Factoring Polynomials with Rational Coefficients.
Lenstra, A.K.; Lenstra, H.W. jr.; Lovász, L.
pp. 515 - 534

NIEDERSÄCHSISCHE STAATS- UND UNIVERSITÄTSBIBLIOTHEK GÖTTINGEN

Terms and Conditions

The Göttingen State and University Library provides access to digitized documents strictly for noncommercial educational, research and private purposes and makes no warranty with regard to their use for other purposes. Some of our collections are protected by copyright. Publication and/or broadcast in any form (including electronic) requires prior written permission from the Goettingen State- and University Library.
Each copy of any part of this document must contain there Terms and Conditions. With the usage of the library's online system to access or download a digitized document you accept there Terms and Conditions.
Reproductions of material on the web site may not be made for or donated to other repositories, nor may be further reproduced without written permission from the Goettingen State- and University Library
For reproduction requests and permissions, please contact us. If citing materials, please give proper attribution of the source.

Contact:

Niedersächsische Staats- und Universitätsbibliothek
Digitalisierungszentrum
37070 Goettingen
Germany
Email: gdz@www.sub.uni-goettingen.de

Purchase a CD-ROM

The Goettingen State and University Library offers CD-ROMs containing whole volumes / monographs in PDF for Adobe Acrobat. The PDF-version contains the table of contents as bookmarks, which allows easy navigation in the document. For availability and pricing, please contact:
Niedersaechisische Staats- und Universitaetsbibliothek Goettingen - Digitalisierungszentrum
37070 Goettingen, Germany, Email: gdz@www.sub.uni-goettingen.de

Math. Ann. 261, 515–534 (1982)

Mathematische Annalen
© Springer-Verlag 1982

Factoring Polynomials with Rational Coefficients

A. K. Lenstra[1], H. W. Lenstra, Jr.[2], and L. Lovász[3]

1 Mathematisch Centrum, Kruislaan 413, NL-1098 SJ Amsterdam, The Netherlands
2 Mathematisch Instituut, Universiteit van Amsterdam, Roetersstraat 15, NL-1018 WB Amsterdam, The Netherlands
3 Bolyai Institute, A. József University, Aradi vértanúk tere 1, H-6720 Szeged, Hungary

In this paper we present a polynomial-time algorithm to solve the following problem: given a non-zero polynomial $f \in \mathbb{Q}[X]$ in one variable with rational coefficients, find the decomposition of f into irreducible factors in $\mathbb{Q}[X]$. It is well known that this is equivalent to factoring *primitive* polynomials $f \in \mathbb{Z}[X]$ into irreducible factors in $\mathbb{Z}[X]$. Here we call $f \in \mathbb{Z}[X]$ primitive if the greatest common divisor of its coefficients (the *content* of f) is 1.

Our algorithm performs well in practice, cf. [8]. Its running time, measured in bit operations, is $O(n^{12} + n^9(\log|f|)^3)$. Here $f \in \mathbb{Z}[X]$ is the polynomial to be factored, $n = \deg(f)$ is the degree of f, and

$$\left|\sum_i a_i X^i\right| = \left(\sum_i a_i^2\right)^{1/2}$$

for a polynomial $\sum_i a_i X^i$ with real coefficients a_i.

An outline of the algorithm is as follows. First we find, for a suitable small prime number p, a p-adic irreducible factor h of f, to a certain precision. This is done with Berlekamp's algorithm for factoring polynomials over small finite fields, combined with Hensel's lemma. Next we look for the irreducible factor h_0 of f in $\mathbb{Z}[X]$ that is divisible by h. The condition that h_0 is divisible by h means that h_0 belongs to a certain lattice, and the condition that h_0 divides f implies that the coefficients of h_0 are relatively small. It follows that we must look for a "small" element in that lattice, and this is done by means of a basis reduction algorithm. It turns out that this enables us to determine h_0. The algorithm is repeated until all irreducible factors of f have been found.

The basis reduction algorithm that we employ is new, and it is described and analysed in Sect. 1. It improves the algorithm given in a preliminary version of [9, Sect. 3]. At the end of Sect. 1 we briefly mention two applications of the new algorithm to diophantine approximation.

The connection between factors of f and reduced bases of a lattice is treated in detail in Sect. 2. The theory presented here extends a result appearing in [8, Theorem 2]. It should be remarked that the latter result, which is simpler to prove, would in principle have sufficed for our purpose.

0025-5831/82/0261/0515/$04.00

Section 3, finally, contains the description and the analysis of our algorithm for factoring polynomials.

It may be expected that other irreducibility tests and factoring methods that depend on diophantine approximation (Cantor [3], Ferguson and Forcade [5], Brentjes [2, Sect. 4A], and Zassenhaus [16]) can also be made into polynomial-time algorithms with the help of the basis reduction algorithm presented in Sect. 1.

Splitting an arbitrary non-zero polynomial $f \in \mathbb{Z}[X]$ into its *content* and its *primitive part*, we deduce from our main result that the problem of factoring such a polynomial is polynomial-time reducible to the problem of factoring positive integers. The same fact was proved by Adleman and Odlyzko [1] under the assumption of several deep and unproved hypotheses from number theory.

The generalization of our result to algebraic number fields and to polynomials in several variables is the subject of future publications.

1. Reduced Bases for Lattices

Let n be a positive integer. A subset L of the n-dimensional real vector space $\mathbb{R}^n$ is called a *lattice* if there exists a basis $b_1, b_2, \ldots, b_n$ of $\mathbb{R}^n$ such that

$$L = \sum_{i=1}^{n} \mathbb{Z} b_i = \left\{ \sum_{i=1}^{n} r_i b_i : r_i \in \mathbb{Z} (1 \leqq i \leqq n) \right\}.$$

In this situation we say that $b_1, b_2, \ldots, b_n$ form a *basis* for L, or that they *span* L. We call n the *rank* of L. The *determinant* $d(L)$ of L is defined by

$$d(L) = |\det(b_1, b_2, \ldots, b_n)|, \tag{1.1}$$

the b_i being written as column vectors. This is a positive real number that does not depend on the choice of the basis [4, Sect. I.2].

Let $b_1, b_2, \ldots, b_n \in \mathbb{R}^n$ be linearly independent. We recall the Gram-Schmidt orthogonalization process. The vectors b_i^* $(1 \leqq i \leqq n)$ and the real numbers μ_{ij} $(1 \leqq j < i \leqq n)$ are inductively defined by

$$b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{ij} b_j^*, \tag{1.2}$$

$$\mu_{ij} = (b_i, b_j^*)/(b_j^*, b_j^*), \tag{1.3}$$

where $(,)$ denotes the ordinary inner product on $\mathbb{R}^n$. Notice that b_i^* is the projection of b_i on the orthogonal complement of $\sum_{j=1}^{i-1} \mathbb{R} b_j$, and that $\sum_{j=1}^{i-1} \mathbb{R} b_j = \sum_{j=1}^{i-1} \mathbb{R} b_j^*$, for $1 \leqq i \leqq n$. It follows that $b_1^*, b_2^*, \ldots, b_n^*$ is an orthogonal basis of $\mathbb{R}^n$.

In this paper, we call a basis $b_1, b_2, \ldots, b_n$ for a lattice L *reduced* if

$$|\mu_{ij}| \leqq 1/2 \quad \text{for} \quad 1 \leqq j < i \leqq n \tag{1.4}$$

and

$$|b_i^* + \mu_{i\,i-1} b_{i-1}^*|^2 \geqq \tfrac{3}{4} |b_{i-1}^*|^2 \quad \text{for} \quad 1 < i \leqq n, \tag{1.5}$$

where $||$ denotes the ordinary Euclidean length. Notice that the vectors $b_i^* + \mu_{i\,i-1} b_{i-1}^*$ and b_{i-1}^* appearing in (1.5) are the projections of b_i and b_{i-1} on the orthogonal complement of $\sum_{j=1}^{i-2} \mathbb{R} b_j$. The constant $\frac{3}{4}$ in (1.5) is arbitrarily chosen, and may be replaced by any fixed real number y with $\frac{1}{4} < y < 1$.

(1.6) **Proposition.** *Let $b_1, b_2, \ldots, b_n$ be a reduced basis for a lattice L in $\mathbb{R}^n$, and let $b_1^*, b_2^*, \ldots, b_n^*$ be defined as above. Then we have*

$$|b_j|^2 \leqq 2^{i-1} \cdot |b_i^*|^2 \quad \text{for} \quad 1 \leqq j \leqq i \leqq n, \tag{1.7}$$

$$d(L) \leqq \prod_{i=1}^{n} |b_i| \leqq 2^{n(n-1)/4} \cdot d(L), \tag{1.8}$$

$$|b_1| \leqq 2^{(n-1)/4} \cdot d(L)^{1/n}. \tag{1.9}$$

Remark. If $\frac{3}{4}$ in (1.5) is replaced by y, with $\frac{1}{4} < y < 1$, then the powers of 2 appearing in (1.7), (1.8) and (1.9) must be replaced by the same powers of $4/(4y-1)$.

Remark. From (1.8) we see that a reduced basis is also reduced in the sense of [9, (7)].

Proof of (1.6). From (1.5) and (1.4) we see that

$$|b_i^*|^2 \geqq (\tfrac{3}{4} - \mu_{i\,i-1}^2) \cdot |b_{i-1}^*|^2 \geqq \tfrac{1}{2} \cdot |b_{i-1}^*|^2$$

for $1 < i \leqq n$, so by induction

$$|b_j^*|^2 \leqq 2^{i-j} \cdot |b_i^*|^2 \quad \text{for} \quad 1 \leqq j \leqq i \leqq n.$$

From (1.2) and (1.4) we now obtain

$$\begin{aligned} |b_i|^2 &= |b_i^*|^2 + \sum_{j=1}^{i-1} \mu_{ij}^2 |b_j^*|^2 \\ &\leqq |b_i^*|^2 + \sum_{j=1}^{i-1} \tfrac{1}{4} 2^{i-j} |b_i^*|^2 \\ &= (1 + \tfrac{1}{4}(2^i - 2)) \cdot |b_i^*|^2 \\ &\leqq 2^{i-1} \cdot |b_i^*|^2. \end{aligned}$$

It follows that

$$|b_j|^2 \leqq 2^{j-1} \cdot |b_j^*|^2 \leqq 2^{i-1} \cdot |b_i^*|^2$$

for $1 \leqq j \leqq i \leqq n$. This proves (1.7).

From (1.1), (1.2) it follows that

$$d(L) = |\det(b_1^*, b_2^*, \ldots, b_n^*)|$$

and therefore, since the b_i^* are pairwise orthogonal

$$d(L) = \prod_{i=1}^{n} |b_i^*|.$$

From $|b_i^*| \leqq |b_i|$ and $|b_i| \leqq 2^{(i-1)/2} \cdot |b_i^*|$ we now obtain (1.8). Putting $j = 1$ in (1.7) and taking the product over $i = 1, 2, \ldots, n$ we find (1.9). This proves (1.6).

Remark. Notice that the proof of the inequality

$$d(L) \leqq \prod_{i=1}^{n} |b_i| \tag{1.10}$$

did not require the basis to be reduced. This is *Hadamard's inequality.*

(1.11) **Proposition.** *Let $L \subset \mathbb{R}^n$ be a lattice with reduced basis $b_1, b_2, \ldots, b_n$. Then*

$$|b_1|^2 \leqq 2^{n-1} \cdot |x|^2$$

for every $x \in L$, $x \neq 0$.

Proof. Write $x = \sum_{i=1}^{n} r_i b_i = \sum_{i=1}^{n} r_i' b_i^*$ with $r_i \in \mathbb{Z}$, $r_i' \in \mathbb{R}$ $(1 \leqq i \leqq n)$. If i is the largest index with $r_i \neq 0$ then $r_i' = r_i$, so

$$|x|^2 \geqq r_i'^2 \cdot |b_i^*|^2 \geqq |b_i^*|^2 .$$

By (1.7), we have $|b_1|^2 \leqq 2^{i-1} \cdot |b_i^*|^2 \leqq 2^{n-1} \cdot |b_i^*|^2$. This proves (1.11).

(1.12) **Proposition.** *Let $L \subset \mathbb{R}^n$ be a lattice with reduced basis $b_1, b_2, \ldots, b_n$. Let $x_1, x_2, \ldots, x_t \in L$ be linearly independent. Then we have*

$$|b_j|^2 \leqq 2^{n-1} \cdot \max\{|x_1|^2, |x_2|^2, \ldots, |x_t|^2\}$$

for $j = 1, 2, \ldots, t$.

Proof. Write $x_j = \sum_{i=1}^{n} r_{ij} b_i$ with $r_{ij} \in \mathbb{Z}$ $(1 \leqq i \leqq n)$ for $1 \leqq j \leqq t$. For fixed j, let $i(j)$ denote the largest i for which $r_{ij} \neq 0$. Then we have, by the proof of (1.11)

$$|x_j|^2 \geqq |b_{i(j)}^*|^2 \tag{1.13}$$

for $1 \leqq j \leqq t$. Renumber the x_j such that $i(1) \leqq i(2) \leqq \ldots \leqq i(t)$. We claim that $j \leqq i(j)$ for $1 \leqq j \leqq t$. If not, then $x_1, x_2, \ldots, x_j$ would all belong to $\mathbb{R}b_1 + Rb_2 + \ldots + \mathbb{R}b_{j-1}$, a contradiction with the linear independence of $x_1, x_2, \ldots, x_t$. From $j \leqq i(j)$ and (1.7) we obtain, using (1.13):

$$|b_j|^2 \leqq 2^{i(j)-1} \cdot |b_{i(j)}^*|^2 \leqq 2^{n-1} \cdot |b_{i(j)}^*|^2 \leqq 2^{n-1} \cdot |x_j|^2$$

for $j = 1, 2, \ldots, t$. This proves (1.12).

Remark. Let $\lambda_1, \lambda_2, \ldots, \lambda_n$ denote the successive minima of $|\ |^2$ on L, see [4, Chap. VIII], and let $b_1, b_2, \ldots, b_n$ be a reduced basis for L. Then (1.7) and (1.12) easily imply that

$$2^{1-i} \lambda_i \leqq |b_i|^2 \leqq 2^{n-1} \lambda_i \quad \text{for} \quad 1 \leqq i \leqq n,$$

so $|b_i|^2$ is a reasonable approximation of λ_i.

(1.14) *Remark.* Notice that the number 2^{n-1} may in (1.11) be replaced by $\max\{|b_1|^2/|b_i^*|^2 : 1 \leqq i \leqq n\}$ and in (1.12) by $\max\{|b_j|^2/|b_i^*|^2 : 1 \leqq j \leqq i \leqq n\}$.

(1.15) We shall now describe an algorithm that transforms a given basis $b_1, b_2, \ldots, b_n$ for a lattice L into a reduced one. The algorithm improves the

algorithm given in a preliminary version of [9, Sect. 3]. Our description incorporates an additional improvement due to J. J. M. Cuppen, reducing our running time estimates by a factor n.

To initialize the algorithm we compute b_i^* $(1 \leqq i \leqq n)$ and μ_{ij} $(1 \leqq j < i \leqq n)$ using (1.2) and (1.3). In the course of the algorithm the vectors $b_1, b_2, \ldots, b_n$ will be changed several times, but always in such a way that they form a basis for L. After every change of the b_i we shall update the b_i^* and μ_{ij} in such a way that (1.2) and (1.3) remain valid.

At each step of the algorithm we shall have a current subscript $k \in \{1, 2, \ldots, n+1\}$. We begin with $k=2$.

We shall now iterate a sequence of steps that starts from, and returns to, a situation in which the following conditions are satisfied:

$$|\mu_{ij}| \leqq \tfrac{1}{2} \quad \text{for} \quad 1 \leqq j < i < k, \tag{1.16}$$

$$|b_i^* + \mu_{i\,i-1} b_{i-1}^*|^2 \geqq \tfrac{3}{4} |b_{i-1}^*|^2 \quad \text{for} \quad 1 < i < k. \tag{1.17}$$

These conditions are trivially satisfied if $k=2$.

In the above situation one proceeds as follows. If $k=n+1$ then the basis is reduced, and the algorithm terminates. Suppose now that $k \leqq n$. Then we first achieve that

$$|\mu_{k\,k-1}| \leqq \tfrac{1}{2} \quad \text{if} \quad k > 1. \tag{1.18}$$

If this does not hold, let r be the integer nearest to $\mu_{k\,k-1}$, and replace b_k by $b_k - r b_{k-1}$. The numbers μ_{kj} with $j < k-1$ are then replaced by $\mu_{kj} - r\mu_{k-1\,j}$, and $\mu_{k\,k-1}$ by $\mu_{k\,k-1} - r$. The other μ_{ij} and all b_i^* are unchanged. After this change (1.18) holds.

Next we distinguish two cases.

Case 1. Suppose that $k \geqq 2$ and

$$|b_k^* + \mu_{k\,k-1} b_{k-1}^*|^2 < \tfrac{3}{4} |b_{k-1}^*|^2. \tag{1.19}$$

Then we interchange b_{k-1} and b_k, and we leave the other b_i unchanged. The vectors b_{k-1}^* and b_k^* and the numbers $\mu_{k\,k-1}$, $\mu_{k-1\,j}$, μ_{kj}, $\mu_{i\,k-1}$, μ_{ik}, for $j < k-1$ and for $i > k$, have now to be replaced. This is done by formulae that we give below. The most important one of these changes is that b_{k-1}^* is replaced by $b_k^* + \mu_{k\,k-1} b_{k-1}^*$; so the new value of $|b_{k-1}^*|^2$ is less than $\frac{3}{4}$ times the old one. These changes being made, we replace k by $k-1$. Then we are in the situation described by (1.16) and (1.17), and we proceed with the algorithm from there.

Case 2. Suppose that $k=1$ or

$$|b_k^* + \mu_{k\,k-1} b_{k-1}^*|^2 \geqq \tfrac{3}{4} |b_{k-1}^*|^2. \tag{1.20}$$

In this case we first achieve that

$$|\mu_{kj}| \leqq \tfrac{1}{2} \quad \text{for} \quad 1 \leqq j \leqq k-1. \tag{1.21}$$

[For $j = k-1$ this is already true, by (1.18).] If (1.21) does not hold, let l be the largest index $< k$ with $|\mu_{kl}| > \frac{1}{2}$, let r be the integer nearest to μ_{kl}, and replace b_k by

$b_k - rb_l$. The numbers μ_{kj} with $j<l$ are then replaced by $\mu_{kj}-r\mu_{lj}$, and μ_{kl} by $\mu_{kl}-r$; the other μ_{ij} and all b_i^* are unchanged. This is repeated until (1.21) holds.

Next we replace k by $k+1$. Then we are in the situation described by (1.16) and (1.17), and we proceed with the algorithm from there.

Notice that in the case $k=1$ we have done no more than replacing k by 2.

This finishes the description of the algorithm. Below we shall prove that the algorithm terminates.

(1.22) For the sake of completeness we now give the formulae that are needed in case 1. Let $b_1, b_2, \ldots, b_n$ be the current basis and b_i^*, μ_{ij} as in (1.2) and (1.3). Let k be the current subscript for which (1.16), (1.17), (1.18), and (1.19) hold. By c_i, c_i^*, and v_{ij} we denote the vectors and numbers that will replace b_i, b_i^*, and μ_{ij}, respectively. The new basis $c_1, c_2, \ldots, c_n$ is given by

$$c_{k-1}=b_k, \quad c_k=b_{k-1}, \quad c_i=b_i \quad \text{for} \quad i\neq k-1,k.$$

Since c_{k-1}^* is the projection of b_k on the orthogonal complement of $\sum_{j=1}^{k-2} \mathbb{R}b_j$ we have, as announced:

$$c_{k-1}^*=b_k^*+\mu_{k\,k-1}b_{k-1}^*$$

[cf. the remark after (1.5)]. To obtain c_k^* we must project b_{k-1}^* on the orthogonal complement of $\mathbb{R}c_{k-1}^*$. That leads to

$$\begin{aligned} v_{k\,k-1} &= (b_{k-1}^*, c_{k-1}^*)/(c_{k-1}^*, c_{k-1}^*) \\ &= \mu_{k\,k-1}|b_{k-1}^*|^2/|c_{k-1}^*|^2, \\ c_k^* &= b_{k-1}^* - v_{k\,k-1}c_{k-1}^*. \end{aligned}$$

For $i\neq k-1, k$ we have $c_i^*=b_i^*$. Let now $i>k$. To find $v_{i\,k-1}$ and v_{ik} we substitute

$$\begin{aligned} b_{k-1}^* &= v_{k\,k-1}c_{k-1}^*+c_k^* \\ b_k^* &= (1-\mu_{k\,k-1}v_{k\,k-1})c_{k-1}^*-\mu_{k\,k-1}c_k^* \\ &= (|b_k^*|^2/|c_{k-1}^*|^2)\cdot c_{k-1}^*-\mu_{k\,k-1}c_k^* \end{aligned}$$

in $b_i=b_i^*+\sum_{j=1}^{i-1}\mu_{ij}b_j^*$. That yields

$$\begin{aligned} v_{i\,k-1} &= \mu_{i\,k-1}v_{k\,k-1}+\mu_{ik}|b_k^*|^2/|c_{k-1}^*|^2 \\ v_{ik} &= \mu_{i\,k-1}-\mu_{ik}\mu_{k\,k-1}. \end{aligned}$$

Finally, we have

$$v_{k-1\,j}=\mu_{kj}, \quad v_{kj}=\mu_{k-1\,j}$$

for $1\leqq j<k-1$, and $v_{ij}=\mu_{ij}$ if $1\leqq j<i\leqq n$, $\{i,j\}\cap\{k-1,k\}=\emptyset$.

We remark that after the initialization stage of the algorithm it is not necessary to keep track of the vectors b_i^*. It suffices to keep track of the numbers $|b_i^*|^2$, in addition to μ_{ij} and the vectors b_i. Notice that $|c_k^*|^2=|b_{k-1}^*|^2\cdot|b_k^*|^2/|c_{k-1}^*|^2$ in the above, and that the left hand side of (1.19), (1.20) equals $|b_k^*|^2+\mu_{k\,k-1}^2|b_{k-1}^*|^2$.

The entire algorithm is represented in Fig. 1, in which $B_i=|b_i^*|^2$.

$$\left.\begin{array}{l} b_i^* := b_i; \\ \left.\begin{array}{l} \mu_{ij} := (b_i, b_j^*)/B_j; \\ b_i^* := b_i^* - \mu_{ij} b_j^* \end{array}\right\} \text{ for } j = 1, 2, \ldots, i-1; \\ B_i := (b_i^*, b_i^*) \end{array}\right\} \text{ for } i = 1, 2, \ldots, n;$$

$k := 2;$

(1) perform (*) for $l = k-1$;

if $B_k < (\frac{3}{4} - \mu_{k\,k-1}^2) B_{k-1}$, go to (2);

perform (*) for $l = k-2, k-3, \ldots, 1$;

if $k = n$, terminate;

$k := k+1$;

go to (1);

(2) $\mu := \mu_{k\,k-1}$; $B := B_k + \mu^2 B_{k-1}$; $\mu_{k\,k-1} := \mu B_{k-1}/B$;

$B_k := B_{k-1} B_k / B$; $B_{k-1} := B$;

$$\begin{pmatrix} b_{k-1} \\ b_k \end{pmatrix} := \begin{pmatrix} b_k \\ b_{k-1} \end{pmatrix};$$

$$\begin{pmatrix} \mu_{k-1\,j} \\ \mu_{kj} \end{pmatrix} := \begin{pmatrix} \mu_{kj} \\ \mu_{k-1\,j} \end{pmatrix} \text{ for } j = 1, 2, \ldots, k-2;$$

$$\begin{pmatrix} \mu_{i\,k-1} \\ \mu_{ik} \end{pmatrix} := \begin{pmatrix} 1 & \mu_{k\,k-1} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & -\mu \end{pmatrix} \begin{pmatrix} \mu_{i\,k-1} \\ \mu_{ik} \end{pmatrix} \text{ for } i = k+1, k+2, \ldots, n;$$

if $k > 2$, then $k := k-1$;

go to (1).

(*) If $|\mu_{kl}| > \frac{1}{2}$, then:

$$\begin{cases} r := \text{integer nearest to } \mu_{kl}; \; b_k := b_k - r b_l; \\ \mu_{kj} := \mu_{kj} - r \mu_{lj} \text{ for } j = 1, 2, \ldots, l-1; \\ \mu_{kl} := \mu_{kl} - r. \end{cases}$$

Fig. 1. The reduction algorithm

(1.23) To prove that the algorithm terminates we introduce the quantities

$$d_i = \det((b_j, b_l))_{1 \leqq j, l \leqq i} \tag{1.24}$$

for $0 \leqq i \leqq n$. It is easily checked that

$$d_i = \prod_{j=1}^{i} |b_j^*|^2 \tag{1.25}$$

for $0 \leqq i \leqq n$. Hence the d_i are positive real numbers. Notice that $d_0 = 1$ and $d_n = d(L)^2$. Put

$$D = \prod_{i=1}^{n-1} d_i.$$

By (1.25), the number D only changes if some b_i^* is changed, which only occurs in case 1. In case 1, the number d_{k-1} is reduced by a factor $< \frac{3}{4}$, by (1.25), whereas the other d_i are unchanged, by (1.24); hence D is reduced by a factor $< \frac{3}{4}$. Below we prove that there is a positive lower bound for d_i that only depends on L. It follows

that there is also a positive lower bound for D, and hence an upper bound for the number of times that we pass through case 1.

In case 1, the value of k is decreased by 1, and in case 2 it is increased by 1. Initially we have $k=2$, and $k \leqq n+1$ throughout the algorithm. Therefore the number of times that we pass through case 2 is at most $n-1$ more than the number of times that we pass through case 1, and consequently it is bounded. This implies that the algorithm terminates.

To prove that d_i has a lower bound we put

$$m(L)=\min\{|x|^2 : x \in L, x \neq 0\}.$$

This is a positive real number. For $i>0$, we can interpret d_i as the square of the determinant of the lattice of rank i spanned by $b_1, b_2, \ldots, b_i$ in the vector space $\sum_{j=1}^{i} \mathbb{R}b_j$. By [4, Chap. I, Lemma 4 and Chap. II, Theorem I], this lattice contains a non-zero vector x with $|x|^2 \leqq (4/3)^{(i-1)/2} d_i^{1/i}$. Therefore $d_i \geqq (3/4)^{i(i-1)/2} m(L)^i$, as required.

We shall now analyse the running time of the algorithm under the added hypothesis that $b_i \in \mathbb{Z}^n$ for $1 \leqq i \leqq n$. By an *arithmetic operation* we mean an addition, subtraction, multiplication or division of two integers. Let the *binary length* of an integer a be the number of binary digits of $|a|$.

(1.26) **Proposition.** *Let $L \subset \mathbb{Z}^n$ be a lattice with basis $b_1, b_2, \ldots, b_n$, and let $B \in \mathbb{R}$, $B \geqq 2$, be such that $|b_i|^2 \leqq B$ for $1 \leqq i \leqq n$. Then the number of arithmetic operations needed by the basis reduction algorithm described in* (1.15) *is $O(n^4 \log B)$, and the integers on which these operations are performed each have binary length $O(n \log B)$.*

Remark. Using the classical algorithms for the arithmetic operations we find that the number of bit operations needed by the basis reduction algorithm is $O(n^6 (\log B)^3)$. This can be reduced to $O(n^{5+\varepsilon} (\log B)^{2+\varepsilon})$, for every $\varepsilon > 0$, if we employ fast multiplication techniques.

Proof of (1.26). We first estimate the number of times that we pass through cases 1 and 2. In the beginning of the algorithm we have $d_i \leqq B^i$, by (1.25), so $D \leqq B^{n(n-1)/2}$. Throughout the algorithm we have $D \geqq 1$, since $d_i \in \mathbb{Z}$ by (1.24) and $d_i > 0$ by (1.25). So by the argument in (1.23) the number of times that we pass through case 1 is $O(n^2 \log B)$, and the same applies to case 2.

The initialization of the algorithm takes $O(n^3)$ arithmetic operations with rational numbers; below we shall see how they can be replaced by operations with integers.

For (1.18) we need $O(n)$ arithmetic operations, and this is also true for case 1. In case 2 we have to deal with $O(n)$ values of l, that each require $O(n)$ arithmetic operations. Since we pass through these cases $O(n^2 \log B)$ times we arrive at a total of $O(n^4 \log B)$ arithmetic operations.

In order to represent all numbers that appear in the course of the algorithm by means of *integers* we also keep track of the numbers d_i defined by (1.24). In the initialization stage these can be calculated by (1.25). After that, they are only changed in case 1. In that case, d_{k-1} is replaced by $d_{k-1} \cdot |c^*_{k-1}|^2 / |b^*_{k-1}|^2 = d_{k-2} \cdot |c^*_{k-1}|^2$ [in the notation of (1.22)] whereas the other d_i are unchanged. By (1.24),

the d_i are *integers*, and we shall now see that they can be used as denominators for all numbers that appear:

$$|b_i^*|^2 = d_i/d_{i-1} \quad (1 \leqq i \leqq n), \tag{1.27}$$

$$d_{i-1} b_i^* \in L \subset \mathbb{Z}^n \quad (1 \leqq i \leqq n), \tag{1.28}$$

$$d_j \mu_{ij} \in \mathbb{Z} \quad (1 \leqq j < i \leqq n). \tag{1.29}$$

The first of these follows from (1.25). For the second, we write $b_i^* = b_i - \sum_{j=1}^{i-1} \lambda_{ij} b_j$ with $\lambda_{ij} \in \mathbb{R}$. Solving $\lambda_{i1}, \ldots, \lambda_{i\,i-1}$ from the system

$$(b_i, b_l) = \sum_{j=1}^{i-1} \lambda_{ij} (b_j, b_l) \quad (1 \leqq l \leqq i-1)$$

and using (1.24) we find that $d_{i-1}\lambda_{ij} \in \mathbb{Z}$, whence (1.28). Notice that the same argument yields

$$d_{i-1}\left(b_k - \sum_{j=1}^{i-1} \mu_{kj} b_j^*\right) \in \mathbb{Z}^n \quad \text{for} \quad i \leqq k;$$

this is useful for the calculation of b_k^* at the beginning of the algorithm. To prove (1.29) we use (1.3), (1.27), and (1.28):

$$d_j \mu_{ij} = d_j (b_i, b_j^*)/(b_j^*, b_j^*) = d_{j-1}(b_i, b_j^*) = (b_i, d_{j-1} b_j^*) \in \mathbb{Z}.$$

To finish the proof of (1.26) we estimate all integers that appear. Since no d_i is ever increased we have $d_i \leqq B^i$ throughout the algorithm. This estimates the denominators. To estimate the numerators it suffices to find upper bounds for $|b_i^*|^2$, $|b_i|^2$, and $|\mu_{ij}|$.

At the beginning we have $|b_i^*|^2 \leqq |b_i|^2 \leqq B$, and $\max\{|b_i^*|^2 : 1 \leqq i \leqq n\}$ is non-increasing; to see this, use that $|c_{k-1}^*|^2 < \frac{3}{4}|b_{k-1}^*|^2$ and $|c_k^*|^2 \leqq |b_{k-1}^*|^2$ in (1.22), the latter inequality because c_k^* is a projection of b_{k-1}^*. Hence we have $|b_i^*|^2 \leqq B$ throughout the algorithm.

To deal with $|b_i|^2$ and μ_{ij} we first prove that every time we arrive at the situation described by (1.16) and (1.17) the following inequalities are satisfied:

$$|b_i|^2 \leqq nB \quad \text{for} \quad i \neq k, \tag{1.30}$$

$$|b_k|^2 \leqq n^2 (4B)^n \quad \text{if} \quad k \neq n+1, \tag{1.31}$$

$$|\mu_{ij}| \leqq \tfrac{1}{2} \quad \text{for} \quad 1 \leqq j < i,\ i < k, \tag{1.32}$$

$$|\mu_{ij}| \leqq (nB^j)^{1/2} \quad \text{for} \quad 1 \leqq j < i,\ i > k, \tag{1.33}$$

$$|\mu_{kj}| \leqq 2^{n-k}(nB^{n-1})^{1/2} \quad \text{for} \quad 1 \leqq j < k, \quad \text{if} \quad k \neq n+1. \tag{1.34}$$

Here (1.30), for $i < k$, is trivial from (1.32), and (1.31) follows from (1.34). Using that

$$\mu_{ij}^2 \leqq |b_i|^2/|b_j^*|^2 = d_{j-1}|b_i|^2/d_j \leqq B^{j-1}|b_i|^2 \tag{1.35}$$

we see that (1.33) follows from (1.30), and (1.32) is the same as (1.16). It remains to prove (1.30) for $i > k$ and to prove (1.34). At the beginning of the algorithm we even have $|b_i|^2 \leqq B$ and $\mu_{ij}^2 \leqq B^j$, by (1.35), so it suffices to consider the situation at the

end of cases 1 and 2. Taking into account that k changes in these cases, we see that in case 1 the set of vectors $\{b_i : i \neq k\}$ is unchanged, and that in case 2 the set $\{b_i : i > k\}$ is replaced by a subset. Hence the inequalities (1.30) are preserved. At the end of case 2, the new values for μ_{kj} (if $k \neq n+1$) are the old values of $\mu_{k+1\,j}$, so here (1.34) follows from the inequality (1.33) at the previous stage. To prove (1.34) at the end of case 1 we assume that it is valid at the previous stage, and we follow what happens to μ_{kj}. To achieve (1.18) it is, for $j < k-1$, replaced by $\mu_{kj} - r\mu_{k-1\,j}$, with $|r| < 2|\mu_{k\,k-1}|$ and $|\mu_{k-1\,j}| \leqq \frac{1}{2}$, so

$$|\mu_{kj} - r\mu_{k-1\,j}| \leqq |\mu_{kj}| + |\mu_{k\,k-1}| \leqq 2^{n-k+1}(nB^{n-1})^{1/2} \quad \text{by (1.34)}. \tag{1.36}$$

In the notation of (1.22) we therefore have

$$|\nu_{k-1\,j}| \leqq 2^{n-(k-1)}(nB^{n-1})^{1/2} \quad \text{for} \quad j < k-1$$

and since $k-1$ is the new value for k this is exactly the inequality (1.34) to be proved.

Finally, we have to estimate $|b_i|^2$ and μ_{ij} at the other points in the algorithm. For this it suffices to remark that the maximum of $|\mu_{k1}|, |\mu_{k2}|, \ldots, |\mu_{k\,k-1}|$ is at most doubled when (1.18) is achieved, by (1.36), and that the same thing happens in case 2 for at most $k-2$ values of l. Combining this with (1.34) and (1.33) we conclude that throughout the course of the algorithm we have

$$|\mu_{ij}| \leqq 2^{n-1}(nB^{n-1})^{1/2} \quad \text{for} \quad 1 \leqq j < i \leqq n$$

and therefore

$$|b_i|^2 \leqq n^2(4B)^n \quad \text{for} \quad 1 \leqq i \leqq n.$$

This finishes the proof of (1.26).

(1.37) *Remark.* Let $1 \leqq n' \leqq n$. If k, in the situation described by (1.16) and (1.17), is for the first time equal to $n'+1$, then the first n' vectors $b_1, b_2, \ldots, b_{n'}$ form a reduced basis for the lattice of rank n' spanned by the first n' vectors of the initially given basis. This will be useful in Sect. 3.

(1.38) *Remark.* It is easily verified that, apart from some minor changes, the analysis of our algorithm remains valid if the condition $L \subset \mathbb{Z}^n$ is replaced by the condition that $(x, y) \in \mathbb{Z}$ for all $x, y \in L$; or, equivalently, that $(b_i, b_j) \in \mathbb{Z}$ for $1 \leqq i, j \leqq n$. The weaker condition that $(b_i, b_j) \in \mathbb{Q}$, for $1 \leqq i, j \leqq n$, is also sufficient, but in this case we should clear denominators before applying (1.26).

We close this section with two applications of our reduction algorithm. The first is to simultaneous diophantine approximation. Let n be a positive integer, $\alpha_1, \alpha_2, \ldots, \alpha_n$ real numbers, and $\varepsilon \in \mathbb{R}$, $0 < \varepsilon < 1$. It is a classical theorem [4, Sect.V.10] that there exist integers $p_1, p_2, \ldots, p_n, q$ satisfying

$$|p_i - q\alpha_i| \leqq \varepsilon \quad \text{for} \quad 1 \leqq i \leqq n,$$
$$1 \leqq q \leqq \varepsilon^{-n}.$$

We show that there exists a polynomial-time algorithm to find integers that satisfy a slightly weaker condition.

(1.39) **Proposition.** *There exists a polynomial-time algorithm that, given a positive integer n and rational numbers $\alpha_1, \alpha_2, \ldots, \alpha_n$, ε satisfying $0<\varepsilon<1$, finds integers $p_1, p_2, \ldots, p_n, q$ for which*

$$|p_i - q\alpha_i| \leqq \varepsilon \quad \text{for} \quad 1 \leqq i \leqq n,$$

$$1 \leqq q \leqq 2^{n(n+1)/4}\varepsilon^{-n}.$$

Proof. Let L be the lattice of rank $n+1$ spanned by the columns of the $(n+1)\times(n+1)$-matrix

$$\begin{pmatrix} 1 & 0 & \ldots & 0 & -\alpha_1 \\ 0 & 1 & \ldots & 0 & -\alpha_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & 1 & -\alpha_n \\ 0 & 0 & \ldots & 0 & 2^{-n(n+1)/4}\varepsilon^{n+1} \end{pmatrix}.$$

The inner product of any two columns is rational, so by (1.38) there is a polynomial-time algorithm to find a reduced basis $b_1, b_2, \ldots, b_{n+1}$ for L. By (1.9) we then have

$$|b_1| \leqq 2^{n/4} \cdot d(L)^{1/(n+1)} = \varepsilon.$$

Since $b_1 \in L$, we can write

$$b_1 = (p_1 - q\alpha_1, p_2 - q\alpha_2, \ldots, p_n - q\alpha_n, q \cdot 2^{-n(n+1)/4}\varepsilon^{n+1})^{\mathsf{T}}$$

with $p_1, p_2, \ldots, p_n, q \in \mathbb{Z}$. It follows that

$$|p_i - q\alpha_i| \leqq \varepsilon \quad \text{for} \quad 1 \leqq i \leqq n,$$

$$|q| \leqq 2^{n(n+1)/4}\varepsilon^{-n}.$$

From $\varepsilon<1$ and $b_1 \neq 0$ we see that $q \neq 0$. Replacing b_1 by $-b_1$, if necessary, we can achieve that $q>0$.

This proves (1.39).

Another application of our reduction algorithm is to the problem of finding $\mathbb{Q}$-linear relations among given real numbers $\alpha_1, \alpha_2, \ldots, \alpha_n$. For this we take the lattice L to be $\mathbb{Z}^n$, embedded in $\mathbb{R}^{n+1}$ by

$$(m_1, m_2, \ldots, m_n) \mapsto \left(m_1, m_2, \ldots, m_n, c\sum_{i=1}^{n} m_i\alpha_i'\right);$$

here c is a large constant and α_i' is a good rational approximation to α_i. The first basis vector of a reduced basis of L will give rise to integers $m_1, m_2, \ldots, m_n$ that are not too large such that $\sum_{i=1}^{n} m_i\alpha_i$ is very small.

Applying this to $\alpha_i = \alpha^{i-1}$ we see that our algorithm can be used to test a given real number α for algebraicity, and to determine its irreducible polynomial. Taking for α a zero of a polynomial $f \in \mathbb{Z}[X]$, $f \neq 0$, and generalizing the algorithm to *complex* α, one finds in this way an irreducible factor of f in $\mathbb{Z}[X]$. It is likely that this yields actually a polynomial-time algorithm to factor f in $\mathbb{Q}[X]$, an algorithm that is different from the p-adic method described in Sect. 3.

In a similar way we can test given real numbers $\alpha, \beta, \gamma, \ldots$ for algebraic dependence, taking the α_i to be the monomials in $\alpha, \beta, \gamma, \ldots$ up to a given degree.

2. Factors and Lattices

In this section we denote by p a prime number and by k a positive integer. We write $\mathbb{Z}/p^k\mathbb{Z}$ for the ring of integers modulo p^k, and $\mathbb{F}_p$ for the field $\mathbb{Z}/p\mathbb{Z}$. For $g=\sum_i a_i X^i \in \mathbb{Z}[X]$ we denote by $(g \bmod p^k)$ the polynomial $\sum_i (a_i \bmod p^k) X^i \in (\mathbb{Z}/p^k\mathbb{Z})[X]$.

We fix a polynomial $f \in \mathbb{Z}[X]$ of degree n, with $n>0$, and a polynomial $h \in \mathbb{Z}[X]$ that has the following properties:

(2.1) h has leading coefficient 1,

(2.2) $(h \bmod p^k)$ divides $(f \bmod p^k)$ in $(\mathbb{Z}/p^k\mathbb{Z})[X]$,

(2.3) $(h \bmod p)$ is irreducible in $\mathbb{F}_p[X]$,

(2.4) $(h \bmod p)^2$ does not divide $(f \bmod p)$ in $\mathbb{F}_p[X]$.

We put $l=\deg(h)$; so $0<l\leqq n$.

(2.5) **Proposition.** *The polynomial f has an irreducible factor h_0 in $\mathbb{Z}[X]$ for which $(h \bmod p)$ divides $(h_0 \bmod p)$, and this factor is uniquely determined up to sign. Further, if g divides f in $\mathbb{Z}[X]$, then the following three assertions are equivalent:*

(i) *$(h \bmod p)$ divides $(g \bmod p)$ in $\mathbb{F}_p[X]$,*

(ii) *$(h \bmod p^k)$ divides $(g \bmod p^k)$ in $(\mathbb{Z}/p^k\mathbb{Z})[X]$,*

(iii) *h_0 divides g in $\mathbb{Z}[X]$.*

In particular $(h \bmod p^k)$ divides $(h_0 \bmod p^k)$ in $(\mathbb{Z}/p^k\mathbb{Z})[X]$.

Proof. The existence of h_0 follows from (2.2) and (2.3), and the uniqueness, up to ± 1, from (2.4). The implications (ii) $\Rightarrow$ (i) and (iii) $\Rightarrow$ (i) are obvious. Now assume (i); we prove (iii) and (ii). From (i) and (2.4) it follows that $(h \bmod p)$ does not divide $(f/g \bmod p)$ in $\mathbb{F}_p[X]$. Therefore h_0 does not divide f/g in $\mathbb{Z}[X]$, so it must divide g. This proves (iii). By (2.3) the polynomials $(h \bmod p)$ and $(f/g \bmod p)$ are relatively prime in $\mathbb{F}_p[X]$, so in $\mathbb{F}_p[X]$ we have

$$(\lambda_1 \bmod p)\cdot(h \bmod p)+(\mu_1 \bmod p)\cdot(f/g \bmod p)=1$$

for certain $\lambda_1, \mu_1 \in \mathbb{Z}[X]$. Therefore $\lambda_1 h+\mu_1 f/g=1-pv_1$ for some $v_1 \in \mathbb{Z}[X]$. Multiplying this by $1+pv_1+p^2v_1^2+\ldots+p^{k-1}v_1^{k-1}$ and by g we obtain

$$\lambda_2 h+\mu_2 f \equiv g \bmod p^k\mathbb{Z}[X]$$

for certain $\lambda_2, \mu_2 \in \mathbb{Z}[X]$. Since the left hand side, when taken modulo p^k, is divisible by $(h \bmod p^k)$, the same is true for the right hand side. This proves (ii).

The final assertion of (2.5) follows if we take $g=h_0$. This proves (2.5).

(2.6) In the remainder of this section we fix an integer m with $m \geqq l$, and we let L be the collection of all polynomials in $\mathbb{Z}[X]$ of degree $\leqq m$ that, when taken modulo p^k, are divisible by $(h \bmod p^k)$ in $(\mathbb{Z}/p^k\mathbb{Z})[X]$. This is a subset of the $(m+1)$-dimensional real vector space $\mathbb{R}+\mathbb{R}\cdot X+\ldots+\mathbb{R}\cdot X^m$. This vector space is identified with $\mathbb{R}^{m+1}$ by identifying $\sum_{i=0}^{m} a_i X^i$ with $(a_0, a_1, \ldots, a_m)$. Notice that the length $\left|\sum_{i=0}^{m} a_i X^i\right|$ of a

polynomial, as defined in the introduction, is equal to the ordinary Euclidean length of $(a_0, a_1, \ldots, a_m)$. It is easy to see that L is a lattice in $\mathbb{R}^{m+1}$ and, using (2.1), that a basis of L is given by

$$\{p^k X^i : 0 \leqq i < l\} \cup \{hX^j : 0 \leqq j \leqq m-l\}.$$

From (1.1) it follows that $d(L) = p^{kl}$.

In the following proposition h_0 is as in (2.5).

(2.7) **Proposition.** *Let $b \in L$ satisfy*

$$p^{kl} > |f|^m \cdot |b|^n. \tag{2.8}$$

Then b is divisible by h_0 in $\mathbb{Z}[X]$, and in particular $\gcd(f, b) \neq 1$.

Remark. A weaker version of (2.7), which could also be used to obtain a polynomial-time factoring algorithm for polynomials, asserts that $\gcd(f, b) \neq 1$ under the same conditions. The proof of this version is less complicated than the proof given below, see [8, Theorem 2].

Proof of (2.7). We may assume that $b \neq 0$. Let $g = \gcd(f, b)$. By (2.5) it suffices to show that $(h \bmod p)$ divides $(g \bmod p)$. Suppose that this is not the case. Then by (2.3) we have

$$\lambda_3 h + \mu_3 g = 1 - pv_3 \tag{2.9}$$

for certain λ_3, μ_3, $v_3 \in \mathbb{Z}[X]$. We shall derive a contradiction from this.

Put $e = \deg(g)$ and $m' = \deg(b)$. Clearly $0 \leqq e \leqq m' \leqq m$. We define

$$\begin{aligned} M &= \{\lambda f + \mu b : \lambda, \mu \in \mathbb{Z}[X], \deg(\lambda) < m' - e, \deg(\mu) < n - e\} \\ &\subset \mathbb{Z} + \mathbb{Z} \cdot X + \ldots + \mathbb{Z} \cdot X^{n+m'-e-1}. \end{aligned}$$

Let M' be the projection of M on

$$\mathbb{Z} \cdot X^e + \mathbb{Z} \cdot X^{e+1} + \ldots + \mathbb{Z} \cdot X^{n+m'-e-1}.$$

Suppose that $\lambda f + \mu b$ projects to 0 in M', with λ, μ as in the definition of M. Then $\deg(\lambda f + \mu b) < e$, but g divides $\lambda f + \mu b$, so $\lambda f + \mu b = 0$. From $\lambda \cdot (f/g) = -\mu \cdot (b/g)$ and $\gcd(f/g, b/g) = 1$ it follows that f/g divides μ. But $\deg(\mu) < n - e = \deg(f/g)$, so $\mu = 0$, and therefore also $\lambda = 0$.

This proves that the projections of

$$\{X^i f : 0 \leqq i < m' - e\} \cup \{X^j b : 0 \leqq j < n - e\}$$

on M' are linearly independent. Since these projections span M', it follows that M' is a lattice of rank $n + m' - 2e$. From Hadamard's inequality (1.10) and (2.8) we obtain

$$d(M') \leqq |f|^{m'-e} \cdot |b|^{n-e} \leqq |f|^m \cdot |b|^n < p^{kl}. \tag{2.10}$$

Below we deduce from (2.9) that

$$\{v \in M : \deg(v) < e + l\} \subset p^k \mathbb{Z}[X]. \tag{2.11}$$

Hence, if we choose a basis $b_e, b_{e+1}, \ldots, b_{n+m'-e-1}$ of M' with $\deg(b_j)=j$, see [4, Chap. I, Theorem I.A], then the leading coefficients of $b_e, b_{e+1}, \ldots, b_{e+l-1}$ are divisible by p^k. [Notice that $e+l-1 \leqq n+m'-e-1$ because g divides b and $(h \bmod p)$ divides $(f/g \bmod p)$.] Since $d(M')$ equals the absolute value of the product of the leading coefficients of $b_e, b_{e+1}, \ldots, b_{n+m'-e-1}$ we find that $d(M') \geqq p^{kl}$. Combined with (2.10) this is the desired contradiction.

To prove (2.11), let $v \in M$, $\deg(v) < e+l$. Then g divides v. Multiplying (2.9) by v/g and by $1+pv_3+p^2v_3^2+\ldots+p^{k-1}v_3^{k-1}$ we obtain

$$\lambda_4 h + \mu_4 v \equiv v/g \bmod p^k \mathbb{Z}[X] \tag{2.12}$$

with $\lambda_4, \mu_4 \in \mathbb{Z}[X]$. From $v \in M$ and $b \in L$ it follows that $(v \bmod p^k)$ is divisible by $(h \bmod p^k)$. So by (2.12) also $(v/g \bmod p^k)$ is divisible by $(h \bmod p^k)$. But $(h \bmod p^k)$ is of degree l with leading coefficient 1, while $(v/g \bmod p^k)$ has degree $< e+l-e=l$. Therefore $v/g \equiv 0 \bmod p^k \mathbb{Z}[X]$, so also $v \equiv 0 \bmod p^k \mathbb{Z}[X]$. This proves (2.11).

This concludes the proof of (2.7).

(2.13) **Proposition.** *Let p, k, f, n, h, l be as at the beginning of this section, h_0 as in* (2.5), *and m, L as in* (2.6). *Suppose that $b_1, b_2, \ldots, b_{m+1}$ is a reduced basis for L (see* (1.4) *and* (1.5)), *and that*

$$p^{kl} > 2^{mn/2} \binom{2m}{m}^{n/2} |f|^{m+n}. \tag{2.14}$$

Then we have $\deg(h_0) \leqq m$ *if and only if*

$$|b_1| < (p^{kl}/|f|^m)^{1/n}. \tag{2.15}$$

Proof. The "if"-part is immediate from (2.7), since $\deg(b_1) \leqq m$. To prove the "only if"-part, assume that $\deg(h_0) \leqq m$. Then $h_0 \in L$ by (2.5), and $|h_0| \leqq \binom{2m}{m}^{1/2} \cdot |f|$ by a result of Mignotte [10; cf. 7, Exercise 4.6.2.20]. Applying (1.11) to $x = h_0$ we find that $|b_1| \leqq 2^{m/2} \cdot |h_0| \leqq 2^{m/2} \cdot \binom{2m}{m}^{1/2} \cdot |f|$. By (2.14) this implies (2.15). This proves (2.13).

(2.16) **Proposition.** *Let the notation and the hypotheses be the same as in* (2.13), *and assume in addition that there exists an index $j \in \{1, 2, \ldots, m+1\}$ for which*

$$|b_j| < (p^{kl}/|f|^m)^{1/n}. \tag{2.17}$$

Let t be the largest such j. Then we have

$$\deg(h_0) = m+1-t,$$

$$h_0 = \gcd(b_1, b_2, \ldots, b_t),$$

and (2.17) *holds for all j with $1 \leqq j \leqq t$.*

Proof. Let $J = \{j \in \{1, 2, \ldots, m+1\}$: (2.17) holds$\}$. From (2.7) we know that h_0 divides b_j for every $j \in J$. Hence if we put

$$h_1 = \gcd(\{b_j : j \in J\})$$

then h_0 divides h_1. Each $b_j, j \in J$, is divisible by h_1 and has degree $\leqq m$, so belongs to

$$\mathbb{Z} \cdot h_1 + \mathbb{Z} \cdot h_1 X + \ldots + \mathbb{Z} \cdot h_1 X^{m - \deg(h_1)}.$$

Since the b_j are linearly independent this implies that

$$\# J \leqq m + 1 - \deg(h_1). \tag{2.18}$$

By the result of Mignotte used in the proof of (2.13) we have $|h_0 X^i| = |h_0| \leqq \binom{2m}{m}^{1/2} \cdot |f|$ for all $i \geqq 0$. For $i = 0, 1, \ldots, m - \deg(h_0)$ we have $h_0 X^i \in L$, so from (1.12) we obtain

$$|b_j| \leqq 2^{m/2} \cdot \binom{2m}{m}^{1/2} \cdot |f|$$

for $1 \leqq j \leqq m + 1 - \deg(h_0)$. By (2.14), this implies that

$$\{1, 2, \ldots, m + 1 - \deg(h_0)\} \subset J. \tag{2.19}$$

From (2.18), (2.19) and the fact that h_0 divides h_1 we now see that equality must hold in (2.18) and (2.19), and that

$$\deg(h_0) = \deg(h_1) = m + 1 - t, \qquad J = \{1, 2, \ldots, t\}.$$

It remains to prove that h_0 is equal to h_1, up to sign, and for this it suffices to check that h_1 is primitive. Choose $j \in J$, and let d_j be the content of b_j. Then b_j/d_j is divisible by h_0, and $h_0 \in L$, so $b_j/d_j \in L$. But b_j belongs to a basis for L, so $d_j = 1$ and b_j is primitive, and the same is true for the factor h_1 of b_j. This finishes the proof of (2.16).

Remark. If $t = 1$ then we see from (2.16) that b_1 is an irreducible factor of f, and that no gcd computation is necessary.

Remark. From the proofs of (2.13) and (2.16) we see that (2.14) may be replaced by

$$p^{kl} > \beta^n \gamma^n |f|^m,$$

where $\beta = \max\{|b_j|/|b_i^*| : 1 \leqq j \leqq i \leqq m+1\}$ [cf. (1.14)] and where γ is such that $|g| \leqq \gamma$ for every factor g of f in $\mathbb{Z}[X]$ with $\deg(g) \leqq m$.

3. Description of the Algorithm

Denote by f a primitive polynomial in $\mathbb{Z}[X]$ of degree n, with $n > 0$. In this section we describe an algorithm that factors f into irreducible factors in $\mathbb{Z}[X]$. We begin with two auxiliary algorithms.

(3.1) Suppose that, in addition to f and n, a prime number p, a positive integer k and a polynomial $h \in \mathbb{Z}[X]$ are given satisfying (2.1), (2.2), (2.3), and (2.4). Assume that the coefficients of h are reduced modulo p^k, so

$$|h|^2 \leqq 1 + l p^{2k},$$

where $l=\deg(h)$. Let further an integer $m \geqq l$ be given, and assume that inequality (2.14) is satisfied:

$$p^{kl} > 2^{mn/2} \cdot \binom{2m}{m}^{n/2} \cdot |f|^{m+n}.$$

We describe an algorithm that decides whether $\deg(h_0) \leqq m$, with h_0 as in (2.5), and determines h_0 if indeed $\deg(h_0) \leqq m$.

Let L be the lattice defined in (2.6), with basis

$$\{p^k X^i : 0 \leqq i < l\} \cup \{hX^j : 0 \leqq j \leqq m-l\}.$$

Applying algorithm (1.15) we find a *reduced* basis $b_1, b_2, \ldots, b_{m+1}$ for L. If $|b_1| \geqq (p^{kl}/|f|^m)^{1/n}$ then by (2.13) we have $\deg(h_0) > m$, and the algorithm stops. If $|b_1| < (p^{kl}/|f|^m)^{1/n}$ then by (2.13) and (2.16) we have $\deg(h_0) \leqq m$ and

$$h_0 = \gcd(b_1, b_2, \ldots, b_t)$$

with t as in (2.16). This gcd can be calculated by repeated application of the subresultant algorithm described in [7, Sect. 4.6.1]. This finishes the description of algorithm (3.1).

(3.2) **Proposition.** *The number of arithmetic operations needed by algorithm* (3.1) *is* $O(m^4 k \log p)$, *and the integers on which these operations are performed each have binary length* $O(mk \log p)$.

Proof. We apply (1.26) with $m+1$ in the role of n and with $B = 1 + lp^{2k}$. From $l \leqq n$ and (2.14) we see that $m = O(k \log p)$, so $\log l < l \leqq m$ implies that $\log B = O(k \log p)$. This leads to the estimates in (3.2). It is straightforward to verify that the gcd computation at the end satisfies the same estimates. This proves (3.2).

(3.3) Next suppose that, in addition to f and n, a prime number p and a polynomial $h \in \mathbb{Z}[X]$ are given such that (2.1), (2.2), (2.3), and (2.4) are satisfied with k replaced by 1. Assume that the coefficients of h are reduced modulo p. We describe an algorithm that determines h_0, the irreducible factor of f for which $(h \bmod p)$ divides $(h_0 \bmod p)$, cf. (2.5).

Write $l = \deg(h)$. If $l = n$ then $h_0 = f$, and the algorithm stops. Let now $l < n$. We first calculate the least positive integer k for which (2.14) holds with m replaced by $n-1$:

$$p^{kl} > 2^{(n-1)n/2} \cdot \binom{2(n-1)}{n-1}^{n/2} \cdot |f|^{2n-1}.$$

Next we modify h, without changing $(h \bmod p)$, in such a way that (2.2) holds for the value of k just calculated, in addition to (2.1), (2.3), and (2.4). This can be accomplished by the use of Hensel's lemma, see [7, Exercise 4.6.2.22; 14; 15; 13]. We may assume that the coefficients of h are reduced modulo p^k.

Let u be the greatest integer for which $l \leqq (n-1)/2^u$. We perform algorithm (3.1) for each of the values $m = [(n-1)/2^u]$, $[(n-1)/2^{u-1}]$, $\ldots$, $[(n-1)/2]$, $n-1$ in succession, with $[x]$ denoting the greatest integer $\leqq x$; but we stop as soon as for one of these values of m algorithm (3.1) succeeds in determining h_0. If this does not occur for any m in the sequence then $\deg(h_0) > n-1$, so $h_0 = f$ and we stop. This finishes the description of algorithm (3.3).

(3.4) **Proposition.** *Denote by* $m_0 = \deg(h_0)$ *the degree of the irreducible factor* h_0 *of* f *that is found by algorithm* (3.3). *Then the number of arithmetic operations needed by algorithm* (3.3) *is* $O(m_0(n^5 + n^4 \log|f| + n^3 \log p))$, *and the integers on which these operations are performed each have binary length* $O(n^3 + n^2 \log|f| + n \log p)$.

Proof. From

$$p^{k-1} \leqq p^{(k-1)l} \leqq 2^{(n-1)n/2} \binom{2(n-1)}{n-1}^{n/2} |f|^{2n-1}$$

it follows that

$$k \log p = (k-1)\log p + \log p = O(n^2 + n \log|f| + \log p).$$

Let m_1 be the largest value of m for which algorithm (3.1) is performed. From the choice of values for m it follows that $m_1 < 2m_0$, and that every other value for m that is tried is of the form $[m_1/2^i]$, with $i \geqq 1$. Therefore we have $\sum m^4 = O(m_0^4)$. Using (3.2) we conclude that the total number of arithmetic operations needed by the applications of algorithm (3.1) is $O(m_0^4 k \log p)$, which is

$$O(m_0^4(n^2 + n \log|f| + \log p)),$$

and that the integers involved each have binary length $O(m_1 k \log p)$, which is

$$O(m_0(n^2 + n \log|f| + \log p)).$$

With some care it can be shown that the same estimates are valid for a suitable version of Hensel's lemma. But it is simpler, and sufficient for our purpose, to replace the above estimates by the estimates stated in (3.4), using that $m_0 \leqq n$; then a very crude estimate for Hensel's lemma will do. The straightforward verification is left to the reader. This proves (3.4).

(3.5) We now describe an algorithm that factors a given primitive polynomial $f \in \mathbb{Z}[X]$ of degree $n > 0$ into irreducible factors in $\mathbb{Z}[X]$.

The first step is to calculate the resultant $R(f, f')$ of f and its derivative f', using the subresultant algorithm [7, Sect. 4.6.1]. If $R(f, f') = 0$ then f and f' have a greatest common divisor g in $\mathbb{Z}[X]$ of positive degree, and g is also calculated by the subresultant algorithm. This case will be discussed at the end of the algorithm. Assume now that $R(f, f') \neq 0$.

In the second step we determine the smallest prime number p not dividing $R(f, f')$, and we decompose $(f \bmod p)$ into irreducible factors in $\mathbb{F}_p[X]$ by means of Berlekamp's algorithm [7, Sect. 4.6.2]. Notice that $R(f, f')$ is, up to sign, equal to the product of the leading coefficient of f and the discriminant of f. So $R(f, f') \not\equiv 0 \bmod p$ implies that $(f \bmod p)$ still has degree n, and that it has no multiple factors in $\mathbb{F}_p[X]$. Therefore (2.4) is valid for every irreducible factor $(h \bmod p)$ of $(f \bmod p)$ in $\mathbb{F}_p[X]$.

In the third step we assume that we know a decomposition $f = f_1 f_2$ in $\mathbb{Z}[X]$ such that the complete factorizations of f_1 in $\mathbb{Z}[X]$ and $(f_2 \bmod p)$ in $\mathbb{F}_p[X]$ are known. At the start we can take $f_1 = 1$, $f_2 = f$. In this situation we proceed as follows. If $f_2 = \pm 1$ then $f = \pm f_1$ is completely factored in $\mathbb{Z}[X]$, and the algorithm stops. Suppose now that f_2 has positive degree, and choose an irreducible factor

($h \bmod p$) of ($f_2 \bmod p$) in $\mathbb{F}_p[X]$. We may assume that the coefficients of h are reduced modulo p and that h has leading coefficient 1. Then we are in the situation described at the start of algorithm (3.3), with f_2 in the role of f, and we use that algorithm to find the irreducible factor h_0 of f_2 in $\mathbb{Z}[X]$ for which ($h \bmod p$) divides ($h_0 \bmod p$). We now replace f_1 and f_2 by $f_1 h_0$ and f_2/h_0, respectively, and from the list of irreducible factors of ($f_2 \bmod p$) we delete those that divide ($h_0 \bmod p$). After this we return to the beginning of the third step.

This finishes the description of the algorithm in the case that $R(f, f') \neq 0$. Suppose now that $R(f, f') = 0$, let g be the gcd of f and f' in $\mathbb{Z}[X]$, and put $f_0 = f/g$. Then f_0 has no multiple factors in $\mathbb{Z}[X]$, so $R(f_0, f_0') \neq 0$, and we can factor f_0 using the main part of the algorithm. Since each irreducible factor of g in $\mathbb{Z}[X]$ divides f_0 we can now complete the factorization of $f = f_0 g$ by a few trial divisions. This finishes the description of algorithm (3.5).

(3.6) **Theorem.** *The above algorithm factors any primitive polynomial $f \in \mathbb{Z}[X]$ of positive degree n into irreducible factors in $\mathbb{Z}[X]$. The number of arithmetic operations needed by the algorithm is $O(n^6 + n^5 \log|f|)$, and the integers on which these operations are performed each have binary length $O(n^3 + n^2 \log|f|)$. Here $|f|$ is as defined in the introduction.*

Using the classical algorithms for the arithmetic operations we now arrive at the bound $O(n^{12} + n^9(\log|f|)^3)$ for the number of bit operations that was announced in the introduction. This can be reduced to $O(n^{9+\varepsilon} + n^{7+\varepsilon}(\log|f|)^{2+\varepsilon})$, for every $\varepsilon > 0$, if we employ fast multiplication techniques.

Proof of (3.6). The correctness of the algorithm is clear from its description. To prove the estimates we first assume that $R(f, f') \neq 0$. We begin by deriving an upper bound for p. Since p is the least prime not dividing $R(f, f')$ we have

$$\prod_{q < p,\, q \text{ prime}} q \leqq |R(f, f')|. \tag{3.7}$$

It is not difficult to prove that there is a positive constant A such that

$$\prod_{q < p,\, q \text{ prime}} q > e^{Ap} \tag{3.8}$$

for all $p > 2$, see [6, Sect. 22.2]; by [12] we can take $A = 0.84$ for $p > 101$. From Hadamard's inequality (1.10) we easily obtain

$$|R(f, f')| \leqq n^n |f|^{2n-1}.$$

Combining this with (3.7) and (3.8) we conclude that

$$p < (n \log n + (2n-1) \log|f|)/A \tag{3.9}$$

or $p = 2$. Therefore the terms involving $\log p$ in proposition (3.4) are absorbed by the other terms.

The call of algorithm (3.3) in the third step requires $O(m_0 \cdot (n^5 + n^4 \log|f_2|))$ arithmetic operations, by (3.4), where m_0 is the degree of the factor h_0 that is found. Since f_2 divides f, Mignotte's theorem [10; cf. 7, Exercise 4.6.2.20] that was used in the proof of (2.13) implies that $\log|f_2| = O(n + \log|f|)$. Further the sum $\sum m_0$ of the

degrees of the irreducible factors of f is clearly equal to n. We conclude that the total number of arithmetic operations needed by the applications of (3.3) is $O(n^6 + n^5 \log|f|)$. By (3.4), the integers involved in (3.3) each have binary length $O(n^3 + n^2 \log|f|)$.

We must now show that the other parts of the algorithm satisfy the same estimates. For the subresultant algorithm in the first step and the remainder of the third step this is entirely straightforward and left to the reader. We consider the second step.

Write P for the right hand side of (3.9). Then p can be found with $O(P)$ arithmetic operations on integers of binary length $O(P)$; here one can apply [11] to generate a table of prime numbers $< P$, or alternatively use a table of squarefree numbers, which is easier to generate. From $p < P$ it also follows that Berlekamp's algorithm satisfies the estimates stated in the theorem, see [7, Sect. 4.6.2].

Finally, let $R(f, f') = 0$, and $f_0 = f/\gcd(f, f')$ as in the algorithm. Since f_0 divides f, Mignotte's theorem again implies that $\log|f_0| = O(n + \log|f|)$. The theorem now follows easily by applying the preceding case to f_0.

This finishes the proof of (3.6).

(3.10) For the algorithms described in this section the precise choice of the basis reduction algorithm is irrelevant, as long as it satisfies the estimates of proposition (1.26). A few simplifications are possible if the algorithm explained in Sect. 1 is used. Specifically, the gcd computation at the end of algorithm (3.1) can be avoided. To see this, assume that $m_0 = \deg(h_0)$ is indeed $\leqq m$. We claim that h_0 occurs as b_1 in the course of the basis reduction algorithm. Namely, by (1.37) it will happen at a certain moment that $b_1, b_2, \ldots, b_{m_0+1}$ form a reduced basis for the lattice of rank $m_0 + 1$ spanned by $\{p^k X^i : 0 \leqq i < l\} \cup \{hX^j : 0 \leqq j \leqq m_0 - l\}$. At that moment, we have $h_0 = b_1$, by (2.13) and (2.16), applied with m_0 in the role of m. A similar argument shows that in algorithm (3.3) one can simply try the values $m = l$, $l+1, \ldots, n-1$ in succession, until h_0 is found.

Acknowledgements are due to J. J. M. Cuppen for permission to include his improvement of our basis reduction algorithm in Sect. 1.

References

1. Adleman, L.M., Odlyzko, A.M.: Irreducibility testing and factorization of polynomials, to appear. Extended abstract: Proc. 22nd Annual IEEE Symp. Found. Comp. Sci., pp. 409–418 (1981)
2. Brentjes, A.J.: Multi-dimensional continued fraction algorithms. Mathematical Centre Tracts 145. Amsterdam: Mathematisch Centrum 1981
3. Cantor, D.G.: Irreducible polynomials with integral coefficients have succinct certificates. J. Algorithms **2**, 385–392 (1981)
4. Cassels, J.W.S.: An introduction to the geometry of numbers. Berlin, Heidelberg, New York: Springer 1971
5. Ferguson, H.R.P., Forcade, R.W.: Generalization of the Euclidean algorithm for real numbers to all dimensions higher than two. Bull. Am. Math. Soc. **1**, 912–914 (1979)
6. Hardy, G.H., Wright, E.M.: An introduction to the theory of numbers. Oxford: Oxford University Press 1979
7. Knuth, D.E.: The art of computer programming, Vol. 2, Seminumerical algorithms. Reading: Addison-Wesley 1981

8. Lenstra, A.K.: Lattices and factorization of polynomials, Report IW 190/81. Amsterdam: Mathematisch Centrum 1981
9. Lenstra, H.W., Jr.: Integer programming with a fixed number of variables. Math. Oper. Res. (to appear)
10. Mignotte, M.: An inequality about factors of polynomials. Math. Comp. **28**, 1153–1157 (1974)
11. Pritchard, P.: A sublinear additive sieve for finding prime numbers. Comm. ACM **24**, 18–23 (1981)
12. Barkley Rosser, J., Schoenfeld, L.: Approximate formulas for some functions of prime numbers. Ill. J. Math. **6**, 64–94 (1962)
13. Yun, D.Y.Y.: The Hensel lemma in algebraic manipulation. Cambridge: MIT 1974; reprint: New York: Garland 1980
14. Zassenhaus, H.: On Hensel factorization. I. J. Number Theory **1**, 291–311 (1969)
15. Zassenhaus, H.: A remark on the Hensel factorization method. Math. Comp. **32**, 287–292 (1978)
16. Zassenhaus, H.: A new polynomial factorization algorithm (unpublished manuscript, 1981)

Received July 11, 1982