^*, g_l^* \rangle} = \frac{\langle g_i, g_j^* \rangle}{\langle g_l^*, g_l^* \rangle} - \lambda \frac{\langle g_j, g_l^* \rangle}{\langle g_l^*, g_l^* \rangle} = \mu_{il} - \lambda \mu_{jl},$$

with $\mu_{jj} = \frac{\langle g_j, g_j^* \rangle}{\langle g_j^*, g_j^* \rangle} = 1$ and $\mu_{jl} = \frac{\langle g_j, g_l^* \rangle}{\langle g_l^*, g_l^* \rangle} = 0$ for $j < l \le n$, and the claim follows. □

LEMMA 19.12. *Let g_k, g_k^* for $1 \le k \le n$ and μ_{kl} for $1 \le l < k \le n$ be as in the previous lemma. Fix some i with $1 < i \le n$, and define $h_1, \ldots, h_n \in \mathbb{R}^n$ by $h_{i-1} = g_i$, $h_i = g_{i-1}$, and $h_k = g_k$ if $k \notin \{i-1, i\}$. Let $(h_1^*, \ldots, h_n^*)$ be the GSO of $(h_1, \ldots, h_n)$, and denote the corresponding basis change coefficients by ν_{kl} for $1 \le l < k \le n$. Then the following hold.*

$$\begin{aligned} h_k^* &= g_k^* \text{ for } 1 \le k \le n \text{ with } k \notin \{i-1, i\}, \\ h_{i-1}^* &= g_i^* + \mu_{i,i-1} g_{i-1}^*, \\ h_i^* &= \frac{|g_i^*|^2 g_{i-1}^* - \mu_{i,i-1} |g_{i-1}^*|^2 g_i^*}{|h_{i-1}^*|^2}. \end{aligned}$$

PROOF. The first statement is clear since $\operatorname{span}\{h_1, \ldots, h_{k-1}\} = \operatorname{span}\{g_1, \ldots, g_{k-1}\}$ if $k \notin \{i-1, i\}$. Now h_{i-1}^* is the projection of $h_{i-1} = g_i = g_i^* + \mu_{i,i-1} g_{i-1}^* + \sum_{1 \le j < i-1} \mu_{ij} g_j^*$ onto $\operatorname{span}\{g_1, \ldots, g_{i-2}\}^\perp = \operatorname{span}\{g_1^*, \ldots, g_{i-2}^*\}^\perp$, and hence $h_{i-1}^* = g_i^* + \mu_{i,i-1} g_{i-1}^*$. We do not need the formula for h_i^* in the following, and it is left as an exercise. □

PROOF of Theorem 19.10. The idea is to show that each execution of steps 2 and 3 makes the current basis "a little more reduced", namely that we know (L_1) to hold for more values of i and j than before. If the condition in step 4 is satisfied, we may lose all "reducedness" gained so far. In fact, at first glance it is not obvious that the algorithm terminates at all, since the "gain and loss" of steps 2/3 and step 4 might continue forever. The crucial point then is to exhibit a value D, which is not too large at the beginning and always at least 1, does not change in steps 2 and 3, and strictly decreases by a factor of 3/4 if the condition in step 4 is satisfied. The conclusion then is that the exchange of step 4 occurs only a polynomial number of times.

(i) The preceding lemmas imply that the module generated by $(g_1, \ldots, g_n)$ does not change during the algorithm, and that each time before step 2, step 3, and step 4, $(g_1^*, \ldots, g_n^*)$ is the GSO of $(g_1, \ldots, g_n)$, with basis change coefficients μ_{kl} for $1 \le l < k \le n$ as in (19.1). We now prove that the following invariants hold each time the algorithm passes through step 2.

$$|\mu_{kl}| \le \frac{1}{2} \text{ for } 1 \le l < k < i, \text{ and } |\mu_{il}| \le \frac{1}{2} \text{ for } j < l < i. \tag{19.3}$$

There is nothing to prove if $i = 1$, so fix i and j with $1 \le j < i \le n$ and suppose that (19.3) holds. The values of μ_{kl} do not change in step 3 for $1 \le l < k < i$ and $j < l < k = i$, and in addition $|\mu_{ij}| \le \frac{1}{2}$ after step 3, by definition of $\lceil \cdot \rfloor$, so that (19.3) holds the next time through step 2.

In particular, condition (L_1) of a reduced basis is satisfied before step 4. If the algorithm terminates, then there was no index i violating condition (L_2) in step 4, and $(g_1, \ldots, g_n)$ is a reduced basis of L as claimed.

(ii) The number of operations in $\mathbb{Z}$ used in steps 2 through 4 is $O(n^3)$. To determine the number of iterations of the algorithm, consider

$$D = D(g_1, \ldots, g_n) = |g_1^*|^{2n} |g_2^*|^{2(n-1)} \cdots |g_n^*|^2 = \prod_{1 \le i \le n} d_i,$$

where

$$d_i = \prod_{1 \le j \le i} |g_j^*|^2 = \det(\langle g_k, g_l \rangle)_{1 \le k,l \le i} \in \mathbb{Z}$$

is the Gramian determinant of $g_1, \ldots, g_i$ for $1 \le i \le n$, by Theorem 19.5 (iv). Hence we always have $1 \le D \in \mathbb{Z}$, and initially

$$D_0 = D(f_1, \ldots, f_n) = |f_1^*|^{2n} \cdots |f_n^*|^2 \le |f_1|^{2n} \cdots |f_n|^2 \le (|f_1| \cdots |f_n|)^{2n},$$

since f_i^* is a projection of f_i for $1 \le i \le n$. By Lemma 19.11, D does not change in steps 2 and 3, and with the notation of Lemma 19.12, in step 4 we have

$$\begin{aligned} |h_{i-1}^*|^2 &= \langle h_{i-1}^*, h_{i-1}^* \rangle = \langle g_i^* + \mu_{i,i-1} g_{i-1}^*, \; g_i^* + \mu_{i,i-1} g_{i-1}^* \rangle \\ &= \langle g_i^*, g_i^* \rangle + \mu_{i,i-1}^2 \langle g_{i-1}^*, g_{i-1}^* \rangle = |g_i^*|^2 + \mu_{i,i-1}^2 |g_{i-1}^*|^2 < \frac{3}{4} |g_{i-1}^*|^2, \end{aligned}$$

since (L_2) is violated for i. On the other hand, since both $(g_1, \ldots, g_n)$ and $(h_1, \ldots, h_n)$ generate L over $\mathbb{Z}$,

$$|g_1^*|^2 \cdots |g_n^*|^2 = \det \begin{pmatrix} g_1^* \\ \vdots \\ g_n^* \end{pmatrix}^2 = |L|^2 = \det \begin{pmatrix} h_1^* \\ \vdots \\ h_n^* \end{pmatrix}^2 = |h_1^*|^2 \cdots |h_n^*|^2,$$

and hence $|h_{i-1}^*|^2 |h_i^*|^2 = |g_{i-1}^*|^2 |g_i^*|^2$. Thus d_k remains unchanged for $1 \leq k \leq n$ if $k \neq i-1$, and d_{i-1}—and hence D—decreases at least by a factor of $\frac{3}{4}$ in step 4, so that the number of iterations of the algorithm is bounded by $\log_{4/3} D_0 \in O(n \log(|f_1| \cdots |f_n|))$. We get a total of $O(n^4 \log(|f_1| \cdots |f_n|))$, or $O(n^5 \log B)$ operations in $\mathbb{Z}$, as claimed.

To conclude that this is indeed polynomial time, it remains to show that the numbers occuring in the course of the algorithm stay polynomially bounded. Grötschel *et al.* (1993) prove in section 5.3 that the integers (including numerators and denominators of rational numbers) are absolutely bounded from above by $(2nDB)^n \sqrt{n} B$ at any stage of the algorithm, and hence their size is $O(n \log D_0)$, or $O(n^3 \log B)$. □

The following table traces the values of the d_i and of D on the lattice of Example 19.3.

step	$\binom{g_1}{g_2}$	$\binom{g_1^*}{g_2^*}$	action	d_1	d_2	D
3	$\begin{pmatrix} -2 & 1 \\ 2 & 0 \end{pmatrix}$	$\begin{pmatrix} -2 & 1 \\ \frac{2}{5} & \frac{4}{5} \end{pmatrix}$	row 2 = row 2 + row 1	5	4	20
4	$\begin{pmatrix} -2 & 1 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} -2 & 1 \\ \frac{2}{5} & \frac{4}{5} \end{pmatrix}$	exchange rows 1 and 2	5	4	20
3	$\begin{pmatrix} 0 & 1 \\ -2 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 \\ -2 & 0 \end{pmatrix}$	row 2 = row 2 − row 1	1	4	4
5	$\begin{pmatrix} 0 & 1 \\ -2 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 \\ -2 & 0 \end{pmatrix}$		1	4	4

COROLLARY 19.13. *Given linearly independent vectors $f_1, \ldots, f_n \in \mathbb{Z}^n$, we can compute a "short" vector $g \in L = \sum_{1 \leq i \leq n} \mathbb{Z} f_i$ with*

$$|g| \leq 2^{(n-1)/2} \min\{|f| : 0 \leq f \in L\}$$

using $O\tilde{~}(n^8 \log^2 B)$ bit operations, where $B = \max_{1 \leq i \leq n} |f_i|$.

PROOF. The claim follows immediately from Theorem 19.10, noting that one arithmetic operation in $\mathbb{Z}$ (addition, multiplication, division with remainder, or gcd) on m-bit integers can be performed with $O(\mathsf{M}(m) \log m)$, or $O\tilde{~}(m)$ bit operations. □

We note that $B \leq \sqrt{n}A$ when A is an upper bound on the absolute value of the coefficients of the f_i, and hence $n^8 \log^2 B \in O^{\sim}(n^8 \log^2 A)$, which is indeed polynomial in the input size $n^2 \log_2 A$.

19.1. Euclid and short vectors. It is interesting to note that Euclid's algorithm for the gcd of two integers is a special case of basis reduction. There are two possible senses to this statement. The weaker one is that a special case of the basis reduction algorithm given above achieves what Euclid's algorithms achieves. This is left as an exercise.

The stronger sense is that just from the input/output specifications of the basis reduction algorithm, one can easily recover the output of Euclid's algorithm. We now indicate why that is so; it immediately implies also the weaker sense of reduction.

To compute the gcd $g \in \mathbb{N}$ of $a, b \in \mathbb{N}$, we assume to have a basis reduction algorithm in $\mathbb{Z}^2$ that produces a short vector, up to some factor $c \in \mathbb{N}$. Algorithm 19.9 achieves this with a factor $\sqrt{2}$, so we can use $c = 2$. The basic idea is that $(0, \frac{a}{g}) = \frac{-b}{g} \cdot (a, 0) + \frac{a}{g}(b, 1)$ is a short vector in the 2-dimensional lattice $\mathbb{Z}(a, 0) + \mathbb{Z}(b, 1)$. To make the idea work, we scale the situation appropriately. So let

$$f_1 = (a(2a+1), 0), \quad f_2 = (b(2a+1), 1) \in \mathbb{Z}^2,$$

and suppose our basis reduction algorithm returns a short vector

$$x = (x_1, x_2) \in L = \mathbb{Z}f_1 + \mathbb{Z}f_2 \subseteq \mathbb{Z}^2,$$

so that $|x| \leq 2|y|$ for all $y \in L \setminus \{0\}$. Von zur Gathen (1984b) shows that $x = (0, \frac{\pm a}{g})$ or $x = (0, \frac{\pm 2a}{g})$, and we can compute the gcd as $g = |\frac{a}{x_2}|$ or $g = |\frac{2a}{x_2}|$, respectively. This reduction is a rather theoretical result and *not* meant as a practical alternative to Euclid's algorithm.

EXAMPLE 19.14. *To compute the gcd of $a = 33$ and $b = 12$, using lattice reduction, we set $f_1 = (2211, 0)$ and $f_2 = (804, 1)$. Algorithm 19.9 then runs as in figure 19.2, computes a short vector $x = (0, -11)$, and indeed* $\gcd(33, 12) = 3 = |\frac{1 \cdot 33}{-11}|$.

19.2. Applications of basis reduction over $\mathbb{Z}$.

Factoring polynomials in $\mathbb{Q}[x]$. This is due to Lenstra *et al.* (1982) and will be discussed in the next section.

Breaking knapsack-type cryptosystems. The *knapsack problem* is defined as follows.

> Given a set of $n \in \mathbb{N}$ items labelled from 1 to n, where to each item i is associated a size $a_i \in \mathbb{N}$ and a value $v_i \in \mathbb{N}$, and an integer $s \in \mathbb{N}$, find a subset $S \subseteq \{1, \ldots, n\}$ of the items such that all of them fit into a knapsack of size s and the total value is maximal, i.e., maximize $\sum_{i \in S} v_i$ under the constraint $\sum_{i \in S} a_i \leq s$.

step	$\binom{g_1}{g_2}$	μ_{21}	$\binom{g_1^*}{g_2^*}$	action
4	$\begin{pmatrix} 2211 & 0 \\ 804 & 1 \end{pmatrix}$	$\frac{4}{11}$	$\begin{pmatrix} 2211 & 0 \\ 0 & 1 \end{pmatrix}$	exchange rows 1 and 2
3	$\begin{pmatrix} 804 & 1 \\ 2211 & 0 \end{pmatrix}$	$\frac{1777644}{646417}$	$\begin{pmatrix} 804 & 1 \\ \frac{2211}{646417} & \frac{-1777644}{646417} \end{pmatrix}$	row 2 = row 2 − 3 · row 1
4	$\begin{pmatrix} 804 & 1 \\ -201 & -3 \end{pmatrix}$	$\frac{-161607}{646417}$	$\begin{pmatrix} 804 & 1 \\ \frac{2211}{646417} & \frac{-1777644}{646417} \end{pmatrix}$	exchange rows 1 and 2
3	$\begin{pmatrix} -201 & -3 \\ 804 & 1 \end{pmatrix}$	$\frac{-53869}{13470}$	$\begin{pmatrix} -201 & -3 \\ \frac{737}{4490} & \frac{-49379}{4490} \end{pmatrix}$	row 2 = row 2 + 4 · row 1
4	$\begin{pmatrix} -201 & -3 \\ 0 & -11 \end{pmatrix}$	$\frac{11}{13470}$	$\begin{pmatrix} -201 & -3 \\ \frac{737}{4490} & \frac{-49379}{4490} \end{pmatrix}$	exchange rows 1 and 2
5	$\begin{pmatrix} 0 & -11 \\ -201 & 3 \end{pmatrix}$	$\frac{3}{11}$	$\begin{pmatrix} 0 & -11 \\ -201 & 0 \end{pmatrix}$	

Figure 19.2: Trace of Algorithm 19.9, used to compute the gcd of 33 and 12.

The decision version of this problem (where one has to decide whether $\sum_{i \in S} v_i \geq v$ is possible, for an additional input $v \in \mathbb{N}$) is $\mathcal{NP}$-complete, and remains $\mathcal{NP}$-complete if in addition $a_i = v_i$ for $1 \leq i \leq n$ (cf. Garey & Johnson 1979). The restricted version of the problem is known as the *subset sum problem* and can be rephrased as follows.

Given $a_1, \ldots, a_n, s \in \mathbb{N}$, are there $x_1, \ldots, x_n \in \{0, 1\}$ such that $\sum_{1 \leq i \leq n} x_i a_i = s$? (19.4)

After Diffie & Hellman (1976) invented public key cryptography, Merkle & Hellman (1978) proposed a public key cryptosystem based on the subset sum problem. After that, other schemes of that type were proposed by Graham & Shamir (see Shamir & Zippel (1980) for a definition), Lu & Lee (1979), Niederreiter (1986), Goodman & McAuley (1985), Ong *et al.* (1984). Most of them were broken in the 1980's using lattice reduction, among others by Shamir (1984), Adleman (1983) and Pollard (Pollard & Schnorr 1987). The only knapsack type cryptosystem withstanding all attacks up to now is the Chor & Rivest (1985) scheme.

In order to allow the legitimate user to decode a message, special subset sum problems with "superincreasing" $a_1 \ll a_2 \ll \cdots \ll a_n$ are used in these cryptosystems; a trivial example is $a_i = 2^{i-1}$, where the solution $(x_1, \ldots, x_n)$ is just the binary representation of s. The "easiness" is then hidden by multiplying with a random number modulo another random number. The cryptanalysts' work then showed that this hiding does not work. Of course, the breaking of these schemes does *not* mean that some large instances of an $\mathcal{NP}$-complete problem have been solved; the "superincreasing" subset sum problem is not (known to be) $\mathcal{NP}$-hard.

The connection between the subset sum problem and short vectors is given by the fact that a solution of (19.4) yields a short vector in the lattice $L \subseteq \mathbb{Z}^{n+1}$ generated by the rows $b_1, \ldots, b_{n+1} \in \mathbb{Z}^{n+1}$ of the matrix

$$\begin{pmatrix} 1 & 0 & \cdots & 0 & -a_1 \\ 0 & 1 & \cdots & 0 & -a_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -a_n \\ 0 & 0 & \cdots & 0 & s \end{pmatrix} \in \mathbb{Z}^{(n+1)\times(n+1)}.$$

To see this, let $(x_1, \ldots, x_n) \in \{0,1\}^n$ be a solution of (19.4). Then

$$v = \sum_{1\leq i\leq n} x_i b_i + b_{n+1} = (x_1, \ldots, x_n, 0) \in L$$

is a vector with $|v| \leq \sqrt{n}$, which is very small since the a_i are typically very large numbers. The appoach to break such a cryptosystem is to compute a reduced basis of L and to hope that the resulting short vector is (essentially) v.

Pseudo-random numbers. A *linear congruential pseudo-random number generator* is given by three parameters $a, b, m \in \mathbb{N}$, and for any seed $x_0 \in \{0, \ldots, m-1\}$ successively computes the elements $x_i \in \{0, \ldots, m-1\}$ defined by $x_{i+1} \equiv ax_i + b$ for $i \in \mathbb{N}$. Pseudo-random numbers are used in many applications; the `rand` function in most computer systems is based on some form of these generators. In cryptography, they are used for the generation of keys. Therefore it is necessary that the numbers x_i are not only (almost) uniformly distributed but that it is practically impossible or computationally hard to infer x_i from $x_0, \ldots, x_{i-1}$, and to break a pseudo-random number generator means exactly to do this with a reasonable amount of computation time. Linear congruential generators have been broken by Frieze *et al.* (1988) and Boyar (1989) using lattice reduction algorithms, and hence turned out to be useless for cryptographic purposes. Lagarias (1990), §8, gives a nice overview of this problem.

Simultaneous Diophantine approximation. The problem is, given $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$, to find $p_1, \ldots, p_n, q \in \mathbb{Z}$ such that $|\alpha_i - \frac{p_i}{q}|$ is "small" for all i and q is not too large. We can simply take q to be a power of 10 and p_i an appropriate initial segment of the decimal expansion of α_i. However, we look for much better solutions, with q much smaller than (the inverse of) the approximation error.

For $n = 1$, this is the classical Diophantine approximation problem, and it can be solved by means of continued fractions: Hasse's theorem states that the "best" approximations occur in the continued fraction expansion of α_1. They can be computed with the Extended Euclidean Algorithm (cf. section 9) and yield $|\alpha_1 - p_1/q| < 5^{-1/2}q^{-2}$.

For general n, the problem can be expressed as a short vector problem. Given a rational approximation $\alpha_i \approx \frac{u_i}{v_i}$ with $u_i, v_i \in \mathbb{Z}$ for $1 \leq i \leq n$, and some "small" number $\lambda \in \mathbb{Z}$, let $L \subseteq \mathbb{Z}^{n+1}$ be the lattice generated by the rows $b_0, \ldots, b_n \in \mathbb{Z}^{n+1}$ of the matrix

$$\begin{pmatrix} \lambda & u_1 & u_2 & \cdots & u_n \\ 0 & -v_1 & 0 & \cdots & 0 \\ 0 & 0 & -v_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & -v_n \end{pmatrix} \in \mathbb{Z}^{(n+1)\times(n+1)},$$

and $a = (a_0, \ldots, a_n) = \sum_{0 \leq i \leq n} p_i b_i \in L$, with $p_i \in \mathbb{Z}$ for $0 \leq i \leq n$. Then

$$|a|^2 = \sum_{0 \leq i \leq n} a_i^2 = p_0^2 \lambda^2 + \sum_{1 \leq i \leq n} (u_i p_0 - v_i p_i)^2.$$

If λ is "small enough", then $|a|$ is small if and only if $|p_0|$ and $|u_i p_0 - v_i p_i|$ are small for $1 \leq i \leq n$, or equivalently, $|\frac{u_i}{v_i} - \frac{p_i}{p_0}|$ is small for all i, and this is a solution to the simultaneous Diophantine approximation problem.

This idea can be made into a polynomial-time algorithm which does the following: on input of individual approximations $(u_1/v_1, \ldots, u_n/v_n)$ and N, if there exists a simultaneous Diophantine approximation with denominators at most N and error at most ϵ, it computes an approximation with denominators at most $2^{(n-1)/2}N$ and error at most $\sqrt{5n}2^{(n-1)/2}\epsilon$ (Lagarias 1985).

Disproof of Mertens' conjecture. Recall the Möbius function from number theory:

$$\mu(n) = \begin{cases} (-1)^k & \text{if } n \text{ is squarefree with } k \text{ prime factors,} \\ 0 & \text{if } n \text{ is not squarefree} \end{cases}$$

for $n \in \mathbb{N}$. Here is a table of the values of μ for small n.

n	1	2	3	4	5	6	7	8	9	10
$\mu(n)$	1	-1	-1	0	-1	1	-1	0	0	1

In analytic number theory, one considers the function $M: \mathbb{N} \longrightarrow \mathbb{N}$ with $M(x) = \sum_{n \leq x} \mu(n)$. Mertens' conjecture says that $|M(x)| \leq \sqrt{x}$ for all $x \in \mathbb{N}$. It is known that this would imply the famous Riemann conjecture. However, Odlyzko & te Riele (1985) showed that Mertens' conjecture is false, using basis reduction in a lattice in $\mathbb{R}^{70}$.

Universität-GH Paderborn
FB 17, AG Algorithmische Mathematik
Skript Computeralgebra II, Teil 2
Sommersemester 1996

20. Factoring in $\mathbb{Q}[x]$: the algorithm

In this section, we will present and analyze in detail the algorithm for factoring (primitive) univariate polynomials with integer coefficients sketched in section 17. We will also indicate how the algorithm can be modified so as to also work for bivariate polynomials over a (finite) field. We have collected all the necessary ingredients in the preceding two sections: Hensel lifting and short vectors in lattices. First we will point out why Hensel lifting alone does not yet lead to a polynomial time algorithm, and how short vectors and factors of polynomials are related.

Suppose that $f \in \mathbb{Z}[x]$ is a squarefree primitive polynomial with irreducible factorization $f = f_1 \cdots f_r$ in $\mathbb{Z}[x]$. Let $p \in \mathbb{N}$ be a prime and $\varphi: \mathbb{Z}[x] \longrightarrow \mathbb{Z}_p[x]$ the canonical homomorphism which reduces the coefficient of a polynomial modulo p, and assume that $\varphi(f)$ is squarefree. Then

$$\varphi(f) = \varphi(f_1) \cdots \varphi(f_r) = (g_{1,1} \cdots g_{1,s_1}) \cdots (g_{r,1} \cdots g_{r,s_r})$$

in $\mathbb{Z}_p[x]$, where $g_{i,1} \cdots g_{i,s_i}$ is the irreducible factorization of $\varphi(f_i)$ in $\mathbb{Z}_p[x]$. Suppose we have factored f modulo p, i.e., we have computed the $g_{i,j}$ for $1 \leq i \leq r$ and $1 \leq j \leq s_i$, and we somehow know that $g_{1,1}, \ldots, g_{1,s_1}$ are exactly the irreducible factors of $\varphi(f)$ that belong to $\varphi(f_1)$ (which we do not know). Then Hensel lifting will in fact compute f_1.

Unfortunately, there seems to be no easy way to achieve such knowledge, and one has to try and lift all possible combinations of irreducible factors of f modulo p. This is the algorithm of Zassenhaus (1969), with exponential running time in the worst case. Collins (1979) showed that Zassenhaus' algorithm is polynomial time "on the average".

Examples for "bad" polynomials are given by the *Swinnerton-Dyer polynomials.* These are polynomials of the form

$$f = \prod (x \pm \sqrt{2} \pm \sqrt{3} \pm \sqrt{5} \pm \cdots \pm \sqrt{p_k}),$$

where p_k is the kth prime and the product runs over all 2^k possible combinations of $+$ and $-$ signs. It follows from general Galois theory that f is an irreducible polynomial in $\mathbb{Z}[x]$. But since for any prime p, $\mathbb{F}_{p^2}$ contains all the square roots $\sqrt{2} \bmod p, \ldots, \sqrt{p_k} \bmod p$, the reduction of f modulo p splits into linear factors over $\mathbb{F}_{p^2}$. Hence the irreducible factors of $f \bmod p$ in $\mathbb{F}_p[x]$ are either linear or quadratic, and there are at least $2^{k-1} = \frac{n}{2}$ of them, where $n = 2^k$ is the degree of f.

The example of the Swinnerton-Dyer polynomials shows that factoring f modulo many different primes and recovering the factorization over $\mathbb{Z}$ by means of the Chinese Remainder Algorithm instead of using only one prime and Hensel lifting would be of no help.

This is the reason for considering short vectors, which will actually yield polynomial time algorithms, and we will now turn to the connection between short vectors and factors.

For a fixed $n \in \mathbb{N}$, we identify polynomials in $f \in \mathbb{Z}[x]$ of degree less than n with their coefficient vectors in $\mathbb{Z}^n$, and let $|f|$ denote the Euclidean norm of the corresponding coefficient vector, as in the previous section.

REMARK 20.1. *Let F be a field and $f = \sum_{0\leq i\leq n} f_i x^i$, $g = \sum_{0\leq i\leq m} g_i x^i$ in $\mathbb{F}[x]$ with $\gcd(f,g) = 1$. Then there are polynomials $s = \sum_{0\leq i<m} s_i x^i$ and $t = \sum_{0\leq i<n} t_i x^i$ in $F[x]$ with $sf + tg = 1$. In section 11, we saw that the coefficients of s and t are the (unique) solution to the system*

$$\mathrm{Syl}(f,g)\cdot(s_{m-1},\ldots,s_0,t_{n-1},\ldots,t_0)^T = (0,\ldots,0,1)^T, \tag{20.1}$$

where $\mathrm{Syl}(f,g) \in F^{(n+m)\times(n+m)}$ is the Sylvester matrix of f and g. By Cramer's rule, the s_i and t_i are of the form

$$s_i = \frac{\det S_i}{\mathrm{res}(f,g)} \text{ for } 0 \leq i < m, \quad t_i = \frac{\det T_i}{\mathrm{res}(f,g)} \text{ for } 0 \leq i < n,$$

where S_i is $\mathrm{Syl}(f,g)$ with the ith column replaced by $(0,\ldots,0,1)^T$, and similarly T_i is $\mathrm{Syl}(f,g)$ with the $(m+i)$th column replaced by that vector.

For $f,g \in \mathbb{Z}[x]$, we have that $\det S_i, \det T_i$ and $\mathrm{res}(f,g)$ are in $\mathbb{Z}$. With $\overline{s} = \mathrm{res}(f,g)\cdot s$ and $\overline{t} = \mathrm{res}(f,g)\cdot t$, we therefore obtain

$$\overline{s}f + \overline{t}g = \mathrm{res}(f,g) \text{ in } \mathbb{Z}[x]$$

Note that in general, 1 is not expressible as a linear combination over $\mathbb{Z}$ of $f,g \in \mathbb{Z}[x]$ with $\gcd(f,g) = 1$ in $\mathbb{Z}[x]$ (consider, e.g., $f = x$ and $g = 2$).

LEMMA 20.2. *Let $f,g,h \in \mathbb{Z}[x]$ have positive degrees n,m,k, respectively, and suppose that h is monic and divides both f and g modulo p^N for a prime $p \in \mathbb{N}$ and $N \in \mathbb{N}$. Suppose that $|f|^m|g|^n < p^N$. Then f and g have a nonconstant gcd in $\mathbb{Z}[x]$.*

PROOF. Suppose that $\gcd(f,g) = 1$ in $\mathbb{Q}[x]$. Then there exist $\overline{s},\overline{t} \in \mathbb{Z}[x]$ such that $\overline{s}f + \overline{t}g \equiv \mathrm{res}(f,g) \bmod p^N$, by the above remark. Since h divides both f and g modulo p^N, it divides $\mathrm{res}(f,g)$ modulo p^N. But h is monic and nonconstant, and thus $\mathrm{res}(f,g) \equiv 0 \bmod p^N$.

On the other hand, $|\mathrm{res}(f,g)| \leq |f|^m|g|^n < p^N$, by Hadamard's inequality (12.4), so that $\mathrm{res}(f,g) = 0$. This is a contradiction to our assumption that $\gcd(f,g) = 1$ in $\mathbb{Q}[x]$, and hence $\gcd(f,g) \in \mathbb{Q}[x]$ is nonconstant. By Gauß' Lemma, $\gcd(f,g) \in \mathbb{Z}[x]$ is also nonconstant. □

The idea of the factoring algorithm is as follows. Given a squarefree primitive polynomial $f \in \mathbb{Z}[x]$ of degree n, a prime $p \in \mathbb{N}$ and $N \in \mathbb{N}$, we use Hensel lifting to compute a monic polynomial $H \in \mathbb{Z}[x]$ of degree $k < n$ that divides f modulo p^N. Then we find a "short" polynomial $u \in \mathbb{Z}[x]$, i.e., with $|u|^n \le p^N |f|^{-\deg u}$, that is also divisible by H modulo p^N, and the above lemma gives us a nontrivial factor of f in $\mathbb{Z}[x]$.

To find such an u, we consider the lattice $L \subseteq \mathbb{Z}^n$ generated by (the coefficient vectors of)

$$\{Hx^i : 0 \le i < n-k\} \cup \{p^N x^i : 0 \le i < k\}.$$

An element of L has the form

$$v = qH + rp^N \text{ with } q, r \in \mathbb{Z}[x], \quad \deg q < n-k, \quad \deg r < k, \tag{20.2}$$

and degree less than n. In particular, H divides v modulo p^N. If on the other hand $v \in \mathbb{Z}[x]$ is of degree less than n and divisible by H modulo p^N, we have $v = q'H + r'p^N$ for some $q', r' \in \mathbb{Z}[x]$. Division with remainder by the monic polynomial H yields $q'', r'' \in \mathbb{Z}[x]$ with $q' = q''H + r''$ and $\deg r'' < \deg H$. Letting $q = q' + p^N q''$ and $r = r''$, we see that v has the form (20.2), and conclude that

$$v \in L \iff H \text{ divides } v \text{ modulo } p^N. \tag{20.3}$$

Thus we can use lattice reduction to find a "short" vector $u \in L$ with the desired properties.

It remains to determine N appropriately so that we are guaranteed to find a short vector $u \in L$ if f is reducible. For this purpose, we need a bound for the Euclidean norm of factors of f in terms of $|f|$, i.e., a bound $B \in \mathbb{R}$ such that any factor $g \in \mathbb{Z}[x]$ of f satisfies $|g| \le B$. One might expect that we can take $B = |f|$, but this is not the case. For example, let $f = x^n - 1$ and $g = \Phi_n \in \mathbb{Z}[x]$, the nth cyclotomic polynomial. It is known that the coefficients of Φ_n are unbounded in absolute value if $n \to \infty$ (see, e.g., Knuth (1981)), and hence this is also true for $|g|$.

LEMMA 20.3. *For $f \in \mathbb{C}[x]$ and $z \in \mathbb{C}$, we have*

$$|(x-z)f| = |(\overline{z}x - 1)f|,$$

where $\overline{z}$ denotes the complex conjugate of z.

PROOF. Writing $f = \sum_{0 \le i \le n} f_i x^i$, we calculate

$$\begin{aligned}
|(x-z)f|^2 &= \sum_{0\le i\le n+1} |f_{i-1} - zf_i|^2 = \sum_{0\le i\le n+1} (f_{i-1} - zf_i)(\overline{f_{i-1}} - \overline{z}\overline{f_i}) \\
&= |f|^2(1+|z|^2) - \sum_{0\le i\le n+1} (z\overline{f_{i-1}}f_i + \overline{z}f_{i-1}\overline{f_i}) \\
&= \sum_{0\le i\le n+1} (\overline{z}f_{i-1} - f_i)(z\overline{f_{i-1}} - \overline{f_i}) = \sum_{0\le i\le n+1} |\overline{z}f_{i-1} - f_i|^2 \\
&= |(\overline{z}x - 1)f|^2,
\end{aligned}$$

where we set $f_{-1} = f_{n+1} = 0$. $\square$

Let

$$f = \sum_{0 \le i \le n} f_i x^i = f_n \prod_{1 \le i \le n} (x - z_i),$$

with $f_0, \ldots, f_n, z_1, \ldots, z_n \in \mathbb{C}$. We define $M(f) = |f_n| \cdot \prod_{1 \le i \le n} \max\{1, |z_i|\}$. Note that $M(g) \le M(f)$ if $f, g \in \mathbb{C}[x]$ are monic and $g \mid f$. The following theorem relates $M(f)$ and $|f|$.

THEOREM 20.4. *(Landau's inequality) For any $f \in \mathbb{C}[x]$, we have $M(f) \le |f|$.*

PROOF. We arrange the roots so that $|z_1|, \ldots, |z_k| > 1$ and $|z_{k+1}|, \ldots, |z_n| \le 1$ for some $k \in \{0, \ldots, n\}$. We have $M(f) = |f_n \cdot z_1 \cdots z_k|$. Let

$$g = f_n \prod_{1 \le i \le k} (\overline{z_i} x - 1) \prod_{k < i \le n} (x - z_i) = g_n x^n + \cdots + g_0 \in \mathbb{C}[x].$$

Then

$$\begin{aligned} M(f)^2 &= |f_n \overline{z_1} \cdots \overline{z_k}|^2 = |g_n|^2 \le |g|^2 = \left| \frac{g}{\overline{z_1} x - 1} (x - z_1) \right| = \cdots \\ &= \left| \frac{g}{(\overline{z_1} x - 1) \cdots (\overline{z_k} x - 1)} (x - z_1) \cdots (x - z_k) \right| = |f|^2, \end{aligned}$$

using repeatedly the previous lemma, and the assertion follows. $\square$

THEOREM 20.5. *(Mignotte 1989, §IV.3) If $g = \sum_{0 \le i \le m} g_i x^i \in \mathbb{C}[x]$ of degree m divides $f = \sum_{0 \le i \le n} f_i x^i \in \mathbb{C}[x]$ of degree $n \ge m$, then*

$$|g| \le \left| \frac{g_m}{f_n} \right| 2^m |f|.$$

PROOF. We write $g = g_m \prod_{1 \le i \le m} (x - u_i)$ with $u_i \in \mathbb{C}$ for $1 \le i \le m$, and note that each u_i equals some root z_j of f. By Vietà's rule, we can express the coefficients of g in terms of its roots as

$$g_i = (-1)^{m-i} g_m \sum_{\substack{S \subseteq \{1, \ldots, m\} \\ \#S = m-i}} \prod_{j \in S} u_j$$

(the sum is the i-th elementary symmetric polynomial in $u_1, \ldots, u_m$), and hence

$$|g_i| \le |g_m| \cdot \sum_S \prod_{j \in S} |u_j| \le \binom{m}{i} M(g)$$

for $0 \le i \le m$. Thus

$$|g| \le \sum_{0 \le i \le m} |g_i| \le 2^m M(g) \le \left| \frac{g_m}{f_n} \right| 2^m M(f) \le \left| \frac{g_m}{f_n} \right| 2^m |f|,$$

by Landau's inequality. $\square$

Now we are ready to state the factoring algorithm for $\mathbb{Z}[x]$.

ALGORITHM 20.6. *Factorization of a squarefree primitive polynomial in* $\mathbb{Z}[x]$.
Input: $f \in \mathbb{Z}[x]$ *squarefree primitive of degree* n.
Output: *If* f *is reducible, then a proper primitive divisor* $e \in \mathbb{Z}[x]$ *of* f, *otherwise "*f *is irreducible".*

1. *Set* $a \in \mathbb{Z}$ *to the leading coefficient of* f *and* $B = 2^{2n^2}|f|^{2n} \in \mathbb{N}$.
2. *Find a prime* $p \in \mathbb{N}$ *not dividing* $a \cdot \operatorname{res}(f, f')$, *and set* $N = \lceil \log_p B \rceil \in \mathbb{N}$.
3. (Factorization modulo p) *Compute a factorization* $f \equiv gh \bmod p$ *in* $\mathbb{F}_p[x]$ *with* $g, h \in \mathbb{Z}[x]$, h *monic and irreducible modulo* p. *If* $f \bmod p$ *is irreducible, return "*f *is irreducible" and stop.*
4. (Hensel lifting) *Compute a factorization* $f \equiv GH \bmod p^N$ *with* $G, H \in \mathbb{Z}[x]$, H *monic,* $G \equiv g \bmod p$, $H \equiv h \bmod p$, $\deg H = \deg h$, *and* $\deg G = \deg g$.
5. (Short vector computation) *Set* $k = \deg H$ *and compute a "short" vector* u *in the lattice* $L \subseteq \mathbb{Z}^n$ *generated by the coefficients of*
$$\{Hx^i : 0 \leq i < n - k\} \cup \{p^N x^i : 0 \leq i < k\}.$$
Denote the corresponding polynomial also by u.
6. (Factor extraction) *Compute the monic polynomial* $e_1 = \gcd(f, u) \in \mathbb{Q}[x]$. *If* $e_1 \neq 1$, *then return* $e = \operatorname{pp}(ae_1) \in \mathbb{Z}[x]$, *otherwise return "*$f$ *is irreducible".*

THEOREM 20.7. *The algorithm works correctly and uses* $O^{\sim}(n^{12} + n^{10} \log^2 |f|)$ *bit operations.*

PROOF. It is clear that the algorithm never outputs a proper factor of f in step 6 if f is irreducible and that it halts with the correct output in step 3 if $f \bmod p$ is irreducible. Hence we may assume that both f and $f \bmod p$ are reducible.

Let $f = f_1 f_2$ in $R[x]$ such that f_1 is the (unique) irreducible factor of f with $(h \bmod p) \mid (f_1 \bmod p)$. Write $f_1 \equiv h \cdot h_1 \bmod p$. Then
$$gh \equiv f = f_1 f_2 \equiv h h_1 f_2 \bmod p,$$
and hence $g \equiv h_1 f_2 \bmod p$. Since $f \bmod p$ is squarefree, $f_1 \bmod p$ is also squarefree, and Hensel's Lemma gives $h^*, h_1^* \in \mathbb{Z}[x]$ such that $f_1 \equiv h^* h_1^* \bmod p^N$, $h^* \equiv h \bmod p$, and $h_1^* \equiv h_1 \bmod p$, so that
$$f = f_1 f_2 \equiv h^*(h_1^* f_2) \bmod p^N, \; h^* \equiv h \bmod p, \; h_1^* f_2 \equiv h_1 f_2 \bmod p.$$
On the other hand, we also have
$$f \equiv GH \bmod p^N, \; H \equiv h \bmod p, \; G \equiv g \equiv h_1 f_2 \bmod p.$$

The uniqueness property of Hensel lifting (Theorem 18.11) implies that $H \equiv h^*$ mod p^N, so that $Hh_1^* \equiv h^*h_1^* \equiv f_1 \bmod p^N$.

Now consider the lattice L in step 5. It consists exactly of the polynomials in $\mathbb{Z}[x]$ of degree less than n that H divides modulo p^N. In particular, $f_1 \in L$ and

$$|u| \leq 2^{(n-1)/2} \min\{|v|: 0 \leq v \in L\} \leq 2^{(n-1)/2}|f_1| \leq 2^{(n-1)/2} 2^{\deg f_1}|f| \leq 2^{2n}|f|,$$

by Mignotte's Theorem 20.5. Together with $\deg u < n$, this implies that

$$|f|^{\deg u}|u|^n \leq |f|^n|u|^n \leq 2^{2n^2}|f|^{2n} = B \leq p^N.$$

Then, by Lemma 20.2, the gcd e_1 of f and z in $\mathbb{Q}[x]$ is a nontrivial factor of f in $\mathbb{Q}[x]$, and by Gauß' Lemma, $\mathrm{pp}(ae_1)$ is a proper primitive factor of f in $\mathbb{Z}[x]$.

The expected cost for step 3 of the algorithm is $O\tilde{~}(n^2 + n \log p)$ arithmetic operations in $\mathbb{F}_p$, or $O\tilde{~}(n^2 \log p + n \log p)$ bit operations, by Theorem 16.18. Step 4 can be done with $O\tilde{~}(N^2 n \log p)$ bit operations, by Theorem 18.10, and the cost for the lattice reduction in step 5 is $O\tilde{~}(n^8(N \log p)^2)$, by Corollary 19.13. This certainly dominates the cost for the other steps, and the claim follows, noting that $N \log p \in O(n^2 + n \log |f|)$. $\square$

Remarks. Algorithm 20.6 only computes a factorization of f into two parts. To get the complete factorization into irreducible polynomials, we of course run the algorithm recursively on the two factors. To factor arbitrary polynomials $f \in \mathbb{Z}[x]$, we first compute the squarefree part $f/\gcd(f, f') \in \mathbb{Z}[x]$, where f' is the formal derivative of f, and then apply Algorithm 20.6 to its primitive part.

In their original paper, Lenstra *et al.* (1982) presented an algorithm that computes an *irreducible* factor of a primitive squarefree polynomial in $\mathbb{Z}[x]$. To this end, they use basis reduction in lattices in $\mathbb{Z}^n$ of dimension less than n.

With some modifications, Algorithm 20.6 can be adapted to the factorization of polynomials in $F[y][x]$, where F is a field and we have an algorithm for factoring univariate polynomials over F. This is the case, e.g., when F is a finite field (see section 16), or when $F = \mathbb{Q}$, by the above. Algorithm 20.6 is then merely a (deterministic) polynomial time *reduction* from bivariate to univariate polynomial factorization. Here is a list with the main differences:

- Hensel lifting over $F[y]$ is completely analogous to Hensel lifting over $\mathbb{Z}$; all theorems (except Theorem 18.10) and algorithms in section 18 are still valid when $\mathbb{Z}$ is replaced by $F[y]$ throughout.
- Lattice reduction is much easier over $F[y]$ than over $\mathbb{Z}$. The appropriate norm is the *maximum norm* $|f|_\infty = \max_{1 \leq i \leq n} \deg f_i$ for a vector $f = (f_1, \ldots, f_n) \in F[y]^n$. In this case, one can always find a *shortest* vector in an n-dimensional lattice in $F[y]^n$ in polynomial time.
- An upper bound on the norm of factors of a polynomial $f \in F[y][x]$ is trivially given by $|f|_\infty$.

For a unified discussion of the factoring algorithm over $\mathbb{Z}$ and $F[y]$, we refer to von zur Gathen (1984a).

21. Factoring integers: trial division and Pollard's ρ-method

The problem: Given a natural number N, express N as a product of prime numbers.

In this section, we will discuss two methods:

1. trial division,
2. Pollard's ρ-method,

In the next section, we will present another method:

3. Dixon's quadratic sieve,

Figure 21.1: Some algorithms to factor an n-bit integer.

method	time
trial division	$2^{n/2}$
Pollard's ρ	$2^{n/4}$
Pollard & Strassen	$2^{n/4}$
quadratic sieve	$\exp(O(\sqrt{n \log n}))$

To get an idea of the algorithmic progress, Figure 21.1 gives the running times of the algorithms for factoring an n-bit integer N. An entry t in the time column means that the algorithm can be performed with $O\tilde{}(t)$ bit operations, where the "soft-Oh" notation is used to ignore logarithmic factors:

$$s = O\tilde{}(t) \iff s = O(t \cdot (\log t)^k) \text{ for some fixed } k.$$

The running times of some of the algorithms actually depend not just on n, but on the size of the second largest prime factor of N. This is discussed in more detail later. Some of the timing analyses are only heuristic, not rigorously proven.

21.1. Trial Division.

Notation 21.1. *$p^\ell \parallel N$ means $p^\ell \mid N$ and $p^{\ell+1} \nmid N$.*

Algorithm 21.2. *Trial division.*
Input: N.
Output: *All prime power factors of N.*

```
p ⟵ 2
while N is not a prime power do
    if p | N
        let p^ℓ || N; output p^ℓ
        N ⟵ N/p^ℓ
    p ⟵ p + 1
output N
```

This algorithm will terminate when p is the second largest prime factor of N. We assume that a "fast" test for prime powers is used in the while-loop.

EXERCISE 21.3. *Design a probabilistic test whether N is a prime power, using the primality test from section 14. Your test should use $O^\sim(\log^3 N)$ bit operations. There are (at least) two different solutions: a numerical one and a modular one.*

DEFINITION 21.4. *For $N \in \mathbb{N}$, let $S_1(N)$ denote the largest prime divisor of N, and $S_2(N)$ the second largest prime divisor of N.*

The number of steps required by trial division is $S_2(N)(\log N)^{O(1)}$.
Knuth & Trabb Pardo (1976) showed that

$$\Pr[S_1(N) > N^{0.85}] \approx 0.20,$$

$$\Pr[S_2(N) > N^{0.30}] \approx 0.20.$$

Thus, the number of steps needed for the trial division algorithm is $O^\sim(N^{0.30})$ "most of the time".

All algorithms to be discussed later will first remove all "small" factors p from N, say with $p \leq 10^6$.

21.2. Pollard's ρ-method. The main features of this method by Pollard (1975) are:

- Probabilistic.
- The expected running time is $S_2(N)^{1/2}(\log N)^{O(1)}$. Thus worst case $O^\sim(N^{1/4})$, since $S_2(N) < \sqrt{N}$. The time bounds are "heuristic" and have not been rigorously established.
- Space $O(\log N)$.

We start with an auxiliary question, the **birthday problem:**
Assuming peoples' birthdays occur randomly, how many people do we need to get together before we can expect at least two people to have the same birthday. Surprising answer: only 23 are sufficient. This means that with 23 or more people at a party, the probability of two coinciding birthdays is at least 50.7%. (EXERCISE: 23 is the correct answer when we ignore leap years, so that each year has exactly

365 days, all equally likely as birthdays (ignoring effects of Fasnacht, springtime, blackouts, etc.). Calculate the correct answer taking leap years into account; recall that year n is a leap year if and only if ($4 \mid n$ and $100 \nmid n$) or ($1000 \mid n$). Here you may make the (unrealistic) assumption that the birthdays of all people are evenly distributed among all days of the preceding 1000 years.)

THEOREM 21.5. *Consider random choices, with replacement, among k labelled balls. The expected number of choices until a collision occurs is $O(\sqrt{k})$.*

PROOF. Let s be the number of choices until collision occurs. This is a random variable. For $j \geq 2$, we have

$$\begin{aligned}
\Pr[s \geq j] &= \frac{1}{k^{j-1}} \prod_{1\leq i<j} (k-(i-1)) = \prod_{1\leq i<j} \left(1 - \frac{i-1}{k}\right) \\
&\leq \prod_{1\leq i<j} e^{-(i-1)/k} = e^{-(j-1)(j-2)/2k} \leq e^{-(j-2)^2/2k},
\end{aligned}$$

where we have used that $1 - x \leq e^{-x}$. It follows that

$$\begin{aligned}
E(s) &= \sum_{j\geq 1} \Pr[s \geq j] \leq 1 + \sum_{j=0}^{\infty} e^{-j^2/2k} \\
&\leq 1 + 1 + \int_0^{\infty} e^{-x^2/2k} dx \leq 2 + \sqrt{2k} \underbrace{\int_0^{\infty} e^{-x^2} dx}_{\text{converges to } \sqrt{\pi}/2} . \quad \Box
\end{aligned}$$

Floyd's 1-step/2-step cycle detection trick:
Given a function $f\colon \{1, \ldots, k\} \to \{1, \ldots, k\}$ and $x_0 \in \{1, \ldots, k\}$, examine the sequence $x_0, x_1, \ldots$ defined by

$$x_{i+1} = f(x_i)$$

for $i \geq 0$. This is an infinite sequence from a finite set, so that at some point the values repeat. This results in a cycle of length t such that $x_\ell = x_{\ell+t}$ for some ℓ.
Problem: Find $i \neq j$ such that $x_i = x_j$.

An obvious method is to write down the sequence until a value repeats itself, but this requires space $O(\ell + t)$.

1-step/2-step trick:

```
x ⟵ x_0
y ⟵ x_0
i ⟵ 0
repeat until x = y
    x ⟵ f(x)
    y ⟵ f(f(y))
    i ⟵ i + 1
```

This trick will find i such that $x_i = x_{2i}$ in at most $\ell + t$ steps and with space $O(1)$.

Back to factoring N: Pollard's ρ-method for factoring N generates a sequence $x_0, x_1, \ldots \in \{0 \ldots N-1\}$. We pick x_0 at random and define $x_{i+1} = x_i^2 + 1 \bmod N$. The choice of this generating function is black magic, but functions not of the form $x^2 + c$ don't seem to work as well.

Let p be the smallest prime dividing N. Then for $i \geq 0$, we have

$$x_{i+1} \equiv x_i^2 + 1 \bmod p.$$

The birthday problem and heuristic reasoning which says that the x_i's "look" random imply that we will have a collision mod p. This duplicate can be detected by using Floyd's cycle detection trick.

We expect to find a small i, namely $i = \mathrm{O}(\sqrt{p})$, such that $x_i \equiv x_{2i} \bmod p$. We hope that $x_i \not\equiv x_{2i} \bmod N$ and then we can split N. If N is not a prime power, then this latter condition will be likely: if q is a different prime divisor of N, and the x_i's are random residues mod N, then $x_i \bmod p$ and $x_i \bmod q$ are independent random variables, by the Chinese Remainder Theorem. If N is a prime power, there are much better ways to factor N.

Algorithm 21.6. *Pollard's ρ-method.*
Input: $N \in \mathbb{N}$, *not a prime power.*
Output: *Either a divisor of N, or failure.*

```
Pick x ∈ {0, ..., N − 1} at random.
y ⟵ x
repeat
    x ⟵ x² + 1 mod N,
    y ⟵ (y² + 1)² + 1 mod N,
    d ⟵ gcd(x − y, N),
until d ≠ 1
if d ≠ N then
    we succeeded in finding a non-trivial divisor.
else
    failure ... we need to start again.
```

Theorem 21.7. *Let $N \in \mathbb{N}$ be composite and p its smallest prime factor. Under the assumption that the function $f(x) = x^2+1$ behaves like a random function $\mathbb{Z}/p\mathbb{Z} \longrightarrow \mathbb{Z}/p\mathbb{Z}$, the expected running time of Pollard's algorithm for finding one nontrivial factor of N is $O(\sqrt{p})(\log N)^{O(1)}$. By iterating the algorithm, N can be completely factored in time $S_2(N)^{1/2}(\log N)^{O(1)}$, or $O^{\sim}(N^{1/4})$. The space requirement of the algorithm is $O(\log N)$.*

Note that the input size is $\log N$.

Example 21.8. *We want to factor $N = 82123$. Starting with $x = x_0 = 631$, we find the following sequence:*

i	x_i mod N	x_i mod 41
0	*631*	*16*
1	*69670*	*11*
2	*28986*	*40*
3	*69907*	*2*
4	*13166*	*5*
5	*64027*	*26*
6	*40816*	*21*
7	*80802*	*32*
8	*20459*	*0*
9	*71874*	*1*
10	*6685*	*2*

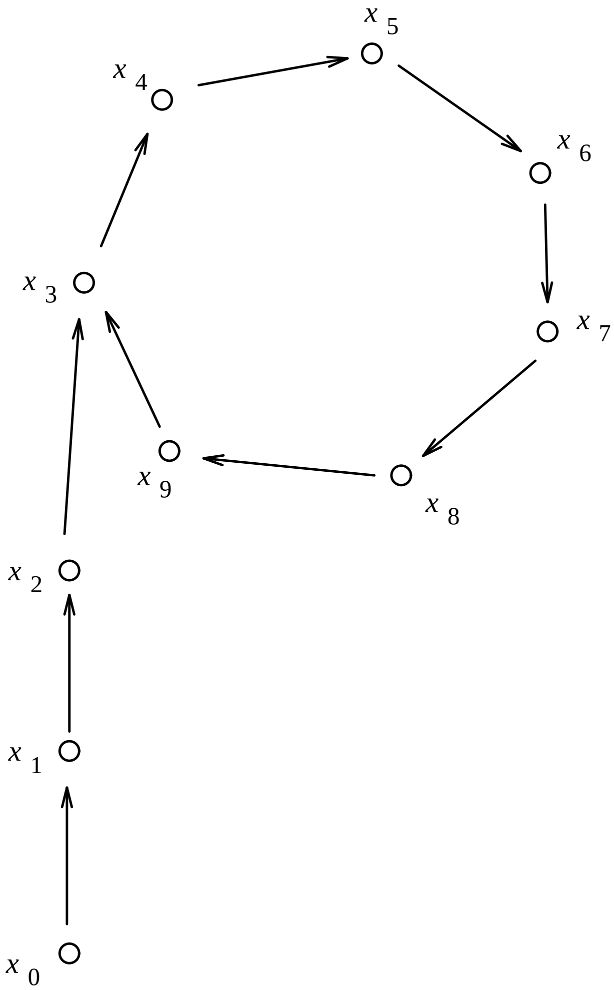


Figure 21.2: Pollard's rho method: $p = 41$, $x_0 = 16$.

This behaviour is illustrated in Figure 21.2. It leads to the factor

$$\gcd(x_3 - x_{10}, N) = 41.$$

The algorithm calculates in tandem x_i and $y_i = x_{2i}$ and does the gcd *test each time. Already after seven 2-steps it catches up with the 1-steps:*

i	$x_i \bmod N$	$x_i \bmod 41$	$y_i \bmod N$	$y_i \bmod 41$
0	631	16	631	16
1	69670	11	28986	40
2	28986	40	13166	5
3	69907	2	40816	21
4	13166	5	20459	0
5	64027	26	6685	2
6	40816	21	75835	26
7	80802	32	17539	32

The factorization $N = 41 \cdot 2003$ is found as $\gcd(x_7 - y_7, N) = 41$. Of course the sequence also repeats modulo 2003, and in fact $x_{38} \equiv 4430 \equiv x_{143} \bmod N$. If this repetition occurred simultaneous with the one modulo 41, we would not find a factor: an "unlucky" x_0.

22. Factoring integers: the quadratic sieve

In this section, we present Dixon's (1981) quadratic sieve method for factoring integers. It was the first method faster than $\exp(\epsilon \cdot n)$ for any $\epsilon > 0$. We do this in two parts. The first part, also given in class, describes the algorithm and analysis, the latter under the unproven Assumption 22.3 that random squares modulo N are smooth—i. e., have only small prime factors—with roughly the same probability as arbitrary random numbers modulo N. The second part, not given in class, presents a complete analysis without any assumption; this is rather technical.

The fast primality tests usually answer "composite" without actually producing a factor of the integer being tested, so we would like a method for producing factors of integers known to be composite. In the following description we make the simplifying assumption that N is not a prime power

We begin by observing that the equations

$$\begin{aligned} N &= s^2 - t^2 = (s+t)(s-t), \\ N &= a \cdot b = \left(\frac{a+b}{2}\right)^2 - \left(\frac{a-b}{2}\right)^2 \end{aligned}$$

describe a bijection between factorizations of N and representations of N as a difference of two squares. This immediately suggests a crude factorization algorithm: for $t = \lceil \sqrt{N} \rceil, \lceil \sqrt{N} \rceil + 1, \ldots$, check whether $t^2 - N$ is a perfect square. If we find such a square, then we can factor N. This algorithm will work well if $N = ab$ with $|a-b|$ small, i. e., the running time is dependent on $|a-b|$. A modification is to let $t = \lceil \sqrt{kN} \rceil, \lceil \sqrt{kN+1} \rceil \ldots$ and then check if $t^2 - kN$ is a perfect square. If we choose $k < N$ and $t^2 - kN = s^2$, then we see that $\gcd(s+t, N)$ is a factor of N, hopefully non-trivial, i. e., $s \not\equiv \pm t \bmod N$. Of course in practice we choose k much smaller than N.

Example 22.1. *Let $N = 2183$. Suppose that we have deduced the system of congruences*

$$\begin{aligned} 453^2 &\equiv 7 \bmod N, \\ 1014^2 &\equiv 3 \bmod N, \\ 209^2 &\equiv 21 \bmod N. \end{aligned}$$

Then we obtain $(453 \cdot 1014 \cdot 209)^2 \equiv 21^2 \bmod N$, *or* $687^2 \equiv 21^2 \bmod N$. *This yields the factors* $37 = \gcd(687 - 21, N)$ *and* $59 = \gcd(687 + 21, N)$*; in fact* $N = 37 \cdot 59$ *is the prime factorization of* N.

22.1. The quadratic sieve algorithm. The above example suggests a systematic approach: choose b^2 at random and hope that $b^2 \bmod N$ is a product of small primes. When enough have been found, we will get a congruence $s^2 \equiv t^2 \bmod N$. We now describe this approach in detail.

A **factor base** is a set $B = \{p_1, p_2, \ldots, p_h\}$ of distinct prime numbers. A number b is a B-**number** if $b^2 \bmod N$ is a product of numbers from B. Thus, in Example 22.1,

$453, 1014$, and 209 are B-numbers, where $B = \{2, 3, 5, 7\}$ and $N = 2183$. A variant of the algorithm allows $p_1 = -1$ and considers the **least absolute residue** a of $b \in \mathbb{Z}$, defined by $b \equiv a \bmod N$ and $-N/2 < a \leq N/2$.

For every B-number b, write $b^2 \equiv p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_h^{\alpha_h} \bmod N$, and associate with b the binary exponent vector

$$\epsilon = (\alpha_1 \bmod 2, \alpha_2 \bmod 2, \ldots, \alpha_h \bmod 2) \in \mathbb{F}_2^h.$$

Now suppose that we have $b_1, b_2, \ldots, b_l$ such that $\epsilon_1 + \epsilon_2 + \cdots + \epsilon_l = 0$ in $\mathbb{F}_2^h$. Writing $b_i^2 \equiv \prod_{1 \leq j \leq h} p_j^{\alpha_{ij}} \bmod N$, we see that

$$\Big(\prod_{1 \leq i \leq l} b_i\Big)^2 \equiv \prod_{1 \leq j \leq h} p_j^{\sum_{1 \leq i \leq l} \alpha_{ij}} = \prod_{1 \leq j \leq h} p_j^{2\gamma_j} = \Big(\prod_{1 \leq j \leq h} p_j^{\gamma_j}\Big)^2 \bmod N,$$

where $\gamma_j = \frac{1}{2} \sum_{1 \leq i \leq l} \alpha_{ij} \in \mathbb{N}$. This gives the desired congruence $s^2 \equiv t^2 \bmod N$, where $s = \prod_{1 \leq i \leq l} b_i$, and $t = \prod_{1 \leq j \leq h} p_j^{\gamma_j}$. Note that we have to generate no more than $h + 1$ B-numbers, i. e., we may always take $l \leq h + 1$, since any set of $h + 1$ vectors in $\mathbb{F}_2^h$ is linearly dependent over $\mathbb{F}_2$.

Having obtained $s^2 \equiv t^2 \bmod N$ we hope that $s \not\equiv \pm t \bmod N$. If N has $r \geq 2$ distinct prime factors (N is not a prime power!), then the Chinese Remainder Theorem shows that every square in $\mathbb{Z}_N$ has exactly 2^r square roots in $\mathbb{Z}_N$. Therefore, if s is a random square root of t^2 we have

$$\operatorname{prob}[s \equiv \pm t \bmod N] = \frac{2}{2^r} \leq \frac{1}{2}.$$

Here is the resulting algorithm.

ALGORITHM 22.2. *Dixon's quadratic sieve.*
Input: N, h.
Output: *A factor of N, hopefully.*

1. *Construct the factor base $B = \{p_1, p_2, \ldots, p_h\}$ consisting of the first h primes.*
2. *Generate random numbers $b_1, b_2, \ldots$, and collect those that are B-numbers.*
3. *When we have $h + 1$ such numbers we find some $b_1, b_2, \ldots, b_l$ with $\epsilon_1 + \epsilon_2 + \cdots + \epsilon_l = 0$ in $\mathbb{F}_2^h$. This can be done by solving an $(h + 1) \times h$ system of linear equations over $\mathbb{F}_2$.*
4. *Form s and t and compute $\gcd(s + t, N)$, hoping to obtain a non-trivial factor (with probability $\geq \frac{1}{2}$.)*

We now estimate the cost of this quadratic sieve, and determine the right choice for h. Let

$$\begin{aligned} \Psi(x, y) &= \{a \in \mathbb{N}\colon 1 \leq a \leq x, \forall p \text{ prime } p \mid a \Longrightarrow p \leq y\}, \\ \psi(x, y) &= \#\Psi(x, y). \end{aligned} \tag{22.1}$$

Numbers all of whose prime factors are not greater than y are called y-smooth. Suppose that we are factoring N using the base $B = \{p_1, p_2, \ldots, p_h\}$ consisting of the first h primes. Then

$$b \text{ is a } B\text{-number } \iff b^2 \bmod N \in \Psi(N, p_h).$$

Clearly the crux of the problem is to choose h wisely: if h is too small, then B-numbers take a long time to find, and if h is too large, then it takes a long time to test a prospective B-number and the linear system is large.

We begin with an estimate of the probability that a random integer $a \in (0, x)$ is y-smooth, with $y = p_h$. Put $u = \log(x)/\log(y)$ and $v = \lfloor u \rfloor$. Let $(a_1, a_2, \ldots, a_v) \in B^v$ and $a = a_1 a_2 \cdots a_v$. Then $a \le p_h^v \le y^u = x$, hence $a \in \Psi(x, y)$. Each a comes from at most $v!$ vectors in B^v, hence

$$\psi(x,y) \ge \frac{h^v}{v!} \ge \left(\frac{h}{v}\right)^v \gtrsim \left(\frac{y}{\log(y)}\right)^v \cdot v^{-v} \sim \left(\frac{y}{\log(y)}\right)^u \cdot u^{-u} = x(u\log(y))^{-u},$$

where we have used the Prime Number Theorem which says that the number $\pi(x)$ of primes up to x is about $x/\log x$, and that the nth prime number p_n is about $n \log n$. (All logs are natural.) More precisely,

$$\frac{x}{\log x} < \pi(x) < \frac{x}{\log x}\left(1 + \frac{3}{2\log x}\right) \tag{22.2}$$

for $x \ge 17$, and

$$n(\log n + \operatorname{loglog} n - \frac{3}{2}) < p_n < n(\log n + \operatorname{loglog} n)$$

for $n \ge 6$ (Rosser & Schoenfeld 1962). For $1 \le a \le x$ we have

$$\operatorname{prob}(a \text{ is } y-\text{smooth}) = \frac{\psi(x,y)}{x} \gtrsim (u\log(y))^{-u}.$$

In fact, the true order of this probability is u^{-u}; this function is called *Dickman's ρ-function.* So the expected number of trials necessary to find a single B-number is u^u. For our purposes, the estimate proven above is good enough.

We now analyze the cost of the algorithm. Let $s = \log p_h = \log y$ and $n = \log N = \log x$. We expect to need at most $(u \log p_h)^u$ b's.

ASSUMPTION 22.3. *For randomly chosen $b \in \{0, \ldots, N-1\}$, the probability that $b^2 \bmod N$ is y-smooth is about $\psi(N, y)/N$.*

Then the total expected number of trials is $(h+1)(us)^u \sim h(us)^u$.

The cost of one trial is $O(M(n))$ bit operations to compute $b^2 \bmod N$, plus $O(hns)$ for trial divisions by all elements of B to check smoothness. The linear system of equations can be solved with $O(h^3)$ operations.

Thus the total cost of the algorithm is

$$O(h(us)^u(n^2 + hns) + h^3). \tag{22.3}$$

We want to find the right value of h so that the two summands in (22.3) are roughly the same size. It is ok to be sloppy in this heuristic derivation, it would be nice to have a rigorous argument then based on this value; this is done in the second part. We can ignore the n^2 term in (22.3) since it is small compared to the other summand. Thus, we want

$$h(us)^u hns = h^3 \quad \text{or} \quad (us)^u ns = h.$$

We have $\log x = n$ and $u = n/s$, and hence

$$n^{n/s} ns = h \sim \frac{e^s}{s},$$

since $e^s = p_h$. Ignoring the smaller factors gives

$$e^{\frac{n \log n}{s}} = n^{n/s} \sim e^s \quad \text{or} \quad s \sim \sqrt{n \log n}.$$

If we replace h^3 by h^2 to account for fast sparse matrix arithmetic (see Wiedemann 1986), then minimizing (22.3) leads to minimizing $\frac{n}{s} e^{2s + \frac{n \log n}{s}}$, and again to $s = \mathrm{const}\sqrt{n \log n}$.

Set $s = \sqrt{n \log n}$. Then for some small $\varepsilon > 0$ we have $n < s < e^{\varepsilon s}$. So the total cost is

$$O\Big(\frac{e^s}{s} n^{n/s} (e^{2\varepsilon s} + \frac{e^s}{s} ns) + \frac{e^{3s}}{s^3}\Big)$$

bit operations. The term $e^{2\varepsilon s}$ can be ignored as it is small compared to the other terms. The time then is

$$O(e^{2s + \frac{n \log n}{s}} \frac{n}{s} + \frac{e^{3s}}{s^3}) \quad \text{or} \quad O(e^{3s + \varepsilon s}) \quad \text{or} \quad e^{(3+o(1))\sqrt{n \log n}}.$$

Define

$$L(N) = e^{\sqrt{\log N \log \log N}}. \tag{22.4}$$

Then the running time is $L(N)^{3+o(1)}$ under Assumption 22.3, and $L(N)^{2+o(1)}$ if fast matrix operations are used.

The space requirement is $h \log h$ to store the factor base, and roughly the same for the linear algebra, since the matrix has h sparse rows, and Wiedemann's algorithm can be executed in the same space.

22.2. Quadratic sieve without assumption. This section presents a version of Dixon's algorithm which is closer to Dixon's original paper and rigorously proven.

Throughout n is an odd, positive integer that is not a prime power; we shall attempt to factor n (called N in the previous subsection) given a list L of l integers in the range $[1, n]$. The set of all possible lists L is denoted $\mathcal{L}$; note that $|\mathcal{L}| = n^l$.

We make $\mathcal{L}$ into a *probability space* by setting $\text{prob}(L) = n^{-l}$ for each $L \in \mathcal{L}$. A real valued function X defined on the space $\mathcal{L}$ is called a *random variable*. By definition the mean and variance of X are

$$\mu(X) = \sum_{L \in \mathcal{L}} X(L)\text{prob}\{\, L \,\},$$
$$var(X) = \sum_{L \in \mathcal{L}} (X(L) - \mu(X))^2.$$

In this setting Chebyshev's inequality states that

$$\text{prob}\{\, L\colon |X(L) - \mu(X)| \geq t \,\} \leq var(X)/t^2,$$

where t is a positive real number. We shall use the Prime Number Theorem (22.2) and Stirling's inequality, which states that for sufficiently large n

$$\sqrt{2n\pi}(n/e)^n < n! < \sqrt{2n\pi}(n/e)^n[1 + 1/(12n - 1)].$$

[See *Handbook of Mathematical Tables and Formulas* by R. S. Burington, Handbook Publishers inc., Sandusky, Ohio.]

We shall also use the fact that each element of $(\mathbb{Z}^*_{q^l})^2$ has precisely two square roots in $\mathbb{Z}^*_{q^l}$ when q is an odd prime number. This generalizes Fact 14.5 (iii), which states this for $l = 1$. To see this it is enough to show that $1 \in \mathbb{Z}^*_{q^l}$ has precisely 2 square roots in $\mathbb{Z}^*_{q^l}$. Suppose that $u^2 \equiv 1 \bmod q^l$. Then $(u-1)(u+1) \equiv 0 \bmod q^l$, so $q^j | (u-1)$ and $q^k | (u+1)$ where $j + k = l$. Then $q^{\min(j,k)} | (u+1) - (u-1) = 2$, and since q is odd we must have $j = 0$ or $k = 0$, that is $u \equiv \pm 1 \bmod q^l$, as desired.

Let $m^2 \bmod n$ denote the unique integer $r \in \mathbb{N}$ satisfying $0 \leq r < n$ and $r \equiv m^2 \bmod n$. For $x, y \in \mathbb{R}$ with $x \geq 0$ let $\Psi(x, y) = \{m \in \mathbb{Z}\colon\ p|m \text{ and } p \text{ prime} \Rightarrow p \leq y\}$ be the set of y-smooth integers up to x, and write $\psi(x, y) = |\Psi(x, y)|$. Also, let $\Psi^{\square}(n, v) = \{k \in [1, n]\colon\ k^2 \bmod n \in \Psi(n, v)\}$. We define the *quadratic character* $\chi(k) = (\epsilon_1, \epsilon_2, \ldots, \epsilon_d)$ of k by $\epsilon_i = 1$ if k is a quadratic residue modulo $q_i^{l_i}$ and $e_i = -1$ otherwise, where n has the canonical factorization $q_1^{l_1} q_2^{l_2} \cdots q_d^{l_d}$ with $d \geq 2$. We partition $\Psi(n, v)$ into 2^d disjoint sets $\Psi_i(n, v)$ according to the values that the quadratic character takes. Finally, let Q stand for the set of integers prime to n that are quadratic residues modulo n.

Given $L \in \mathcal{L}$ the algorithm which attempts to factor n is described in the flowchart of Figure 22.2. In Step 2 of the chart, $p_1, p_2, \ldots, p_h$ are the first h primes, where $h = \pi(v)$. This v will be chosen later to optimize the running time of "most" algorithms. At the start of the algorithm B and Z are initially empty lists. By an *operation* we mean any arithmetic operation $+, -, *, /$ between two positive integers not larger than n. If T_i denotes the number of times A_L executes step i, then clearly $T + 1 \geq T_1 \geq T_2 \geq T_3$ and $T_4 = T_5 = \max(T_3 - h, 0)$. In Step 2, factorization of w requires $O(h \log n)$ operations since each a_i is not more than $\log n$. In Step 4, the determination of c and the calculation of the coefficients f_b can be carried out by Gaussian elimination and thus requires at most $O(h^3)$ operations. In Step 5, the gcd can be computed in $O^{\sim}(\log n)$ operations Hence, if we let $R(L)$ stand for the total number of operations executed by A_L, we have

$$\begin{aligned} R(L) &= T_1 \cdot O(1) + T_2 \cdot O(h \log n) + T_3 \cdot O(1) + T_4 \cdot O(h^3) + T_5 \cdot O(h + \log n) \\ &= O(T_1 h \log n + T_4 h^3). \end{aligned}$$

THEOREM 22.4. *Let n be an odd integer divisible by at least two distinct primes. Put $v = \exp\{(2 \log n \operatorname{loglog} n)^{1/2}\}$ and $l = \lfloor v^2 + 1 \rfloor$. Then the average number of operations required in the execution of A_L ($L \in \mathcal{L}$) is*

$$\mu(R) = O(\exp\{3(2 \log n \operatorname{loglog} n)^{1/2}\}),$$

and the proportion of algorithms A_L which fail to find a proper factor of n is $O(l^{-1/2})$.

To prove the theorem, we need a few lemmas.

LEMMA 22.5. *Let $k \geq 2$ be an integer and let $v \geq 2$ be real. Then*

$$\psi(v^k, v) \geq \binom{\pi(v) + k}{k}.$$

PROOF. We have

$$p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_{\pi(v)}^{\alpha_{\pi(v)}} \leq v^k \quad \Longleftrightarrow \quad \sum_{1 \leq i \leq \pi(v)} \alpha_i \log p_i \leq k \log v,$$

where $p_1, p_2, \ldots, p_{\pi(v)} = 2, 3, \ldots$ are the primes up to v. Since $\log p_i \leq \log v$ for each i, the inequality $\sum_{1 \leq i \leq \pi(v)} \alpha_i \leq k$ is more stringent than the preceding one, i. e., it is satisfied by fewer choices of α_i. The $\pi(v)$-tuples α which satisfy this latter inequality are in 1-1 correspondance with strings of symbols of length $\pi(v) + k$ consisting of $\pi(v)$ symbols "|" and k symbols "$*$":

$$\underbrace{**}_{\alpha_1} | \underbrace{***}_{\alpha_2} | \cdots | \underbrace{****}_{\alpha_{\pi(v)}},$$

and there are $\binom{\pi(v)+k}{k}$ such strings. $\square$

LEMMA 22.6. *There is a constant $c_0 > 0$ such that $|\Psi^{\square}(n, v)| \geq n(\log n)^{-4r}$ whenever the following three conditions hold.*

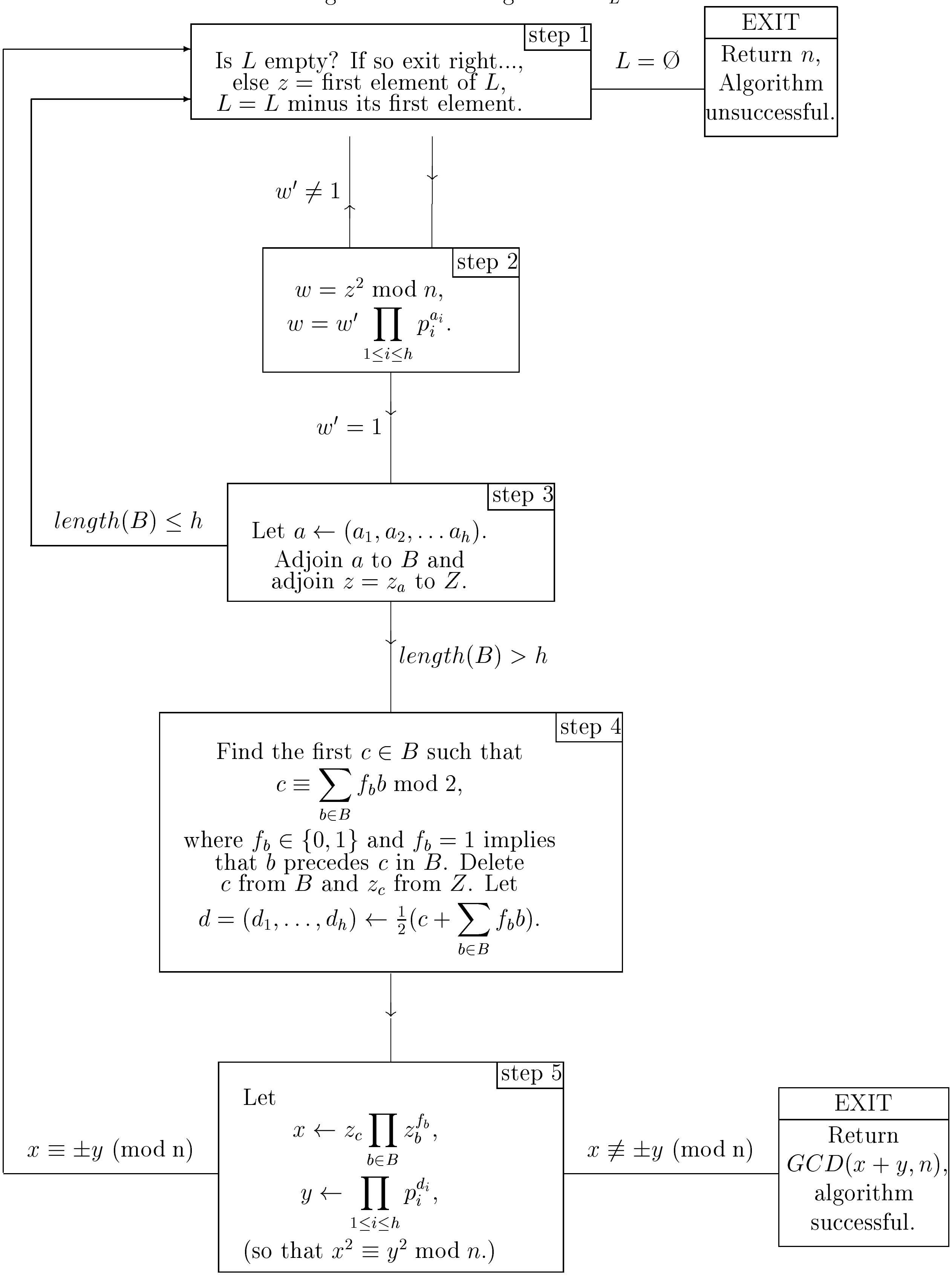
(i) $v \geq n^{1/2r}, \quad v \in \mathbb{R},\ n, r \in \mathbb{N}$,

(ii) $c_0^{-1} \log n \geq r \geq \operatorname{loglog} n$,

(iii) all prime factors of n are greater than v.

(Note that dividing n successively by $p_1, \ldots, p_{\pi(v)}$ allows us to assume (iii).)

Figure 22.3: The algorithm A_L



PROOF. We define some sums:

$$S = \sum_{\substack{s \le \sqrt{n} \\ s \in \Psi(n,v)}} \tau(s)^{-1}, \quad S_i = \sum_{\substack{s \le \sqrt{n} \\ s \in \Psi_i(n,v)}} \tau(s)^{-1}, \quad S' = \sum_{\substack{s \le \sqrt{n} \\ s \in \Psi(n,v)}} \tau(s).$$

where $\tau(t) = \sum_{d|t} 1$ (number of divisors of t.) We prove two strings of inequalities, the first of which is

$$|\Psi^{\square}(n,v)| \overset{[1]}{=} 2^d |\Psi(n,v) \cap Q| \overset{[2]}{\ge} 2^d \sum_{i=1}^{2^d} S_i^2 \overset{[3]}{\ge} S^2 \overset{[4]}{\ge} \frac{\psi(\sqrt{n},v)^4}{S'^2} \overset{[5]}{\ge} n^{-1}(\log n)^{-2} \psi(\sqrt{n},v)^4.$$

[1]. By the Chinese Remainder Theorem $\mathbb{Z}_n^* \cong \mathbb{Z}^*_{{q_1}^{l_1}} \times \mathbb{Z}^*_{{q_2}^{l_2}} \times \cdots \times \mathbb{Z}^*_{{q_d}^{l_d}}$, and each $y \in (\mathbb{Z}^*_{{q_i}^{l_i}})^2$ has exactly two square roots in $\mathbb{Z}^*_{{q_i}^{l_i}}$ for $i = 1, 2, \ldots, d$; so [1] follows from assumption (iii) by considering the mapping $\Psi^{\square}(n,v) \to \Psi(n,v) \cap Q$ given by $z \mapsto z^2 \bmod n$.

[2]. We carry out the calculation

$$\begin{aligned}
\sum_{1 \le i \le 2^d} S_i^2 &= \sum_{1 \le i \le 2^d} \Big(\sum_{\substack{s \le \sqrt{n} \\ s \in \Psi_i(n,v)}} \tau(s)^{-1} \Big)^2 = \sum_{1 \le i \le 2^d} \sum_{\substack{s,s' \le \sqrt{n} \\ s,s' \in \Psi_i(n,v)}} (\tau(s)\tau(s'))^{-1} \\
&= \sum_{t \le n} \Bigg(\sum_{1 \le i \le 2^d} \sum_{\substack{(s,s') \\ ss' = t \\ s,s' \le \sqrt{n} \\ s,s' \in \Psi_i(n,v)}} \tau(t)^{-1} \Bigg) = \sum_{t \le n} c(t) \le |\Psi(n,v) \cap Q|,
\end{aligned}$$

where $c(t)$ is implicitly defined. We have used the fact that $0 \le c(t) \le 1$ and the fact that $c(t) > 0 \Rightarrow \chi(t) = \chi(s)\chi(s') = (1, 1, \ldots, 1) \Rightarrow t \in Q \cap \Psi(n,v)$.

[3]. We apply the Cauchy-Schwartz inequality:

$$S^2 = \Big(\sum_{1 \le i \le 2^d} 1 \cdot S_i \Big)^2 \le \Big(\sum_{1 \le i \le 2^d} 1^2 \Big) \Big(\sum_{1 \le i \le 2^d} S_i^2 \Big) = 2^d \sum_{1 \le i \le 2^d} S_i^2.$$

[4]. Another application of Cauchy-Schwartz:

$$\begin{aligned}
\psi(\sqrt{n},v)^2 &= \Bigg(\sum_{\substack{s \le \sqrt{n} \\ s \in \Psi(n,v)}} 1 \Bigg)^2 = \Bigg(\sum_{\substack{s \le \sqrt{n} \\ s \in \Psi(n,v)}} \tau(s)^{-1/2} \tau(t)^{1/2} \Bigg)^2 \\
&\le \Bigg(\sum_{\substack{s \le \sqrt{n} \\ s \in \Psi(n,v)}} \tau(s)^{-1} \Bigg) \Bigg(\sum_{\substack{s \le \sqrt{n} \\ s \in \Psi(n,v)}} \tau(s) \Bigg) = SS'.
\end{aligned}$$

[5]. Dirichlet's inequality states that $\sum_{t \le x} \tau(t) = x \log x + (2\gamma - 1)x + O(\sqrt{x})$, where γ is Euler's constant (Apostol 1976, p. 57), hence

$$S' \le \sum_{t \le \sqrt{n}} \tau(t) \le \sqrt{n} \log n$$

for sufficiently large n.

The net inequality we have proved so far is

$$|\Psi^{\square}(n,v)| \geq n^{-1}(\log n)^{-2}\psi(\sqrt{n},v)^4. \tag{22.5}$$

Continuing the proof, we now show the string of inequalities

$$\psi(\sqrt{n},v) \overset{[1]}{\geq} \frac{\pi(v)^r}{r!} \overset{[2]}{\geq} v^r(r\log v)^{-r} \overset{[3]}{\geq} \sqrt{n}(\frac{1}{2}\log n)^{-r}.$$

[1]. The inequality $\psi(\sqrt{n},v) \geq \psi(v^r,v)$ fails once $v \geq n^{1/2r}+\epsilon$, therefore we take $v = n^{1/2r}$, and have by Lemma 22.5

$$\psi(\sqrt{n},v) = \psi(v^r,v) \geq \binom{\pi(v)+r}{r} \geq \frac{\pi(v)^r}{r!}.$$

This introduces a technicality in applying Lemma 22.6 to the main theorem since now $r = r(n)$ does not appear to be integer valued (see Lemma 22.7).

[2]. If v and r are sufficiently large then the Prime Number Theorem and Stirling's formula allow us to deduce

$$\pi(v) \geq v(2\log v)^{-1}, \qquad r! \leq \left(\frac{r}{2}\right)^r,$$

since [2] follows immediately from these we need to see that $v, r \to \infty$ as $c_0 \to \infty$, and this follows from the calculations below (refer to assumptions (ii) and (iii))

$$\log n \geq c_0 r \Rightarrow n^{1/2r} \geq e^{c_0/2},$$

$$\begin{aligned}\log n \geq c_0 r &\Rightarrow \log\log n \geq \log c_0 + \log r \\ &\Rightarrow r \geq \log c_0 + \log r \\ &\Rightarrow r \geq \log c_0\end{aligned}$$

[3]. Since $x(\log x)^{-1}$ is increasing on (e,∞), we have

$$\begin{aligned}v(\log v)^{-1} &\geq n^{1/2r}\left(\frac{\log n}{2r}\right)^{-1}, \\ v(r\log v)^{-1} &\geq n^{1/2r}\left(\frac{1}{2}\log n\right)^{-1}, \\ v^r(r\log v)^{-r} &\geq \sqrt{n}\left(\frac{1}{2}\log n\right)^{-r},\end{aligned}$$

The net inequality is

$$\psi(\sqrt{n},v) \geq \sqrt{n}(\frac{1}{2}\log n)^{-r}. \tag{22.6}$$

Combining all of these,

$$\begin{aligned}
c_0 \text{ large} \Rightarrow |\Psi^{\Box}(n,v)| &\overset{(22.5)}{\geq} n^{-1}(\log n)^{-2}\psi(\sqrt{n},v)^4 \\
&\overset{(22.6)}{\geq} n^{-1}(\log n)^{-2}\left[\sqrt{n}\left(\frac{1}{2}\log n\right)^{-r}\right]^4, \\
&= n(\log n)^{-4r}\left(\frac{4^r}{\log n}\right)^2 \\
&\geq n(\log n)^{-4r},
\end{aligned}$$

since $4^r \geq e^r \geq \log n$. $\Box$

LEMMA 22.7. *Let* $v = \exp((2\log n \operatorname{loglog} n)^{1/2})$ *and* $\lambda = n^{-1}\Psi^{\Box}(n,v)$. *Then for sufficiently large* n, *we have* $\lambda \geq v^{-1}$.

PROOF. Let

$$y = \frac{1}{4}(2\log n \operatorname{loglog} n)^{1/2}(\operatorname{loglog} n - \log 2)^{-1},$$

$$x = (\log n)^{1/2}(8 \operatorname{loglog} n)^{-1/2}.$$

Then, setting $\alpha = \log 2$, we have

$$\begin{aligned}
\lim_{n\to\infty}(y-x) &= \lim_{t\to\infty}\frac{1}{4}\left(\frac{(2te^t)^{1/2}-2t}{t-\alpha} - \frac{(2e^t)^{1/2}}{t^{1/2}}\right) \\
&= \lim_{t\to\infty}\frac{(2te^t)^{1/2}}{4}\left(\frac{1}{t-\alpha}-\frac{1}{t}\right) - \frac{t}{2(t-\alpha)} = \infty,
\end{aligned}$$

so that $[x,y]\cap\mathbb{Z} \neq \emptyset$, say $r \in [x,y]\cap\mathbb{Z}$. Put $v' = n^{1/2r}$, then $c_0^{-1} > x > y > t$ for t large (keeping $t = \operatorname{loglog} n$), and also $v' \leq n^{1/2r}$. Thus n, v' and r asymptotically satisfy the (sharper) hypotheses of Lemma 22.6, since $v' \leq v$ and all prime factors of n are greater than v. Recalling the proof of Lemma 22.6, we have

$$\lambda \geq (\log n)^{-4r}\left(\frac{4^r}{\log n}\right)^2 \geq (\log n)^{-4y}\left(\frac{4^y}{\log n}\right)^2 = v^{-1}. \qquad \Box$$

LEMMA 22.8. *prob*$\{\, L \in \mathcal{L}(n,l) : A_L$ executes Step 5 at least k times $\} \leq 2^{-k}$.

PROOF. Let $L_0 = \text{—} z_{i_1} \text{—} z_{i_2} \text{—} \cdots \text{—} z_{i_k} \text{—}$ be a bad list, where the z_{i_j} are the values that occur for z_c in step 5 and "—" abbreviates the (fixed) sub-list of elements seperating succesive z_{i_j}. Then it is clear that the 2^k lists
$\text{—} \pm z_{i_1} \text{—} \pm z_{i_2} \text{—} \cdots \text{—} \pm z_{i_k} \text{—}$
are also bad, and if we consider the larger collection of 2^{dk} lists
$\text{—} \alpha_{i_1} \text{—} \alpha_{i_2} \text{—} \cdots \text{—} \alpha_{i_k} \text{—}$,
where $\alpha_{i_j}^2 \equiv z_{i_j}^2 \bmod n$, we see that these (previous) 2^k lists are the only bad ones

in the new batch of 2^{dk} (of course Lemma 22.6 (iii) is in force.) Since $d \geq 2$ the proportion of bad lists is $2^k/2^{dk} \leq 2^{-k}$. Observe that these sets of size 2^{dk} we have introduced are disjoint from one another, so partially partitioning $\mathcal{L}(n,l)$ into subsets of size 2^{dk} in this manner proves our lemma. □

Proof of the Main Theorem. Recall the special choices for the variables:

$$\begin{aligned} h &= \pi(v), \\ v &= e^{(2\log n \operatorname{loglog} n)^{1/2}}, \\ l &= \lfloor v^2+1 \rfloor = \lfloor e^{2(2\log n \operatorname{loglog} n)^{1/2}} + 1 \rfloor. \end{aligned}$$

Note that these choices satisfy $n \geq l \geq 4hv$; we shall prove first that whenever n, l and h satisfy this inequality then

$$\mathrm{prob}\{\, L\colon A_L \text{ fails to split } n \,\} = O(l^{-1/2}).$$

Clearly $X(L)$ is also the number of z in L that lie in $\Psi^{\square}(n,v)$, and some easy calculations show that $\mu(X) = \lambda l$ and $\sigma^2 = var(X) = \lambda(1-\lambda)l$, where $\lambda = n^{-1}\Psi^{\square}(n,v)$ (mean and variance of a binomial distribution). We compute

$$\begin{aligned} &\mathrm{prob}\{\, L : A_L \text{ fails} \,\} \\ &= \mathrm{prob}\{\, L : A_L \text{ fails } \& \ X(L) \leq \tfrac{1}{2}\mu \,\} + \mathrm{prob}\{\, L : A_L \text{ fails } \& \ X(L) > \tfrac{1}{2}\mu \,\} \\ &\leq \mathrm{prob}\{\, L : X(L) \leq \tfrac{1}{2}\mu \,\} + \mathrm{prob}\{\, L : A_L \text{ fails } \& \ X(L) > \tfrac{1}{2}\mu \,\} \\ &\leq \mathrm{prob}\{\, L : |X(L) - \tfrac{1}{2}\mu| \geq \tfrac{1}{2}\mu \,\} + \mathrm{prob}\{\, L : A_L \text{ fails } \& \ X(L) > 2h \,\}, \end{aligned}$$

where we have used Lemma 22.7 and our assumption to get $\mu = \frac{1}{2}\lambda l \geq \frac{1}{2}v^{-1}(4hv) = 2h$. By Lemma 22.8, the last probability is at most 2^{-h}, because if A_L fails to find a factor and L contains more than $2h$ Z-numbers, then this means that A_L passes through Step 5 more than h times. Also, one readily applies Chebyshev's inequality to get $\mathrm{prob}\{\, L : X(L) \leq \frac{1}{2}\mu \,\} \leq \sigma^2(\frac{1}{2}\mu)^{-2} = 4(1-\lambda)\mu^{-1} \leq 4\mu^{-1}$. Combining these gives

$$\mathrm{prob}\{\, L : A_L \text{ fails} \,\} \leq 4\mu^{-1} + 2^{-h} = O(vl^{-1}) + O(n^{-1}) = O(l^{-1/2}).$$

This proves the first part of the theorem.

For the second part we consider T_4 and T_1 as random variables on the space $\mathcal{L}(n,l)$. We have

$$\begin{aligned} \mu(T_1) &= \sum_{\substack{L \\ T_1(L) > 4hv+2}} T_1(L)\mathrm{prob}(L) + \sum_{\substack{L \\ T_1(L) \leq 4hv+2}} T_1(L)\mathrm{prob}(L) \\ &\leq (l+1)\mathrm{prob}\{\, L\colon T_1(L) > 4hv+2 \,\} + (4hv+2)\mathrm{prob}\{\, L\colon T_1(L) \leq 4hv+2 \,\} \\ &= (l+1)O(v^{-1}) + O(4hv+2) \\ &= O(vh), \end{aligned}$$

and

$$\begin{aligned}\mu(T_4) &= \sum_L T_4(L)\mathrm{prob}(L) = \sum_{1\le k\le l-h} \sum_{\substack{L\\ T_4(L)=k}} T_4(L)\mathrm{prob}(L)\\ &= \sum_{1\le k\le l-h} k\,\mathrm{prob}\{\, L : T_4(L)=k \,\} \le \sum_{1\le k\le l-h} k2^{1-k} \le \sum_{k\ge 1} k2^{1-k} = O(1),\end{aligned}$$

where we have used Lemma 22.8 in the last calculation. For the special value $h = \pi(v)$ (above) we have $\mu(T_1 h \log n) = O(vh^2 \log n) = O(v^3)$, since $h = O(v/\log v)$ by the Prime Number Theorem. Moreover $\mu(T_4 h^3) = O(h^3) = o(v^3)$ holds for h. Let $R(L)$ be the number of operations executed by A_L. The discussion before Figure 22.2 shows that $R = O(T_1 h \log n + T_4 h^3)$, therefore

$$\mu(R) = O(\mu(T_1 h \log n) + \mu(T_4 h^3)) = O(O(v^3) + o(v^3)) = O(v^3),$$

which is the average number of operations for A_L claimed in the main theorem. $\square$

Universität-GH Paderborn
FB 17, AG Algorithmische Mathematik
Skript Computeralgebra II, Teil 3
Sommersemester 1996

23. Gröbner bases

In this section, we will present an important algorithmic approach to dealing with polynomials in several variables. Hironaka introduced a special type of basis for polynomial ideals, called "standard basis". Buchberger re–invented them independently in his Ph.D. thesis, and named them *Gröbner bases* after his thesis advisor W. Gröbner. They are a vital tool in modern computational algebraic geometry. The following example illustrates some of the problems in applications that can be solved with the help of Gröbner bases.

Example 23.1. **Robotics**. *Suppose we have a two-armed robot whose first arm is fixed at one end with a joint to a point (say, the origin of the Cartesian plane), and with another joint between the two arms. The first arm has length* 2*, the joint between the two arms is in position* (x, y)*, the second arm has length* 1*, and its endpoint is in position* (z, w)*. Furthermore, there is a line* $L = \{(u, \lambda u + \mu) : u \in \mathbb{R}\}$ *with some fixed parameters* $\lambda, \mu \in \mathbb{R}$*. Figure 23.4 illustrates the situation.*

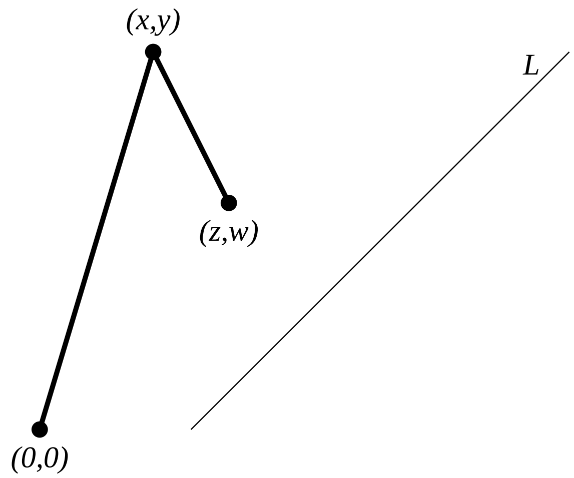


Figure 23.4: A two-armed robot in the plane

One possible question is: can the robot reach the line? The possible positions $(x, y, z, w) \in \mathbb{R}^4$ *of the robot are charcterized by the algebraic equations*

$$x^2 + y^2 = 4 \text{ and } (z - x)^2 + (w - y)^2 = 1, \tag{23.1}$$

and an answer to the question is either a quadruple (x, y, z, w) satisfying (23.1) and the additional equation

$$w = \lambda z + \mu,$$

or a proof that no such quadruple can exist.

Let F be a field, $R = F[x_1, \ldots, x_n]$ the polynomial ring in n variables over F, and $f_1, \ldots, f_s \in R$. In this section, we use the notation $I = \langle f_1, \ldots, f_s \rangle = \{\sum a_i f_i \colon a_i \in R\}$ for the *ideal generated by* $f_1, \ldots, f_s$, and $V(I) = \{u \in F^n \colon f(u) = 0 \ \text{ for all } f \in I\}$ is the *variety of* I. Interesting questions about I include:

- Is $V(I) \neq \emptyset$?
- How "big" is $V(I)$?
- Membership problem: given $g \in R$, is $g \in I$?
- Triviality: Is $I = R$?

A Gröbner basis of I is a special "basis" for I, in which these questions are easy to answer. A good reference for this section is Cox *et al.* (1992), where also the proofs that we omit for space considerations can be found.

Everything is easy in the case $n = 1$: Since $F[x]$ is a Euclidean domain (cf. section 9),

$$\langle f_1, \ldots, f_s \rangle = \langle \gcd(f_1, \ldots, f_s) \rangle \tag{23.2}$$

so that we may always assume $s = 1$. Now let $f, g \in F[x]$, and divide g by f with remainder, yielding $q, r \in F[x]$ with $g = qf + r$ and $\deg r < \deg f$. Then $g \in \langle f \rangle$ if and only if $r = 0$, and $V(f) = \{u_1, \ldots, u_s\}$ if $x - u_1, \ldots, x - u_s$ are the distinct linear factors of f in $F[x]$. (23.2) is no longer true in the case of two variables or more. E.g., $\gcd(x, y) = 1$ in $F[x, y]$, but (x, y) is different from $(1) = F[x, y]$. (Prove this!)

Much of the effort of Gröbner bases goes into restoring some of these apparently lost properties, such as division with remainder.

23.1. Monomial orderings and multivariate division with remainder. We introduce an analogue of division with remainder in the case of many variables. In order to do that, we have to say what the "leading term" of a polynomial should be in the general case.

Recall that an *order* $<$ on a set S satisfies $\neg((\alpha < \beta) \wedge (\beta < \alpha))$ and $\alpha < \beta < \gamma \Longrightarrow \alpha < \gamma$, is *total* if and only if either $\alpha = \beta$ or $\alpha < \beta$ or $\beta < \alpha$, for all $\alpha, \beta, \gamma \in S$, and is a *well–ordering* if every non–empty subset of S has a smallest element.

We identify the vector $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}^n$ with the monomial

$$\boldsymbol{x}^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n} \in R.$$

Definition 23.2. A monomial ordering *in $R = F[x_1, \ldots, x_n]$ is a relation $\prec$ on $\mathbb{N}^n$ such that* *(i)* $\prec$ *is a total order,*

(ii) $\alpha \prec \beta \Rightarrow \alpha + \gamma \prec \beta + \gamma$ *for all* $\alpha, \beta, \gamma \in \mathbb{N}^n$,

(iii) $\prec$ *is a well–ordering.*

Under the above identification, this gives an ordering on the monomials in R. If $n = 1$, e.g., the natural order on $\mathbb{N} = \mathbb{N}^1$ is a monomial ordering, and the corresponding univariate monomials are ordered by their degree.

If (i) holds, then (iii) is equivalent to the condition that there be no infinite descending chain $\alpha_1 \succ \alpha_2 \succ \alpha_3 \succ \cdots$ in $\mathbb{N}^n$.

EXERCISE 23.3. *Show that* $0 = (0, \ldots, 0) \in \mathbb{N}^n$ *is the smallest element with respect to a monomial ordering* $\prec$, *i.e.,* $0 = \min_{\prec} \mathbb{N}^n$.

EXAMPLE 23.4. *Here are three standard examples for monomial orderings:*

1. Lexicographic order.

$$\alpha \prec_{\mathrm{lex}} \beta \Longleftrightarrow \text{ the leftmost nonzero entry in } \alpha - \beta \text{ is negative.}$$

For example, if $n = 3, \alpha_1 = (0, 3, 5), \alpha_2 = (1, 2, 0), \alpha_3 = (1, 5, 2)$, *and* $\alpha_4 = (4, 1, 3)$, *then* $\alpha_1 \prec_{\mathrm{lex}} \alpha_2 \prec_{\mathrm{lex}} \alpha_3 \prec_{\mathrm{lex}} \alpha_4$.

2. Graded lexicographic order. *Let* $\alpha = (\alpha_1, \ldots, \alpha_n), \beta = (\beta_1, \ldots, \beta_n) \in \mathbb{N}^n$. *Then*

$$\alpha \prec_{\mathrm{grlex}} \beta \Longleftrightarrow \sum_i \alpha_i < \sum_i \beta_i \text{ or } \Big(\sum_i \alpha_i = \sum_i \beta_i \text{ and } \alpha \prec_{\mathrm{lex}} \beta\Big).$$

With $\alpha_1, \ldots, \alpha_4$ *as above, we have* $\alpha_2 \prec_{\mathrm{grlex}} \alpha_1 \prec_{\mathrm{grlex}} \alpha_3 \prec_{\mathrm{grlex}} \alpha_4$.

3. Graded reverse lexicographic order.

$$\begin{aligned} \alpha \prec_{\mathrm{grevlex}} \beta \quad \Longleftrightarrow \quad & \sum_i \alpha_i < \sum_i \beta_i \text{ or } \Big(\sum_i \alpha_i = \sum_i \beta_i \text{ and the} \\ & \text{rightmost nonzero entry in } \alpha - \beta \in \mathbb{Z}^n \text{ is positive}\Big). \end{aligned}$$

Then $\alpha_2 \prec_{\mathrm{grevlex}} \alpha_1 \prec_{\mathrm{grevlex}} \alpha_4 \prec_{\mathrm{grevlex}} \alpha_3$, *where* $\alpha_1, \ldots, \alpha_4$ *are as before.*

In any of the three examples, the variables are ordered as $x_n \prec x_{n-1} \prec \ldots \prec x_2 \prec x_1$. "Graded" refers to the fact that the total degree $\sum \alpha_i$ is the main criterion.

Once we have a monomial order on R, we can sort terms of a polynomial according to $\prec$.

EXAMPLE 23.5. *Let* $f = 4xy^2z + 4z^3 - 5x^3 + 7x^2z^2 \in \mathbb{Q}[x, y, z]$ *(we always identify* x, y, z *with* x_1, x_2, x_3*). Then the orderings of* f *with respect to* $\prec_{\mathrm{lex}}$, $\prec_{\mathrm{grlex}}$, *and* $\prec_{\mathrm{grevlex}}$ *are:* $-5x^3 + 7x^2z^2 + 4xy^2z + 4z^2$, $7x^2z^2 + 4xy^2z - 5x^3 + 4z^2$, *and* $4xyz^2 + 7x^2z^2 - 5x^3 + 4z^2$, *respectively.*

In the case $n = 1$, we have $\prec_{\text{lex}} = \prec_{\text{grlex}} = \prec_{\text{grevlex}}$.

THEOREM 23.6. *$\prec_{\text{lex}}$, $\prec_{\text{grlex}}$,, and $\prec_{\text{grevlex}}$, are monomial orderings.*

PROOF. The proof is a simple check; we give some details only for $\prec_{\text{grevlex}}$. Condition (i) is satisfied, since for each $\alpha, \beta \in \mathbb{N}^n$ with $\alpha \neq \beta$ we have either $\alpha \prec_{\text{grevlex}} \beta$ or $\beta \prec_{\text{grevlex}} \alpha$.

For Condition (ii), we have

$$\sum_i \alpha_i < \sum \beta_i \Longleftrightarrow \sum_i (\alpha_i + \gamma_i) < \sum_i (\beta_i + \gamma_i),$$

and similarly for " $=$", and $\alpha - \beta = (\alpha + \gamma) - (\beta + \gamma)$. For Condition (iii), if $S \subseteq \mathbb{N}^n$ is nonempty and $T \subseteq S$ the set of monomials of smallest total degree in S, then T is finite (since for any $m \in \mathbb{N}$ there are only finitely many monomials of total degree m) and $\min T = \min S$. $\square$

The antilexicographic ordering $\prec_{\text{alex}}$ on $\mathbb{N}^2$, with

$$\alpha \prec_{\text{alex}} \beta \Longleftrightarrow \beta \prec_{\text{lex}} \alpha,$$

is an example where Condition (iii) is violated: $S = \mathbb{N} \times \{0\}$ has no smallest element, since

$$(\alpha_1, 0) \prec_{\text{alex}} (\beta_1, 0) \Longleftrightarrow \beta_1 < \alpha_1.$$

DEFINITION 23.7. *Let $f = \sum_{\alpha \in \mathbb{N}^n} a_\alpha \boldsymbol{x}^\alpha \in R$ be a nonzero polynomial with all $a_\alpha \in F$, and $\prec$ a monomial ordering.*

(i) The multidegree *of f is*

$$\text{mdeg}(f) = \max\nolimits_\prec \{\alpha \in \mathbb{N}^n : a_\alpha \neq 0\},$$

where $\max_\prec$ means the maximum with respect to $\prec$.

(ii) The leading coefficient *of f is*

$$\text{lc}(f) = a_{\text{mdeg}(f)} \in F.$$

(iii) The leading monomial *of f is $\text{lm}(f) = \boldsymbol{x}^{\text{mdeg}(f)} \in R$.*

(iv) The leading term *of f is $\text{lt}(f) = \text{lc}(f) \cdot \text{lm}(f) \in R$.*

EXAMPLE 23.8. *To illustrate this, we take the polynomial $f = 4xy^2z + 4z^3 - 5x^3 + 7x^2z^2 \in \mathbb{Q}[x, y, z]$ from Example 23.5, and the three orders from Example 23.4.*

	$\prec_{\text{lex}}$	$\prec_{\text{grlex}}$	$\prec_{\text{grevlex}}$
$\text{mdeg}(f)$	$(3,0,0)$	$(2,0,2)$	$(1,2,1)$
$\text{lc}(f)$	-5	7	4
$\text{lm}(f)$	x^3	x^2z^2	xy^2z
$\text{lt}(f)$	$-5x^3$	$7x^2z^2$	$4xy^2z$

If we wanted to extend these definitions to the zero polynomial, we would formally adjoin the element $-\infty$ to $\mathbb{N}^n$ and let $\mathrm{lt}(0) = \mathrm{lm}(0) = \mathrm{lc}(0) = 0$ and $\mathrm{mdeg}(0) = -\infty$, with the obvious interpretation.

LEMMA 23.9. *Fix a monomial order $\prec$ on R, and let $f, g \in R \setminus \{0\}$.*

(i) $\mathrm{mdeg}(fg) = \mathrm{mdeg}(f) + \mathrm{mdeg}(g)$.

(ii) If $f + g \neq 0$ *then* $\mathrm{mdeg}(f+g) \leq \max\{\mathrm{mdeg}(f), \mathrm{mdeg}(g)\}$, *with equality if* $\mathrm{mdeg}(f) \neq \mathrm{mdeg}(g)$.

Now we are ready to state an algorithm for division with remainder in R. Given $f, f_1, \ldots, f_s \in R$, we want to write $f = a_1 f_1 + \ldots + a_s f_s + r$ with $a_1, \ldots, a_s, r \in R$.

EXAMPLE 23.10. *Let* $\prec = \prec_{\mathrm{lex}}$, $f = xy^2 + 1$, $f_1 = xy + 1$, $f_2 = y + 1$.

	$xy+1$	$y+1$
xy^2+1	y	
$-(xy^2+y)$		
$-y+1$		-1
$-(-y-1)$		
2		

	$xy+1$	$y+1$
xy^2+1		xy
$-(xy^2+xy)$		
$-xy+1$		$-x$
$-(-xy-x)$		
$x+1$		

In the left-hand tableau, division is performed as in the univariate case, with the difference that we have two divisors instead of one. The quotients that we get in each step are recorded in the column below the respective divisor. In the last line, 2 is not divisible by the leading term of f_1 or f_2, and the process stops. Hence $f = y \cdot f_1 - 1 \cdot f_2 + 2$. Note that there is some non-determinism here because we might as well have chosen f_2 instead of f_1 in the first step; this is illustrated in the right-hand tableau, where we find that $f = 0 \cdot f_1 + (xy - x) \cdot f_2 + (x+1)$, and no term in $x+1$ is divisible by $\mathrm{lc}(f_1)$ *or* $\mathrm{lc}(f_2)$.

EXAMPLE 23.11. *Let* $\prec=\prec_{\mathrm{lex}}, = f = x^2y + xy^2 + y^2$, $f_1 = xy - 1$, $f_2 = y^2 - 1$.

	$xy-1$	y^2-1	remainder
$x^2y+xy^2+y^2$	x		
$-(x^2y-x)$			
xy^2+x+y^2	y		
$-(xy^2-y)$			
$x+y^2+y$			x
$-x$			
y^2+y		1	
$-(y^2-1)$			
$y+1$			

Here, a phenomenon occurs that cannot happen in the univariate case. In the third step, the leading term x is not divisible by the leading term of f_1 or f_2 and is moved to the remainder column. Afterwards, the division may continue further, and the result is $f = (x+y) \cdot f_1 + 1 \cdot f_2 + (x+y+1)$.

Theorem 23.12. *(Division with remainder in $F[x_1, \ldots, x_n]$). Let $\prec$ be a monomial order in R, and $f, f_1, \ldots, f_s \in R$. Then there exist $a_1, \ldots, a_s, r \in R$, such that $f = a_1 f_1 + \ldots + a_s f_s + r$ and no monomial occurring in r is divisible by any of $\mathrm{lt}(f_1), \ldots, \mathrm{lt}(f_s)$. These $a_1, \ldots, a_s$ are the* quotients *and r is the* remainder *of the division. Furthermore, if $a_i f_i \neq 0$, then $\mathrm{mdeg}(f) \geq \mathrm{mdeg}(a_i f_i)$.*

This kind of division with remainder need not be unique: there may be a choice in what divisor polynomial to take when the leading term of f is divisible by more than one leading term of the divisor polynomials.

Example 23.13. *If we choose f_2 instead of f_1 in the second step of Example 23.11, we get*

	$xy - 1$	$y^2 - 1$	*remainder*
$x^2y + xy^2 + y^2$	x		
$-(xy^2 - x)$			
$xy^2 + x + y^2$		x	
$-(xy^2 - x)$			
$2x + y^2$			$2x$
$-2x$			
y^2		1	
$-(y^2 - 1)$			
1			

and $f = x \cdot f_1 + (x+1) \cdot f_2 + (2x + 1)$.

If, however, we fix the order of the trial divisions to always be $f_1, \ldots, f_s$, then the process is uniquely determined and therefore also its outcome, the *quotients* $a_1, \ldots, a_s$ and the *remainder* r, for which we write

$$r = f \ \mathrm{rem}\ (f_1, \ldots, f_s). \tag{23.3}$$

For $s = 1$, division with remainder solves the ideal membership problem: $f \in \langle f_1 \rangle$ if and only if the remainder is zero. This fails in general for $s \geq 2$, as the following example shows.

Example 23.14. *Consider dividing $f = xy^2 - x$ by $f_1 = xy + 1$ and $f_2 = y^2 - 1$:*

	$xy + 1$	$y^2 - 1$
$xy^2 - x$	y	
$-(xy^2 + y)$		
$-x - y$		

so that $f = y \cdot f_1 + 0 \cdot f_2 + (-x - y)$. But on the other hand, $f = 0 \cdot f_1 + x \cdot f_2 + 0$, which shows that $f \in \langle f_1, f_2 \rangle$.

Our goal is therefore to find a special basis of an arbitrary ideal such that the remainder on division by that basis is unique and thus gives the correct answer to the ideal membership problem. At first sight, it is not clear whether such a type of basis exists at all.

23.2. Monomial ideals and Hilbert's Basis Theorem.

DEFINITION 23.15. *A* monomial ideal $I \subseteq R$ *is an ideal generated by monomials in R, i.e.,*

$$\exists A \subseteq \mathbb{N}^n \quad I = \langle \boldsymbol{x}^A \rangle = \langle \boldsymbol{x}^\alpha : \alpha \in A\} \rangle.$$

LEMMA 23.16. *Let* $I = \langle \boldsymbol{x}^A \rangle \subseteq R$ *be a monomial ideal and* $\alpha \in \mathbb{N}^n$. *Then*

$$\boldsymbol{x}^\alpha \in I \quad \Longleftrightarrow \quad \exists \beta \in A \quad \boldsymbol{x}^\beta \mid \boldsymbol{x}^\alpha.$$

PROOF. "$\Longleftarrow$" is clear. Conversely, let $\alpha_1, \ldots, \alpha_s \in A$ and $h_1, \ldots, h_s \in R$ such that $\boldsymbol{x}^\alpha = \sum_i h_i \boldsymbol{x}^{\alpha_i}$. Each monomial on the right hand side is divisible by some $\boldsymbol{x}^\gamma$, with $\gamma \in A$, and hence also each monomial on the left hand side. □

LEMMA 23.17. *Let* $I \subseteq R$ *be a monomial ideal and* $f \in R$. *Then the following are equivalent:*

(i) $f \in I$,

(ii) each term of f *is in* I,

(iii) f *is an* F*–linear combination of monomials in* I.

PROOF. (i) $\Longrightarrow$ (ii): As in the previous proof, a representation $f = \sum_i h_i \boldsymbol{x}^{\alpha_i}$ implies that each monomial of f is divisible by some $\boldsymbol{x}^\gamma$, with $\gamma \in A$. (ii) $\Longrightarrow$ (iii) $\Longrightarrow$ (i) are clear, and, in fact, hold for any ideal I. □

Taking the three leading terms from Example 23.8, so that

$$I = \langle -5x^3, 7x^2z^2, 4xy^2z \rangle,$$

the lemma implies that $8x^4y - 9x^2y^2z \in I$ and $x^2z^2 - y^8 \notin I$. The implication (i) $\Rightarrow$ (ii) is false for some ideals, as shown in Example 23.22 below.

COROLLARY 23.18. *Two monomial ideals are identical if and only if they contain the same monomials.*

THEOREM 23.19. *(Dickson's Lemma). Every monomial ideal is generated by a finite set of monomials. More precisely, for every* $A \subseteq \mathbb{N}^n$ *there exists a finite subset* $B \subseteq A$ *such that* $\langle \boldsymbol{x}^A \rangle = \langle \boldsymbol{x}^B \rangle$.

COROLLARY 23.20. *Let* $\prec$ *be an order on* $\mathbb{N}^n$ *such that*

$$\forall \alpha, \beta, \gamma \in \mathbb{N}: \alpha \prec \beta \Longrightarrow \alpha + \gamma \prec \beta + \gamma.$$

Then $\prec$ *is a well–ordering if and only if* $\alpha \succeq 0$ *for all* $\alpha \in \mathbb{N}^n$.

PROOF. ("$\Longleftarrow$" only, "$\Longrightarrow$" is Exercise 23.3.) Let $A \subseteq \mathbb{N}^n$ be non–empty, and $I = \langle \boldsymbol{x}^A \rangle \subseteq R$. Then I is finitely generated, by Dickson's Lemma, i.e.,

$$\exists \alpha_1, \ldots, \alpha_s \in A \quad I = \langle \boldsymbol{x}^{\alpha_1}, \ldots, \boldsymbol{x}^{\alpha_s} \rangle.$$

We order them so that $\alpha_1 \prec \alpha_2 \prec \cdots \prec \alpha_s$. We now claim that $\min_\prec A = \alpha_1$. It is clear that $\min_\prec A \preceq \alpha_1$. For the reverse direction, let $\alpha = \min_\prec A$. By Lemma 23.16, there exist $i \leq s$ and $\gamma \in \mathbb{N}^n$ with $\alpha = \alpha_i + \gamma$, so that $\alpha = \alpha_i + \gamma \succ \alpha_1 + \gamma \succeq \alpha_1 + 0 = \alpha_1$. $\square$

So we can replace the condition (iii) in the definition of monomial orders by

(iii)' $\quad \forall \alpha \in \mathbb{N}^n \quad \alpha \succeq 0.$

DEFINITION 23.21. *Let $G \subseteq R$ be any set different from $\emptyset, \{0\}$. Then we define* $\mathrm{lt}(G) = \{\mathrm{lt}(g) : g \in G\}$.

If $I \subseteq R$ is an ideal, then there is a finite subset $G \subseteq I$ such that $\langle \mathrm{lt}(I) \rangle = \langle \mathrm{lt}(G) \rangle$, by Dickson's Lemma. However, it can happen that a finite set G generates I but $\langle \mathrm{lt}(I) \rangle \supsetneqq \langle \mathrm{lt}(G) \rangle$, as in the following example.

EXAMPLE 23.22. *Let $f = x^3 - 2xy, g = x^2y - 2y^2 + x \in \mathbb{Q}[x, y]$, $\prec = \prec_{\mathrm{grlex}}$, $G = \{f, g\}$, and $I = \langle G \rangle$. Then $x^2 = -y \cdot f + x \cdot g$ and $x^2 \in \langle \mathrm{lt}(I) \rangle$ but $x^2 \notin \langle \mathrm{lt}(G) \rangle = \langle x^3, x^2y \rangle$.*

THEOREM 23.23. *(Hilbert's Basis Theorem) Every ideal in $R = F[x_1, \ldots, x_n]$ is finitely generated.*

PROOF. Let $I \subseteq R$ be an ideal, $I \neq \{0\}$, and $g_1, \ldots, g_s \in I$ such that $\langle \mathrm{lt}(I) \rangle = \langle \mathrm{lt}(g_1), \ldots, \mathrm{lt}(g_s) \rangle$; these exist by Dickson's Lemma. If f is an arbitrary polynomial in I, then division with remainder yields $f = a_1 g_1 + \cdots + a_s g_s + r$, with $a_1, \ldots, a_s, r \in R$, and either $r = 0$ or such that no term of r is divisible by the leading term of any g_i. But $r = f - a_1 g_1 - \cdots - a_s g_s \in I$, and hence $\mathrm{lt}(r) \in \mathrm{lt}(I) = \langle \mathrm{lt}(g_1), \cdots, \mathrm{lt}(g_s) \rangle$. This together with Lemma 23.16 implies that $r = 0$, and thus $f \in \langle g_1, \cdots, g_s \rangle$. $\square$

23.3. Gröbner bases and S–polynomials.

DEFINITION 23.24. *Fix a monomial order. A finite set $G \subseteq R$ is a* Gröbner basis *(*GB*) for the ideal $I \subseteq R$ if $\langle \mathrm{lt}(I) \rangle = \langle \mathrm{lt}(G) \rangle$.*

With the convention that $\langle \rangle = \langle \emptyset \rangle = \{0\}$, our proof of Hilbert's Basis Theorem implies the following.

THEOREM 23.25. *Every ideal I in $R = F[x_1, \ldots, x_n]$ has a Gröbner basis. Every such basis also generates I.*

In Example 23.22, G is not a GB for I, but $\{f, g, x^2, 2xy, -2y^2 + x\}$ is.

Theorem 23.26. *(Ascending Chain Condition, ACC) Let* $I_1 \subseteq I_2 \subseteq I_3 \subseteq \ldots$ *be an ascending chain of ideals in* R*. Then the chain stabilizes, i.e.,* $I_n = I_{n+1} = I_{n+2} = \ldots$ *for some* $n \in \mathbb{N}$.

Proof. Let $I = \bigcup_{j\geq 1} I_j$. Then I is an ideal, which is finitely generated, by Hilbert's Basis Theorem, say $I = \langle f_1, \ldots, f_s \rangle$. With $n = \min\{j \geq 1 \colon f_1 \ldots, f_s \in I_j\}$, we then have $I_n = I_{n+1} = \cdots = I$. □

In general, a ring satisfying the ascending chain condition is called *Noetherian*. Thus $F[x_1, \ldots, x_n]$ is Noetherian, if F is a field.

We now want to show that division with remainder by a GB is a valid ideal membership test.

Lemma 23.27. *Let* G *be a GB for some ideal* $I \subseteq R$ *and* $f \in R$*. Then there is a unique polynomial* $r \in R$ *with*

1. $f - r \in I$,
2. *no term of* r *is divisible by any of* $\mathrm{lt}(G)$.

Proof. Existence follows from Theorem 23.12. For the uniqueness, suppose that $f = h_1 + r_1 = h_2 + r_2$ with $h_1, h_2 \in I$ and no term of r_1 or r_2 divisible by any of $\mathrm{lt}(G)$. Then $r_1 - r_2 = h_2 - h_1 \in I$, and $\mathrm{lt}(r_1 - r_2)$ is divisible by some $\mathrm{lt}(g)$ with $g \in G$, by Lemma 23.16. Hence $h_2 - h_1 = 0$ and $r_1 = r_2$. □

Thus the remainder on division by G does not depend on the order of the elements of G. Extending the notation (23.3), we then write

$$r = f \text{ rem } G \in R.$$

Theorem 23.28. *Let* G *be a GB for the ideal* $I \subseteq R$*, and* $f \in R$*. Then* $f \in I$ *if and only if the remainder of* f *on division by* G *is zero.*

This property is equivalent to the definition of GB.

In order to construct a GB for an ideal I from a given basis G, we have to investigate how G can fail to be a GB. One possible reason is that a linear combination $a\boldsymbol{x}^\alpha f + b\boldsymbol{x}^\beta g$ of two polynomials $f, g \in G$ with $a, b \in F$ and $\alpha, \beta \in \mathbb{N}^n$ may yield a polynomial whose leading term is not divisible by any of $\mathrm{lt}(G)$, by cancellation of leading terms. E.g., in Example 23.22, $(-y) \cdot f + x \cdot g = x^2$, and $\mathrm{lt}(x^2) = x^2$ is not divisible by any of $\mathrm{lt}(G) = \{x^3, x^2y\}$.

Definition 23.29. *Let* $f, g \in R \setminus \{0\}$, $\alpha = (\alpha_1, \ldots, \alpha_n) = \mathrm{mdeg}(f)$, $\beta = (\beta_1, \ldots, \beta_n) = \mathrm{mdeg}(g)$, *and* $\gamma = (\gamma_1, \ldots, \gamma_n) = \max\{\alpha, \beta\} \in \mathbb{N}^n$, *with* $\gamma_i = \max\{\alpha_i, \beta_i\}$ *for all* $i \leq n$*. The* S-polynomial *of* f *and* g *is*

$$S(f, g) = \frac{\boldsymbol{x}^\gamma}{\mathrm{lt}(f)} f - \frac{\boldsymbol{x}^\gamma}{\mathrm{lt}(g)} g.$$

Clearly $S(g,f) = -S(f,g)$, and since $\boldsymbol{x}^\gamma/\mathrm{lt}(f), \boldsymbol{x}^\gamma/\mathrm{lt}(g) \in R$, we have $S(f,g) \in \langle f,g\rangle$. In Example 23.22, we have

$$\alpha = (3,0), \beta = (2,1), \gamma = (3,1),$$

$$S(f,g) = \frac{x^3y}{x^3}f - \frac{x^3y}{x^2y}g = -x^2.$$

The following lemma says that when cancellation of leading terms occurs in a linear combination of polynomials in G, it necessarily comes from S-polynomials.

LEMMA 23.30. *Let $g_1,\ldots,g_s \in R$, $\alpha_1,\ldots,\alpha_s \in \mathbb{N}^n$, $c_1,\ldots,c_s \in F \setminus \{0\}$,*

$$f = \sum_{1\le i\le s} c_i \boldsymbol{x}^{\alpha_i} g_i \in R,$$

and $\delta \in \mathbb{N}^n$ such that $\alpha_i + \mathrm{mdeg}(g_i) = \delta$ for $1 \le i \le s$, and $\mathrm{mdeg}(f) < \delta$. Then $\boldsymbol{x}^{\gamma_{ik}} \mid \boldsymbol{x}^\delta$, where $\boldsymbol{x}^{\gamma_{ik}} = \mathrm{lcm}(\mathrm{lm}(g_i), \mathrm{lm}(g_k))$, and there are $c_{ik} \in F$ such that

$$f = \sum_{1\le i<k\le s} c_{ik}\boldsymbol{x}^{\delta-\gamma_{ik}} S(g_i,g_k)$$

and $\mathrm{mdeg}(\boldsymbol{x}^{\delta-\gamma_{ik}} S(g_i,g_k)) < \delta$ for all $1 \le i < k \le s$.

The following theorem gives us an easy test for GB.

THEOREM 23.31. *Let $G \subseteq R$ be finite. Then G is a GB (of the ideal $\langle G\rangle$) if and only if*

$$S(g,h) \text{ rem } G = 0 \text{ for all } g,h \in G. \tag{23.4}$$

PROOF. "$\Longrightarrow$" is clear since $S(g,h) \in I = \langle G\rangle$ for $g,h \in G$, and the remainder of an element of I on division by a GB is zero. For the reverse direction, we let $f \in I\setminus\{0\}$, and have to show that

$$\mathrm{lt}(f) \in \langle \mathrm{lt}(G)\rangle.$$

Write

$$f = \sum_{g\in G} a_g g \tag{23.5}$$

where all $a_g \in R$. Then $\mathrm{mdeg}(f) \preceq \max_{g\in G}\{\mathrm{mdeg}(a_g g)\}$. If strict inequality holds, then some cancellation of leading terms occurs in (23.5). By Lemma 23.30, we can rewrite (23.5) in terms of S-polynomials, and since remainders of $S(g,h)$ with respect to G are zero, we get $a'_g \in R$ for $g \in G$ with $f = \sum_{g\in G} a'_g g$ and $\mathrm{mdeg}(f) \preceq \max_\prec\{\mathrm{mdeg}(a'_g g) \colon g \in G\} \prec \max_\prec\{\mathrm{mdeg}(a_g g) \colon g \in G\}$. Continuing this replacement process if necessary, we find $a_g \in R$ with (23.5) and $\mathrm{mdeg}(f) = \max_\prec\{\mathrm{mdeg}(a_g g)\}$, i.e., there exists a $g \in G$ such that $\mathrm{mdeg}(f) = \mathrm{mdeg}(a_g g)$. Then $\mathrm{lt}(f) = \sum_{\text{all such } g} \mathrm{lt}(a_g)\cdot \mathrm{lt}(g) \in \langle \mathrm{lt}(G)\rangle$. □

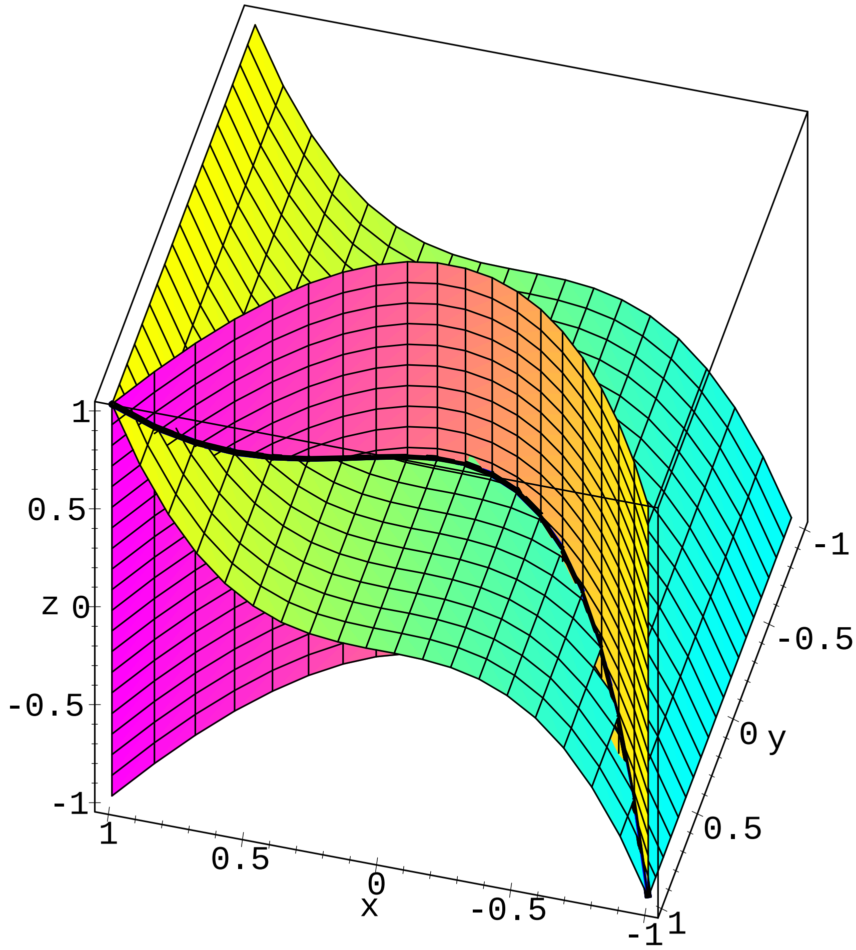


Figure 23.5: The twisted cubic in $\mathbb{R}^3$

EXAMPLE 23.32. *The* twisted cubic C *is the curve in* F^3 *given by* $G = \{y - x^2, z - x^3\}$, *i.e.,* $C = \{(a, a^2, a^3): a \in F\} \subseteq F^3$. *In* $\mathbb{R}^3$, C *is the intersection of the two cylindrical surfaces* $V(y - x^2)$ *and* $V(z - x^3)$, *as illustrated in Figure 23.5.* G *is a GB for* $I = \langle G \rangle$ *with respect to the lexicographic order and* $y \succ z \succ x$, *by Theorem 23.31, since*

$$\begin{aligned} S(y - x^2, z - x^3) &= z(y - x^2) - y(z - x^3) \\ &= yx^3 - zx^2 \\ &= x^3(y - x^2) + (-x^2)(z - x^3) + 0. \end{aligned}$$

EXERCISE 23.33. *Show that* G *is not a GB with respect to* $\prec_{\text{lex}}$ *and* $x \succ y \succ z$.

23.4. Buchberger's algorithm. We want to describe an algorithm for computing a GB, and start by looking at Example 23.22. The basic idea is that, whenever (23.4) is violated, we add the offending S-polynomial to our basis. Let $\prec = \prec_{\text{grlex}}$,

$f_1 = x^3 - 2xy$, and $f_2 = x^2y - 2y^2 + x \in \mathbb{Q}[x, y, z]$. $G = \{f_1, f_2\}$ is not a GB since $S(f_1, f_2) = -x^2$ and $\mathrm{lt}(S(f_1, f_2)) = -x^2 \notin \langle x^3, x^2y\rangle = \langle \mathrm{lt}(G)\rangle$. Now $f_3 = S(f_1, f_2)$ rem $(f_1, f_2) = -x^2$, and $S(f_1, f_2)$ rem $(f_1, f_2, f_3) = 0$.

Next,

$$S(f_1, f_3) = 1 \cdot f_1 - (-x) \cdot f_3 = -2xy,$$

$$S(f_1, f_3) \text{ rem } (f_1, f_2, f_3) = -2xy = f_4,$$

which is adjoined to our basis, so that $S(f_1, f_3)$ rem $(f_1, \ldots, f_4) = 0$. Now

$$S(f_1, f_4) = y \cdot f_1 - (-\frac{1}{2}x^2) \cdot f_4 = -2xy^2 = y \cdot f_4,$$

so that $S(f_1, f_4)$ rem $(f_1, \ldots, f_4) = 0$, and

$$S(f_2, f_3) = 1 \cdot f_2 - (-y)f_3 = -2y^2 + x.$$

After adjoining $f_5 = S(f_2, f_3)$ rem $(f_1, \ldots, f_4) = -2y^2 + x$, one checks that $S(f_i, f_j)$ rem $(f_1, \ldots, f_5) = 0$ for $1 \le i < j \le 5$, and $\{f_1, \ldots, f_5\}$ is a GB.

We now present a simplified version of Buchberger's (1965) algorithm.

ALGORITHM 23.34. *Gröbner basis computation.*
Input: $f_1, \ldots, f_s \in R = F[x_1, \ldots, x_n]$, *and a monomial ordering* $\prec$.
Output: *A GB* $G \subseteq R$ *for* $I = \langle f_1, \ldots, f_s\rangle$, *with* $f_1, \ldots, f_s \in G$.

1. $G = \{f_1, \ldots, f_s\}$.
2. *Repeat steps 3 to 6.*
3. *Set* $\mathcal{S} = \emptyset$.
4. *Do step 5 for all* $g, h \in G$.
5. *Let* $G = \{g_1, \ldots, g_r\}$, *set* $S = S(g, h)$ *rem* $(g_1, \ldots, g_r)$, *and if* $S \neq 0$ *then add* S *to* $\mathcal{S}$: $\mathcal{S} \leftarrow \mathcal{S} \cup \{S\}$.
6. *If* $S = \emptyset$, *then return* G, *else replace* G *by* $G \cup S$.

THEOREM 23.35. *Algorithm 23.34 works correctly.*

PROOF. First we show partial correctness. At any stage of the algorithm, the set G in step 2 is a basis of I and $f_1, \ldots, f_s \in G$, since this is true initially and only elements of I; namely the remainders of S–polynomials on divison by elements of I, are added to G in the course of the algorithm. If the algorithm terminates, the remainders of all the S–polynomials on division by G are zero, and G is a GB by Theorem 23.31.

It remains to show that the algorithm terminates. If G, G' correspond to successive passes through step 2, then $\langle \mathrm{lt}(G')\rangle \supseteq \langle \mathrm{lt}(G)\rangle$. Hence the ideals $\langle \mathrm{lt}(G)\rangle$ from successive passes through step 2 form an ascending chain, which stabilizes by the ACC

(Theorem 23.26). Thus, after a finite number of steps we have $\langle \mathrm{lt}(G') \rangle = \langle \mathrm{lt}(G) \rangle$. We claim that then $G = G'$. Assume the contrary and suppose that the remainder $r = S(g,h)$ rem G is nonzero for some $g, h \in G$. By definition of the remainder, none of its terms is divisible by any of $\mathrm{lt}(G)$, in particular $\mathrm{lt}(r) \notin \langle \mathrm{lt}(G) \rangle$. But $r \in G'$, i.e., $\mathrm{lt}(r) \in \langle \mathrm{lt}(G') \rangle = \langle \mathrm{lt}(G) \rangle$, and we conclude from this contradiction that $G = G'$. □

THEOREM 23.36. *Ideal membership testing via division with remainder works (with Gröbner bases).*

COROLLARY 23.37. *The following problems are solvable:*

1. *ideal containment testing,*

2. *ideal equality testing.*

In general, the GB computed by Buchberger's algorithm is neither minimal nor unique. One can, however, manipulate it so that both properties hold.

LEMMA 23.38. *If G is a GB of $I \subseteq R$, $g \in G$, and $\mathrm{lt}(g) \in \langle \mathrm{lt}(G \setminus \{g\}) \rangle$, then $G \setminus \{g\}$ is a GB of I.*

DEFINITION 23.39. *A subset $G \subseteq R$ is a* minimal *GB for I if it is a GB for I and for all $g \in G$*

1. $\mathrm{lc}(g) = 1$,

2. $\mathrm{lt}(g) \notin \langle \mathrm{lt}(G \setminus \{g\}) \rangle$.

DEFINITION 23.40. *A minimal GB G for $I \subseteq R$ is* reduced *if for all $g \in G$ no monomial of g is in $\langle \mathrm{lt}(G \setminus \{g\}) \rangle$.*

THEOREM 23.41. *Every ideal has a unique reduced GB.*

PROOF. Idea of the proof:

1. Existence: Repeatedly applying Lemma 23.38 if necessary, we may start with a minimal GB G. For $g \in G$, we then replace G by $(G \setminus \{g\}) \cup \{h\}$ where $h = g$ rem $G \setminus \{g\}$.

2. Uniqueness: Suppose that G and G' are reduced GB's of I. Then $\mathrm{lt}(G) = \mathrm{lt}(G')$. Let $g \in G$. Then there exists some $g' \in G'$ such that $\mathrm{lt}(g) = \mathrm{lt}(g')$. Now $0 = g - g'$ rem $G = g - g'$, since no monomial in $g - g'$ is divisible by any of $\mathrm{lt}(G)$.

Both Gaussian elimination and Euclid's algorithm for gcds in $F[x]$ are special cases of Buchberger's algorithm.

23.5. Applications. Algebra provides a powerful general language which can express a wide range of problems. Then a general–purpose solution strategy like Gröbner bases is available for the solution of these problems. Of course, this approach has its limitations, e.g., with regards to efficiency. We now give four sample applications. The first two are rather naturally formulated as geometric problems, while the next two are computer science questions whose geometric formulation is not obvious (but easy to produce, once we set our mind on it).

Implicitization. Let $f_1, \ldots, f_n \in F[t_1, \ldots, t_m]$, and suppose that the affine algebraic variety $V \subseteq F^n$ is given in parametrized form by

$$\begin{aligned} x_1 &= f_1(t_1, \ldots, t_m), \\ &\vdots \\ x_n &= f_n(t_1, \ldots, t_m), \end{aligned}$$

so that V is "explicitly" described as $V = \{a \in F^n \colon \exists b \in F^m \; a = (f_1(b), \ldots, f_n(b))\}$. The task is now to find polynomials $g_1, \ldots, g_s \in F[x_1, \ldots, x_n]$ such that V has the "implicit" representation $V = V(I)$, where $I = \langle g_1, \ldots, g_s \rangle$.

EXAMPLE 23.42. *The twisted cubic C from Example 23.32 can be parametrized by*

$$x = t, \quad y = t^2, \quad z = t^3.$$

An implicitization for C is $g_1 = y - x^2, g_2 = z - x^3$. The curve is illustrated in Figures 23.5 and 23.6. The latter corresponds to the explicit representation of C, while the former depicts the implicit form as the intersection of the two surfaces defined by g_1 and g_2.

EXAMPLE 23.43. *Let $V \subseteq F^3$ be parametrized by $x = t^2$, $y = t^3$, $z = t^4$. Then an implicitization for V is $g_1 = z - x^2, g_2 = y^2 - x^3$.*

To solve the implicitation problem, we let $J = \langle x_1 - f_1, \ldots, x_n - f_n \rangle \subseteq F[t_1, \ldots, t_m, x_1, \ldots, x_n]$, order the variables so that $t_1 \succ \cdots \succ t_m \succ x_1 \succ \cdots \succ x_n$, and compute a GB G for J with respect to $\prec_{\text{lex}}$. Then some $g \in G$ will depend only on $x_1, \ldots, x_n$. These are candidates for the g_i's.

In Examples 23.42 and 23.43, the reduced Gröbner bases for J w.r.t. $t \succ z \succ y \succ x$ are

$$\{t - x, z - x^3, y - x^2\}$$

and

$$\{t^2 - x, ty - x^2, tx - y, z - x^2, y^2 - x^3\},$$

respectively.

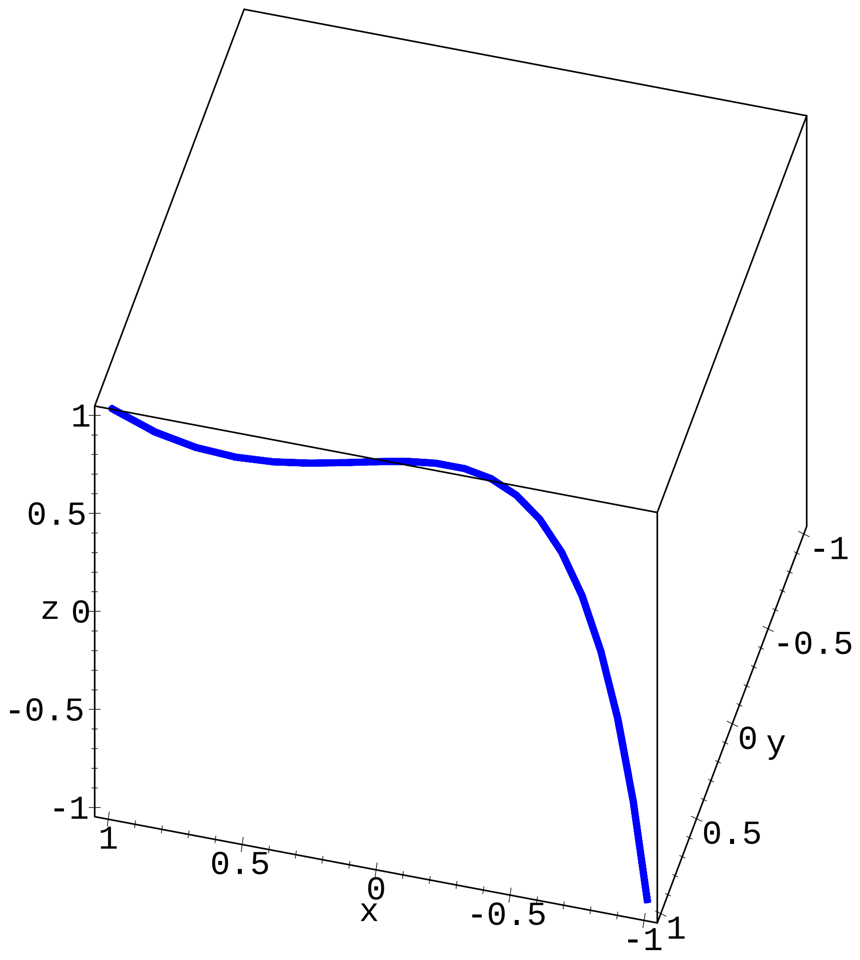


Figure 23.6: The twisted cubic

Solving systems of polynomial equations. This is the most natural application of GBs, and they were originally invented for this. In general, we have polynomials $f_1, \ldots, f_m \in F[x_1, \ldots, x_n]$ and want to answer questions about the corresponding *affine variety*

$$V = \{a \in F^n : f_1(a) = \cdots = f_m(a) = 0\}.$$

Examples of such questions are: Is $V \neq \emptyset$? If so, please find a point on V. Given another polynomial $g \in F[x_1, \ldots, x_n]$, is $g(a) = 0$ for all $a \in V$?

The structure of these varieties is the subject of *algebraic geometry*, a deep mathematical theory whose most recent success is the solution of the 350–year old Fermat problem. We give a few pointers to the recent exciting work in *computational algebraic geometry* at the end of this section.

One of the simplest situations occurs when V is *zero–dimensional*, i.e., it consists only of finitely many isolated points. An example is the intersection of two plane curves. We saw in section 9 how to use resultants for this problem: one first finds a polynomial $r \in F[x]$ in one variable which the x–coordinates of all intersection points satisfy, and then has to solve, for each such x–coordinate, the original bivariate equations (or their gcd)). This resultant approach has a (non–trivial) generalization to higher dimensions.

GBs provide an alternate approach. For a zero–dimensional problem, the GB will often include some polynomials only containing x_n, some only with x_{n-1}, x_n, and so on. These can then be solved by back substitution: first for x_n, then for x_{n-1}, and so on.

Application areas such as robot kinematics and motion planning provide many such interesting geometric problems. We now look at a slightly different type of example, namely the following nonlinear optimization problem: Determine the maximum of $f = x^3 + 2xyz - z^2 \in \mathbb{R}[x, y, z]$ on the unit sphere $S = \{x^2 + y^2 + z^2 - 1 = 0\} \subseteq \mathbb{R}^3$.

In numerical analysis, such a problem is solved with the aid of *Lagrange multipliers*: if $\nabla f = (f_x, f_y, f_z)$ and $\nabla g = (g_x, g_y, g_z)$ are the Jacobians of f and $g = x^2 + y^2 + z^2 - 1$, respectively, then the equality $\nabla f = \lambda \nabla g$ holds at a local maximum or minimum of f on S for some $\lambda \in \mathbb{R}$. In our example, this leads to the following system of polynomial equations:

$$\begin{aligned} 3x^2 + 2yz &= 2\lambda x \\ 2xz &= 2\lambda y \\ 2xy - 2z &= 2\lambda z \\ x^2 + y^2 + z^2 &= 1 \end{aligned}$$

One can solve this system by numerical methods, e.g., Newton's method. If f, g have rational coefficients, Gröbner bases can in some cases help to find *exact* solutions. This works by computing a GB with respect to $\prec_{\text{lex}}$ with $\lambda \succ x \succ y \succ z$ for the ideal $I \subseteq \mathbb{Q}[\lambda, x, y, z]$ generated by the polynomial equations.

Buchberger's algorithm with respect to lexicographic order often gives a basis for I in which one can eliminate the variables one after the other by "back substitution". In our example, the reduced GB for I consists of the eight polynomials

$$\lambda - \frac{3}{2}x - \frac{3}{2}yz - \frac{167616}{3835}z^6 + \frac{36717}{590}z^4 - \frac{134419}{7670}z^2,$$

$$x^2 + y^2 + z^2 - 1, \quad xy - \frac{19584}{3835}z^5 + \frac{1999}{295}z^3 - \frac{6403}{3835}z,$$

$$xz + yz^2 - \frac{1152}{3835}z^5 - \frac{108}{295}z^3 + \frac{2556}{3835}z, \quad y^3 + yz^2 - y - \frac{9216}{3835}z^5 + \frac{906}{295}z^3 - \frac{2562}{3835}z,$$

$$y^2z - \frac{6912}{3835}z^5 + \frac{827}{295}z^3 - \frac{3839}{3835}z, \quad yz^3 - yz - \frac{576}{59}z^6 + \frac{1605}{118}z^4 - \frac{453}{118}z^2,$$

$$z^7 - \frac{1763}{1152}z^5 + \frac{655}{1152}z^3 - \frac{11}{288}z.$$

It contains one polynomial in all four variables, three polynomials in x, y, z, three in y and z, and one polynomial in z only. One then solves the last equation for z, substitutes the values in the remaining equations, then solves for y, and so on. Finally, this yields all solutions of the system of equations, provided there are only finitely many.

23.6. Cost of Buchberger's algorithm. Very expensive:

– many polynomials
– high degree
– large coefficients

Difficult to analyze the algorithm.
Results for ≤ 3 variables.

Mayr & Meyer (1982): ideal membership problem over $\mathbb{Q}$ is $\mathcal{EXPSPACE}$–complete.

Bayer & Stillman (1988): Buchberger's algorithm with $\prec_{\mathrm{grevlex}}$ is polynomial in the input length and the Castelnuovo regularity $c(I)$. Intuitively, $c(I)$ is small for natural geometric problems.

Kühnle & Mayr (1996): Gröbner basis computation is in $\mathcal{EXPSPACE}$.

Practical recommendation: use $\prec_{\mathrm{grevlex}}$!

References

L. M. Adleman, On breaking generalized knapsack public key cryptosystems. In *Proc. of the 15th ACM Symp. on Theory of Computing*, 1983, 402–412.

T. M. Apostol, *Introduction to Analytic Number Theory.* Springer-Verlag, New York NY, 1976.

D. Bayer and M. Stillman, On the complexity of computing syzygies. *J. Symbolic Computation* **6** (1988), 135–147.

J. Boyar, Inferring sequences produced by pseudorandom number generators. *J. A.C.M.* **36** (1989), 129–141.

E. Brieskorn, *Lineare Algebra und analytische Geometrie*, vol. II. Vieweg & Sohn, Wiesbaden / Braunschweig, 1985.

B. Buchberger, *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal.* PhD thesis, University of Innsbruck, Austria, 1965.

B. Chor and R. Rivest, A knapsack type public key cryptosystem based on arithmetic in finite fields. In *Advances in Cryptology; Proceedings of CRYPTO 84*, New York NY, 1985, Springer, 54–65.

G. E. Collins, Factoring univariate integral polynomials in polynomial average time. In *Proc. EUROSAM 1979*, 1979, 316–329.

D. Cox, L. Little, and D. O'Shea, *Ideals, varieties and algorithms: an introduction to computational algebraic geometry and commutative algebra.* Springer, 1992.

W. Diffie and M. Hellman, New directions in cryptography. *IEEE Trans. Inform. Theory* **22** (1976), 644–654.

J. D. Dixon, Asymptotically fast factorization of integers. *Math. Comp.* **36** (1981), 255–260.

A. Frieze, J. Hastad, R. Kannan, J.C. Lagarias, and A. Shamir, Reconstructing truncated integer variables satisfying linear congruences. *SIAM J. Comput.* **17** (1988), 262–280.

M. R. Garey and D. S. Johnson, *Computers and intractability: A guide to the theory of NP-Completeness.* W. H. Freeman, San Francisco CA, 1979.

J. von zur Gathen, Hensel and Newton methods in valuation rings. *Math. Comp.* **42** (1984a), 637–661.

J. von zur Gathen, Parallel algorithms for algebraic problems. *SIAM J. Comput.* **13** (1984b), 802–824.

R. M. Goodman and A. J. McAuley, A new trapdoor knapsack public key cryptosystem. In *Advances in Cryptology, Proc. Eurocrypt 84*, New York NY, 1985, Springer, 150–158.

M. Grötschel, L. Lovász, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*. Algorithms and Combinatorics **2**. Springer, Berlin, Heidelberg, 2nd edition, 1993.

R. Kannan, Algorithmic geometry of numbers. *Ann. Rev. Comput. Sci.* **2** (1987), 231–267.

D. E. Knuth, *The Art of Computer Programming, Vol.2, Seminumerical Algorithms*. Addison-Wesley, Reading MA, 2 edition, 1981.

D. E. Knuth and L. Trabb Pardo, Analysis of a simple factorization algorithm. *Theoretical Computer Science* **3** (1976), 321–348.

K. Kühnle and E. W. Mayr, Exponential space computation of Gröbner bases. In *Proc. ISSAC '96, Zürich, Switzerland*. ACM press, 1996, 63–71.

J. C. Lagarias, The computational complexity of simultaneous Diophantine approximation problems. *SIAM J. Comput.* **14** (1985), 196–209.

J. C. Lagarias, Pseudorandom number generators in cryptography and number theory. In *Cryptology and Computational Number Theory*, ed. C. Pomerance, vol. 42 of *Proceedings of symposia in applied mathematics*. American Mathematical Society, 1990, 115–144.

A. K. Lenstra, H. W. Lenstra, and L. Lovász, Factoring polynomials with rational coefficients. *Math. Ann.* **261** (1982), 515–534.

S. C. Lu and L. N. Lee, A simple and effective public-key cryptosystem. *COMSAT Tech. Rev.* (1979), 15–24.

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

R. C. Merkle and M. E. Hellman, Hiding information and signatures in trapdoor knapsacks. *IEEE Trans. Informat. Theory* **24**(5) (1978), 525–530.

M. Mignotte, *Mathématiques pour le calcul formel*. Presses Universitaires de France, 1989.

H. Niederreiter, Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory* **15** (1986), 159–166.

A. M. Odlyzko and H. J. J. te Riele, Disproof of the Mertens conjecture. *Journal für die Reine und Angewandte Mathematik* **357** (1985), 138–160.

H. Ong, C. P. Schnorr, and A. Shamir, An efficient signature scheme based on quadratic equations. In *Proceedings of 16th ACM Symp. on Theory of Computing*, 1984, 208–216.

J. M. Pollard, A Monte Carlo method for factorisation. *BIT* **15** (1975), 331–334.

J. M. Pollard and C. P. Schnorr, An efficient solution of the congruence $x^2 + ky^2 = m(\mathrm{mod} n)$. *IEEE Trans. Informat. Theory* **IT-33**(5) (1987), 702–709.

J. B. Rosser and L. Schoenfeld, Approximate formulas for some functions of prime numbers. *Ill. J. Math.* **6** (1962), 64–94.

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

A. Shamir, A polynomial time algorithm for breaking the basic Merkle-Hellman cryptosystem. *IEEE Trans. Informat. Theory* **IT-30**(5) (1984), 699–704.

A. Shamir and R. Zippel, On the security of the Merkle-Hellman cryptographic scheme. *IEEE Trans. Informat. Theory* **26**(3) (1980), 339–340.

D. Wiedemann, Solving sparse linear equations over finite fields. *IEEE Trans. Inf. Theory* **32** (1986), 54–62.

H. Zassenhaus, On Hensel factorization, I. *J. Number Theory* **1** (1969), 291–311.

Index

The index entries should be understood as follows:

Page numbers are underlined (e.g.10) when they represent the definition or the main source of information about the index entry. A page number is given in italics (e.g.*10*) if it refers to a table or a figure.