Contents

Chapter 14
Integer Programming and Algorithmic Geometry of Numbers
A tutorial

Friedrich Eisenbrand

Abstract This chapter surveys a selection of results from the interplay of integer programming and the geometry of numbers. Apart from being a survey, the text is also intended as an entry point into the field. I therefore added exercises at the end of each section to invite the reader to delve deeper into the presented material.

14.1 Lattices, integer programming and the geometry of numbers

The central objects of study of the *geometry of numbers* are lattices. A *lattice* is a set $\Lambda = \{y \in \mathbb{R}^n : y = Ax,\ x \in \mathbb{Z}^n\}$, where $A \in \mathbb{R}^{n\times n}$ is a nonsingular matrix. We say that the lattice is *generated by* A and write $\Lambda = \Lambda(A)$. The matrix A is called a *basis* of the lattice Λ. If A is a rational matrix, i.e., $A \in \mathbb{Q}^{n\times n}$, then Λ is a *rational lattice*.

A very important problem, which has also received a lot of attention in computer science and optimization, is the *shortest vector problem* with respect to the ℓ_p-norm for some $p \in \mathbb{N}_+ \cup \{\infty\}$. It is as follows.

> Given a rational lattice-basis $A \in \mathbb{Q}^{n\times n}$, compute a nonzero vector $v \in \Lambda(A)$ with minimal norm $\|v\|_p$.

If the norm is not specified, we implicitly assume the ℓ_2-norm and denote the length of a shortest vector w.r.t. the ℓ_2-norm as $SV(\Lambda)$. The shortest vector problem can (in fixed dimension) be solved efficiently with *lattice basis reduction*. In varying dimension the shortest vector problem is NP-hard under randomized reductions [2]. Still, the fastest algorithms [43, 3] for this problem rely on basis reduction.

Friedrich Eisenbrand
Department of Mathematics, EPFL, Lausanne, Switzerland
e-mail: friedrich.eisenbrand@epfl.ch

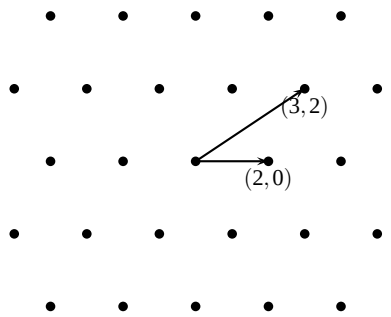


Fig. 14.1 The lattice generated by $(2,0)$ and $(3,2)$.

Lenstra [56] has shown that *integer programming* can be solved in polynomial time, if the dimension is fixed. His algorithm is based on lattice basis reduction. Why do lattices and the algorithmic geometry of numbers come into play in integer programming?

The *greatest common divisor* of two integers $a_0, a_1 \in \mathbb{Z}$ where not both a_0 and a_1 are zero, is the largest integer that divides both a_0 and a_1. It is denoted by $\gcd(a_0, a_1)$. The greatest common divisor can be efficiently computed with the *Euclidean algorithm*, see, e.g. [51, 1]. It computes the remainder sequence $a_0, a_1, \ldots, a_{k-1}, a_k \in \mathbb{N}_+$, where a_i, $i \geqslant 2$ is given by $a_{i-2} = a_{i-1}q_{i-1} + a_i$, $q_i \in \mathbb{N}$, $0 < a_i < a_{i-1}$, and a_k divides a_{k-1} exactly. Then $a_k = \gcd(a_0, a_1)$. The Euclidean algorithm can be interpreted as a reduction algorithm and we will see that the 2-dimensional basis reduction algorithm of Lagrange (see Section 14.2) works along the same lines.

Now, the connection between integer linear programming with algorithmic number theory reveals itself already in the following theorem, which is proved at the beginning of every course on elementary number theory, see, e.g. [65].

Theorem 14.1. *Let $a, b \in \mathbb{Z}$ be integers that are not both equal to zero. The greatest common divisor $\gcd(a,b)$ is the smallest integer in the set*

$$\{ax + by \colon x, y \in \mathbb{Z},\, ax + by \geqslant 1\}. \tag{14.1}$$

The problem to find the minimum in (14.1) can be modeled as an integer program in two variables and one constraint.

$$\begin{aligned} \min \quad & ax + by \\ & ax + by \geqslant 1 \\ & x, y \in \mathbb{Z}. \end{aligned}$$

This already explains why efficient methods for integer programming with a fixed number of variables incorporate *reduction techniques*, which in a basic form appear already in the Euclidean algorithm. Such reduction techniques are in the focus of this chapter.

14.2 Informal introduction to basis reduction

Informally, *lattice basis reduction* is about transforming a lattice basis B into a lattice basis B' that generates the same lattice, $\Lambda(B) = \Lambda(B')$ and from which a shortest vector can (in fixed dimension) be easily determined. Before we make this more precise, we have to understand what valid transformations are, i.e., when $\Lambda(B) = \Lambda(B')$ holds. Recall that an integer matrix $U \in \mathbb{Z}^{n\times n}$ is called *unimodular* if $\det(U) = \pm 1$. Thus $U \in \mathbb{Z}^{n\times n}$ is unimodular if and only if U is non-singular and U^{-1} is a matrix with integer entries, see exercise 1.

Lemma 14.1. *Let $B, B' \in \mathbb{R}^{n\times n}$ be two rational non-singular matrices. One has $\Lambda(B) = \Lambda(B')$ if and only if there exists a unimodular matrix $U \in \mathbb{Z}^{n\times n}$ with $B' = B \cdot U$.*

Proof. Suppose $\Lambda(B) = \Lambda(B')$. Then, every column of B is in $\Lambda(B')$ which implies that there exists an integral matrix $U \in \mathbb{Z}^{n\times n}$ with $B = B' \cdot U$. Similarly, there exists an integral matrix $V \in \mathbb{Z}^{n\times n}$ with $B' = B \cdot V$. From this it follows that $B = B \cdot V \cdot U$ and since B is non-singular, this implies that $V \cdot U = I_n$, where I_n is the $n \times n$ identity matrix. This implies that U is unimodular since $1 = \det(V \cdot U) = \det(V) \cdot \det(U)$ and since both $\det(V)$ and $\det(U)$ are integers, one has $\det(U) = \pm 1$.

On the other hand, if $B' = B \cdot U$ with an integral U, then $\Lambda(B') \subseteq \Lambda(B)$. If U is unimodular, then $B = B' \cdot U^{-1}$ and U^{-1} is integral, which implies $\Lambda(B) \subseteq \Lambda(B')$. □

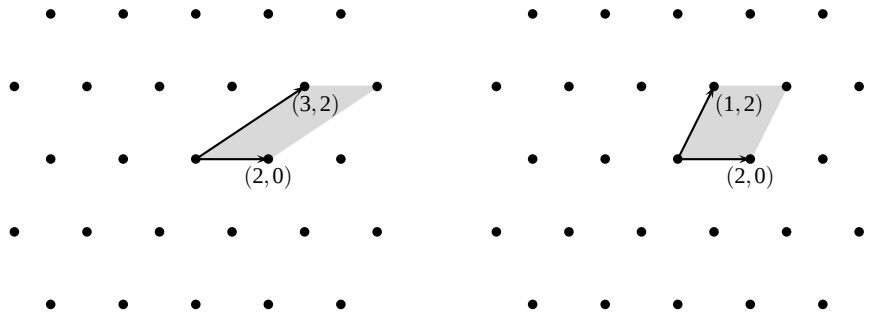


Fig. 14.2 A lattice, two different bases and the lattice determinant, which is depicted as the volume of the parallelepipeds defined by the bases respectively.

Lemma 14.1 implies that the absolute value $|\det(B)|$ of the determinant of a basis B of Λ is an invariant of Λ. This value is called the *determinant* of Λ. The set $\Pi(B) = \{B\lambda : \lambda \in \mathbb{R}^n, 0 \leqslant \lambda < 1\}$ is called the *parallelepiped* spanned by the

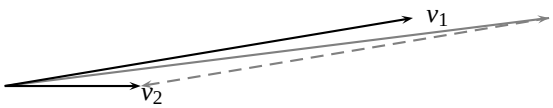
columns of B. The volume of this parallelepiped is the absolute value of the determinant of B. Thus the lattice determinant is the volume of the parallelepiped defined by the basis elements of any basis, see Figure 14.2.

The result of the multiplication of B by a unimodular matrix U can also be obtained by a sequence of *elementary column operations* on B:

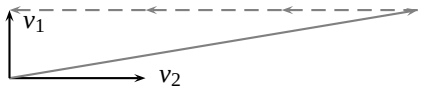
i) Swap of two columns.
ii) Multiplication of a column with -1.
iii) Addition of an *integral* multiple of one column to *another column*.

Now that we know which operations to apply, let us consider an example of lattice basis reduction. Suppose that we want to determine the shortest vector in the lattice which is generated by the following two column-vectors v_1 and v_2.

It is difficult to guess what the shortest vector of this lattice could be. We subtract v_1 from v_2 and replace v_2 with the outcome of this operation. The result is a new basis.



Still, it is not easy to determine the shortest vector. Next we subtract 3-times v_2 from v_1 and replace v_1 with the outcome of this operation.



Now the shortest vector reveals itself easily. Since the obtained basis of our lattice consists of two orthogonal vectors, the shortest vector of the lattice is the shortest vector of the basis itself. In fact, we have traced above the reduction algorithm of Lagrange which was also described by Gauß [29].

Intuitively it seems clear that the shortest vector problem should still be easy, if the basis is *almost orthogonal*. We will deliver a precise definition of this in Section 14.5, where we describe the LLL-algorithm.

Exercises

1) Prove that $U \in \mathbb{Z}^{n\times n}$ is unimodular if and only if U is non-singular and U^{-1} is a matrix with integer entries.

2) Let $B \in \mathbb{Q}^{n\times n}$ be a lattice basis that consists of pairwise orthogonal vectors. Prove that the shortest vector of $\Lambda(B)$ is the shortest column vector of B.
3) Let $\Lambda, \Lambda' \subseteq \mathbb{Z}^n$ be lattices and suppose that $\Lambda \supseteq \Lambda'$. Show that $\det(\Lambda)$ divides $\det(\Lambda')$.
4) Consider three points $v_1, v_2, v_3 \in \mathbb{Z}^2$ that are not co-linear, i.e., there is no line containing all three points. Show that the triangle spanned by v_1, v_2 and v_3 does not contain an integer point apart from v_1, v_2 and v_3 itself, if and only if the matrix $(v_2 - v_1, v_3 - v_2)$ is unimodular.
   A similar statement cannot be made in $\mathbb{R}^3$. Provide an example of linearly independent integer vectors $v_1, v_2, v_3 \in \mathbb{Z}^3$ such that the simplex $\operatorname{conv}\{0, v_1, v_2, v_3\}$ does not contain an integer point apart from $0, v_1, v_2$ and v_3 and $\det(v_1, v_2, v_3) \neq \pm 1$.
5) (Picks formula) The convex hull $P = \operatorname{conv}\{v_1, \ldots, v_n\}$ of integer points $v_i \in \mathbb{Z}^2$, $i = 1, \ldots, n$ is a convex lattice polygon. Let A, I and B be the area, number of integer points in the interior and boundary of P respectively. Prove Picks formula

$$A = I + B/2 - 1.$$

   Hint: Exercise 4).

14.3 The Hermite normal form

The section is devoted to the algorithmic solution of the following problem

Given a rational matrix $A \in \mathbb{Q}^{m\times n}$ of full row-rank and a rational vector $b \in \mathbb{Q}^m$, decide whether there exists an integral vector $x \in \mathbb{Z}^n$ with $Ax = b$.

This problem is solved with the Hermite normal form of a rational matrix which is central to this chapter. We will also see that the set $\{Ax \colon x \in \mathbb{Z}^n\}$ is also a lattice, namely the lattice that is generated by the Hermite normal form of A.

We motivate the Hermite normal form first via certificates of unsolvability of rational linear equations. Suppose we are given a system $Ax = b$ of linear equations over the reals. How can one certify that this system does not have a solution? The following well known theorem from linear algebra provides a certificate.

Theorem 14.2. *Let $A \in \mathbb{R}^{m\times n}$ be a matrix and $b \in \mathbb{R}^m$ be a vector. The system $Ax = b$ does not have a solution $x \in \mathbb{R}^n$ if and only if there exists a $\lambda \in \mathbb{R}^m$ with $\lambda^T A = 0$ and $\lambda^T b \neq 0$.*

Proof. If x is a solution to the system, then $\lambda^T A = 0$ implies $\lambda^T b = \lambda^T Ax = 0^T x = 0$.

On the other hand, if $Ax = b$ does not have a solution then b is not in the vector space

$$Z = \{z \in \mathbb{R}^m \colon z = Ax, x \in \mathbb{R}^n\}.$$

The vector space Z is the kernel of a matrix $C \in \mathbb{R}^{k\times m}$,

$$Z = \{z \in \mathbb{R}^m : Cz = 0\}.$$

We have $CA = 0$ and $Cb \neq 0$. We can choose a y with $y^T Cb \neq 0$. We then have $y^T CA = 0$, so $\lambda = y^T C$ serves our purpose. □

Now let $A \in \mathbb{Q}^{m \times n}$ be a rational matrix and let $b \in \mathbb{Q}^m$ be a rational vector. What happens if we ask, whether a system $Ax = b$ has an integer solution $x \in \mathbb{Z}^n$?

Clearly $Ax = b$ has an integer solution if and only if $(\alpha \cdot A)x = \alpha \cdot b$ has an integer solution for $\alpha \in \mathbb{R} - \{0\}$. This means that we can multiply A and b with the least common multiple of the denominators of their entries and obtain an integral matrix and an integral vector.

The simplest nontrivial system of rational equations over the integers is one equation and two variables. This case was already considered by Gauß [29], who mentions the following theorem. Recall the notation $u \mid v$ which stands for *u divides v* for integers u and v.

Theorem 14.3. *Let a, b and c be integers where a or b are nonzero. The system*

$$ax + by = c \tag{14.2}$$

has a solution with integers x and y if and only if $\gcd(a,b) \mid c$.

Proof. Let d be a common divisor of a and b, i.e., $d \cdot a' = a$ and $d \cdot b' = b$ with integers a' and b'. If $ax + by = c$, then

$$d(a'x + b'y) = c$$

which implies that $d \mid c$. Thus if (14.2) has an integral solution, then $\gcd(a,b) \mid c$.

On the other hand let $\gcd(a,b) \cdot k = c$ for some integer k. Recall that there exist integers x' and y' with $ax' + by' = \gcd(a,b)$. An integral solution of (14.2) is then $x = kx'$ and $y = ky'$. □

In other words, (14.2) does not have an integral solution, if and only if there exists an $\alpha \in \mathbb{R}$ such that $\alpha \cdot (a,b) \in \mathbb{Z}^2$ and $\alpha \cdot c \notin \mathbb{Z}$. It turns out that this simple observation can be generalized.

Theorem 14.4. *Let $A \in \mathbb{Q}^{m \times n}$ and $b \in \mathbb{Q}^m$. The system*

$$Ax = b \tag{14.3}$$

does not have an integral solution, if and only if there exists a $\lambda \in \mathbb{R}^m$ with $\lambda^T A \in \mathbb{Z}^n$ and $\lambda^T b \notin \mathbb{Z}$.

To prove this theorem, we now introduce the Hermite normal form. A rational matrix $A \in \mathbb{Q}^{m \times n}$ of full row rank is said to be in *Hermite normal form (HNF)* if it has the form $[B \mid 0]$, where B is a nonsingular, lower triangular matrix with non-negative entries, in which each row has a unique maximal entry, located on the diagonal. The following is an example of a matrix in HNF

$$\begin{pmatrix} 2 & 0 & 0 \\ 1 & 3 & 0 \end{pmatrix}$$

Theorem 14.5. *Each rational matrix $A \in \mathbb{Q}^{m\times n}$ of full row-rank can be brought into Hermite normal form by a finite series of elementary column operations.*

Proof. By scaling all entries of A with a positive common multiple α of the denominators, we obtain an integral matrix A'. If $[H' \mid 0]$ is a HNF of A', then $(1/\alpha)[H' \mid 0]$ is an HNF of A. Therefore we can assume without loss of generality that A is integral.

Algorithm 1 computes the Hermite normal form and a corresponding unimodular matrix of an integral matrix with full row rank with the use of the extended Euclidean algorithm. Here the term "update columns i and j" with a 2×2-matrix $\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$ means that the new columns c_i and c_j result from their old values by multiplying (c_i, c_j) with $\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$, i.e., $(c_i, c_j) := (c_i, c_j)\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$.

The function **exggT**(a,b) implements the *extended Euclidean algorithm*, which computes the triple (g,x,y), where $g = \gcd(a,b)$ and x and y are integers with $g = xa + yb$, see, e.g. [51]. Since the matrix $\left(\begin{smallmatrix} x & -b/g \\ y & -a/g \end{smallmatrix}\right)$ is unimodular, the updating step in the algorithm corresponds to a sequence of elementary column operations.

Algorithm 1 HNF of A

Input: $A \in \mathbb{Z}^{m\times n}$ full row rank
Return: $[H \mid 0] \in \mathbb{Z}^{m\times n}$ HNF of A, unimodular matrix $U \in \mathbb{Z}^{n\times n}$ with $A\cdot U = [H \mid 0]$

```
H ← A
U ← I_n
for i = 1 to m do
    for j = i+1 to n do
        if H_{i,j} ≠ 0 then
            (g,x,y) = exggT(H_{i,i}, H_{i,j})
            update columns i and j of H and U with ( x  -H_{i,j}/g )
                                                   ( y   H_{i,i}/g )
        end if
    end for
    for j = 1 to i-1 do
        H_{i,j} = q · H_{i,i} + r (division with remainder, r non-negative)
        update columns j and i of H and U with ( 1  0 ) {reduce entries to the left of diagonal
                                               ( -q 1 )  element H_{i,i}}
    end for
end for
```

□

It is an easy exercise to show that the following invariant holds after each "update columns" operation of Algorithm 1

$$A\cdot U = H.$$

Let us trace the algorithm, when it is executed on the matrix $\left(\begin{smallmatrix}2&3&4\\2&4&6\end{smallmatrix}\right)$. The greatest common divisor of $(2,3)$ is $1=(-1)\cdot 2+1\cdot 3$. Thus we update column 1 and 2 with the matrix $\left(\begin{smallmatrix}-1&-3\\1&2\end{smallmatrix}\right)$, obtaining $\left(\begin{smallmatrix}1&0&4\\2&2&6\end{smallmatrix}\right)$. The transforming matrix U yields $\left(\begin{smallmatrix}-1&-3&0\\1&2&0\\0&0&1\end{smallmatrix}\right)$. Eliminating 4 yields $H=\left(\begin{smallmatrix}1&0&0\\2&2&-2\end{smallmatrix}\right)$ and $U=\left(\begin{smallmatrix}-1&-3&4\\1&2&-4\\0&0&1\end{smallmatrix}\right)$.
Eliminating -2 yields $H=\left(\begin{smallmatrix}1&0&0\\2&2&0\end{smallmatrix}\right)$, $U=\left(\begin{smallmatrix}-1&-4&1\\1&4&-2\\0&-1&1\end{smallmatrix}\right)$
Now reducing 2 in the lower left corner yields the Hermite normal form $H=\left(\begin{smallmatrix}1&0&0\\0&2&0\end{smallmatrix}\right)$ and the unimodular matrix transformation matrix $U=\left(\begin{smallmatrix}3&-4&1\\-3&4&-2\\1&-1&1\end{smallmatrix}\right)$ with $A\cdot U=H$.

Proof (of Theorem 14.4). Suppose $Ax=b$ has an integral solution $x\in\mathbb{Z}^n$. Then with $\lambda^T A\in\mathbb{Z}^n$ one has $\lambda^T b=\lambda^T Ax\in\mathbb{Z}$.

Suppose now that $Ax=b$ does not have an integral solution. If $Ax=b$ is not solvable in $\mathbb{R}^n$, then there exists a $\lambda\in\mathbb{R}^m$ with $\lambda^T A=0$ and $\lambda^T b\neq 0$ and thus a λ with $\lambda^T A=0$ and $\lambda^T b=1/2$. Thus we can assume that $Ax=b$ has a solution in $\mathbb{R}^n$ and therefore we can assume that A has full row-rank.

There exists a unimodular matrix $U\in\mathbb{Z}^{n\times n}$ such that $A\cdot U=[B\mid 0]$ is in HNF. Thus $B^{-1}A$ is an integral matrix. We claim that $B^{-1}b$ is not an integral vector and thus that there exists a row λ^T of B^{-1} with $\lambda^T A\in\mathbb{Z}^n$ and $\lambda^T b\notin\mathbb{Z}$.

If $B^{-1}b$ is integral, then

$$A\cdot U\begin{pmatrix}B^{-1}b\\0\end{pmatrix}=[B\mid 0]\begin{pmatrix}B^{-1}b\\0\end{pmatrix}=b$$

shows that $x=U\cdot\begin{pmatrix}B^{-1}b\\0\end{pmatrix}$ is an integral solution of $Ax=b$, which is a contradiction. □

The next theorem shows that the HNF is unique. For this we extend the notation $\Lambda(A)=\{Ax\colon x\in\mathbb{Z}^n\}$ also for the case where $A\in\mathbb{Q}^{m\times n}$ is a rational matrix of full row-rank which is not necessarily a basis. Such a set is also a lattice, since $\Lambda(A)=\Lambda(B)$, where $[B\mid 0]$ is the HNF of A and B is nonsingular.

Theorem 14.6. *Let A and A' be integral matrices of full row-rank, with HNF $[B\mid 0]$ and $[B'\mid 0]$ respectively. Then $\Lambda(A)=\Lambda(A')$ if and only if $B=B'$.*

Proof. If $B=B'$, the clearly $\Lambda(A)=\Lambda(A')$.

It remains to prove that if B and B' are different, then $\Lambda(B)\neq\Lambda(B')$. So assume that B and B' nevertheless generate the same lattice Λ. Let i be the smallest row-index such that row i of B differs from row i of B'. Without loss of generality assume that $B_{i,j}>B'_{i,j}$ holds for some $1\leqslant j<i$. Observe that the i-th components of elements of Λ whose first $i-1$ components are 0, are integral multiples of $B_{i,i}$. Notice that the first $i-1$ components of the vector $\beta=B^{(j)}-B'^{(j)}\in\Lambda$ are 0, where $B^{(j)}$ and $B'^{(j)}$ denote the j-th column of B and B' respectively. But the i-th component of β is strictly between 0 and $B_{i,i}$. This is a contradiction. □

Corollary 14.1. *The Hermite normal form of a rational matrix with full row-rank is unique.*

Clearly, Algorithm 1 requires $O(m \cdot n)$ extended gcd-computations and $O(m^2 \cdot n)$ arithmetic operations. But is it a polynomial algorithm? This cannot be argued, since the binary encoding length of the numbers could grow exponentially. The *size* of an integer z is the number of bits which are required to encode z. We define $\text{size}(z) = 1 + \lceil \log_2(|z|+1) \rceil$. Likewise, the size of a matrix $A \in \mathbb{Z}^{m \times n}$ is the number of bits needed to encode A, i.e., $\text{size}(A) = mn + \sum_{i,j} \text{size}(a_{ij})$, see [75, p. 29]. In fact Fang and Havas [23] provide examples of pivoting schemes, where the size of the numbers in the intermediate steps of the HNF-algorithm above grows exponentially.

Nevertheless, it can be shown that the HNF of a rational matrix can be computed in polynomial time, see [47, 76]. The key to Schrijver's method [76] is the following lemma.

Lemma 14.2. *Let $A \in \mathbb{Z}^{m \times n}$ be a matrix of full row rank and let $d = \det(\Lambda(A))$ be the lattice determinant of $\Lambda(A)$ and let D be a multiple of d. Then*

$$\Lambda(A) = \Lambda([A \mid D \cdot I_m]).$$

Proof. Clearly $\Lambda(A) \subseteq \Lambda([A \mid D \cdot I_m])$. Let Λ denote $\Lambda(A)$. For the reverse inclusion we simply have to show that $D \cdot e_i \in \Lambda$ for each unit vector e_i, $i = 1, \ldots, m$. Let $B \in \mathbb{Z}^{m \times m}$ be a basis of Λ and let $\widetilde{B} \in \mathbb{Z}^{m \times m}$ be the adjoint of B. All column vectors of $B \cdot \widetilde{B}$ are elements of Λ. But Cramers rule says $B^{-1} = (1/\det(B))\widetilde{B}$. Therefore $B \cdot \widetilde{B} = \det(B) \cdot I_m$. Since $d = |\det(B)|$ we see that $d \cdot e_i \in \Lambda$, for each unit vector e_i, $i = 1, \ldots, n$. Since $d \mid D$ we also have that $D \cdot e_i \in \Lambda$. □

Corollary 14.2. *Let $[H \mid 0]$ be the HNF of $A \in \mathbb{Z}^{m \times n}$, where A has full row-rank and let $d = \det(\Lambda(A))$ be the determinant of $\Lambda(A)$. Let $[H' \mid 0]$ be the HNF of $[A \mid d \cdot I_m] \in \mathbb{Z}^{m \times (n+m)}$. Then $H' = H$.*

Theorem 14.7. *There exists a polynomial time algorithm which computes the HNF of a rational matrix of full row-rank.*

Proof. Since we can scale A with the product of the denominators of A we can assume without loss of generality that A is an integral matrix. We start by identifying m linearly independent columns of A and by computing $D = |\det(A)|$. This can be done with Gaussian elimination in polynomial time [17]. The fact that the encoding length of D is polynomial follows from the Hadamard bound (14.9) which we discuss later. Exercise 14.2.3) shows that $D \mid \det(\Lambda(A))$.

We compute the HNF of $[A \mid D \cdot I_m]$ as in Algorithm 1, but keeping numbers in a row reduced modulo D until we eliminate D in this row stemming from $D \cdot I_m$.

The important observation is this. The first i rows of H remain unchanged after the i-th run through the first **for** loop. The remaining rows of H can be kept reduced modulo D, since the last $m - i$ columns of H are of the form $\begin{pmatrix} 0 \\ d \cdot I_{m-i} \end{pmatrix}$. This procedure requires a polynomial amount of extended-gcd computations and arithmetic operations on numbers of size at most $\text{size}(D)$. □

Theorem 14.8. *There exists a polynomial algorithm which computes the HNF $[H \mid 0]$ of a rational matrix $A \in \mathbb{Q}^{m \times n}$ and the corresponding unimodular matrix $U \in \mathbb{Z}^{n \times n}$ with $A \cdot U = [H \mid 0]$.*

Proof. Select m linearly independent columns of A using Gaussian elimination. Assume without loss of generality that those are the first m columns. The matrix $[H \mid 0]$ is the HNF of A if and only if the HNF of $A' = \begin{bmatrix} A \\ 0 \mid I_{n-m} \end{bmatrix}$ is of the form $H' = \begin{bmatrix} H \mid 0 \\ B \end{bmatrix}$ with some matrix B. Compute H' with the polynomial modification of Algorithm 1. We have $A' \cdot U = H'$ and since A' is nonsingular we have $U = H' \cdot A'^{-1}$. Clearly $A \cdot U = [H \mid 0]$. □

The proof of the next theorem is left as an excise.

Theorem 14.9. *There exists a polynomial algorithm for the following problem*

Given an integral matrix $A \in \mathbb{Z}^{m \times n}$ and an integral vector $b \in \mathbb{Z}^m$, compute a solution $\hat{x} \in \mathbb{Z}^n$ of $Ax = b$ or establish that no such $\hat{x}$ exists.

Notes

Showing that Gaussian elimination is in fact a polynomial-time algorithm is not trivial. Edmonds [17] has shown that, if one keeps the numerators and denominators in the intermediate steps of the algorithm gcd-free, then the size of the numbers remains polynomial. Von zur Gathen and Sieveking [82] showed that systems of rational equations over the integers can be solved in polynomial time.

In fact, there is an arsenal of ways to do linear algebra in polynomial time, one of them being a modular method again. If one wants to compute the determinant of an integral matrix $A \in \mathbb{Z}^{n \times n}$ for example, one could apply the Hadamard bound (14.9) to compute a $D \in \mathbb{Z}$ with $2 \cdot |\det(A)| < D$. If we now know a number $x \in \mathbb{Z}$, $0 \leqslant x < D$ with $x \equiv \det(A) \pmod{D}$, then we can retrieve $\det(A)$ from x. In fact, if $x \leqslant D/2$, then $\det(A) = x$ and if $x > D/2$, then $\det(A) = x - D$. One computes with the sieve of Erathostenes the first k prime numbers $p_1, \ldots, p_k$ such that $p_1 \cdots p_k \geqslant D$ holds. The prime-number theorem guarantees that the *value* of p_k is polynomial in the encoding length of D. One can then compute the value $\det(A) \pmod{p_i}$ for each p_i with Gaussian elimination in the field $\mathbb{Z}_{p_i}$ and reconstruct x with the Chinese remainder theorem, see, e.g., [81].

Exercises

1) Compute by hand the HNF of $\left(\begin{smallmatrix} 3 & 5 & 7 & 2 \\ 2 & 4 & 9 & 3 \end{smallmatrix}\right)$.
2) Write a computer program which implements Algorithm 1.
3) Prove Theorem 14.9.
4) Let $D \in \mathbb{N}_+$ be an integer which is not necessarily prime and let $A \in \mathbb{Z}^{m \times n}$ and $b \in \mathbb{Z}^m$ be an integer matrix and vector respectively. Show that there is a polynomial algorithm which computes a solution to the system

$$Ax = b \pmod{D}, x \in \mathbb{Z}^n$$

or asserts that the system does not have a solution.

5) Describe a unimodular matrix G such that the step "update columns" j and i of H and U with $\left(\begin{smallmatrix} 1 & 0 \\ -q & 1 \end{smallmatrix}\right)$ in Algorithm 1 corresponds to the multiplication of H and U with G from the right.
6) Modify Algorithm 1 such that it detects on the fly whether A has full row-rank and possibly determines a row which is in the span of the other rows.

14.4 Minkowski's theorem

At the end of the 19-th century, Minkowski opened the stage for the geometry of numbers with his famous book *Geometrie der Zahlen* [61]. One of his main results is described in this section. He used geometric methods to prove upper bounds on numbers which are representable by positive definite quadratic forms. In the setting of lattices his result means that a lattice $\Lambda \subseteq \mathbb{R}^n$ contains a nonzero lattice point whose norm is bounded roughly by $\sqrt{\frac{2}{\pi e}n}\det(\Lambda)^{1/n}$. The simplicity and elegance of his approach is stunning. His result was preceded by a sequence of bounds which we briefly discuss. Lagrange and Gauß [29] proved that a 2-dimensional lattice has a nonzero lattice point whose norm is bounded by $\sqrt{4/3}\det(\Lambda)^{1/2}$. In his *Disquisitiones Arithmeticae* [29] Gauß proved that a 3-dimensional lattice has a nonzero lattice vector of norm bounded by $\sqrt{4/3}\det(\Lambda)^{1/3}$. Hermite [37] generalized this result to arbitrary dimension and provided the upper bound $(4/3)^{(n-1)/4}\det(\Lambda)^{1/n}$. All these results were algorithmic, in the sense that they provided algorithms computing nonzero lattice vectors achieving these bounds. It is remarkable that these algorithms run in polynomial time, if the dimension is fixed. The bound obtained by Minkowski is much stronger than the one of Hermite. It is however not known how to compute a nonzero lattice vector satisfying this bound in polynomial time, if the dimension is not fixed.

A *convex body* is a compact and full-dimensional convex set $K \subseteq \mathbb{R}^n$. In its simplest form, Minkowski's theorem is as follows.

Theorem 14.10. *Let $K \subseteq \mathbb{R}^n$ be a convex body which is symmetric around the origin ($x \in K$ implies $-x \in K$). If $\mathrm{vol}(K) \geqslant 2^n$, then K contains a nonzero integral vector $v \in \mathbb{Z}^n \setminus \{0\}$.*

Proof. We first argue that it suffices to assume that $\mathrm{vol}(K) > 2^n$ holds. If $\mathrm{vol}(K) = 2^n$, then, since K is compact, there exists an $\varepsilon > 0$ such that the distance to K of each integer point that is not contained in K is at least ε. This means that there exists a constant $\delta > 0$ such that the sets $(1+\delta)\cdot K$ and K contain the same integer points. The volume of $(1+\delta)\cdot K$ however is strictly larger than 2^n.

Suppose now that the theorem does not hold. Consider the set $S = 1/2 \cdot K$ and the translates of S with integer vectors

$$S + v, v \in \mathbb{Z}^n.$$

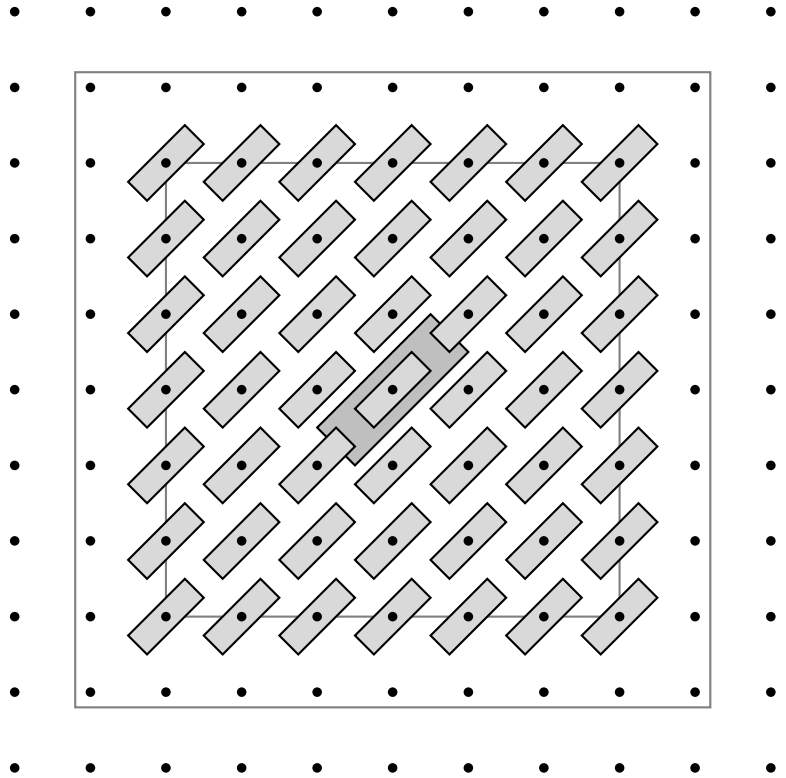


Fig. 14.3 An illustration of the proof of Minkowski's theorem, where the dark rectangle stands for K and the light rectangles for the sets $S+v$. The inner square represents the points $x \in \mathbb{R}^2$ with $\|x\|_\infty \leqslant M$ and the outer square represents the set (14.5), being the points $x \in \mathbb{R}^2$ with $\|x\|_\infty \leqslant M+D$.

If $S+v_1 \cap S+v_2 \neq \emptyset$ for some integral vectors $v_1 \neq v_2$, then there exist $k_1, k_2 \in K$ such that

$$0 \neq v_1 - v_2 = 1/2(k_2 - k_1).$$

Due to the symmetry of K around the origin we have $-k_1 \in K$ and due to the convexity of K we have $1/2(k_2-k_1) \in K$, which is a nonzero integral point. This shows that the translates $S+v$, $v \in \mathbb{Z}^n$ do not intersect.

Now we consider the volume

$$V_M = \text{vol}\left(\bigcup_{\substack{v \in \mathbb{Z}^n \\ \|v\|_\infty \leqslant M}} S+v \right)$$

for some $M \in \mathbb{N}$. Since the $S+v$ do not intersect we have

$$\begin{aligned} V_M &= \sum_{v \in \mathbb{Z}^n,\, \|v\|_\infty \leqslant M} \mathrm{vol}(S) \\ &= (2 \cdot M + 1)^n \cdot \mathrm{vol}(S). \end{aligned} \tag{14.4}$$

Since S is bounded, S has finite diameter $D \in R$. This means that the union

$$\bigcup_{\substack{v \in \mathbb{Z}^n \\ \|v\|_\infty \leqslant M}} S + v$$

is a subset of

$$\{x \in \mathbb{R}^n \colon \|x\|_\infty \leqslant M + D\}. \tag{14.5}$$

The volume of the set in (14.5) is $2^n \cdot (M+D)^n$. Therefore the inequality

$$2^n \cdot (M+D)^n \geqslant (2 \cdot M + 1)^n \cdot \mathrm{vol}(S)$$

must hold. As M tends to infinity, the expression

$$\frac{(2 \cdot M + 2 \cdot D)^n}{(2 \cdot M + 1)^n}$$

tends to one. This is a contradiction, since $\mathrm{vol}(S) > 1$. □

Minkowski's theorem has also a version for general lattices. Let $\Lambda(B) \subseteq \mathbb{R}^n$ be a lattice and $K \subseteq \mathbb{R}^n$ be a convex set which is symmetric around the origin with $\mathrm{vol}(K) \geqslant 2^n \det(\Lambda)$. The mapping $\phi(x) = B^{-1}x$ maps Λ to $\mathbb{Z}^n$ and $\phi(K)$ is a convex body which is symmetric around the origin. The volume of $\phi(K)$ is $\mathrm{vol}(\phi(K)) = (1/\det(B))\,\mathrm{vol}(K)$ and thus $\mathrm{vol}(\phi(K)) \geqslant 2^n$. Theorem 14.10 implies that $\phi(K)$ contains a nonzero integer vector or equivalently K contains a nonzero lattice vector from Λ.

Theorem 14.11 (Minkowski's convex body theorem [61]). *Let $\Lambda \subseteq \mathbb{R}^n$ be a lattice and let $K \subseteq \mathbb{R}^n$ be a convex body of volume $\mathrm{vol}(K) \geqslant 2^n \det(\Lambda)$ that is symmetric about the origin. Then K contains a nonzero lattice point.*

As announced in the introduction of this section, Minkowski's theorem can be used to derive an upper bound on the length of a shortest vector of a lattice Λ in terms of the determinant of Λ. Let V_n be the volume of the n-dimensional unit ball. By scaling the unit ball with $\alpha \in \mathbb{R}_+$ one obtains a ball of volume $\alpha^n \cdot V_n$. This is greater or equal to $2^n \det(\Lambda)$ for $\alpha \geqslant 2 \cdot \sqrt[n]{\det(\Lambda)/V_n}$. This has the following consequence.

Theorem 14.12. *A lattice $\Lambda \subseteq \mathbb{R}^n$ has a nonzero lattice point of length less than or equal to $2 \cdot \sqrt[n]{\det(\Lambda)/V_n}$.*

The formula for V_n is

$$V_n = \frac{\pi^{\lfloor n/2 \rfloor} 2^{\lceil n/2 \rceil}}{\prod_{0 \leqslant 2i \leqslant n} (n - 2i)}.$$

Using Stirling's formula ($n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$) one sees that this is roughly $\left(\frac{2\pi e}{n}\right)^{n/2}$. The bound of Theorem 14.12 is thus roughly $\sqrt{\frac{2}{\pi e}n}\det(\Lambda)^{1/n}$.

Exercises

1) Show that a lattice Λ has a nonzero lattice point v with $\|v\|_\infty \leqslant \sqrt[n]{\det(\Lambda)}$.
2) Show that Minkowski's theorem holds also for convex sets that are full-dimensional and closed, i.e., the boundedness condition is not really necessary.
3) Let $K \subseteq \mathbb{R}^n$ be a convex body of volume $\mathrm{vol}(K) \geqslant k \cdot 2^n$. Show that K contains at least $2 \cdot k$ nonzero integer points.
4) (Two squares theorem) In this exercise you will prove that a prime number p with $p \equiv 1 \pmod 4$ can be written as the sum of two square numbers $p = a^2 + b^2$ for $a, b \in \mathbb{N}$.
   a) Show that the equation $q^2 \equiv -1 \pmod p$ has a solution.
   b) Consider the lattice Λ generated by $\left(\begin{smallmatrix} 1 & 0 \\ q & p \end{smallmatrix}\right)$ and the disk of radius $\sqrt{p \cdot 2 - \varepsilon}$ around 0 for a small $\varepsilon > 0$.
      i) Show that $\|v\|^2$ is divisible by p for each $v \in \Lambda$.
      ii) Show that there exists a $v \in \Lambda \setminus \{0\}$ with $\|v\|^2 = p$.
      iii) Conclude that p is the sum of two squares.

   Hints: Wilson's theorem, see, e.g. [65] states $(p-1)! \equiv -1 \pmod p$. If $q^2 \equiv -1$ does not have a solution, then $\mathbb{Z}_p^*$ can be paired (perfectly matched) into $(p-1)/2$ pairs, where the product of each pair is congruent to -1. But $(p-1)/2$ is even ($p \equiv 1 \pmod 4$), implying $(p-1)! \equiv 1 \pmod p$, contradicting Wilson's theorem.

14.5 The LLL algorithm

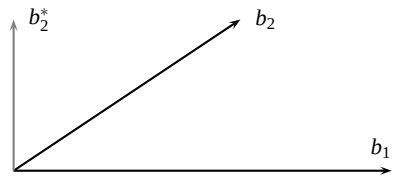
Recall the intuition from the introductory section on basis reduction. We want to transform a lattice basis into an equivalent one that is almost orthogonal. The hope is that a shortest vector can be determined from such a basis more easily. For a clear understanding of what almost orthogonal should mean we first recall the Gram-Schmidt procedure.

Let $U \subseteq \mathbb{R}^n$ be a subspace of $\mathbb{R}^n$ and let $v \in \mathbb{R}^n$. The *projection* of v *onto the orthogonal complement of* U is a vector $v^* = v - h$, where $h \in U$ and $\langle v - h, u\rangle = 0$ for each $u \in U$. If $U =< x_1, \ldots, x_k >$ is the subspace generated by $x_1, \ldots, x_k$, then $v - h$ is also called the projection of v onto the orthogonal complement of $x_1, \ldots, x_k$.

The Gram-Schmidt procedure computes a set of vectors $b_1^*, \ldots, b_n^*$ such that the following conditions hold.

i) The vectors $b_1, \ldots, b_k$ span the same subspace as $b_1^*, \ldots, b_k^*$ for each $k = 1, \ldots, n$.

Fig. 14.4 The projection b_2^* of b_2 onto to the orthogonal complement of the subspace generated by b_1.



ii) The vectors $b_1^*, \ldots, b_n^*$ are pairwise orthogonal.

The procedure is described below. It is easy to see that i) and ii) hold for B^*.

Algorithm 2 Gram-Schmidt orthogonalization

Input: $B = (b_1, \ldots, b_n) \in \mathbb{R}^{n \times n}$ nonsingular
Return: $B^* = (b_1^*, \ldots, b_n^*) \in \mathbb{R}^{n \times n}$ satisfying conditions i) and ii)
  $b_1^* \leftarrow b_1$
  for $j = 2, \ldots, k$ **do**
    $b_j^* \leftarrow b_j - \sum_{i=1}^{j-1} \mu_{ji} b_i^*$, where $\mu_{ji} = \langle b_j, b_i^* \rangle / \|b_i^*\|^2$
  end for

The Gram-Schmidt orthogonalization procedure decomposes the matrix $B = (b_1, \ldots, b_n)$ into

$$B = B^* \cdot R \tag{14.6}$$

where B^* is a matrix with pairwise orthogonal columns $b_1^*, \ldots, b_n^*$ and

$$R = \begin{pmatrix} 1 & & \mu \\ & \ddots & \\ 0 & & 1 \end{pmatrix}$$

is an upper triangular matrix with diagonal elements 1. The decomposition (14.6) is the *Gram-Schmidt orthogonalization (GSO)* of B.

From the Gram-Schmidt orthogonalization procedure, we can also deduce the so-called Hadamard bound. Notice that

$$\det(B) = \det(B^* \cdot R) = \det(B^*). \tag{14.7}$$

Since $\det(B^* \cdot B^{*T}) = \|b_1^*\|^2 \cdots \|b_n^*\|^2$ and since $\det(B^* \cdot B^{*T}) = \det(B^*)^2$ we conclude that

$$|\det(B)| = \prod_{i=1}^{n} \|b_i^*\|. \tag{14.8}$$

Since we have $\|b_i\| \geqslant \|b_i^*\|$ for $i = 1, \ldots, n$ we obtain the *Hadamard bound*

$$|\det(B)| \leqslant \prod_{i=1}^{n} \|b_i\|. \tag{14.9}$$

We can use equation (14.9) now to measure how far a lattice basis B deviates from being orthogonal. The *orthogonality defect* of the lattice basis B is the number γ such that

$$\gamma \cdot |\det(B)| = \prod_{i=1}^{n} \|b_i\| \tag{14.10}$$

holds. The columns of B are pairwise orthogonal if and only if $\gamma = 1$. With the next theorem, we connect the computation of a shortest vector to this orthogonality defect. It implies that a shortest vector can be found among $(2 \cdot \gamma + 1)^n$ candidates.

Theorem 14.13. *Let $B \in \mathbb{Q}^{n \times n}$ be a lattice basis with orthogonality defect γ. A shortest non-zero vector $v \in \Lambda(B)$ is of the form*

$$v = \sum_{i=1}^{n} x_i \cdot b_i, \text{ with } x_i \in \mathbb{Z}, -\gamma \leqslant x_i \leqslant \gamma. \tag{14.11}$$

Proof. Since $\|b_j\| \geqslant \|b_j^*\|$ for $j = 1, \ldots, n$ and since

$$\|b_1\| \cdots \|b_n\| = \gamma \cdot \|b_1^*\| \cdots \|b_n^*\|,$$

we can conclude that

$$\|b_n\| \leqslant \gamma \cdot \|b_n^*\| \tag{14.12}$$

holds. Consider now $v = B \cdot x$ in (14.11). Since $B \cdot x = B^* \cdot R \cdot x$, where B^* is the Gram-Schmidt orthogonalization of B, and since the last component of $R \cdot x$ is equal to x_n, we conclude that

$$\|B \cdot x\| \geqslant |x_n| \cdot \|b_n^*\| \geqslant (|x_n|/\gamma)\|b_n\|. \tag{14.13}$$

From this we can conclude that, if v is a shortest vector, then x_n satisfies $-\gamma \leqslant x_n \leqslant \gamma$.

The orthogonality defect is invariant under the permutation of columns of B. Therefore each vector in the basis can play the role of being the last vector in the above argument. This implies that each component x_i in (14.11) satisfies $-\gamma \leqslant x_i \leqslant \gamma$ which implies the assertion. □

This implies that the shortest vector problem can be solved in fixed dimension, if we can compute a lattice basis B' of $\Lambda(B)$ whose orthogonality defect is bounded by a constant, depending on the dimension only. Therefore we call a lattice basis $B \in \mathbb{Q}^{n \times n}$ *reduced* if its orthogonality defect is bounded by a constant γ_n, depending only on the dimension n.

The LLL-algorithm[55] is an algorithm which reduces a lattice basis in polynomial time. The orthogonality defect of an LLL-reduced basis can be bounded by $2^{n(n-1)/4}$. Consequently, the shortest vector problem can be solved in time $2^{O(n^3)}$ times a polynomial in the binary encoding length of the input.

Normalization

Let $B = B^* \cdot R$ be the GSO of B. If R would be the identity matrix, then B would be B^* and thus orthogonal. The first step of the LLL-algorithm is *normalization*: One applies elementary column operations to transform the upper triangular matrix R into a matrix that is as close as possible to the identity matrix. Normalization is in the literature also sometimes referred to as *size reduction*.

Let r_{ij} be the j-th entry of the i-th row of R. By subtracting $\lfloor r_{ij} \rceil$ times the i-th column of R from the j-th column, the new entry r'_{ij} at position ij will satisfy $-1/2 < r'_{ij} \leqslant 1/2$. Notice that the entries in a row below the i-th row of R remain unchanged. Thus working our way from the last to the first row, we obtain a basis $B' = B^* \cdot R'$ with

$$-1/2 < r'_{ij} \leqslant 1/2, \text{ for } 1 \leqslant i < j \leqslant n. \tag{14.14}$$

This procedure is called a *normalization step*.

Swapping

The LLL-algorithm iterates normalization and swapping steps. More precisely it normalizes the basis and then searches for two consecutive basis elements which should be swapped. This its continued, until a certain condition holds.

Algorithm 3 LLL algorithm

Repeat the following two steps, as long as there exists a j, $1 \leqslant j \leqslant n-1$ with

$$\|b^*_{j+1} + \mu_{j+1,j} b^*_j\|^2 < 3/4 \|b^*_j\|^2 : \tag{14.15}$$

Normalize B
Swap b_j and b_{j+1}

Notice that the condition (14.15) is invariant under normalization, since B^* is left untouched by the normalization procedure. Let us shed some light on the swapping operation and on this condition (14.15) that has to hold when swapping is applied. The vector $b^*_{j+1} + \mu_{j+1,j} b^*_j$ is the new j-th vector of B^* after the swap because

$$b^*_{j+1} = b_{j+1} - \sum_{i=1}^{j} \mu_{j+1,i} b^*_i. \tag{14.16}$$

Thus the vector $b^*_{j+1} + \mu_{j+1j} b^*_j$ is the projection of b_{j+1} onto the orthogonal complement of $b_1, \ldots, b_{j-1}$.

The condition (14.15) ensures that the norm of this new j-th column has decreased by a factor of $3/4$ at least. Since the vectors b^*_μ for $\mu \neq j, j+1$ remain the same (see exercise 14.5.2), the only side effect is an increase of the norm of the $j+1$-st column of B^*. The rest of the GSO remains unchanged.

More precisely, if the norm j-th column decreases by a factor of α then the $j+1$-st column increases by a factor of $1/\alpha$ since the product of the norms of the columns of B^* is equal to $|\det(B)|$ which is left invariant by a swap of columns in B.

Analysis

The above observation allows us now to show that the algorithm runs in polynomial time. The *potential* of a lattice basis B is defined as

$$\phi(B) = \|b_1^*\|^{2n}\|b_2^*\|^{2(n-1)}\|b_3^*\|^{2(n-2)}\cdots\|b_n^*\|^2. \tag{14.17}$$

A normalization step does not change the potential, since B^* remains unchanged. How does a swap affect the potential? Let B' be the basis after a swap operation, where the j-th and $j+1$-st column are swapped. We have (see exercise 2)

$$\begin{aligned}\frac{\phi(B)}{\phi(B')} &= \frac{\|b_j^*\|^{2\cdot(n-j+1)}\|b_{j+1}^*\|^{2(n-j)}}{\|{b'_j}^*\|^{2\cdot(n-j+1)}\|{b'_{j+1}}^*\|^{2(n-j)}} \\ &= \frac{\|b_j^*\|^{2\cdot(n-j)}\|b_{j+1}^*\|^{2(n-j)}}{\|{b'_j}^*\|^{2\cdot(n-j)}\|{b'_{j+1}}^*\|^{2(n-j)}} \cdot \frac{\|b_j^*\|^2}{\|{b'_j}^*\|^2} \\ &= \frac{\|b_j^*\|^2}{\|{b'_j}^*\|^2} \\ &\geqslant \frac{4}{3}.\end{aligned} \tag{14.18}$$

This shows that the potential drops at every iteration by at least $3/4$. Next we show that the potential of a lattice basis $B \in \mathbb{Z}^{n\times n}$ is an integer.

Let B_i be the matrix consisting of the first i columns of B. Then we have

$$\det(B_i^T \cdot B_i) = \|b_1^*\|^2\cdots\|b_i^*\|^2 \in \mathbb{N}. \tag{14.19}$$

Consequently we have

$$\phi(B) = \prod_{i=1}^{n}\det(B_i^T \cdot B_i) \in \mathbb{N}. \tag{14.20}$$

This shows that the potential is bounded from below by 1 and the algorithm terminates in $O(\log\phi(B))$ steps. Clearly $\phi(B) \leqslant \det(B)^{2\cdot n} \leqslant (\sqrt{n}M)^{2\cdot n^2}$, where M is an upper bound on the absolute value of an entry in B.

We thus have the following theorem.

Theorem 14.14. *The LLL-algorithm terminates in $O(n^2(\log n + s))$ iterations, where s is the largest binary encoding length of a coefficient of $B \in \mathbb{Z}^{n\times n}$.*

In order to conclude that the LLL-algorithm runs in polynomial time, we also have to bound the binary encoding length of the numbers which occur in the intermediate steps of the algorithm. This is very important but a bit tedious and we don't do it here and refer, for example to [55]. We conclude with the following theorem.

Theorem 14.15. *The LLL-algorithm runs in polynomial time in the input encoding length of the initial lattice basis.*

Orthogonality defect and approximation of the shortest vector

The next theorem bounds the length of a shortest vector from below by means of the GSO. Here, $SV(\Lambda)$ denotes the length of a shortest nonzero vector of Λ.

Theorem 14.16. *Let B be a lattice basis and let $B^* = (b_1^*, \ldots, b_n^*)$ be its Gram-Schmidt orthogonalization, then $SV(\Lambda(B)) \geqslant \min_{i=1,\ldots,n} \|b_i^*\|_2$.*

Proof. Let $0 \neq v \in \Lambda$, then $v = Bx$ and let $k \leqslant n$ be the largest index with $x_k \neq 0$. With $B = B^* \cdot R$ we have

$$\begin{aligned} v &= B^* \cdot R \cdot x \\ &= x_k b_k^* + \sum_{i=1}^{k-1} \lambda_i b_i^*, \text{ for some } \lambda_i \in \mathbb{R}. \end{aligned}$$

This implies $\|v\| \geqslant |x_k| \, \|b_k^*\|$ and the theorem follows. □

With this lower bound, we can show that the first vector of an LLL-reduced basis is an approximation of the shortest vector, which is exponential in the dimension only.

Upon termination of the LLL-algorithm we have that each $\mu_{j+1,j}^2 \leqslant 1/4$. Since also $\|b_{j+1}^* + \mu_{j+1,j} b_j^*\|^2 \geqslant 3/4 \|b_j^*\|^2$ and since $\|b_{j+1}^* + \mu_{j+1,j} b_j^*\|^2 = \|b_{j+1}^*\|^2 + \mu_{j+1,j}^2 \|b_j^*\|^2$ we have

$$\|b_{j+1}^*\|^2 \geqslant 1/2 \|b_j^*\|^2 \tag{14.21}$$

for each $j = 1, \ldots, n-1$. Thus it follows by induction that

$$\|b_j^*\|^2 \geqslant 2^{i-j} \|b_i^*\|^2, \text{ for } 1 \leqslant i < j \leqslant n. \tag{14.22}$$

From this we can conclude that

$$\|b_1\|^2 = \|b_1^*\|^2 \leqslant 2^{n-1} SV(\Lambda(B)). \tag{14.23}$$

Also since

$$b_j = b_j^* + \sum_{i=1}^{j-1} \mu_{ji} b_i^* \tag{14.24}$$

and since each μ_{ji} has absolute value less that $1/2$ we have

$$\|b_j\|^2 \leqslant \|b_j^*\|^2 + 1/4 \sum_{i=1}^{j-1} \|b_i^*\| \tag{14.25}$$

and by applying (14.22) we obtain

$$\|b_j\|^2 \leqslant \|b_j^*\|^2 (1 + 1/4 \sum_{i=1}^{j-1} 2^{j-i}) \leqslant 2^{j-1} \|b_j^*\|^2. \tag{14.26}$$

This implies the following for the orthogonality defect

$$\|b_1\| \cdots \|b_n\| \leqslant 2^{n(n-1)/4} \|b_1^*\| \cdots \|b_n^*\| = 2^{n(n-1)/4} |\det(B)|. \tag{14.27}$$

Together with Theorem 14.13 we thus have the next theorem.

Theorem 14.17. *A shortest vector of an integral lattice can be computed in time* $2^{O(n^3)}$ *times a polynomial in the length of the input encoding.*

Notes

The LLL-algorithm requires $O(n^5 \log B)$ arithmetic operations on rational numbers of size $O(n + \log B)$. Here B is an upper bound on the norm of the basis vectors. Schnorr [72] presented an floating-point variant of the LLL algorithm which requires $O(n^4 \log B)$ arithmetic operations on rationals of size $O(n + \log B)$. Improvements on the algorithm itself and its analysis were given by Kaltofen [40] and Storjohann [79] among others. Using naive arithmetic, the analysis of the LLL-algorithm presented here amounts to a bit-complexity of $O(n^4 \log B(n + \log B)^2)$. Recently, Nguyen and Stehlé [63] have presented a floating-point variant of the LLL-algorithm which requires $O(n^5(n + \log B) \log B)$ bit-operations. In fixed dimension, this matches the bit-complexity of the Euclidean algorithm. The greatest common divisor of two s-bit integers can be computed with $O(M(s) \log s)$ bit-operations [73], where $M(s)$ is the number of bit-operations, which are required for the multiplication of two s-bit numbers. For a long time, the fastest method for integer multiplication was the one by Schönhage and Strassen [74], requiring $O(s \log s \log\log s)$ bit-operations. Martin Fürer [26] improved this complexity recently to $O(s \log s \cdot 2^{O(\log^* s)})$. It is an interesting open problem, whether a shortest vector in fixed dimension can be computed with $O(M(s) \log s)$ bit-operations. Eisenbrand and Rote [20] showed that one can compute a shortest vector in fixed dimension n using $O(M(s) \log^{n-1} s)$ bit-operations. Recently Gama and Nguyen [27] proved that using a shortest vector oracle in dimension k, one can compute a $((1+\varepsilon)\gamma_k)^{(n-k)/(k-1)}$ approximation of the shortest vector, where γ_k is the so-called *Hermite constant*.

Exercises

1. Prove that $\|b_i\| \geqslant \|b_i^*\|$ holds for each $i = 1, \ldots, n$ for the GSO of B.
2. Let
$$B = (b_1, \ldots, b_{i-1}, b_i, b_{i+1}, b_{i+2}, \ldots, b_n)$$
and
$$C = (b_1, \ldots, b_{i-1}, b_{i+1}, b_i, b_{i+2}, \ldots, b_n)$$
be two lattice bases. Notice that C originates from B via swapping the i-th and $i+1$-st column. Prove that B^* and C^* only differ in the i-th and $i+1$-st column. Show further that $\|b_i^*\| \cdot \|b_{i+1}^*\| = \|c_i^*\| \cdot \|c_{i+1}^*\|$ holds.

3. Let $B \in \mathbb{R}^{n\times n}$ be a matrix. Prove that $|\det(B)| \leqslant (\sqrt{n}M)^n$, where M is an upper bound on the absolute values of the entries in B. *Hint: Hadamard bound!*
4. Estimate the total number of arithmetic operations which are performed by the LLL-algorithm in terms of $\phi(B)$.
5. Let α be a fixed constant and let $\Lambda = \Lambda(A)$, $A \in \mathbb{Q}^{n\times n}$ be a rational lattice in fixed dimension n.
   a) Prove that one can enumerate all vectors $v \in \Lambda(A)$ with $\|v\| \leqslant \alpha \cdot SV(\Lambda)$ in polynomial time.
   b) Let $\|\cdot\|$ be any fixed norm. Show that a shortest vector of Λ w.r.t. $\|\cdot\|$ can be computed in polynomial time, if $\|v\|$ can be evaluated in polynomial time in the binary input encoding of v for any $v \in \Lambda$.

14.6 Kannan's shortest vector algorithm

As we have mentioned above, one can compute a shortest vector of a lattice that is represented by a LLL-reduced basis $b_1, \ldots, b_n$ in $2^{O(n^3)}$ steps via enumerating the candidates $\sum_{j=1}^{n} \lambda_j b_j$, where $|\lambda_j| \leqslant 2^{n(n-1)/4}$ and choosing the shortest nonzero vector from this set.

Kannan [42, 43] provided an algorithm for the shortest vector problem, whose dependence on the dimension is $2^{O(n\log n)}$. Helfrich [36] improved Kannan's algorithm. Recently, Ajtai, Kumar and Sivakumar [3] presented a randomized algorithm for the shortest vector problem, with an expected dependence of $2^{O(n)}$ which is the subject of the next section. In this section, we describe Kannan's algorithm.

Korkine-Zolotareff reduction

A lattice basis $b_1, \ldots, b_n$ is *Korkine-Zolotareff reduced*, or *K-Z reduced* for short, if the following conditions hold.

i) The vector b_1 is a shortest vector of the lattice generated by $b_1, \ldots, b_n$.
ii) The numbers μ_{jk} in the Gram-Schmidt orthogonalization of $b_1, \ldots, b_n$ satisfy $|\mu_{jk}| \leqslant 1/2$.
iii) If $b'_2, \ldots, b'_n$ denote the projections of $b_2, \ldots, b_n$ onto the orthogonal complement of the space generated by b_1, then $b'_2, \ldots, b'_n$ is Korkine-Zolotareff reduced.

A two-dimensional lattice basis that is K-Z reduced is also called *Gauß* reduced, see [29]. The algorithm of Kannan computes a Korkine-Zolotareff reduced basis in dimension n by first computing a partially Korkine-Zolotareff reduced lattice basis, from which a shortest vector is among $2^{O(n\log n)}$ candidates. The basis is partially Korkine-Zolotareff reduced with the help of an algorithm for Korkine-Zolotareff reduction in dimension $n-1$.

With a shortest vector at hand, one can then compute a fully K-Z reduced basis by K-Z reducing the projection along the orthogonal complement of this shortest vector. A lattice basis $b_1,\ldots,b_n$ is *partially Korkine-Zolotareff reduced* or *partially K-Z reduced* for short, if it satisfies the following properties.

1. If $b'_2,\ldots,b'_n$ denotes the projection of $b_2,\ldots,b_n$ onto the orthogonal complement of the space generated by b_1, then $b'_2,\ldots,b'_n$ is Korkine-Zolotareff reduced.
2. The numbers μ_{jk} in the Gram-Schmidt orthogonalization of $b_1,\ldots,b_n$ satisfy $|\mu_{jk}| \leqslant 1/2$.
3. $\|b'_2\| \geqslant 1/2\,\|b_1\|$.

Notice that, once Conditions 1 and 3 hold, Condition 2 can be satisfied via a normalization step. Normalization does not destroy Conditions 1 and 3. Condition 1 can be satisfied by applying Kannan's algorithm for full K-Z reduction to $b'_2,\ldots,b'_n$, and applying the transformation to the original vectors $b_2,\ldots,b_n$. Then if Condition 3 is not satisfied, then Helfrich [36] has proposed to replace b_1 and b_2 with the *Gauß-reduction* of this pair, or equivalently its K-Z reduction. Clearly, if b_1,b_2 is Gauß-reduced, which means that $\|b_1\| \leqslant \|b_2\|$ and the angle enclosed by b_1 and b_2 is at least 60° and at most 120°, then Condition 3 holds.

The following algorithm computes a partially K-Z reduced basis from a given input basis $b_1,\ldots,b_n$. It uses as a subroutine an algorithm to K-Z reduce the lattice basis $b'_2,\ldots,b'_n$.

Algorithm 4 Partial K-Z reduction

1. Apply the LLL-algorithm to $b_1,\ldots,b_n$.
2. K-Z reduce $b'_2,\ldots,b'_n$ and apply the corresponding transformation to $b_2,\ldots,b_n$.
3. Perform normalization step on $b_1,\ldots,b_n$.
4. If $\|b'_2\| < 1/2\,\|b_1\|$, then replace b_1,b_2 by its Gauß reduction and go to Step 2.

We show in a moment that we can extract a shortest vector from a partially K-Z reduced basis in $2^{O(n\log n)}$ steps, but before, we analyze the running time of the algorithm.

Theorem 14.18 ([36]). *Step 4 of Algorithm 4 is executed at most* $\log n + 6$ *times.*

Proof. Let v be a shortest vector and let $b_1,\ldots,b_n$ be the lattice basis immediately before Step 4 of Algorithm 4 and let $b'_2,\ldots,b'_n$ denote the projection of $b_2,\ldots,b_n$ onto the orthogonal complement of b_1.

If Step 4 is executed, then v is not equal to b_1. Then clearly, the projection of v onto the orthogonal complement of b_1 is nonzero. Since $b'_2,\ldots,b'_n$ is K-Z reduced it follows that $\|v\| \geqslant \|b'_2\|$ holds. Denote the Gauß reduction of b_1,b_2 by $\widetilde{b}_1,\widetilde{b}_2$. The determinant of $\Lambda(b_1,b_2)$ is equal to $\|b_1\|\,\|b'_2\|$. After the Gauß reduction in Step 4, we have therefore

$$\|\widetilde{b}_1\| \leqslant 2\sqrt{\|b_1\|\,\|b_2'\|} \tag{14.28}$$

$$\leqslant 2\sqrt{\|b_1\|\,\|v\|}. \tag{14.29}$$

Dividing this inequality by $\|v\|$ gives

$$\frac{\|\widetilde{b}_1\|}{\|v\|} \leqslant 2\sqrt{\frac{\|b_1\|}{\|v\|}}.$$

Thus, if $b_1^{(i)}$ denotes the first basis vector after the i-th execution of Step 4, one has

$$\frac{\|b_1^{(i)}\|}{\|v\|} \leqslant 4\left(\frac{\|b_1^{(0)}\|}{\|v\|}\right)^{(1/2)^i}. \tag{14.30}$$

Since we start with a LLL-reduced basis, we know that $\|b_1^{(0)}\|/\|v\| \leqslant 2^{(n-1)/2}$ holds, and consequently that $\|b_1^{(\log n)}\|/\|v\| \leqslant 8$. Each further Gauß reduction decreases the length of the first basis vector by at least $3/4$. Therefore the number of runs through Step 4 is bounded by $\log n + 6$. □

Extracting a shortest vector

We now argue that with such a partially K-Z reduced basis $b_1, \ldots, b_n$ at hand, one only needs to check $n^{O(n)}$ candidates for the shortest vector. Let $v = \sum_{j=1}^n \lambda_j b_j$ be a shortest vector. After rewriting each b_j in terms of the Gram-Schmidt orthogonalization one obtains

$$\begin{aligned} v &= \sum_{j=1}^{n}\sum_{k=1}^{j}(\lambda_j \mu_{jk} b_k^*) \\ &= \sum_{k=1}^{n}(\sum_{j=k}^{n}\lambda_j \mu_{jk}) b_k^*, \end{aligned}$$

where the μ_{jk} are as in Algorithm 2.

The length of v satisfies

$$\|v\|^2 = \sum_{k=1}^{n}(\sum_{j=k}^{n}(\lambda_j \mu_{jk}))^2 \|b_k^*\|^2. \tag{14.31}$$

Consider the coefficient $c_n = |\lambda_n \mu_{nn}| = |\lambda_n|$ of $\|b_n^*\|$ in (14.31). We can bound this absolute value by $|\lambda_n| \leqslant \|v\|/\|b_n^*\| \leqslant \|b_1\|/\|b_n^*\|$. This leaves us $1+2\|b_1\|/\|b_n^*\|$ possibilities for λ_n. Suppose now that we picked $\lambda_n, \ldots, \lambda_{j+1}$ and inspect the coefficient c_j of $\|b_j^*\|$ in (14.31), which is

$$\begin{aligned} c_j &= |\sum_{k=j}^{n} (\lambda_k \mu_{kj})| \\ &= |\lambda_j + \sum_{k=j+1}^{n} (\lambda_k \mu_{kj})|. \end{aligned}$$

Since the inequality $c_j \leqslant \|b_1\|/\|b_j^*\|$ must hold, this leaves only $1 + 2\|b_1\|/\|b_j^*\|$ possibilities to pick λ_j. Thus by choosing the coefficients $\lambda_n, \ldots, \lambda_1$ in this order, one has at most $\prod_{j=1}^{n}(1 + 2\|b_1\|/\|b_j^*\|)$ candidates.

Suppose $\|b_j^*\| > \|b_1\|$ for some j. Then b_j can never have a nonzero coefficient λ_j in a shortest vector representation $v = \sum_{j=1}^{n} \lambda_j b_j$. Because in that case, v has a nonzero component in its projection to the orthogonal complement of $b_1\mathbb{R} + \ldots + b_{i-1}\mathbb{R}$ and since $b_2', \ldots, b_n'$ is K-Z reduced, this implies that $\|v\| \geqslant \|b_j^*\| > \|b_1\|$, which is impossible. Thus we can assume that $\|b_j^*\| \leqslant \|b_1\|$ holds for all $j = 1, \ldots, n$. Otherwise, b_j can be discarded. Therefore the number of candidates N for the tuples $(\lambda_1, \ldots, \lambda_n)$ satisfies

$$\begin{aligned} N &\leqslant \prod_{j=1}^{n} (1 + 2\|b_1\|/\|b_j^*\|) \\ &\leqslant \prod_{j=1}^{n} (3\|b_1\|/\|b_j^*\|) \\ &= 3^n \|b_1\|^n / \det(\Lambda). \end{aligned}$$

Next we give an upper bound for $\|b_1\|$. If b_1 is a shortest vector, then Minkowski's theorem, (Theorem 14.11) guarantees that $\|b_1\| \leqslant \sqrt{n}\det(\Lambda)^{1/n}$ holds. If b_1 is not a shortest vector, then the shortest vector v has a nonzero projection onto the orthogonal complement of $b_1\mathbb{R}$. Since $b_2', \ldots, b_n'$ is K-Z reduced, this implies that $\|v\| \geqslant \|b_2'\| \geqslant 1/2\|b_1\|$, since the basis is partially K-Z reduced. In any case we have $\|b_1\| \leqslant 2\sqrt{n}\det(\Lambda)^{1/n}$ and thus that $N \leqslant 6^n n^{n/2}$.

Now it is clear how to compute a K-Z reduced basis and thus a shortest vector. With an algorithm for K-Z reduction in dimension $n-1$, one uses Algorithm 4 to partially K-Z reduce the basis and then one checks all possible candidates for a shortest vector. Then one performs K-Z reduction on the basis for the projection onto the orthogonal complement of the shortest vector. Kannan [43] has shown that this procedure for K-Z reduction requires $n^{O(n)}\varphi$ operations, where φ is the binary encoding length of the initial basis and where the operands during the execution of the algorithm have at most $O(n^2\varphi)$ bits.

Theorem 14.19 ([43]). *Let B be a lattice basis of binary encoding length φ. There exists an algorithm which computes a K-Z reduced basis and requires $n^{O(n)} \cdot p(\varphi)$ arithmetic operations on rationals of size $O(n^2\varphi)$, where $p(\cdot)$ is a fixed polynomial.*

Notes

Kannan [44] also developed an algorithm for the *closest vector problem* whose running time is $2^{O(n\log n)}$ times a polynomial in the encoding length of the input. Here, one is given a rational vector $x \in \mathbb{Q}^n$ and a lattice $\Lambda \subseteq \mathbb{Q}^n$. The task is to compute a lattice point $v \in \Lambda$ which is closest to x, i.e., minimizing $\|v-x\|$. Kannan also showed that the integer programming feasibility problem can be solved within this complexity bound. Furthermore he showed that one can compute an approximate solution of the closest vector problem with a polynomial number of queries to an oracle which solves the shortest vector problem of an $n+1$-dimensional lattice. Blömer [9] showed that there exists an algorithm for the closest vector problem which runs in time $n!$ times a polynomial in the input encoding length. This means an exponential improvement over Kannan's algorithm [44] and its subsequent improvement by Helfrich [36]. Hanrot and Stehlé [33] improved this further to $n^{0.5\cdot n+o(n)}$ and showed that the shortest vector problem can be solved in time $n^{0.184\cdot n+o(n)}$ times a polynomial in the input length. Stehlé and Pujol [66] improve the arithmetic complexity of Kannan's algorithm.

Exercises

1. Prove that a 2-dimensional lattice basis is Gauß reduced if and only if it is LLL-reduced.
2. Prove that the shortest vector can be extracted in time $2^{O(n^2)}$ out of a LLL-reduced basis by adapting the arguments given in this section.

14.7 A randomized simply exponential algorithm for shortest vector

Ajtai, Kumar and Sivakumar [3] described a randomized method which outputs a shortest vector of a lattice with very high probability and has running time $2^{O(n)}$ times a polynomial in the binary input encoding length of the lattice basis. We follow the description of their algorithm in Oded Regev's excellent lecture notes [68]. At first sight, the algorithm is quite different from the shortest-vector algorithms that are based on reduction and candidate-trial, as Kannan's algorithm.

Our task is to compute the shortest vector of a lattice $\Lambda(B)$, for a nonsingular $B \in \mathbb{Q}^{n\times n}$. We can assume that $2 \leqslant SV(\Lambda) < 3$ holds, see exercise 14.7.1. The idea is to sample points in $\mathbb{R}^n$ and to translate these samples into lattice points. How is this translation done? A point $x \in \mathbb{R}^n$ can be represented as a linear combination of the basis vectors

$$x = \sum_{i=1}^{n} \lambda_i b_i, \tag{14.32}$$

for some $\lambda_i \in \mathbb{R}$. If we round all λ_i down, we obtain a lattice point

$$\sum_{i=1}^{n} \lfloor \lambda_i \rfloor \cdot b_i \in \Lambda(B). \tag{14.33}$$

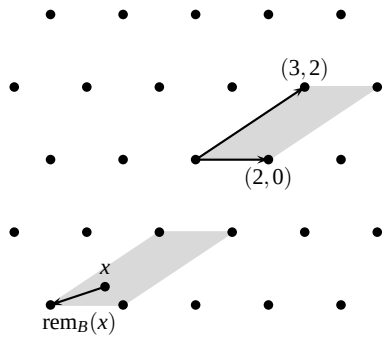


Fig. 14.5 The remainder of a point w.r.t. a basis B.

The *remainder* of x modulo the *basis* B is the vector

$$\mathrm{rem}_B(x) = \sum_{i=1}^{n} (\lambda_i - \lfloor \lambda_i \rfloor) b_i.$$

The lattice point in (14.33) is $x - \mathrm{rem}_B(x)$ and we associate it to x. Clearly one has

$$\mathrm{rem}_B(x) = \mathrm{rem}_B(x+v) \tag{14.34}$$

for each $v \in \Lambda(B)$.

Sieving

What we discussed above suggests the following approach. We sample a number of points $x_1, \ldots, x_k$ uniformly at random from $B_2(0)$, the ball of radius 2 around 0, and compute their remainders $y_i = \mathrm{rem}_B(x_i)$. The norms of these remainders are bounded by $R = \sum_{i=1}^{n} \|b_i\|$.

If we sample enough points (and the next lemma provides a bound on k), then there will be two such sampled points x_i, x_j such that their remainders y_i and y_j have distance at most $R/2$. The length of the lattice vector $x_i - y_i - (x_j - y_j)$ is then bounded by $R/2 + 4$.

Lemma 14.3 (Sieving Lemma). *Let $y_1, \ldots, y_k \in \mathbb{R}^n$ be points contained in the ball $B_R(0)$. There is a polynomial algorithm that computes a mapping $\tau : \{1, \ldots, k\} \to \{1, \ldots, k\}$ with the following properties:*

i) The cardinality of the image of τ is bounded by 5^n.
ii) For each $i = 1,\ldots,k$ one has $\|y_i - y_{\tau(i)}\| \leqslant R/2$.

In other words, we can identify $\leqslant 5^n$ *centers* among the $y_1,\ldots,y_k$ such that the balls of radius $R/2$ around these centers cover all of the remaining y_i. The mapping τ associates the y_i to a center of a ball that contains y_i.

Proof (of Lemma 14.3). We describe the simple procedure. In the beginning, all points are colored black. We iterate the following steps until there are no black points left: Choose any black point y_i and define $\tau(j) = i$, for each j such that y_j is black and $\|y_i - y_j\| \leqslant R/2$ holds. Color all points at distance at most $R/2$ from y_i red.

Call the points y_i with $\tau(i) = i$ *centers*. The number of centers is the size of the image of τ. Two centers have distance at least $R/2$ from each other. This means that the balls of radius $R/4$ around the centers do not intersect. On the other hand, these balls all fit into the ball of radius $R + R/4$. This implies that the number of centers is bounded by

$$\mathrm{vol}(B_{5\cdot R/4}(0))/\mathrm{vol}((B_{R/4}(0)) = 5^n,$$

and the assertion follows. □

The randomized algorithm for shortest vector

The sieving lemma provides a procedure to generate lattice vectors whose length is roughly $R/2$. This idea is now iterated in the random sampling algorithm for shortest vector.

Algorithm 5 Randomized algorithm for shortest vector

1. $R_0 \leftarrow \sum_{i=1}^n \|b_i\|$
   Choose $N = 2^{8n}\log R_0$ points $x_1,\ldots,x_N$ uniformly at random from $B_2(0)$
   Initialize a set of tuples $L = \{(x_i, y_i)\colon y_i = \mathrm{rem}_B(x_i),\ i = 1,\ldots,N\}$
   $R \leftarrow R_0$
2. **While** $R > 6$ **do**
   Apply the sieving algorithm to the vectors y_i for each $(x_i, y_i) \in L$
   Remove from L all tuples (x_i, y_i), where y_i is a center of the sieving procedure
   Replace each of the remaining (x_j, y_j) with $(x_j, y_j - (y_{\tau(j)} - x_{\tau(j)}))$
   $R \leftarrow R/2 + 2$.
3. For any two pairs (x_i, y_i), (x_j, y_j) in L compute the difference $x_i - y_i - (x_j - y_j)$ and output the shortest nonzero vector among these differences

Lemma 14.4. *At the beginning of each iteration of the* **while** *loop one has for each $(x_i, y_i) \in L$*

i) $x_i - y_i \in \Lambda(B)$
ii) $\|y_i\| \leqslant R$.

Proof. At the beginning of the first iteration, these invariants hold, as we discussed above.

We now show that these conditions hold after the instructions of the while-loop have been executed, if they were true at the beginning of that iteration. We only need to consider a tuple (x_j, y_j), where y_j is not a center in the sieving procedure. Let y_i be the center of y_j, i.e., $\tau(j) = i$. Since $x_j - y_j \in \Lambda(B)$ and $x_i - y_i \in \Lambda(B)$ we conclude that $x_j - y_j - (x_i - y_i) \in \Lambda(B)$, which implies condition i).

Since y_i is the center of y_j we have $\|y_j - y_i\| \leqslant R/2$. From this we conclude

$$\|y_j - (y_i - x_i)\| \leqslant \|y_j - y_i\| + \|x_i\| \leqslant R/2 + 2.$$

□

Lemma 14.4 and Exercise 14.7.4 imply that the algorithm runs in time $2^{O(n)}$ times a polynomial in the input encoding of the lattice basis B. Furthermore, at the end of the algorithm, there are at least

$$2^{8n} \log R_0 - 5^n \log R_0 \geqslant 2^{7n} \log R_0 \tag{14.35}$$

tuples (x_i, y_i) in L. The lattice vectors $x_i - y_i$ are short, i.e. $\|x_i - y_i\| \leqslant 2 + 6 = 8$. But how can we be sure that they are not all zero? Exercise 3 shows that, even when L is initialized, roughly half of these lattice vectors could be zero.

It turns out that the following observation that follows from (14.34) is crucial.

> Consider a tuple (x_i, y_i) in L before Step 3 of the algorithm. The algorithm would behave just the same until this point if x_i was replaced by $x_i + v_i$ after the initialization (Step 1), where $v_i \in \Lambda(B)$ is an arbitrary lattice vector.

When do we have to really *know* a point x_i and not just $y_i = \mathrm{rem}_B(x_i)$? The value of x_i is needed only when y_i was a center of the sieving procedure and tuples (x_j, y_j) are replaced by $(x_j, y_j - (y_i - x_i))$. Now we shed some light on why we sample the points $x_1, \ldots, x_N$ from $B_2(0)$ and not from a ball of other radius. This is connected to the fact that $2 \leqslant SV(\Lambda(B)) < 3$ and this crucial observation from above.

Let $v \in \Lambda(B)$ be a shortest vector. Consider the set $C_1 = B_2(0) \cap B_2(v)$ and the set $C_2 = B_2(0) \cap B_2(-v)$.

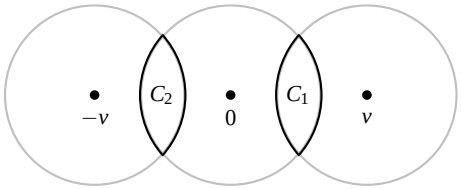


Fig. 14.6 The sets C_1 and C_2.

The mapping

$$f(x) = \begin{cases} x - v, & \text{if } x \in C_1 \\ x + v, & \text{if } x \in C_2 \\ x, & \text{if } x \in B_2(0) \setminus (C_1 \cup C_2) \end{cases} \tag{14.36}$$

is a bijection of $B_2(0)$ which maps C_1 to C_2 and vice-versa. A point $x \in B_2(0)$ which is chosen uniformly at random, remains to be chosen uniformly at random after we toss a fair coin and map x to $f(x)$ in the case that the coin shows heads.

This shows, together with the observation above, that we could, just before we enter Step 3 of the algorithm, toss a fair coin for each x_i with $(x_i, y_i) \in L$ and replace x_i by $f(x_i)$ if the coin shows head.

The next theorem implies that a shortest vector of $\Lambda(B)$ is returned with high probability.

Theorem 14.20. *At the end of Step 2 there exists with high probability a subset $L' \subseteq L$ of size 2^n which satisfies the following conditions*

a) $x_i - y_i = x_j - y_j$ *for each* $(x_i, y_i), (x_j, y_j) \in L'$.
b) $x_i \in C_1 \cup C_2$ *for each* $(x_i, y_i) \in L'$.

Before we prove the theorem we derive the main result of this section.

Theorem 14.21. *The random sampling algorithm outputs a shortest vector with high probability.*

Proof. Consider the set L' from Theorem 14.20. For each $(x_i, y_i) \in L'$, we toss a coin and replace x_i with $f(x_i)$ if the coin shows head. If we consider a difference

$$x_i - y_i - (x_j - y_j), \text{ where } (x_i, y_i) \neq (x_j, y_j) \in L'$$

then the probability that this difference becomes $\pm v$ is exactly $1/2$. (One coin shows head and coin shows tail). Since the cardinality of L' is 2^n with high probability, the algorithm will output a shortest vector of $\Lambda(B)$. □

We now prove Theorem 14.20. Recall that the set L contains at least $2^{7 \cdot n}$ tuples at the end of Step 2. The next lemma shows that the expected amount of these tuples, whose first component belongs to $C_1 \cup C_2$ is at least $2^{5 \cdot n + 1}$.

Lemma 14.5. *Consider the sets C_1 and C_2 described above. One has*

$$\text{vol}(C_1)/\text{vol}(B_2(0)) = \text{vol}(C_2)/\text{vol}(B_2(0)) \geqslant 2^{-2n}. \tag{14.37}$$

Lemma 14.5 shows that the expected number of tuples (x_i, y_i) with $x_i \in C_1 \cap C_2$ is $2^{6n+1} \log R_0$. The probability that this number of tuples is smaller than $2^{6n} \log R_0$ at the end of Step 1 is exponentially small.

Since we delete in each iteration 5^n tuples and perform at most $\log R_0$ many iterations, L contains at the end of Step 2, 2^{5n} tuples (x_i, y_i) with $x_i \in C_1 \cup C_2$ with high probability. Notice that at this point, a difference $x_i - y_i$ is a vector, whose length is bounded by 8.

Lemma 14.6.

$$|\Lambda(B) \cap B_8(0)| \leqslant 2^{4n}. \tag{14.38}$$

Lemma 14.6 and the previous discussion show that there exists a lattice vector $w \in \Lambda(B)$ such that $x_i - y_i = w$ and $x_i \in C_1 \cup C_2$ for at least 2^n of these tuples. This shows Theorem 14.20.

Notes

Ajtai et al. [4] also showed that there is a randomized algorithm to compute K-Z reduced bases in time $2^{O(n)}$. Together with the block-reduction algorithm of Schnorr [72], this implies the existence of a randomized polynomial time approximation algorithm for shortest vector, with an approximation factor of $2^{n \log\log n / \log n}$. Schnorr [72] previously showed that one approximate the shortest vector within a factor of $2^{n \log\log^2 n / \log n}$.

We did not mention how the sampling of points in $B_2(0)$ is done. A more general procedure for this task (sampling in convex bodies) was presented by Dyer, Kannan and Frieze [16]. Here the authors show how to sample points whose statistical distance to the uniform distribution is exponentially small. This serves the purpose of algorithm 5, see also exercise 7.

Blömer and Naewe [10] modified the sampling procedure described above to compute shortest vectors which are outside a given subspace. Under some length conditions on the target vector, they achieve a simply exponential algorithm for closest vector.

The shortest vector problem is also very interesting from the viewpoint of computational complexity. Van Emde Boas [22] proved that the shortest vector problem with respect to the ℓ_∞ norm is NP-hard, and he conjectured that it is NP-hard with respect to the Euclidean norm. In the same paper he proved that the closest vector problem is NP-hard for any ℓ_p norm. Ajtai [2] proved that the shortest vector problem with respect to the ℓ_2-norm is NP-hard for randomized problem reductions. This means that the reduction makes use of results of a probabilistic algorithm. Ajtai also showed that approximating the length of a shortest vector in a given lattice within a factor $1 + 1/2^{n^c}$ is NP-hard for some constant c. The non-approximability factor was improved to $(1 + 1/n^\varepsilon)$ by Cai and Nerurkar [12]. Micciancio [60] improved this factor substantially by showing that it is NP-hard to approximate the shortest vector in a given lattice within any constant factor less that $\sqrt{2}$ for randomized problem reductions, and that the same result holds for deterministic problem reductions (the "normal" type of reductions used in an NP-hardness proof) under the condition that a certain number theoretic conjecture holds. Goldreich and Goldwasser [30] proved that it is not NP-hard to approximate the shortest vector, or the closest vector, within a factor $\sqrt{n}$ unless the polynomial-time hierarchy collapses. Dinur [15] showed that it is NP-hard to approximate shortest vectors w.r.t. ℓ_∞ within almost polynomial factors. Khot [50] showed that the shortest vector problem w.r.t. the ℓ_p-norm is hard to approximate within any constant if $p > 1$. Using norm-embeddings

and a reduction to the ℓ_2 norm, Regev and Rosen [69] showed that this also holds for $p = 1$ and $p = \infty$. The currently strongest inapproximability result for shortest vector is due to Haviv and Regev [34].

Exercises

1. Justify the assumption $2 \leqslant SV(\Lambda) \leqslant 3$. *Hint: Suppose that $B \in \mathbb{Z}^{n\times n}$ and consider the lattices $\Lambda\left((3/2)^i/\|b_1\| \cdot B\right)$.*
2. Show that $\|\mathrm{rem}_B(x)\|$ is bounded by $\sum_{i=1}^n \|b_i\|$.
3. Suppose that x is chosen uniformly at random from $B_2(0)$. Show that the probability that
$$x - \mathrm{rem}_B(x) = 0 \tag{14.39}$$
holds, is at most $1/2$. For each fixed n, provide a family of lattice bases B_k^n with $2 \leqslant SV(\Lambda(B_k^n)) < 3$ such that the probability for the event (14.39) tends to $1/2$ for $k \to \infty$.
4. Prove that the number of iterations through the **while** loop of algorithm 5 is bounded by $\log R_0 - 1$.
5. Prove Lemma 14.5.
6. Prove Lemma 14.6. *Hint: Packing argument!*
7. How many bits does one have to know of each sampled point x_i in Algorithm 5? Show that the number of bits which are necessary to represent the numbers in the intermediate steps of Algorithm 5 is polynomial in the input encoding of the lattice basis.

8*. A prominent research question is the one whether there exists a deterministic algorithm for shortest vector with running time $2^{O(n)}$ times a polynomial in the input encoding.

9*. Is there a (randomized) $2^{O(n)}$-algorithm which computes a shortest nonnegative lattice vector, i.e. a $v \in \Lambda - \{0\}$ with $v \geqslant 0$ and $\|v\|$ minimal?

14.8 Integer programming in fixed dimension

In this section, we will describe Lenstra's algorithm [56] which solves the following integer feasibility problem.

> Given $A \in \mathbb{Z}^{m\times n}$ and $b \in \mathbb{Z}^m$, decide whether there exists an integer point that satisfies the inequality system $Ax \leqslant b$.

The algorithm runs in polynomial time if the dimension n is fixed. The integer feasibility problem is in NP [11]. In other words, if there exists an integer solution of $Ax \leqslant b$, then there exists one, whose size is polynomial in the size of the largest absolute value of a coefficient of the system $Ax \leqslant b$ and n, see, e.g. [76].

We start with the flatness theorem, which is the key concept of any polynomial time algorithm for integer programming in fixed dimension. The flatness theorem

shows that the feasibility problem can be decided with a variant of branch-and-bound and in particular that the feasibility problem can be solved in polynomial time, if the dimension n is fixed.

Let $K \subseteq \mathbb{R}^n$ be a nonempty closed subset of $\mathbb{R}^n$ and let $d \in \mathbb{R}^n$ be a vector. The *width* of K *along* d is the number

$$w_d(K) = \max\{d^T x \colon x \in K\} - \min\{d^T x \colon x \in K\}.$$

If the maximum or minimum does not exist, then $w_d(K)$ is defined to be infinity.

Suppose now that $K \subseteq \mathbb{R}^n$ is a convex body which does not contain an integer point. The *flatness theorem*, attributed to Khinchin 1948, ensures that there exists a nonzero integer vector $d \in \mathbb{Z}^n$ such that $w_d(K)$ is bounded by a constant. The *width* of P is defined as

$$w(P) = \min_{d \in \mathbb{Z}^n \setminus \{0\}} w_d(P).$$

A $d \in \mathbb{Z}^n \setminus \{0\}$ which minimizes $w_d(P)$ is called a *flat direction* of P.

Theorem 14.22 (Khinchine's flatness theorem [49]). *Let $K \subseteq \mathbb{R}^n$ be a convex body. Either K contains an integer point, or $w(K) \leqslant \omega(n)$, where $\omega(n)$ is a constant depending on the dimension only.*

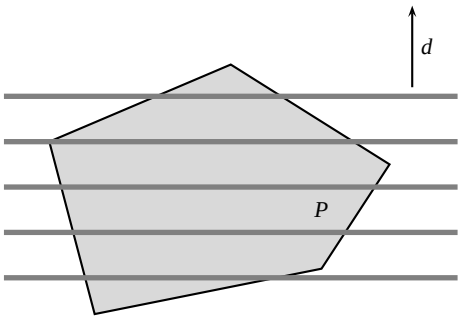


Fig. 14.7 Branching on a flat direction. The vector $d \neq 0$ is integral and the grey lines represent the lines $d^T x = \delta$ where $\delta \in \mathbb{Z}$ and $\max_{x \in P} d^T x \geqslant \delta \geqslant \min_{x \in P} d^T x$. The feasibility of P can be checked by checking the feasibility of the five 1-dimensional polytopes which are obtained from the intersection of P with the gray lines.

If $w_d(P) > \omega(n)$ for a flat direction d of P, then the flatness theorem guarantees the existence of an integer point in P. If, on the other hand, $w_d(P) \leqslant \omega(n)$, then an

integer point $x^* \in P \cap \mathbb{Z}^n$ must lie on one of the constant number of hyperplanes

$$c^T x = \delta, \text{ where } \delta \in \mathbb{Z}, \text{ and } \min\{c^T x \colon x \in P\} \leqslant \delta \leqslant \max\{c^T x \colon x \in P\}.$$

Later on in this section, we prove the flatness theorem. If the reader has understood how the flatness theorem is proved, he can also solve exercises 6 and 7. These exercises ask you to prove that the width and a flat direction of a rational polyhedron in fixed dimension can be computed in polynomial time.

Algorithm 6 Lenstra's algorithm

Input: $A \in \mathbb{Z}^{m \times n}$ and $b \in \mathbb{Z}^m$ with $P = \{x \in \mathbb{R}^n \colon Ax \leqslant b\}$ full-dimensional
Return: "Yes" if P is integer feasible and "No" otherwise.
  Compute $w(P)$ and a flat direction d of P
  if $w(P) > \omega(n)$ **then**
    return "Yes"
  end if
  for $\delta = \lceil \min_{x \in P} d^T x \rceil, \ldots, \lfloor \max_{x \in P} d^T x \rfloor$ **do**
    Compute $C \in \mathbb{Z}^{m \times (n-1)}$ and $f \in \mathbb{Z}^m$ such that $Cx \leqslant f$ is integer feasible if and only if $Ax \leqslant b$, $d^T x = \delta$ is integer feasible (See exercise 2)
    Recursively solve the integer feasibility problem for $Cx \leqslant f$
  end for

Exercises 2, 6 and 7 show that Lenstra's algorithm runs in polynomial time in the input encoding length.

Theorem 14.23. *Lenstra's algorithm runs in polynomial time in the input encoding length if the dimension is fixed.*

Proving the flatness theorem

Let us first discuss how to compute a flat direction of an ellipsoid. An ellipsoid is the image $f(B^n)$ of the n-dimensional unit ball $B^n = \{x \in \mathbb{R}^n \colon \|x\| \leqslant 1\}$ under an affine map $f(x) = A^{-1}x + a$, where $A \in \mathbb{R}^{n \times n}$ is a non-singular matrix, and $a \in \mathbb{R}^n$ is a vector. This ellipsoid can also be described as $E(A,a) = \{x \in \mathbb{R}^n \colon \|A(x-a)\| \leqslant 1\}$.

Consider now an ellipsoid $E(A,a)$ and a direction $d \in \mathbb{Z}^n \setminus \{0\}$. We first want to compute the width of $E(A,a)$ along d and the discuss how to find an integral vector $d \neq 0$ along which the width is minimal.

Since the width is invariant under translation, we can assume that $a = 0$ holds. The width of $E(A,0)$ along d is

$$\max_{x_1, x_2 \in E(A,0)} d^T(x_1 - x_2). \tag{14.40}$$

We have $d^T x_1 - d^T x_2 = d^T A^{-1}(Ax_1 - Ax_2)$ and Ax_1 and Ax_2 are contained in the unit ball if and only if $x_1, x_2 \in E(A,0)$. Consequently (14.40) is simply $2 \cdot \|d^T A^{-1}\|$.

Keep in mind that we want to compute a flat direction of $E(A,0)$ or, in other words, that we are interested in an integral d such that $2 \cdot \|d^T A^{-1}\|$ is as small as possible. This is a shortest vector problem in the lattice $\Lambda(A^{-1^T})$.

The *dual lattice* Λ^* of a lattice $\Lambda \subseteq \mathbb{R}^n$ is defined as

$$\Lambda^* = \{x \in \mathbb{R}^n : x^T y \in \mathbb{Z} \text{ for all } y \in \Lambda\}.$$

Exercise 5 shows that $\Lambda^*(A) = \Lambda(A^{-1^T})$ holds. Our discussion above implies the following theorem.

Theorem 14.24. *A nonzero vector $d \in \mathbb{Z}^n$ is a flat direction of the ellipsoid $E(A,a)$ if and only if $A^{-1^T} d$ is a shortest nonzero vector of $\Lambda^*(A)$. A flat direction of an ellipsoid can be computed in polynomial time in fixed dimension. If the dimension is varying, one can compute an integer vector $d \in \mathbb{Z}^n \setminus \{0\}$ in polynomial time such that $w_d(E(A,a)) \leqslant 2^{n-1} w(E(A,a))$ holds.*

The *covering radius* $\mu(\Lambda)$ of a lattice $\Lambda \in \mathbb{R}^n$ is the smallest number α such that the balls of radius α centered at the lattice points cover $\mathbb{R}^n$. Alternatively, the covering radius is the largest distance of a point $x \in \mathbb{R}^n$ to the lattice Λ.

The *packing radius* $\rho(\Lambda)$ of Λ is the largest number β such that the balls of radius β around the lattice points do not properly intersect. Alternatively, the packing radius is $SV(\Lambda)/2$.

We are now very close to proving the flatness theorem for ellipsoids. What if $E(A,a)$ does not contain an integer point? Inspecting the definition of $E(A,a) = \{x \in \mathbb{R}^n : \|A(x-a)\| \leqslant 1\}$ we see that this is the case if and only if the distance of Aa to the lattice $\Lambda(A)$ is larger than 1, i.e., $\mu(\Lambda(A)) > 1$. If we can now infer from this that the packing radius of $\Lambda^*(A)$ is bounded by a constant, depending only on the dimension, we are done, since the width of $E(A,a)$ is exactly $2 \cdot SV(\Lambda^*(A)) = 4 \cdot \rho(\Lambda^*(A))$. This is established in the next theorem via basis reduction.

Theorem 14.25. *Let $\Lambda \subseteq \mathbb{R}^n$ be a lattice and Λ^* be its dual. One has*

$$\mu(\Lambda) \cdot \rho(\Lambda^*) \leqslant f(n),$$

where $f(n)$ is a constant depending only on n which satisfies $f(n) \leqslant n/4 \cdot 2^{n(n-1)/4}$.

Proof. Suppose that $B = (b_1, \ldots, b_n) \in \mathbb{Q}^{n \times n}$ is a basis of the rational lattice $\Lambda(B)$ with orthogonality defect γ, i.e., one has

$$\|b_1\| \cdots \|b_n\| = \gamma \cdot |\det(B)|. \tag{14.41}$$

Assume that the longest basis vector is b_n, in other words that $\|b_n\| \geqslant \|b_j\|$ for $j = 1, \ldots, n-1$. This assumption can be made without loss of generality, since the orthogonality defect is invariant under swapping of columns.

Let $u \in \mathbb{R}^n$ be a point whose distance to Λ is $\mu(\Lambda)$. Since B is a basis of $\mathbb{R}^n$ we can write $u = \sum_{i=1}^n \lambda_i b_i$, with $\lambda_i \in \mathbb{R}$. The vector $v = \sum_{i=1}^n \lfloor \lambda_i \rceil b_i$ belongs to the lattice $\Lambda(B)$, where $\lfloor \lambda_i \rceil$ denotes the closest integer to λ_i. Clearly $\|v - u\| \geqslant \mu(\Lambda)$.

We also have

$$\begin{aligned} \|v-u\| &= \|\sum_{i=1}^{n}(\lfloor\lambda_i\rceil-\lambda_i)b_i\| \\ &\leqslant \sum_{i=1}^{n}\|(\lfloor\lambda_i\rceil-\lambda_i)b_i\| \\ &\leqslant \tfrac{1}{2}\sum_{i=1}^{n}\|b_i\| \\ &\leqslant \tfrac{n}{2}\|b_n\|, \end{aligned} \tag{14.42}$$

where the last inequality in (14.42) follows from the fact that the last basis vector b_n is the longest one in the basis. Since $\|v-u\| \geqslant \mu(\Lambda)$ we therefore have

$$\|b_n\| \geqslant 2\,\mu(\Lambda)/n. \tag{14.43}$$

Let $B = B^* \cdot R$ be the GSO of B. The following facts (see exercise 14.5.1)

$$\begin{aligned} &\|b_1\|\cdots\|b_n\| = \gamma\cdot\|b_1^*\|\cdots\|b_n^*\| \\ &\|b_j\| \geqslant \|b_j^*\|, \quad j = 1,\ldots,n. \end{aligned}$$

imply $\|b_n\| \leqslant \gamma\cdot\|b_n^*\|$, which together with (14.43) implies

$$\|b_n^*\| \geqslant 2\,\mu(\Lambda)/(n\cdot\gamma).$$

Now b_n^* is orthogonal to the vectors b_j^*, $j = 1,\ldots,n-1$. Since R is an upper triangular matrix with only 1's on its diagonal, it follows that $(b_n^*)^T B^*\cdot R = (0,\ldots,0,\|b_n^*\|^2)$ from which we can conclude that

$$d = b_n^*/\|b_n^*\|^2 \in \Lambda^*(B)^{\cdot}$$

The norm of d is the reciprocal of the norm of b_n^* and thus satisfies

$$\|d\| \leqslant n\cdot\gamma/(2\,\mu(\Lambda)).$$

This bounds the length of a shortest vector of $\Lambda^*(B)$ and consequently one has $\rho(\Lambda^*) \leqslant n\cdot\gamma/(4\,\mu(\Lambda))$ implying

$$\rho(\Lambda^*)\mu(\Lambda) \leqslant n\cdot\gamma/4.$$

By using the LLL-bound (14.27) on γ we obtain

$$\rho(\Lambda^*)\mu(\Lambda) \leqslant (n/4)2^{n(n-1)/4}.$$

□

Theorem 14.26 (Flatness theorem for ellipsoids). *Let $E(A,a) \subseteq \mathbb{R}^n$ be an ellipsoid which does not contain an integer point, then $w(E(A,a)) \leqslant 4\cdot f(n)$.*

Proof. If $E(A,a)$ does not contain an integer point, then the covering radius of $\Lambda(A)$ is at least one. Since $\mu(\Lambda(A))\cdot\rho(\Lambda^*(A)) \leqslant f(n)$ it follows thus that $\rho(\Lambda^*(A)) \leqslant f(n)$ and consequently that $w(E(A,a)) \leqslant 4\cdot f(n)$. □

The flatness theorem 14.22 for general convex bodies $K \subseteq \mathbb{R}^n$ follows from the fact that K can be well approximated by an ellipsoid. John [39] showed that there

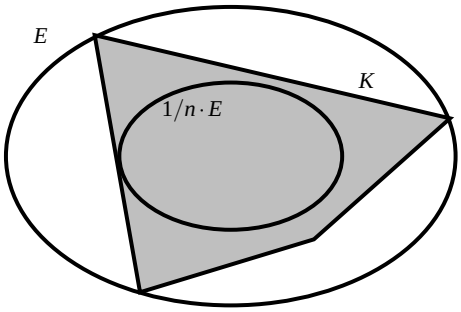


Fig. 14.8 A John-pair of ellipsoids.

exists an ellipsoid E containing K such that if E is centrally scaled by $1/n$, then it is contained in K. In other words one has

$$E/n \subseteq K \subseteq E.$$

This implies the flatness theorem with $\omega(n) \leqslant 4 \cdot n \cdot f(n)$.

Notes

The bound $f(n) \leqslant (n/4)2^{n(n-1)/4}$ of Theorem 14.25 is far from optimal. Lagarias, Lenstra and Schnorr [54] proved that $f(n) = O(n^{3/2})$. Banaszcyk [5] proved that $f(n) = O(n)$ holds.

Grötschel, Lovász and Schrijver [32] have shown that there exists a polynomial time algorithm which computes an ellipsoid E such that $1/(n(n+1)) \cdot E \subseteq K \subseteq E$ holds, if K is a convex body which is equipped with a *weak separation oracle*. Nesterov and Nemirovski [62] describe a polynomial algorithm which computes E such that $((1+\varepsilon)n)^{-1}E \subseteq P \subseteq E$ holds, where $P \subseteq \mathbb{R}^n$ is a full-dimensional polytope. Here $\varepsilon > 0$ is a parameter of the algorithm and the algorithm runs in polynomial time in the binary encoding length of P (rational data) and the binary encoding length of ε. A faster algorithm was provided by Khachiyan [48].

Finding a maximum volume ellipsoid inscribed in a polyhedron is a convex programming problem of *LP-type*. This means that such an ellipsoid can be computed in linear time in the RAM-model of computation if the dimension is fixed, see [59]. In the RAM-model, all numbers have input length one and basic arithmetic operations take constant time. The binary encoding length of the numbers in the input do not contribute to the input length. In fact those could even be real numbers. Gärtner [28] provided a subexponential algorithm for problems of LP-type in varying dimension, and thus also a subexponential algorithm to compute the maximum volume ellipsoid of a polytope. His algorithm is a generalization of the linear programming algorithm of Matoušek, Sharir and Welzl [59].

If the dimension is fixed, then using the algorithms for the largest enclosed ellipsoid by Matoušek et al. [59] Lenstra's algorithm requires $O(m \cdot s)$ arithmetic operations on rational numbers of size $O(s)$, where s is the largest binary encoding length of a coefficient of $Ax \leqslant b$. If the number of constraints m is fixed, then this matches the running time of the Euclidean algorithm, see, e.g. [51]. If one uses the floating point variant of the LLL-algorithm of Nguyen and Stehlé [64], then the bit-complexity for fixed dimension and number of constraints is $O(s^2)$ which also matches the bit-complexity of the Euclidean algorithm.

Exercises

1. Show how to compute a feasible point $x \in \mathbb{Z}^n$ of $Ax \leqslant b$ if it exists, by using a polynomial number of calls to an oracle solving an integer feasibility problem on an input which is of polynomial size in size(A), size(b) and n.
2. Consider a system of inequalities

$$Ax \leqslant b, d^T x = \beta, \tag{14.44}$$

   where $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, $d \in \mathbb{Z}^n$ and $\beta \in \mathbb{Z}$.
   Show how to reduce the feasibility problem of (14.44) to a feasibility problem in dimension $n-1$ involving m inequalities. This new system should be of polynomial size in the size of the system (14.44).
   Hint: Hermite normal form
3. Consider the polyhedron $P = \{x \in \mathbb{R}^n : Ax \leqslant b\}$, where $A \in \mathbb{Z}^{m \times n}$ and $b \in \mathbb{Z}^m$. Suppose that P has a flat direction whose first component is nonzero. Formulate a mixed integer program to compute the width and a flat direction of P. Conclude that the encoding length of a flat direction is polynomial in the encoding length of $Ax \leqslant b$.
4. Let $d \in \mathbb{R}^n \setminus \{0\}$ and let B^n be the n-dimensional unit ball. Show that $w_d(B^n) = 2 \cdot \|d\|$.
5. Prove that the dual lattice of $\Lambda(A)$ is $\Lambda((A^{-1})^T)$, i.e., $\Lambda^*(A) = \Lambda((A^T)^{-1})$.
6. Let $A \in \mathbb{Q}^{n \times n}$ be a nonsingular rational matrix and $a \in \mathbb{Q}^n$ be a rational vector. Show that there exists a polynomial algorithm which outputs either an integer vector in $E(A,a)$, or a nonzero integer vector $d \in \mathbb{Z}^n$ with $w_d(E(A,a)) \leqslant n \cdot 2^{n(n-1)/4}$.
7. Let $P = \{x \in \mathbb{R}^n : Ax \leqslant b\}$ be a rational polyhedron in fixed dimension n. Show that the width and a flat direction of P can be computed in polynomial time. *Hint: Exercise 5 of Section 14.5.*

14.9 The integer linear optimization problem

In this section we want to consider the integer optimization problem in fixed dimension

$$\max\{c^T x\colon Ax \leqslant b, x \in \mathbb{Z}^n\}. \tag{14.45}$$

In the analysis of the algorithm that follows, we use the parameters m and s, where m is the number of inequalities of the system $Ax \leqslant b$ and s is an upper bound on the binary encoding length of a coefficient of A, b and c.

The *greatest common divisor* of two integers a and b can be computed with the Euclidean algorithm with $O(s)$ arithmetic operations, where s is an upper bound on the binary encoding length of the integers a and b. On the other hand we have the following well known formula

$$\gcd(a,b) = \min\{a x_1 + b x_2\colon a x_1 + b x_2 \geqslant 1, x_1, x_2 \in \mathbb{Z}\}. \tag{14.46}$$

This implies that the greatest common divisor can be computed with an algorithm for the integer optimization problem in dimension 2 with one constraint.

The integer optimization problem can be reduced to the integer feasibility problem with binary search. One has a complexity of $O(m+s)$ for the integer feasibility problem in fixed dimension. This follows from an analysis of Lenstra's algorithm in combination with an efficient algorithm to compute a Löwner-John ellipsoid, see [59, 83]. With binary search for an optimal point, one obtains a running time of $O(m \cdot s + s^2)$. If, in addition to the dimension, also the number of constraints is fixed, this results in an $O(s^2)$ algorithm for the integer optimization problem, which is in contrast to the linear running time of the Euclidean algorithm.

Clarkson [13] has shown that the integer optimization problem with m constraints can be solved with an expected number of $O(m)$ arithmetic operations and $O(\log m)$ calls to an oracle solving the integer optimization problem on a constant size subset of the input constraints. Therefore we concentrate now on the integer optimization problem with a fixed number of constraints. In this section we outline an algorithm which solves the integer optimization problem in fixed dimension with a fixed number of constraints with $O(s)$ arithmetic operations on rational numbers of size $O(s)$. The algorithm relies on the LLL-algorithm.

The first step is to reduce the integer optimization problem over a full-dimensional polytope with a fixed number of facets to a disjunction of integer optimization problems over a constant number of *two-layer simplices*. A two-layer simplex is a full-dimensional simplex, whose vertices can be partitioned into two sets V and W, such that the objective function of the elements in each of the sets V and W agree, i.e., for all $v_1, v_2 \in V$ one has $c^T v_1 = c^T v_2$ and for all $w_1, w_2 \in W$ one has $c^T w_1 = c^T w_2$.

How can one reduce the integer optimization problem over a polytope P to a sequence of integer optimization problems over two-layer simplices? Simply consider the hyperplanes $c^T x = c^T v$ for each vertex v of P. If the number of constraints defining P is fixed, then these hyperplanes partition P into a constant number of polytopes, whose vertices can be grouped into two groups, according to their objec-

tive function value. Thus we can assume that the vertices of P can be partitioned into two sets V and W, such that the objective function values of the elements in each of the sets V and W agree. Carathéodory's theorem, see Schrijver [76, p. 94], implies that P is covered by the simplices that are spanned by the vertices of P. These simplices are two-layer simplices. Therefore, the integer optimization problem in fixed dimension with a fixed number of constraints can be reduced to a constant number of integer optimization problems over a two-layer simplex by applying a constant number of arithmetic operations.

The key idea is then to let the objective function slide into the two-layer simplex, until the width of the truncated simplex exceeds the flatness bound. In this way, one can be sure that the optimum of the integer optimization problem lies in the truncation, which is still flat. Thereby one has reduced the *integer optimization problem* in dimension n to a constant number of *integer optimization problems* in dimension $n-1$ and binary search can be avoided.

How do we determine a parameter π such that the truncated two-layer simplex $\Sigma \cap (c^T x \geqslant \pi)$ just exceeds the flatness bound? We explain the idea with the help of the 3-dimensional example in Figure 14.9.

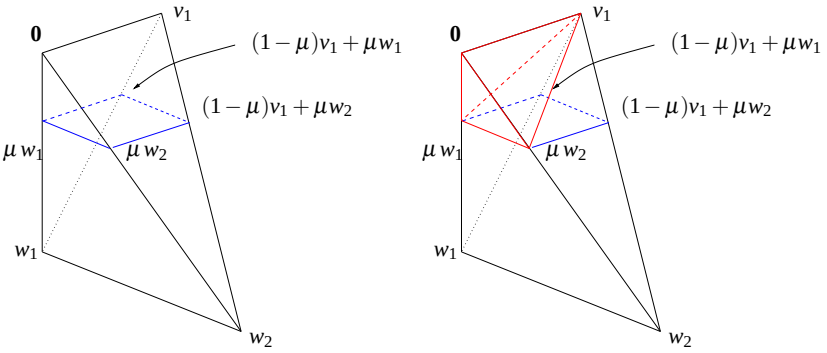


Fig. 14.9 Reduction to the parametric lattice width problem.

Here we have a two-layer simplex Σ in 3-space. The set V consists of the points $\mathbf{0}$ and v_1 and W consists of w_1 and w_2. The objective is to find a highest point in the vertical direction. The picture on the left describes a particular point in time, where the objective function slid into Σ. So we consider the truncation $\Sigma \cap (c^T x \geqslant \pi)$ for some $\pi \geqslant c^T w_1$. This truncation is the convex hull of the points

$$\mathbf{0}, v_1, \mu w_1, \mu w_2, (1-\mu)v_1+\mu w_1, (1-\mu)v_1+\mu w_2, \tag{14.47}$$

where $\mu = \pi / c^T w_1$. Now consider the simplex $\Sigma_{V,\mu W}$, which is spanned by the points $\mathbf{0}, v_1, \mu w_1, \mu w_2$. This simplex is depicted on the right in Figure 14.9. If this simplex is scaled by 2, then it contains the truncation $\Sigma \cap (c^T x \geqslant \pi)$. This is easy to see, since the scaled simplex contains the points $2(1-\mu)v_1$, $2\mu w_1$ and $2\mu w_2$. So we have the condition $\Sigma_{V,\mu W} \subseteq \Sigma \cap (c^T x \geqslant \pi) \subseteq 2\Sigma_{V,\mu W}$. From this we can infer the

important observation

$$w(\Sigma_{V,\mu W}) \leqslant w(\Sigma \cap (c^T x \geqslant \pi)) \leqslant 2\, w(\Sigma_{V,\mu W}). \tag{14.48}$$

This means that we essentially determine the correct π by determining a $\mu \geqslant 0$, such that the width of the simplex $\Sigma_{V,\mu W}$ just exceeds the flatness bound. The width of $\Sigma_{V,\mu W}$ is roughly (up to a constant factor) the length of the shortest vector of the lattice $L(A_\mu)$, where A_μ is the matrix

$$A_\mu = \begin{pmatrix} \mu w_1^T \\ \mu w_2^T \\ v_1 \end{pmatrix}.$$

Thus we have to find a parameter μ, such that the shortest vector of $L(A_\mu)$ is sandwiched between $\omega(n)+1$ and $\gamma \cdot (\omega(n)+1)$ for some constant γ. This problem can be understood as a *parametric shortest vector* problem.

To describe this problem let us introduce some notation. We define for an $n \times n$-matrix $A = (a_{ij})_{\forall i,j}$, the matrix $A^{\mu,k} = (a_{ij})^{\mu,k}_{\forall i,j}$, as

$$a_{ij}^{\mu,k} = \begin{cases} \mu \cdot a_{ij}, & \text{if } i \leqslant k, \\ a_{ij}, & \text{otherwise.} \end{cases} \tag{14.49}$$

In other words, the matrix $A^{\mu,k}$ results from A by scaling the first k rows with μ. The parametric shortest vector problem is now defined as follows.

> Given a nonsingular matrix $A \in \mathbb{Z}^{n\times n}$ and some $U \in \mathbb{N}$, find a parameter $p \in \mathbb{N}$ such that $U \leqslant \mathrm{SV}(L(A^{p,k})) \leqslant 2^{n+1/2} \cdot U$ or assert that $\mathrm{SV}(L) > U$.

It turns out that the parametric shortest vector problem can be solved in linear time when the dimension is fixed with a cascaded LLL-algorithm. From this, it follows that the integer optimization problem in fixed dimension with a fixed number of constraints can be solved in linear time. Together with Clarkson's result we obtain.

Theorem 14.27 ([18]). *The integer optimization problem* (14.45) *can be solved with an expected number of $O(m + s\log m)$ arithmetic operations on rationals of size $O(s)$.*

Notes

A polynomial time algorithm for the *two-variable integer programming problem* was presented by Hirschberg and Wong [38] and Kannan [46] for special cases and by Scarf [70, 71] for the general case. Then, Lenstra [56] proved that integer programming in arbitrary fixed dimension can be solved in polynomial time. Afterwards, various authors were looking for faster algorithms for the two-dimensional case [24, 85, 41]. A linear-time algorithm in the arithmetic model was presented by Eisenbrand and Laue [19].

Exercises[1]

1*) Can one solve an integer program in fixed dimension with a fixed number of constraints in quadratic time in the bit-model of complexity ?
Recent results on the bit-complexity of the LLL-algorithm [63] could help answering this question
2*) Is the shortest vector problem solvable in time $O(M(s)\log s)$ if the dimension is fixed?
See also the Notes section of Chapter 14.5.

14.10 Diophantine approximation and strongly polynomial algorithms

In this section we review a very important appliction of the LLL-algorithm in optimization, namely the rounding algorithm of Frank and Tardos [25]. The input of a combinatorial optimization problem usually consists of a combinatorial structure, like a graph or a subset system and some numbers, which reflect weights on the edges or costs of subsets or alike. An algorithm is *strongly polynomial* if

A) it consist only of basic arithmetic operations $+,-,*,/$ and comparisons $<,>,=$;

B) the number of such basic operations which are carried out is polynomial in the input length, where the numbers in the input have length one;
C) the encoding lengths of the numbers in the intermediate steps of the algorithm are polynomial in the binary encoding length of the input, where numbers in the input account with their binary encoding length.

An algorithm is weakly polynomial if it satisfies A) and C) but instead of B) satisfies the weaker condition that the number of basic operations is bounded in the binary encoding length of the input.

Algorithms which are based on *bit-scaling* of the involved numbers usually have the property to be weakly polynomial only. It is an outstanding open problem, whether linear programming, which can be solved in weakly polynomial time, can also be solved in strongly polynomial time.

A *0/1-optimization problem* is an integer program, where the integer variables are restricted to be either 0 or 1. The result that we survey now shows a facet of the tremendous importance of the LLL-algorithm in the area of algorithms and complexity. In fact it holds in greater generality, see [25, 31]. We treat a less general version here, to keep the exposition as simple as possible.

Theorem 14.28 ([25]). *If a 0/1 optimization problem can be solved in weakly polynomial time, then it can be solved in strongly polynomial time.*

[1] Those are not really ”exercises” but rather research questions.

The starting point of the method leading to this result is Dirichlet's theorem. Suppose that you have an n-dimensional vector $\alpha \in \mathbb{R}^n$ which is represented by n real numbers $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ and you are interested in a vector $p \in \mathbb{Z}^n$ with integer components of small absolute value and which encloses a small angle with α. Such a p could be a candidate to replace the linear objective function $\alpha^T x$ with the linear objective function $p^T x$, hoping that the optimal 0/1-solutions are the same.

Theorem 14.29 (Dirichlet). *Let $Q \geqslant 1$ be a real number and let $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$. There exists an integer q and integers $p_1, \ldots, p_n$ with*

i) $1 \leqslant q \leqslant Q^n$ *and*
ii) $|q \cdot \alpha_i - p_i| \leqslant 1/Q$ *for* $i = 1, \ldots, n$.

Proof. Consider the lattice Λ which is generated by the following matrix

$$\begin{pmatrix} 1 & & & \alpha_1 \\ & 1 & & \alpha_2 \\ & & \ddots & \vdots \\ & & 1 & \alpha_n \\ & & & \frac{1}{Q^{n+1}} \end{pmatrix}. \tag{14.50}$$

The determinant of Λ is $1/Q^{n+1}$. The cube $K = \{x \in \mathbb{R}^{n+1} : -1/Q \leqslant x \leqslant 1/Q$ is symmetric around 0 and has volume $2^{n+1} \cdot 1/Q^{n+1}$. Minkowski's theorem implies that this set contains a nonzero lattice point [2]

$$\begin{pmatrix} -p_1 + q \cdot \alpha_1 \\ \vdots \\ -p_n + q \cdot \alpha_n \\ q/Q^{n+1} \end{pmatrix}.$$

We can assume q to be positive, since the negative of this vector is also contained in K. The integers p_i, $i = 1, \ldots, n$ and q are as it is stated in the theorem. □

If Q in the Dirichlet theorem is a positive integer, then one can replace condition ii) by the slightly stronger condition

ii') $|q \cdot \alpha_i - p_i| < 1/Q$ for $i = 1, \ldots, n$,

see exercise 1).

Suppose now that we want to solve a 0/1 optimization problem

$$\max_{x \in \mathcal{F}} \alpha^T x \tag{14.51}$$

[2] Because, if this cube would not contain a nonzero lattice point, then one could scale it with a factor larger than one and still it would not contain a nonzero lattice point. The volume of this scaled cube is strictly larger than $2^{n+1} \cdot 1/Q^{n+1}$. This would contradict Minkowski's theorem.

where $\mathscr{F} \subseteq \{0,1\}^n$. The set $\mathscr{F}$ could, for example, be the characteristic vectors of matchings of a graph $G = (V,E)$ or characteristic vectors of other combinatorial structures.

Our goal is to replace α by an integer vector v whose binary encoding length is polynomial in the dimension n such that the optimal points in $\mathscr{F}$ w.r.t. the objective functions $\alpha^T x$ and $v^T x$ are the same.

To this end, consider an optimal solution $\overline{x}$ of problem (14.51), i.e., an $\overline{x}$ with

$$\alpha^T(\overline{x} - x) \geqslant 0 \text{ for all } x \in \mathscr{F}.$$

The points $\overline{x} - x$ above have components in $\{0, \pm 1\}$. Our integer vector $v \in \mathbb{Z}^n$ will have the following property.

For each $y \in \{0, \pm 1\}^n$ one has

$$\alpha^T y \geqslant 0 \text{ if and only if } v^T y \geqslant 0.$$

If this holds, then we can safely replace α by v in (14.51).

Let us assume for a moment that Dirichlet's theorem has an efficient implementation, i.e., that the q and p_i satisfying conditions i) and ii) can be computed in polynomial time. Assume further that the largest absolute value of a component in α is one, or in other words that $\|\alpha\|_\infty = 1$. If this would not hold, then we could simply scale α by $1/\|\alpha\|_\infty$. Set Q in Dirichlet's theorem to $Q := n$ and let q and p be the resulting integer and integer vector respectively.

How large are the absolute values of the components of p? Since $1 \leqslant q \leqslant n^n$ (i) and $|q\,\alpha_i - p_i| < 1/n$ (ii') it follows that $\|p\|_\infty \leqslant q\|\alpha\|_\infty \leqslant n^n$. The binary encoding length of p is therefore polynomial in n, which is one of the objectives that we want an approximation of α to satisfy.

How close comes p in terms of being a valid replacement v of α? The next lemma shows, that this vector comes very close already to what we want. Let $\operatorname{sign}(x)$ denote the sign of a real number x, i.e.,

$$\operatorname{sign}(x) = \begin{cases} 1, & \text{if } x > 0, \\ -1, & \text{if } x < 0, \\ 0, & \text{if } x = 0. \end{cases}$$

Lemma 14.7. *Let $y \in \{0, \pm 1\}^n$ be a vector. If $\operatorname{sign}(p^T y) \neq 0$, then $\operatorname{sign}(p^T y) = \operatorname{sign}(\alpha^T y)$.*

Proof. Suppose that $p^T y > 0$. Since p and y are integer vectors, it follows that $p^T y \geqslant 1$ holds. Property ii) of the Dirichlet theorem implies $|q\,\alpha_i - p_i| < 1/n$ for each i. Thus $|q\,\alpha^T y - p^T y| = |(q\,\alpha - p)^T y| < n/n = 1$. In other words, the distance of $p^T y$ to $q\,\alpha^T y$ is less than one, implying that $q\,\alpha^T y > 0$ and consequently $\alpha^T y > 0$.

The case $p^T y < 0$ is analogous. □

If $\operatorname{sign}(p^T y) = 0$, then we cannot infer anything about $\operatorname{sign}(\alpha^T y)$. The idea is now to apply recursion. If $\operatorname{sign}(p^T y) = 0$, then clearly

$$\operatorname{sign}(\alpha^T y) = \operatorname{sign}\left((q\cdot\alpha - p)^T y\right)$$

Since one component of α is 1, say component i, we clearly have $p_i = q$ and thus the vector $q\cdot\alpha - p$ has at least one more zero component than α. Let v' be an integer vector which has been recursively computed and satisfies for each $y \in \{0,\pm 1\}^n$ $\operatorname{sign}(v'^T y) = \operatorname{sign}\left((q\cdot\alpha - p)^T y\right)$.

Let M be a large weight. The above discussion shows that

$$\forall y \in \{0,\pm 1\}^n\colon\ \operatorname{sign}(\alpha^T y) = \operatorname{sign}\left((M\cdot p + v')^T y\right).$$

Thus we set $v := M\cdot p + v'$.

Let us deduce a bound on M. This number M has to be large enough so that $M\cdot p^T y$ dominates the sign of $(M\cdot p + v')^T y$ if $p^T y \neq 0$. This clearly holds if M is at least as large as the absolute value of $v'^T y$. To this end, let $t(n)$ be an upper bound on the largest absolute value of a component of v which is constructed by the above described recursive procedure. The absolute value of $v'^T y$ is then bounded by $n\cdot t(n-1)$. Thus we set M to this value. How large is $\|v\|_\infty$? For this we have the recursive formula

$$t(n) \leqslant n\cdot t(n-1)\cdot n^n + t(n-1),$$

which shows a crude bound of $t(n) \leqslant (n+1)^{n^2+n}$. The binary encoding length of v is thus $O(n^2\log n)$.

What about our assumption from the beginning that Dirichlet's theorem has an efficient implementation? In fact, it is an outstanding open problem, whether the q and the p_i as in Dirichlet's theorem can be found in polynomial time. However, we have the LLL-algorithm. The proofs of the next two lemmas are left as an exercise.

Lemma 14.8. *Let $Q \geqslant 1$ be an integer and $\alpha_1,\ldots,\alpha_n \in \mathbb{Q}$ be rational numbers. There exists a polynomial time algorithm that computes an integer q and integers $p_1,\ldots,p_n$ with*

I) $1 \leqslant q \leqslant \sqrt{n+1}\,2^{(n-1)/2}\cdot Q^n$ *and*
II) $|q\cdot\alpha_i - p_i| < \sqrt{n+1}\,2^{(n-1)/2}\cdot 1/Q$ *for* $i = 1,\ldots,n$.

Lemma 14.9. *Let $Q \geqslant 1$ be an integer and $\alpha_1,\ldots,\alpha_n \in \mathbb{Q}$ be rationals numbers. There exists a polynomial algorithm that computes an integer q and integers $p_1,\ldots,p_n$ with*

a) $1 \leqslant q \leqslant 2^{n^2}Q^n$ *and*
b) $|q\cdot\alpha_i - p_i| < 1/Q$ *for* $i = 1,\ldots,n$.

If we replace the guarantees of Dirichlet's theorem with the ones obtained in Lemma 14.9, then the binary encoding length of v is still polynomial in n. This means that we have an algorithm to round an objective function vector $\alpha \in \mathbb{Q}^n$ to an equivalent integer vector $v \in \mathbb{Z}^n$. The only trouble is that the LLL-algorithm is not *strongly polynomial* and our goal is to develop a strongly polynomial time algorithm. This is finally achieved by rounding the α_i first individually before the algorithm behind Lemma 14.9 is applied. The rounding operation $\lfloor\cdot\rfloor$ is not part of

the standard set of operations on which a strongly polynomial time algorithm can rely. Let $\beta \in \mathbb{R}$ be a real number with $0 < \beta \leqslant 1$ and let $K \geqslant 2$ be a positive integer. In exercise 4, you are asked to compute $\lfloor K \cdot \beta \rfloor$ with a linear (in the binary encoding length of K) number of elementary operations $(+, -, *, /, <, >, =)$.

Theorem 14.30. *There exists a strongly polynomial algorithm that, given n* real numbers $\alpha_1, \ldots, \alpha_n$ *with* $\|\alpha\|_\infty = 1$, *computes integers $p_1, \ldots, p_n$ and q with*

$$|q \cdot \alpha_i - p_i| < 1/n,\ i = 1, \ldots, n, \text{ and } 1 \leqslant q \leqslant 2^{n^2+n} n^n.$$

Proof. For each i compute:

$$\alpha_i' = \frac{\lfloor \alpha_i 2^{n^2+n+1} n^{n+1} \rfloor}{2^{n^2+n+1} n^{n+1}}$$

in strongly polynomial time (see exercise 4) and run the algorithm behind Theorem 14.9 on input α' and $Q' = 2 \cdot n$. This algorithm returns q and p.

We now have to show the following claim.

$$|q \cdot \alpha_i - p_i| \leqslant 1/n,\ i = 1, \ldots, n, \text{ and } q \leqslant 2^{n^2+n} n^n. \tag{14.52}$$

Clearly one has $1 \leqslant q \leqslant 2^{n^2}(2n)^n = 2^{n^2+n} n^n$. On the other hand we have

$$\begin{aligned} |q\alpha_i - p_i| &\leqslant |q\alpha_i - q\alpha_i'| + |q\alpha_i' - p_i| \\ &< \frac{2^{n^2+n} n^n}{2^{n^2+n+1} n^{n+1}} + 1/(2n) \\ &= 1/n. \end{aligned}$$

which shows the claim. □

If we adapt the argument from before with the new parameters of Theorem 14.30 we obtain the result of Frank and Tardos [25].

Theorem 14.31. *There exists a strongly polynomial algorithm that, given $\alpha \in \mathbb{R}^n$, computes a $v \in \mathbb{Z}^n$ whose binary encoding length is polynomial in n such that* $\operatorname{sign}(v^T y) = \operatorname{sign}(\alpha^T y)$ *for all $y \in \{0, \pm 1\}^n$.*

Consequently, a $0/1$-optimization problem can be solved in polynomial time if and only if it can be solved in strongly polynomial time.

Notes

The result of Tardos and Frank [25] is more general than for the case restricted to $0/1$-problems described above. If one wants to optimize a linear function over the convex hull of a set of rational points, then the linear function can be replaced by an integer linear objective function, whose binary encoding length is polynomial in the

largest binary encoding length of a point in the set, see also [58, 31]. Tardos [80] has shown that a linear program $\max\{c^T x\colon Ax \leqslant b\}$ can be solved within a number of elementary operations bounded by a polynomial in the binary encoding length of A. This can also be shown with a rounding argument, see [31]. Diophantine approximation is also discussed in a recent survey of Lenstra [57].

Exercises

1. Show that condition ii) can be strengthened to $|q\cdot\alpha_i - p_i| < 1/Q$ for $i = 1,\ldots,n$.

2. Prove Lemma 14.8.
   Hint: The shortest vector of the lattice generated by the matrix (14.50) *has ℓ_2-norm at most $\sqrt{n+1}/Q$ and the LLL-algorithm finds a vector x which is a $2^{(n-1)/2}$-approximation of this shortest vector. Bound the ℓ_∞-norm of x.*
3. Prove Lemma 14.9.
   Hint: Set $Q' := \lceil\sqrt{n+1}\cdot 2^{(n-1)/2}\cdot Q\rceil$ and apply Lemma 14.8 using α and Q'. The bound a) holds for all but a finite number of n. For this fixed number of n one can afford to find a shortest vector w.r.t. the ℓ_∞-norm (see exercise 14.5.5).
4. Let $\beta \in \mathbb{R}$ be a real number with $0 < \beta \leqslant 1$ and let $K \geqslant 2$ be a positive integer. Show how to compute $\lfloor K\cdot\beta\rfloor$ with a linear (in the binary encoding length of K) number of elementary RAM operations $(+,-,*,/,<,>,=)$.
5. Show that for a $0/1$-optimization problem (14.51) there exists a $v \in \mathbb{Z}^n$ with $\|v\|_\infty = 2^{O(n\log n)}$ such that an $x \in \mathscr{F}$ is optimal w.r.t. $\alpha^T x$ if and only if x is optimal w.r.t. $v^T x$.
   Hint: Use linear programming duality and the Hadamard bound.

14.11 Parametric integer programming

The *Frobenius problem* is as follows. Given n integers $a_1,\ldots,a_n$ whose greatest common divisor is one, compute the largest $t \in \mathbb{N}$ which cannot be written as

$$x_1\cdot a_1 + \cdots + x_n\cdot a_n = t, \quad x_1,\ldots,x_n \in \mathbb{N}_0.$$

The Frobenius problem is NP-complete [67]. Kannan [45] showed that it can be solved in polynomial time, if n is fixed. In fact, Kannan showed something more general. The Frobenius problem can be solved with binary search, if the validity of the following *forall/exist*-statement can be decided in polynomial time.

$$\forall y \in \mathbb{Z}, y \geqslant N \quad \exists x_1,\ldots,x_n \in \mathbb{N}_0 \quad : \quad y = x_1\cdot a_1 + \cdots + x_n a_n \tag{14.53}$$

Kannan's result is a polynomial time procedure to decide forall/exist statements of the following form, where $Q \subseteq \mathbb{R}^m$ is a polyhedron, $A \in \mathbb{Z}^{m\times n}$ is a matrix and $t \in \mathbb{N}$ is an integer.

$$\forall b \in \left(Q \cap (\mathbb{R}^{m-t} \times \mathbb{Z}^t)\right) \quad Ax \leqslant b \text{ is IP-feasible.} \tag{14.54}$$

More precisely, he showed that such statements can be decided in polynomial time, if n, t and the affine dimension of Q are fixed.

Theorem 14.32 ([45]). *If n,t and the affine dimension of Q are fixed, then $\forall\exists$-statements can be decided in polynomial time.*

In this section we review the basic ideas of his procedure and the generalization of Eisenbrand and Shmonin [21], which only assumes n and t to be fixed whereas the dimension of Q can vary.

Theorem 14.33 ([21]). *If n,t are fixed, then $\forall\exists$-statements can be decided in polynomial time.*

Suppose that the width of a polyhedron $P \subseteq \mathbb{R}^n$ is the width along the integer direction c, i.e., $w(P) = w_c(P)$ with $c \in \mathbb{Z}^n$. Let $\beta = \min\{c^T x\colon x \in P\}$. If $w(P) > \omega(n)$, then we can scale and translate P, to obtain a polyhedron P' which is sandwiched between the hyperplanes $c^T x = \beta$ and $c^T x = \beta + \omega(n)$, see figure 14.10.

Theorem 14.34. *If $w(P) > \omega(n)$, then there exists an integer point in*

$$P \cap \left(\beta \leqslant c^T x \leqslant \beta + \omega(n)\right).$$

The width of P' is exactly $\omega(n)$ and by the flatness theorem (Theorem 14.22), P' contains an integer point. From this it follows that there exists an integer point on one of the constant number of lower dimensional polyhedra

$$P \cap c^T x = \lceil \beta \rceil + i, \text{ for } i = 0, \ldots, \omega(n). \tag{14.55}$$

Theorem 14.35 (Strengthening of flatness theorem). *Let $P \subseteq \mathbb{R}^n$ be a polyhedron with $w(P) = w_c(P)$ for an integer vector $c \in \mathbb{Z}^n \setminus \{0\}$. The polyhedron P contains an integer point if and only if at least one of the polyhedra*

$$P \cap \left(c^T x = \lceil \beta \rceil + i\right) \quad i = 0, \ldots, \omega(n)$$

contains an integer point.

We continue with a description of Kannan's ideas in dimension 2. Suppose that $A \in \mathbb{Z}^{m\times 2}$ and $Q \subseteq \mathbb{R}^m$ and that we wish to decide the $\forall\exists$-statement (14.54) with $t = 0$. We then would have to decide, whether there exists a $b \in Q$ such that $Ax \leqslant b$ is *integer infeasible*. We denote the polyhedron $\{x \in \mathbb{R}^2\colon Ax \leqslant b\}$ by P_b. Let us make the following simplifying assumptions.

I) The flat direction of P_b is always the first unit vector e_1.

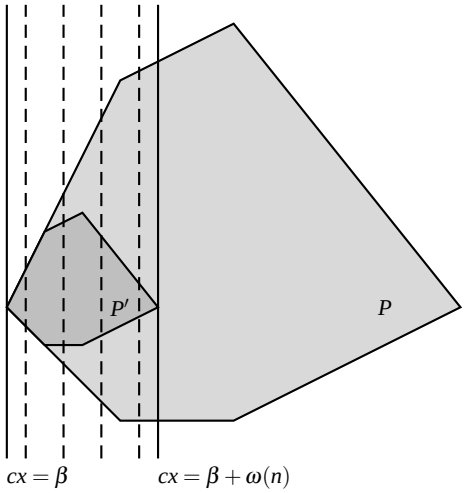


Fig. 14.10 An illustration of the strengthening of the flatness theorem. There must be an integer point on one of the dashed lines $c^T x = \lceil \beta \rceil + i$, $i = 1, \ldots, \omega(n)$.

II) The optimum basis of the linear program $\min\{e_1^T x \colon x \in \mathbb{R}^2, Ax \leqslant b\}$ is the same for all $b \in Q$.

It follows from II) that the optimum solution of the linear program $\min\{e_1^T x \colon x \in P_b\}$ is of the form $G \cdot b$ for a fixed matrix $G \in \mathbb{R}^{2\times 2}$.

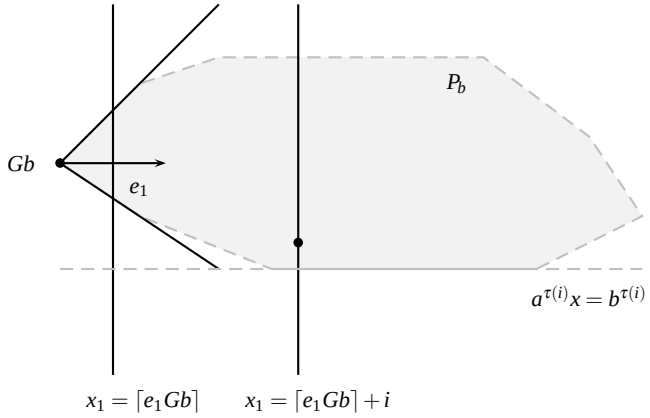


Fig. 14.11 Illustration in dimension 2.

The strengthening of the flatness theorem tells that P_b contains an integral point if and only if there exists an integral point on the lines $x_1 = \lceil e_1 Gb \rceil + j$ for $j = 0, \ldots, \omega(2)$. The intersections of these lines with the polyhedron P_b are 1-

dimensional polyhedra. Some of the constraints $ax \leqslant b$ of $Ax \leqslant b$ are "pointing upwards", i.e., $ae_2 < 0$, where e_2 is the second unit-vector. Let $a_1x_1 + a_2x_2 \leqslant b$ be a constraint pointing upwards such that the intersection point $(\lceil e_1Gb \rceil + j, y)$ of $a_1x_1 + a_2x_2 = \beta$ with the line $x_1 = \lceil e_1Gb \rceil + j$ has the largest second component. The line $x_1 = \lceil e_1Gb \rceil + j$ contains an integral point in P_b if and only if $(\lceil e_1Gb \rceil + j, \lceil y \rceil)$ is contained in P_b. This point is illustrated in Figure 14.11. We assume that this "highest upwards constraint" for the line $x_1 = z + i$ is always fixed, lets say it is the constraint $a^{\tau(i)}x \leqslant b^{\tau(i)}$ for a fixed mapping $\tau : \{0, \ldots, \omega(2)\} \to \{1, \ldots, m\}$. We make this assumption explicit.

III) The highest intersection point of a line $a^j x = b^j$ with the line $x_1 = z + i$ is the intersection point of the line $a^{\tau(i)}x = b^{\tau(i)}$, where $\tau : \{0, \ldots, \omega(2)\} \to \{1, \ldots, m\}$ is a fixed mapping.

We now formulate a mixed-integer program from which we can extract the integer solution of P_b on the line $x_1 = z + i$, assuming that I-III) holds.

$$\begin{array}{c} (Gb)_1 \leqslant z < (Gb)_1 + 1 \\ x_1^i = z + i \\ y = (b^{\tau(i)} - a_1^{\tau(i)} x_1)/a_2^{\tau(i)} \\ y \leqslant x_2^i < y + 1 \\ x_1^i, x_2^i, z \in \mathbb{Z}. \end{array} \tag{14.56}$$

For a given $b \in Q$, the line $x_1 = z + i$ contains an integer point in P_b if and only if $x^i = (x_1^i, x_2^i) \in P_b$.

Recall that we want to decide the forall/exist-statement (14.54)) by searching a $b \in Q$ such that P_b does not contain an integer point. In other words, we are looking for a $b \in Q$ for which each of the x^i for $i = 0, \ldots, \omega(2)$ violates some constraint of $Ax \leqslant b$. To do that, we again fix a mapping $\mu : \{0, \ldots, \omega(2)\} \to \{1, \ldots, m\}$, where $a^{\mu(i)}x \leqslant b^{\mu(i)}$ is supposed to be the constraint of $Ax \leqslant b$ which is violated by x^i. The number of such mappings μ is $m^{\omega(2)+1}$, which is polynomial in m, since $\omega(2)$ is a constant.

We enumerate all these mappings μ and append to each of the $\omega(2)+1$ MIP's (14.56)) the constraint

$$a^{\mu(i)}x^i > b^{\mu(i)}. \tag{14.57}$$

The conjunction of all these MIP's is a MIP with a constant number of integer variables, which can be solved with Lenstra's algorithm [56] in polynomial time. This shows that we can decide forall/exist statements, given that the conditions I-III) hold.

We justify condition I, at the heart of which lies the following theorem.

Theorem 14.36. *Let n be fixed and $A \in \mathbb{Z}^{m \times n}$ be a matrix of full column rank. There exist a set $D \subseteq \mathbb{Z}^n \setminus \{0\}$ of polynomial (in the encoding length of A) cardinality such that for each $b \in \mathbb{R}^m$ there exists a $d \in D$ with $w(P_b) = w_d(P_b)$. Such a set D can be computed in polynomial time.*

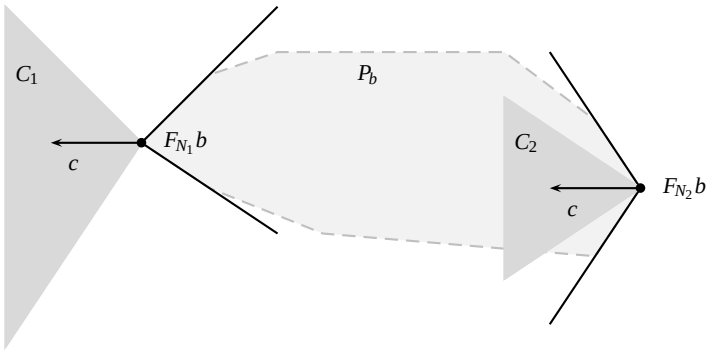


Fig. 14.12 The flat direction c and the two cones C_1 and C_2.

Proof. Suppose that P_b is non-empty and let $c \in \mathbb{Z}^n$ be its width direction. Then there are two bases N_1 and N_2 such that

$$\max\{cx : Ax \leqslant b\} = cF_{N_1}b \quad \text{and} \quad \min\{cx : Ax \leqslant b\} = cF_{N_2}b \tag{14.58}$$

and c belongs to the cones C_1 and C_2 defined by the optimal bases w.r.t. $\max\{c^T x\colon x \in P_b\}$ and $\min\{c^T x\colon x \in P_b\}$, see Figure 14.12. Here F_{N_1} and F_{N_2} are the inverses of the basis matrices of N_1 and N_2 respectively.

In fact, equations (14.58) hold for any vector c in $C_1 \cap C_2$. Thus, the lattice width of P_b is equal to the optimum value of the following optimization problem:

$$\min\{c(F_{N_1} - F_{N_2})b : c \in C_1 \cap C_2 \cap \mathbb{Z}^n \setminus \{0\}\}. \tag{14.59}$$

The latter can be viewed as an integer programming problem. Indeed, the cones C_1 and C_2 can be represented by some systems of linear inequalities, say $cD_1 \leqslant 0$ and $cD_2 \leqslant 0$, respectively, where $D_1, D_2 \in \mathbb{Z}^{n\times n}$. The minimum (14.59) is taken over all integral vectors c satisfying $cD_1 \leqslant 0$ and $cD_2 \leqslant 0$, except the origin. Since both cones C_1 and C_2 are simplicial, i.e., generated by n linearly independent vectors, the origin is a vertex of $C_1 \cap C_2$ and therefore can be cut off by a single inequality, for example, $cD_1\mathbf{1} \leqslant -1$, where $\mathbf{1}$ denotes the n-dimensional all-one vector. It is important that all other integral vectors c in $C_1 \cap C_2$ satisfy this inequality and therefore remain feasible. Thus, the problem (14.59) can be rewritten as

$$\min\{c(F_{N_1} - F_{N_2})b : cD_1 \leqslant 0, cD_2 \leqslant 0, cD_1\mathbf{1} \leqslant -1\, c \in \mathbb{Z}^n\}.$$

For a given b, this is an integer programming problem. Therefore, the optimum value of (14.59) is attained at some vertex of the integer hull of the underlying polyhedron

$$\{c : cD_1 \leqslant 0, cD_2 \leqslant 0, cD_1\mathbf{1} \leqslant -1\} \tag{14.60}$$

Shevchenko [77] and Hayes and Larman [35] proved that the number of vertices of the integer hull of a rational polyhedron is polynomial in fixed dimension. Tight bounds for this number were presented by Cook et al. [14] and Bárány et al. [6]. □

To justify conditions I,II) and III) one then partitions the parameter polyhedron Q into a polynomial number of polyhedra such that for those b in one partition, the width direction is always invariant. Via a unimodular transformation one can assume that this flat direction is the first unit vector, see exercises 1 and 2) for further details.

Notes

The problem to decide forall/exist statements belongs to the second level of the polynomial hierarchy and is Π_2^p-complete, see [78, 84]. Barvinok and Woods [7] extended the counting algorithm of Barvinok [8] such that it computes a short rational generating function for an integer projection of the set of integer points in a polytope. The algorithm is based on Kannan's partitioning lemma. The variant of the partitioning theorem which we described was used and further refined for in implementation of the algorithm [7] by Köppe et al. [52]. Köppe and Verdoolaege [53] presented an algorithm which computes a formula for the number of integer points in a parameterized polyhedron.

Exercises

1. Consider the polyhedra $P_b = \{x \in \mathbb{R}^n : Ax \leqslant b\}$ for $b \in Q$ and let $d \in \mathbb{Z}^n - \{0\}$, where $A \in \mathbb{Z}^{m\times n}$ has full column rank and n is fixed. Write down a polynomial number of inequalities which describe those $b \in Q$ such the $w(P_b) = w_d(P_b)$.
   Hint: Associate to each $d \in D$ from Theorem 14.36 an "entering basis" and "leaving basis".
2. Write down a linear program which partitions Q further into parameters such that condition III) holds.

3*. I s there a polynomial algorithm which, given a matrix $A \in \mathbb{Z}^{m\times n}$, where n is fixed and a polyhedron $Q \subseteq \mathbb{R}^m$ determines a $b \in Q$ such that the number of integer points in P_b is minimal?

Acknowledgments

I am grateful to Damien Stehlé for numerous comments and suggestions which helped me a lot to improve this manuscript. I also want to thank Johannes Blömer for several discussions on shortest and closest vectors.

References

1. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The design and analysis of computer algorithms*, Addison-Wesley, Reading, 1974.
2. M. Ajtai, *The shortest vector problem in L2 is NP-hard for randomized reductions*, Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC-98) (New York), ACM Press, 1998, pp. 10–19.
3. M. Ajtai, R. Kumar, and D. Sivakumar, *A sieve algorithm for the shortest lattice vector problem*, Proceedings of the thirty-third annual ACM symposium on Theory of computing, ACM Press, 2001, pp. 601–610.
4. M. Ajtai, R. Kumar, and D. Sivakumar, *Sampling short lattice vectors and the closest lattice vector problem*, Proceedings of the 17th IEEE Annual Conference on Computational Complexity (Montreal, Canada), 2002, pp. 53–67.
5. W. Banaszczyk, *Inequalities for convex bodies and polar reciprocal lattices in* $\mathbf{R}^n$. *II. Application of K-convexity*, Discrete Comput. Geom. **16** (1996), no. 3, 305–311. MR MR1410163 (97h:11073)
6. I. Bárány, R. Howe, and L. Lovász, *On integer points in polyhedra: A lower bound*, Combinatorica **12** (1992), no. 2, 135–142.
7. A. Barvinok and K. Woods, *Short rational generating functions for lattice point problems*, Journal of the American Mathematical Society **16** (2003), no. 4, 957–979 (electronic). MR MR1992831 (2004e:05009)
8. A. I. Barvinok, *A polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed*, Mathematics of Operations Research **19** (1994), no. 4, 769–779. MR MR1304623 (96c:52026)
9. J. Blömer, *Closest vectors, successive minima, and dual HKZ-bases of lattices*, Automata, Languages and Programming, 27th International Colloquium, ICALP 2000, Geneva, Switzerland, July 9-15, 2000, Proceedings (U. Montanari, J. D. P. Rolim, and E. Welzl, eds.), Lecture Notes in Computer Science, vol. 1853, Springer, 2000, pp. 248–259.
10. J. Blömer and S. Naewe, *Sampling methods for shortest vectors, closest vectors and successive minima*, Proceedings of the 34th International Colloquium on Automata, Languages and Programming, ICALP 2007 (L. Arge, C. Cachin, T. Jurdzinski, and A. Tarlecki, eds.), Lecture Notes in Computer Science, vol. 4596, Springer, 2007, pp. 65–77.
11. I. Borosh and L. B. Treybig, *Bounds on positive integral solutions of linear diophantine equations*, Proceedings of the American Mathematical Society **55** (1976), 299–304.
12. J.-Y. Cai and A. P. Nerurkar, *Approximating the svp to within a factor* $(1+1/dim^{\varepsilon})$ *is NP-hard under randomized reductions*, Proceedings of the 38th IEEE Conference on Computational Complexity (Pittsburgh), IEEE Computer Society Press, 1998, pp. 46–55.
13. K. L. Clarkson, *Las Vegas algorithms for linear and integer programming when the dimension is small*, Journal of the Association for Computing Machinery **42** (1995), 488–499.
14. W. Cook, M. E. Hartmann, R. Kannan, and C. McDiarmid, *On integer points in polyhedra*, Combinatorica **12** (1992), no. 1, 27 –37.
15. I. Dinur, *Approximating* SVP_∞ *to within almost-polynomial factors is NP-hard*, Theoretical Computer Science **285** (2002), no. 1, 55–71, Algorithms and complexity (Rome, 2000). MR MR1925787 (2003f:68048)
16. M. Dyer, A. Frieze, and R. Kannan, *A random polynomial-time algorithm for approximating the volume of convex bodies*, Journal of the ACM **38** (1991), no. 1, 1–17.
17. J. Edmonds, *Systems of distinct representatives and linear algebra*, Journalof Reseach of the National Bureau of Standards **71B** (1967), 241–245.
18. F. Eisenbrand, *Fast integer programming in fixed dimension*, In Proceedings of the 11th Annual European Symposium on Algorithms, ESA' 2003 (G. D. Battista and U. Zwick, eds.), LNCS, vol. 2832, Springer, 2003, pp. 196–207.
19. F. Eisenbrand and S. Laue, *A linear algorithm for integer programming in the plane*, Mathematical Programming **102** (2005), no. 2, Ser. A, 249–259. MR MR2124445 (2005k:90080)

20. F. Eisenbrand and G. Rote, *Fast reduction of ternary quadratic forms*, Cryptography and Lattices Conference, CALC 2001 (J. Silverman, ed.), LNCS, vol. 2146, Springer, 2001, pp. 32–44.
21. F. Eisenbrand and G. Shmonin, *Parametric integer programming in fixed dimension*, Mathematics of Operations Research (2008), to appear.
22. P. van Emde Boas, *Another NP-complete partition problem and the complexity of computing short vectors in a lattice*, Tech. Report MI-UvA-81-04, Mathematical Institute, University of Amsterdam, Amsterdam, 1981.
23. X. G. Fang and G. Havas, *On the worst-case complexity of integer Gaussian elimination*, Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation (Kihei, HI) (New York), ACM, 1997, pp. 28–31 (electronic). MR MR1809966
24. S. D. Feit, *A fast algorithm for the two-variable integer programming problem*, Journal of the Association for Computing Machinery **31** (1984), no. 1, 99–113.
25. A. Frank and É. Tardos, *An application of simultaneous Diophantine approximation in combinatorial optimization*, Combinatorica **7** (1987), 49–65.
26. M. Fürer, *Faster integer multiplication*, Proceedings of the 39th Annual ACM Symposium on Theory of Computing, STOC 2007 (D. S. Johnson and U. Feige, eds.), ACM, 2007, pp. 57–66.
27. N. Gama and P. Q. Nguyen, *Finding short lattice vectors within mordell's inequality*, Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, STOC 2008 (R. E. Ladner and C. Dwork, eds.), ACM, 2008, pp. 207–216.
28. B. Gärtner, *A subexponential algorithm for abstract optimization problems*, SIAM J. Comput. **24** (1995), no. 5, 1018–1035. MR MR1350756 (96m:68067)
29. C. F. Gauß, *Disquisitiones arithmeticae*, Gerh. Fleischer Iun., 1801.
30. O. Goldreich and S. Goldwasser, *On the limits of non-approximability of lattice problems*, Proceedings of the 30th Annual ACM Symposium on Theory of Computing (New York), ACM Press, 1998, pp. 1–9.
31. M. Grötschel, L. Lovász, and A. Schrijver, *Geometric algorithms and combinatorial optimization*, Algorithms and Combinatorics, vol. 2, Springer, 1988.
32. M. Grötschel, L. Lovász, and A. Schrijver, *Geometric methods in combinatorial optimization*, Progress in Combinatorial Optimization (W. R. Pulleyblank, ed.), Academic Press, Toronto, 1984, pp. 167–183. MR 86d:90127
33. G. Hanrot and D. Stehlé, *Improved analysis of Kannan's shortest lattice vector algorithm (extended abstract)*, Advances in cryptology—CRYPTO 2007, Lecture Notes in Comput. Sci., vol. 4622, Springer, Berlin, 2007, pp. 170–186. MR MR2419600
34. I. Haviv and O. Regev, *Tensor-based hardness of the shortest vector problem to within almost polynomial factors*, STOC'07—Proceedings of the 39th Annual ACM Symposium on Theory of Computing, ACM, New York, 2007, pp. 469–477. MR MR2402472 (2009e:68036)
35. A. C. Hayes and D. G. Larman, *The vertices of the knapsack polytope*, Discrete Applied Mathematics **6** (1983), no. 2, 135–138.
36. B. Helfrich, *Algorithms to construct Minkowski reduced and Hermite reduced lattice bases*, Theoretical Computer Science **41** (1985), no. 2-3, 125–139 (1986). MR MR847673 (87h:11124)
37. C. Hermite, *Extraits de lettres de M. Ch. Hermite à M. Jacobi sur différents objets de la théorie des nombres*, Journal für die reine und angewandte Mathematik **40** (1850).
38. D. S. Hirschberg and C. K. Wong, *A polynomial algorithm for the knapsack problem in two variables*, Journal of the Association for Computing Machinery **23** (1976), no. 1, 147–154.
39. F. John, *Extremum problems with inequalities as subsidiary conditions*, Studies and Essays Presented to R. Courant on his 60th Birthday, January 8, 1948, Interscience Publishers, Inc., New York, N. Y., 1948, pp. 187–204. MR MR0030135 (10,719b)
40. E. Kaltofen, *On the complexity of finding short vectors in integer lattices*, Computer Algebra: Proceedings of EUROCAL '1983 (J. VanHulzen, ed.), Lecture Notes in Computer Science, vol. 162, Springer-Verlag, Berlin, 1983, pp. 236–244. MR 86m:11101
41. N. Kanamaru, T. Nishizeki, and T. Asano, *Efficient enumeration of grid points in a convex polygon and its application to integer programming*, International Journal of Computational Geometry & Applications **4** (1994), no. 1, 69–85.

42. R. Kannan, *Improved algorithms for integer programming and related problems*, Proceedings of the 15th Annual ACM Symposium on Theory of Computing (New York), ACM Press, 1983, pp. 193–206.
43. ———, *Minkowski's convex body theorem and integer programming*, Mathematics of Operations Research **12** (1987), no. 3, 415–440.
44. ———, *Minkowski's convex body theorem and integer programming*, Mathematics of Operations Research **12** (1987), no. 3, 415–440.
45. ———, *Lattice translates of a polytope and the Frobenius problem*, Combinatorica **12** (1992), no. 2, 161–177.
46. R. Kannan, *A polynomial algorithm for the two-variable integer programming problem*, Journal of the Association for Computing Machinery **27** (1980), no. 1, 118–122.
47. R. Kannan and A. Bachem, *Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix*, SIAM Journal on Computing **8** (1979), no. 4, 499–507.
48. L. G. Khachiyan and M. J. Todd, *On the complexity of approximating the maximal inscribed ellipsoid for a polytope*, Math. Programming **61** (1993), no. 2, Ser. A, 137–159. MR MR1240454 (94m:90071)
49. A. Khinchine, *A quantitative formulation of Kronecker's theory of approximation (in russian)*, Izvestiya Akademii Nauk SSR Seriya Matematika **12** (1948), 113–122.
50. S. Khot, *Hardness of approximating the shortest vector problem in high l_p norms*, Journal of Computer and System Sciences **72** (2006), no. 2, 206–219. MR MR2205284 (2007b:68066)
51. D. Knuth, *The art of computer programming*, vol. 2, Addison-Wesley, 1969.
52. M. Köppe, S. Verdoolaege, and K. Woods, *An implementation of the barvinok–woods integer projection algorithm*, Tech. report, Katholieke Universiteit Leuven, 2008.
53. M. Köppe and S. Verdoolaege, *Computing parametric rational generating functions with a primal Barvinok algorithm*, Electron. J. Combin. **15** (2008), no. 1, Research Paper 16, 19. MR MR2383436
54. J. Lagarias, H. Lenstra, and C. Schnorr, *Korkin-zolotarev bases and successive minima of a lattice and its reciprocal lattice*, Combinatorica **10** (1990), no. 4, 333–348.
55. A. K. Lenstra, H. W. Lenstra, and L. Lovász, *Factoring polynomials with rational coefficients*, Math. Annalen **261** (1982), 515 – 534.
56. H. W. Lenstra, *Integer programming with a fixed number of variables*, Mathematics of Operations Research **8** (1983), no. 4, 538 – 548.
57. H. W. Lenstra, Jr., *Lattices*, Algorithmic number theory: lattices, number fields, curves and cryptography, Math. Sci. Res. Inst. Publ., Cambridge Univ. Press, Cambridge, 2008, pp. 127–181. MR MR2467546
58. L. Lovász, *An algorithmic theory of numbers, graphs and convexity*, SIAM, 1986.
59. J. Matoušek, M. Sharir, and E. Welzl, *A subexponential bound for linear programming*, Algorithmica **16** (1996), no. 4-5, 498–516. MR 97f:90052
60. D. Micciancio, *The shortest vector in a lattice is hard to approximate to within some constant*, Proceedings of the 39th Annual Symposium on Foundations of Computer Science (Los Alamitos, CA), IEEE Computer Society, 1998, pp. 92–98.
61. H. Minkowski, *Geometrie der Zahlen*, Teubner, Leipzig, 1896.
62. Y. E. Nesterov and A. S. Nemirovski, *Self–concordant functions and polynomial–time methods in convex programming*, Tech. report, Central Economic and Mathematical Institute, USSR Academy of Science, Moscow, USSR, 1989.
63. P. Q. Nguyen and D. Stehlé, *Floating-point LLL revisited*, Proceedings of the 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 05) (R. Cramer, ed.), Lecture Notes in Computer Science, vol. 3494, Springer, 2005, pp. 215–233.
64. P. Q. Nguyen and D. Stehlé, *Floating-point LLL revisited*, Advances in cryptology—EUROCRYPT 2005, Lecture Notes in Comput. Sci., vol. 3494, Springer, Berlin, 2005, pp. 215–233. MR MR2352190
65. I. Niven, H. S. Zuckerman, and H. L. Montgomery, *An introduction to the theory of numbers, fifth edition*, Wiley, 1991.

66. X. Pujol and D. Stehlé, *Rigorous and efficient short lattice vectors enumeration*, 14th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2008), 2008, pp. 390–405.
67. J. L. Ramírez-Alfonsín, *Complexity of the Frobenius problem*, Combinatorica **16** (1996), no. 1, 143–147. MR MR1394516 (97d:11058)
68. O. Regev, *Lattices in computer science*, Lecture notes, Tel Aviv University, 2004, http://www.cs.tau.ac.il/˜odedr/teaching/lattices_fall_2004/index.html .
69. O. Regev and R. Rosen, *Lattice problems and norm embeddings*, STOC'06: Proceedings of the 38th Annual ACM Symposium on Theory of Computing, ACM, New York, 2006, pp. 447–456. MR MR2277170 (2007i:68034)
70. H. E. Scarf, *Production sets with indivisibilities. Part I: generalities*, Econometrica **49** (1981), 1–32.
71. ———, *Production sets with indivisibilities. Part II: The case of two activities*, Econometrica **49** (1981), 395–423.
72. C.-P. Schnorr, *A hierarchy of polynomial time lattice basis reduction algorithms*, Theoretical Computer Science **53** (1987), no. 2-3, 201–224. MR 89h:11085
73. A. Schönhage, *Schnelle berechnung von kettenbruchentwicklungen*, Acta Informatica **1** (1971), 139–144.
74. A. Schönhage and V. Strassen, *Schnelle multiplikation grosser zahlen*, Computing **7** (1971), 281–292.
75. A. Schrijver, *Theory of linear and integer programming*, John Wiley & Sons, Chichester, 1986.
76. ———, *Theory of linear and integer programming*, John Wiley, 1986.
77. V. N. Shevchenko, *On the number of extreme points in integer programming*, Kibernetika (1981), no. 2, 133–134 (Russian).
78. L. J. Stockmeyer, *The polynomial-time hierarchy*, Theoretical Computer Science **3** (1976), no. 1, 1–22.
79. A. Storjohann, *Faster algorithms for integer lattice reduction*, Tech. Report 249, Department of Computer Science, ETH Zürich, 1996.
80. É. Tardos, *A strongly polynomial algorithm to solve combinatorial linear programs*, Operations Research **34** (1986), no. 2, 250–256. MR MR861043 (87i:90151)
81. J. von zur Gathen and J. Gerhard, *Modern computer algebra*, Cambridge University Press, 1999.
82. J. von zur Gathen and M. Sieveking, *Weitere zum erfüllungsproblem polynomial äquivalente kombinatorische aufgahen*, Komplexität von Entscheidungsproblemen: ein Seminar (E. Specker and V. Strassen, eds.), LNCS, vol. 43, Springer, 1976, pp. 49–71.
83. E. Welzl, *Smallest enclosing disks (balls and ellipsoids)*, New results and new trends in computer science (Graz, 1991), Lecture Notes in Comput. Sci., vol. 555, Springer, Berlin, 1991, pp. 359–370. MR MR1254721
84. C. Wrathall, *Complete sets and the polynomial-time hierarchy*, Theoretical Computer Science **3** (1976), no. 1, 23–33.
85. L. Y. Zamanskij and V. D. Cherkasskij, *A formula for determining the number of integral points on a straight line and its application*, Ehkon. Mat. Metody **20** (1984), 1132–1138, (in Russian).