MPRI 2-12-1: Phong Nguyen **Fall 2024**

Lecture 2: Fundamental Lattice Algorithms

We presented lattices from a mathematical point of view. Here, we introduce the most fundamental lattice algorithms.

In practice, one only deals with *rational lattices*, that is, lattices included in $\mathbb{Q}^n$. In this case, by a suitable multiplication, one only needs to be able to deal with integer lattices, those which are included in $\mathbb{Z}^n$. Such lattices are usually represented by a basis, that is, a matrix with integral coefficients. When we explicitly give such a matrix, we will adopt a row representation: the row vectors of the matrix will be the basis vectors. The size of the lattice is measured by the dimensions of the matrix (the number d of rows, which correspond to the lattice rank, and the number n of columns), and the maximal bit-length $\log B$ of the matrix coefficients: thus, the whole matrix can be stored using $dn \log B$ bits.

We start with a list of elementary problems over lattices:

- Given a generating set of an integer lattice L, find a basis of the lattice L. In other words, given arbitrary vectors $\mathbf{a}_1, \ldots, \mathbf{a}_n \in \mathbb{Z}^m$, find a basis of the lattice $L(\mathbf{a}_1, \ldots, \mathbf{a}_n)$ generated by the $\mathbf{a}_i$'s.
- Given a basis of an integer lattice $L \subseteq \mathbb{Z}^n$ and a target vector $\mathbf{v} \in \mathbb{Z}^n$, decide if $\mathbf{v} \in L$, and if so, find the $\mathbb{Z}$-decomposition of $\mathbf{v}$ with respect to the basis.
- Given bases of two integer lattices L_1 and L_2, decide if $L_1 = L_2$, compute a basis of the intersection lattice $L_1 \cap L_2$.
- Given bases of an integer lattice L and a rational subspace E, compute a basis of the intersection lattice $L \cap E$.

All these problems turn out to be solvable in deterministic polynomial time. They are typically solved by one or a combination of the following important tools: the Hermite normal form, duality and Gram-Schmidt orthogonalization.

1 The Hermite normal form

Unlike real lattices, it turns out that integer lattices have a canonical basis, known as the Hermite normal form. In this section, to simplify the exposition, we will restrict to full-rank integer lattices:

Theorem 1.1 *Let L be a full-rank lattice in $\mathbb{Z}^n$. Then:*

1. *$\mathbb{Z}^n/L$ is a finite abelian group.*
2. *L has a basis whose row representation is a lower-triangular matrix.*
3. *There is one and only basis of L whose row representation is a lower-triangular matrix $(h_{i,j})_{1\leq i,j\leq n}$ such that: $h_{i,i} > 0$ for all i, and $0 \leq h_{i,j} < h_{j,j}$ for all $i > j$.*
4. *The order $[\mathbb{Z}^n : L|$ of $\mathbb{Z}^n/L$ is $\mathrm{covol}(L)$.*

Proof. By definition, $\mathbb{Z}^n/L$ is an abelian group. Let $(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ be a basis of L. Let $\mathcal{P} = \{\sum_{i=1}^n x_i\mathbf{b}_i, 0 \leq x_i < 1\}$ be the parallelepiped spanned by the $\mathbf{b}_i$'s. We claim that any coset $\mathbf{u} + L$ (where $\mathbf{u} \in \mathbb{Z}^n$) intersects $\mathcal{P}$. Indeed, if $\mathbf{u} \in \mathbb{Z}^n$, there exist $u_1, \ldots, u_n \in \mathbb{R}$ such that $\mathbf{u} = \sum_{i=1}^n u_i\mathbf{b}_i$. Therefore $\mathbf{u} - \sum_{i=1}^n \lfloor u_i \rfloor \mathbf{b}_i \in \mathcal{P} \cap (\mathbf{u} + L)$. So any coset $\mathbf{u} + L$ has an element in $\mathbb{Z}^n \cap \mathcal{P}$. However, $\mathbb{Z}^n$ is a lattice, so the intersection $\mathbb{Z}^n \cap \mathcal{P}$ is finite. Hence, $\mathbb{Z}^n/L$ is finite, whose order is denoted by $[\mathbb{Z}^n : L]$.

By Lagrange's theorem, $[\mathbb{Z}^n : L]\mathbb{Z}^n \subseteq L$. Consider the rows of the $n \times n$ identity matrix multiplied by $[\mathbb{Z}^n : L]$: these are n linearly independent vectors in L. In the previous lecture notes, we showed a theorem proving that there exists a lattice basis, which is generating the same subspaces as these linearly independent vectors. This basis is therefore represented by a lower-triangular matrix, which proves 2).

Now, by possibly changing signs, we can always make the diagonal coefficients > 0. Each off-diagonal coefficient can be made ≥ 0 and strictly less than the diagonal coefficient in its row, by subtracting a suitable linear combinations of the previous rows: this may modify the previous off-diagonal coefficient in its row, so we need to start with the most-right coefficient in each row to obtain the existence in 3). The unicity can be proved by induction.

Using this lower-triangular basis, one sees that any coset $\mathbf{u} + L$ has one and only element in $\prod_{i=1}^n \{0, 1, \ldots, h_{i,i-1}\}$. This proves that the order $[\mathbb{Z}^n : L]$ is equal to $\prod_{i=1}^n h_{i,i} = \mathrm{covol}(L)$. □

The one and only basis of item 3) is called the Hermite normal form of L.

We now show to compute the Hermite normal form of L, given an arbitrary basis of a full-rank integer lattice L. The proof of Th. 1.1 shows that it suffices to find a lower-triangular basis. The simplest idea to find such a basis would be to perform an integral variant of Gaussian elimination, but the natural approach is actually not efficient, due to coefficient blow-up. The underlying idea, which will also be used for efficient HNF computations is the following:

Lemma 1.2 *Given $(a, b) \in \mathbb{Z}^2$, one can compute in deterministic polynomial time a unimodular matrix $U \in GL_2(\mathbb{Z})$ such that:*

$$U \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 0 \\ \gcd(a, b) \end{pmatrix}.$$

Proof. If $\gcd(a, b) = 0$, one can take the identity matrix for U. Otherwise, Euclid's extended gcd algorithm outputs $(\alpha, \beta) \in \mathbb{Z}^2$ such that $a\alpha + b\beta = \gcd(a, b)$, so one can take:

$$U = \begin{pmatrix} b/\gcd(a, b) & -a/\gcd(a, b) \\ \alpha & \beta \end{pmatrix}$$

□

This transformation U allows to modify a basis to cancel any chosen coordinate: starting with an arbitrary basis, we can repeatedly apply a transformation U to make all basis vectors ending with a zero coordinate, except the last basis vector. Then we iterate over the first $n-2$ vectors to zero the second-to-last coordinate, and so on until we get a lower-triangular matrix. The algorithm we described gives an alternate proof of the existence of the Hermite normal form. However, this algorithm might be inefficient, because when we apply transformations U repeatedly to zero certain coordinates, nothing prevents other coordinates to blow up, and such a blow up can really occur in practice: if we perform a polynomial number of operations, but each operation doubles the size of the operands, then the total running time is not polynomial. To avoid such a blow up, we just need to be able to reduce the coefficients after applying a transformation U: this can be achieved incrementally, as in Alg. 1, where a given HNF of a sublattice is used to reduce the coefficients of any lattice vector.

Lemma 1.3 *Let L be a full-rank integer lattice of $\mathbb{Z}^d$, and $\mathbf{a} \in \mathbb{Z}^d$. Given $\mathbf{a}$ and the HNF of L, Alg. 1 outputs the HNF of the lattice $L + \mathbb{Z}\mathbf{a}$ in time polynomial in d, $\log \mathrm{covol}(L)$ and the maximal size of the coefficients of $\mathbf{a}$.*

Proof. All the coefficients of the HNF of L are ≥ 0 and $\leq \mathrm{covol}(L)$. By Lemma 1.2, the size of the coefficients of U is polynomial in $\log \mathrm{covol}(L)$ and the maximal size of the coefficients of $\mathbf{a}$. Thus, after Step. 5 the size of each coefficient of $\mathbf{b}$ and $\mathbf{c}$ is also polynomial in $\log \mathrm{covol}(L)$ and the size of $\mathbf{a}$: denote by s_b and s_c the maximal coefficient sizes. Let s_h be the maximal size of a coefficient of the input HNF. In Step. 7, the size of $\lceil b_i/h_{i,i} \rfloor$ is bounded by the size of the current coefficients of $\mathbf{b}$. It follows that Step. 7 increases additively the size of s_b by at most s_h. So during the loop of Step. 6, the size of any coefficient is always bounded by $s_b + ds_h$, and the final size s_b is less than s_h, since the i-th coefficient of $\mathbf{b}$ is always less than $h_{i,i}$. The same reasoning holds for the second loop of Step. 10. It follows that the running time of Alg. 1 is polynomial in d, $\log \mathrm{covol}(L)$ and the maximal size of the coefficients of $\mathbf{a}$. □

Algorithm 1 Incremental HNF.

Input: The HNF $(\mathbf{h}_1, \ldots, \mathbf{h}_d)$ of a full-rank integer lattice $L \subseteq \mathbb{Z}^d$ and a vector $\mathbf{a} \in \mathbb{Z}^d$.

Output: The HNF of the augmented lattice $L + \mathbb{Z}\mathbf{a}$.

1: **if** $d = 1$ **then**
2: **return** $\gcd(h_{1,1}, a_1)$
3: **else**
4: Compute the transformation U from Lemma 1.2 such that $U \begin{pmatrix} h_{d,d} \\ a_d \end{pmatrix} = \begin{pmatrix} 0 \\ \gcd(h_{d,d}, a_d) \end{pmatrix}$.
5: Let $\begin{pmatrix} \mathbf{b} \\ \mathbf{c} \end{pmatrix} \longleftarrow U \begin{pmatrix} \mathbf{h}_d \\ \mathbf{a} \end{pmatrix}$
6: **for** $i = d - 1$ downto 1 **do**
7: $\mathbf{b} \longleftarrow \mathbf{b} - \lceil b_i/h_{i,i} \rfloor \mathbf{h}_i$
8: **end for**
9: Call recursively Incremental HNF on $(\mathbf{h}_1, \ldots, \mathbf{h}_{d-1})$ and $\mathbf{b}$, where the d-th entry of each row vector (which is zero) is ignored. Let $H = (h_{i,j})$ be the $(d-1) \times d$ output HNF, where the last column of H is zero.
10: **for** $i = d - 1$ down 1 **do**
11: $\mathbf{c} \longleftarrow \mathbf{c} - \lceil c_i/h_{i,i} \rfloor \mathbf{h}_i$
12: **end for**
13: **return** the row concatenation of H with $\mathbf{c}$.
14: **end if**

The incremental Alg. 1 gives rise to a simple polynomial-time HNF al-

gorithm: Alg. 2:

Theorem 1.4 *Let L be a full-rank integer lattice of $\mathbb{Z}^d$. Given a basis of L, Alg. 2 outputs the HNF of the lattice L in time polynomial in the size of the input basis.*

Proof. We assume it is known that the determinant can be computed in polynomial time: for instance, one can select a prime number p larger than the determinant (which is bounded by Hadamard's inequality), and do Gaussian elimination to compute the determinant modulo p, which turns out the be the exact value of the determinant. Then it suffices to apply Lemma. 1.3 iteratively, by noticing that during the loop, each coefficient of the temporary HNF is between 0 and $\mathrm{covol}(L)$, so the running time given by Lemma. 1.3 is polynomial in d, the size of $\mathrm{covol}(L)$ and the maximal size of a coefficient of $\mathbf{a}$, where $\mathbf{a}$ is one of the input basis vector. □

Algorithm 2 Computing the HNF.

Input: A basis $B = (\mathbf{b}_1, \ldots, \mathbf{b}_d)$ of a full-rank integer lattice L.
Output: The HNF of L.

1: Compute $V = |\det B|$: if the (group) exponent of $\mathbb{Z}^n/L$ is known, then take the exponent as V.
2: Let H be the $d \times d$ identity matrix, multiplied by V.
3: **for** $i = 1$ to d **do**
4: $\quad H \longleftarrow$ Incremental HNF $(H, \mathbf{b}_i)$
5: **end for**
6: Return H

The HNF has many applications: it can be used to decide if two lattices are identical, and it can be used to efficiently decide if a vector belongs to the lattice or not, and if so, find its decomposition with respect to the HNF:

Theorem 1.5 *Given as input the HNF of a full-rank lattice $L \subseteq \mathbb{Z}^d$ and a vector $\mathbf{a} \in \mathbb{Z}^d$, Alg. 3 runs in polynomial time, and decides if $\mathbf{a} \in L$, and if so, return the decomposition of $\mathbf{a}$ with respect to the HNF.*

Proof. It is clear that the each operation is polynomial time, and using the same argument as in Lemma. 1.3, one shows that each coefficient of $\mathbf{a}$ remains polynomially sized during the loop. □

2 Duality

We introduce the dual lattice:

Algorithm 3 Lattice membership

Input: The HNF $(\mathbf{h}_1, \ldots, \mathbf{h}_d)$ of a full-rank integer lattice $L \subseteq \mathbb{Z}^d$ and a vector $\mathbf{a} = (a_1, \ldots, a_d) \in \mathbb{Z}^d$.

Output: Decide if $\mathbf{a} \in L$, and if so, return $\mathbf{x}_1, \ldots, x_d \in \mathbb{Z}$ such that $\mathbf{a} = \sum_{i=1}^{d} x_i \mathbf{h}_i$.

1: **for** $i = d$ down 1 **do**
2: **if** a_i is not divisible by $h_{i,i}$ **then**
3: **return** $\mathbf{a} \notin L$
4: **else**
5: $x_i \longleftarrow a_i / h_{i,i}$
6: $\mathbf{a} \longleftarrow \mathbf{a} - x_i \mathbf{h}_i$
7: **end if**
8: **end for**
9: **return** $\mathbf{a} \in L$ and $(x_1, \ldots, x_d)$.

Definition 2.1 *Let L be a lattice in $\mathbb{R}^n$. The* dual lattice *of L is defined as:*

$$L^\times = \{\mathbf{y} \in \mathrm{span}(L) \text{ such that} \langle \mathbf{x}, \mathbf{y} \rangle \in \mathbb{Z} \text{ for all } \mathbf{x} \in L\}.$$

Theorem 2.2 *If L is a d-rank lattice of $\mathbb{R}^n$, then $L^\times$ is a d-rank lattice of $\mathbb{R}^n$. If $B = (\mathbf{b}_1, \ldots, \mathbf{b}_d)$ is a (row) basis of L, then $(BB^t)^{-1}B$ is a (row) basis of $L^\times$, called the dual basis of B. And*

$$\mathrm{covol}(L) \times \mathrm{covol}(L^\times) = 1.$$

Proof. By definition, $L^\times$ is a subgroup of $\mathbb{R}^n$. Let $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ be a basis of a lattice L. Let $\mathbf{y} \in L^\times$ be non-zero. Then $\mathbf{y} \in \mathrm{span}(L)$ and $\langle \mathbf{y}, \mathbf{b}_i \rangle \in \mathbb{Z}$ for all $1 \leq i \leq d$. Since $y \neq 0$, there exists $i \in \{1, \ldots, d\}$ such that $|\langle \mathbf{y}, \mathbf{b}_i \rangle| \geq 1$. By Cauchy-Schwarz, this implies that $\|\mathbf{y}\| \geq 1/\|\mathbf{b}_i\|$. Hence, we proved that the intersection of $L^\times$ with the open ball of radius $\min_i 1/\|\mathbf{b}_i\|$ is equal to $\{0\}$. This shows that $L^\times$ is discrete, and therefore a lattice in $\mathbb{R}^n$.

Let $\mathbf{y} \in L^\times$. Then $\langle \mathbf{y}, \mathbf{b}_i \rangle \in \mathbb{Z}$ for all $1 \leq i \leq d$. So $\mathbf{y}B^t \in \mathbb{Z}^d$. Since $L^\times \subseteq \mathrm{span}(L)$, there exists $\mathbf{x} \in \mathbb{R}^d$ such that $\mathbf{y} = \mathbf{x}B$. Therefore $\mathbf{x}BB^t \in \mathbb{Z}^d$, which means that $\mathbf{x} \in \mathbb{Z}^d(BB^t)^{-1}$, thus $\mathbf{y} \in \mathbb{Z}^d(BB^t)^{-1}B$. Reciprocally, each row of $(BB^t)^{-1}B$ belongs to $L^\times$. Hence, $(BB^t)^{-1}B$ is a (row) basis of $L^\times$ and $\mathrm{rank}(L^\times) = d$.

It follows that:

$$\mathrm{covol}(L^\times)^2 = \det(BB^t)^{-1}B[(BB^t)^{-1}B]^t = \det(BB^t)^{-1} = 1/\mathrm{covol}(L)^2.$$

□

Duality also allows to consider sublattices of lower dimension, which can be used in proofs by induction. For instance, if L is a d-rank lattice and $\mathbf{v}$ is a non-zero vector of $L^\times$, then $L \cap H$ is a $(d-1)$-rank sublattice of L, where $H = \mathbf{v}^\perp$ denotes the hyperplane orthogonal to $\mathbf{v}$. And the projection of L over $H^\perp = \mathrm{span}(\mathbf{v})$ is $\mathbb{Z}\mathbf{v}/\|\mathbf{v}\|^2$.

3 Gram-Schmidt orthogonalization

Gram-Schmidt orthogonalization is a classical tool in bilinear algebra to construct orthogonal bases: it is also widely used in lattice algorithms.

Definition 3.1 *Let $\mathbf{b}_1, \ldots, \mathbf{b}_d$ be vectors in $\mathbb{R}^n$. Their Gram-Schmidt orthogonalization (GSO) is the sequence $(\mathbf{b}_1^\star, \ldots, \mathbf{b}_d^\star)$ defined as follows: $\mathbf{b}_1^\star = \mathbf{b}_1$ and more generally $\mathbf{b}_i^\star = \pi_i(\mathbf{b}_i)$ for $1 \leq i \leq d$, where π_i denotes the orthogonal projection over the orthogonal supplement of the linear span of $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$, which means that $\pi_i(\mathbf{x})$ is the component of $\mathbf{x}$ which is orthogonal to $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$.*

Notice that the GSO depends on the order of the vectors. We start with elementary properties:

Theorem 3.2 *Let $\mathbf{b}_1, \ldots, \mathbf{b}_d$ be vectors in $\mathbb{R}^n$, with Gram-Schmidt orthogonalization $(\mathbf{b}_1^\star, \ldots, \mathbf{b}_d^\star)$. Then:*

1. *For all $1 \leq i \leq d$, $\mathrm{span}(\mathbf{b}_1^\star, \ldots, \mathbf{b}_i^\star) = \mathrm{span}(\mathbf{b}_1, \ldots, \mathbf{b}_i)$.*
2. *For all $i \neq j$, $\langle \mathbf{b}_i^\star, \mathbf{b}_j^\star \rangle = 0$.*
3. *The $\mathbf{b}_i^\star$'s which are non-zero are linearly independent.*
4. *For $1 \leq j < i \leq d$, let $\mu_{i,j} = \frac{\langle \mathbf{b}_i, \mathbf{b}_j^\star \rangle}{\|\mathbf{b}_j^\star\|^2}$ if $\mathbf{b}_j^\star \neq 0$, and zero otherwise. Then we have the recursive fomula:*

$$\mathbf{b}_i^\star = \mathbf{b}_i - \sum_{j=1}^{i-1} \mu_{i,j} \mathbf{b}_j^\star, 1 \leq i \leq d \tag{1}$$

5. $\mathrm{vol}(\mathbf{b}_1, \ldots, \mathbf{b}_d) = \prod_{i=1}^d \|\mathbf{b}_i^\star\|$.

6. We have for all $1 \leq i \leq d$*:*

$$\|\mathbf{b}_i\|^2 = \|\mathbf{b}_i^\star\|^2 + \sum_{j=1}^{i-1} \mu_{i,j}^2 \|\mathbf{b}_j^\star\|^2 \tag{2}$$

7. If $1 \leq j < i \leq d$ *and* $\mathbf{b}_j^\star \neq 0$*, then:*

$$\mu_{i,j} = \frac{\langle \mathbf{b}_i, \mathbf{b}_j \rangle - \sum_{k=1}^{j-1} \mu_{j,k}\ \mu_{i,k} \|\mathbf{b}_k^\star\|^2}{\|\mathbf{b}_j^\star\|^2} \tag{3}$$

Proof. The first two items follow from the definition $\mathbf{b}_i^\star = \pi_i(\mathbf{b}_i)$. The third item follows from the second item. Let $1 \leq i \leq d$. If $i = 1$, there is nothing to prove, so assume that $i \geq 2$. Then $\mathbf{b}_i - \mathbf{b}_i^\star \in \mathrm{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}) = \mathrm{span}(\mathbf{b}_1^\star, \ldots, \mathbf{b}_{i-1}^\star)$. Because of item 3), there exist $x_1, \ldots, x_{i-1} \in \mathbb{R}$ such that $\mathbf{b}_i - \mathbf{b}_i^\star = \sum_{j=1}^{i-1} x_j \mathbf{b}_j^\star$. If $1 \leq j < i$, by taking the inner product with $\mathbf{b}_j^\star$, and using item 2), we obtain that $\langle \mathbf{b}_i, \mathbf{b}_j^\star \rangle = x_j \|\mathbf{b}_j^\star\|^2$. If $\mathbf{b}_j^\star \neq 0$, this shows that $x_j = \mu_{i,j}$. This proves (1). The fifth item follows from (1): the Gram determinant of $\mathbf{b}_1, \ldots, \mathbf{b}_d$ is equal to that of $\mathbf{b}_1^\star, \ldots, \mathbf{b}_d^\star$, which is the product of the $\|\mathbf{b}_i^\star\|^2$ by orthogonality. The equation (2) follows from (1) and orthogonality of the $\mathbf{b}_i^\star$'s. And (3) follows by plugging (1) in the formula for $\mu_{i,j}$. □

The HNF allowed us to work with triangular bases. The Gram-Schmidt orthogonalization is widely used in lattice reduction because it also allows us to triangularize the basis, in a different way. More precisely, the family $(\mathbf{b}_1^\star/\|\mathbf{b}_1^\star\|, \ldots, \mathbf{b}_d^\star/\|\mathbf{b}_d^\star\|)$ is an orthonormal basis of $\mathbb{R}^n$. And if we express the row vectors $\mathbf{b}_1, \ldots, \mathbf{b}_d$ with respect to the orthonormal basis $(\mathbf{b}_1^\star/\|\mathbf{b}_1^\star\|, \ldots, \mathbf{b}_d^\star/\|\mathbf{b}_d^\star\|)$ (rather than the canonical basis), we obtain the following lower-triangular matrix, with diagonal coefficients $\|\mathbf{b}_1^\star\|, \ldots, \|\mathbf{b}_d^\star\|$:

$$\begin{pmatrix} \|\mathbf{b}_1^\star\| & 0 & \ldots & 0 \\ \mu_{2,1}\|\mathbf{b}_1^\star\| & \|\mathbf{b}_2^\star\| & \ddots & \\ & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ \mu_{d,1}\|\mathbf{b}_1^\star\| & \ldots & \mu_{d,d-1}\|\mathbf{b}_{d-1}^\star\| & \|\mathbf{b}_d^\star\| \end{pmatrix} \tag{4}$$

This can be summarized by the matrix equality $B = \mu B^\star$, where B is the $d \times n$ matrix whose rows are $\mathbf{b}_1, \ldots, \mathbf{b}_d$, $B^\star$ is the $d \times n$ matrix whose rows are $\mathbf{b}_1^\star, \ldots, \mathbf{b}_d^\star$, and μ is the $d \times d$ lower-triangular matrix, whose diagonal coefficients are all equal to 1, and whose off-diagonal coefficients are the $\mu_{i,j}$'s.

The basis triangularization could have been obtained with other factorizations. For instance, if we had used Iwasa's decomposition of the row matrix B corresponding to $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$, we would have obtained $B = UDO$ where U is a lower-triangular matrix with unit diagonal, D is diagonal, and O is an orthogonal matrix. In other words, U would be the matrix defined by the $\mu_{i,j}$'s (lower-triangular with unit diagonal, where the remaining coefficients are the $\mu_{i,j}$'s), D would be the diagonal matrix defined by the $\|\mathbf{b}_i^\star\|$'s, and O would be the row representation of $(\mathbf{b}_1^\star/\|\mathbf{b}_1^\star\|, \ldots, \mathbf{b}_d^\star/\|\mathbf{b}_d^\star\|)$.

3.1 Relationship with Duality

Gram-Schmidt orthogonalization is related to duality.

For any $i \in \{2, \ldots, d\}$, the vector $\mathbf{b}_i^\star/\|\mathbf{b}_i^\star\|^2$ is orthogonal to $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$, and we have $\langle \mathbf{b}_i^\star/\|\mathbf{b}_i^\star\|^2, \mathbf{b}_i \rangle = 1$, which implies that:

$$\mathbf{b}_i^\star/\|\mathbf{b}_i^\star\|^2 \in \mathcal{L}(\mathbf{b}_1, \ldots, \mathbf{b}_i)^\times.$$

3.2 Computing Gram-Schmidt

The formulas (1), (2) and (3) proves the following by induction:

Lemma 3.3 *If $\mathbf{b}_i \in \mathbb{Q}^n$, then its GSO satisfies: $\mathbf{b}_i^\star \in \mathbb{Q}^n$, $\mu_{i,j} \in \mathbb{Q}$ and $\|\mathbf{b}_i^\star\|^2 \in \mathbb{Q}$.*

It is now natural to ask whether these rational quantities can be computed in polynomial time. First, we need to find out polynomial-sized denominators.

From now on, we assume that $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{Z}^m$ and we let $M = \max_{1 \le i \le n} \|\mathbf{b}_i\|$. We define the following integers:

- $d_0 = 1$.
- $d_i = \mathrm{Gram}(\mathbf{b}_1, \ldots, \mathbf{b}_i) = \prod_{j=1}^{i} \|\mathbf{b}_j^\star\|^2$, so $1 \le d_i \le M^{2i}$.

We obtain good denominators as follows:

Theorem 3.4 *Let $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{Z}^m$. Then for all $1 \le j < i \le n$:*

- *$d_{i-1}\mathbf{b}_i^\star \in \mathcal{L}(\mathbf{b}_1, \ldots, \mathbf{b}_i) \subseteq \mathbb{Z}^m$ where $\|d_{i-1}\mathbf{b}_i^\star\| \le M^{2i-1}$.*
- *$\lambda_{i,j} = d_j \mu_{i,j} \in \mathbb{Z}$ where $|\lambda_{i,j}| \le M^{2j}$.*

Proof. We may assume that $d_{i-1} \neq 0$, otherwise there is nothing to prove.

Let $L = \mathcal{L}((\mathbf{b}_1, \ldots, \mathbf{b}_i)$ and denote by $L^\times$ be its dual lattice. Then $[L^\times : L] = \mathrm{covol}(L)^2 = d_i$. By duality, we know that $\mathbf{b}_i^\star/\|\mathbf{b}_i^\star\|^2 \in L^\times$, so $[L^\times : L]\mathbf{b}_i^\star/\|\mathbf{b}_i^\star\|^2 \in L$ which means that $d_{i-1}\mathbf{b}_i^\star \in L(\mathbf{b}_1, \ldots, \mathbf{b}_i) \subseteq \mathbb{Z}^m$. □

The integers d_i's and $\lambda_{i,j}$ are called the integral Gram-Schmidt orthogonalization of $(\mathbf{b}_1, \ldots, \mathbf{b}_n)$:

Theorem 3.5 *Given as input linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{Z}^m$, one can compute in polynomial time the integers $d_i = \prod_{j=1}^{i} \|\mathbf{b}_j^\star\|^2$ and $\lambda_{i,j} = d_j \mu_{i,j}$ by the following formulas:*

$$d_i = d_{i-1}\|\mathbf{b}_i\|^2 - \sum_{j=1}^{i-1} \frac{d_{i-1}\lambda_{i,j}^2}{d_j d_{j-1}} \tag{5}$$

$$\lambda_{i,j} = d_{j-1}\langle \mathbf{b}_i, \mathbf{b}_j \rangle - \sum_{k=1}^{j-1} \frac{d_{j-1}\ \lambda_{j,k}\lambda_{i,k}}{d_k d_{k-1}} \tag{6}$$

Furthermore:

$$\mathbf{b}_i^\star = \mathbf{b}_i - \sum_{j=1}^{i-1} \frac{\lambda_{i,j}}{d_j}\mathbf{b}_j^\star \tag{7}$$

$$\|\mathbf{b}_i^\star\|^2 = \frac{d_i}{d_{i-1}} \tag{8}$$

Proof. (2) can be rewritten as:

$$\|\mathbf{b}_i^\star\|^2 = \frac{d_i}{d_{i-1}} = \|\mathbf{b}_i^\star\|^2 - \sum_{j=1}^{i-1} \mu_{i,j}^2 \|\mathbf{b}_j^\star\|^2 = \|\mathbf{b}_i^\star\|^2 - \sum_{j=1}^{i-1} \frac{\lambda_{i,j}^2}{d_j^2}\|\mathbf{b}_j^\star\|^2,$$

which becomes (5) by multiplication by d_{i-1}. □

Equations (5) and (6) can be unified if we let $\lambda_{i,i} = d_i$. So let $\lambda_{0,0} = 1$ and for all $1 \leq i \leq d$, let: $\lambda_{i,i} = \prod_{j=1}^{i} \|\mathbf{b}_j^*\|^2 = \mathrm{vol}(\mathbf{b}_1, \ldots, \mathbf{b}_i)^2 \in \mathbb{Z}$. Then we let $\lambda_{i,j} = \mu_{i,j}\lambda_{j,j}$ for all $j < i$, so that $\mu_{i,j} = \frac{\lambda_{i,j}}{\lambda_{j,j}}$. Since $\|\mathbf{b}_i^\star\|^2 = \lambda_{i,i}/\lambda_{i-1,i-1}$, it follows that the GSO can be derived from the integral matrix $\lambda = (\lambda_{i,j})_{1 \leq i,j \leq d}$. Alg. 4 computes the matrix integral matrix $\lambda = (\lambda_{i,j})_{1 \leq i,j \leq d}$ in polynomial time.

3.3 Size reduction

A basis B of a lattice L is said to be *size-reduced* if its GSO satisfies: $|\mu_{i,j}| \leq 1/2$ for all $j < i$. From the triangular representation of the basis, it is very easy to see how to size-reduce a basis in polynomial time (See Alg. 5): the vectors $\mathbf{b}_i$'s are modified, but not their projections $\mathbf{b}_i^\star$. Here, Alg. 5 used the rational coefficients $\mu_{i,j}$, but it is easy to rewrite Alg. 5 using the integral λ matrix.

Algorithm 4 Integral Gram-Schmidt

Input: A set of d linearly independent vectors $[\mathbf{b_1}, ..., \mathbf{b_d}]$ of $\mathbb{Z}^n$

Output: The λ matrix of the GSO of $[\mathbf{b_1}, ..., \mathbf{b_d}]$.

for $i = 1$ to d **do**
  $\lambda_{i,1} \leftarrow \langle \mathbf{b}_i, \mathbf{b}_1 \rangle$
  for $j = 2$ to i **do**
    $S = \lambda_{i,1}\lambda_{j,1}$
    for $k = 2$ to $j - 1$ **do**
      $S \leftarrow (\lambda_{k,k}S + \lambda_{j,k}\lambda_{i,k})/\lambda_{k-1,k-1}$
    end for
    $\lambda_{i,j} \leftarrow \langle \mathbf{b}_i, \mathbf{b}_j \rangle \lambda_{j-1,j-1} - S$
  end for
end for

Algorithm 5 Size reduction.

Input: A basis $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ of a lattice L.

Output: The basis $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ becomes size reduced without changing the Gram-Schmidt vectors.

Compute all the Gram-Schmidt coefficients $\mu_{i,j}$.
for $i = 2$ to d **do**
  for $j = i - 1$ downto 1 **do**
    $\mathbf{b}_i \longleftarrow \mathbf{b}_i - \lceil \mu_{i,j} \rfloor \mathbf{b}_j$
    for $k = 1$ to j **do**
      $\mu_{i,k} \longleftarrow \mu_{i,k} - \lceil \mu_{i,j} \rfloor \mu_{j,k}$
    end for
  end for
end for

3.4 Lattice Membership

The HNF allowed us to decide membership, thanks to its triangular form. We can do the same with the GSO: Alg. 6 is the GSO variant of Alg. 3, and it works for any integer lattice, not necessarily full-rank. We write Alg. 6 using a rational GSO, but it can easily be adapted with the integral GSO.

Theorem 3.6 *Given as input a basis of a lattice $L \subseteq \mathbb{Z}^n$ and a vector $\mathbf{a} \in \mathbb{Z}^n$, Alg. 6 runs in polynomial time, and decides if $\mathbf{a} \in L$, and if so, return the decomposition of $\mathbf{a}$ with respect to the input basis.*

Proof. This is similar arguments as in Alg. 3: one needs to show that each rational coefficient remains polynomially sized during the loop. □

Algorithm 6 Lattice membership using the GSO

Input: A basis $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ of an integer lattice $L \subseteq \mathbb{Z}^n$ and a vector $\mathbf{a} \in \mathbb{Z}^n$.
Output: Decide if $\mathbf{a} \in L$, and if so, return $\mathbf{x}_1, \ldots, x_d \in \mathbb{Z}$ such that $\mathbf{a} = \sum_{i=1}^{d} x_i \mathbf{b}_i$.

Compute the GSO of $(\mathbf{b}_1, \ldots, \mathbf{b}_d, \mathbf{a})$ using Alg. 4.
for $i = d$ downto 1 **do**
  if $\mu_{d+1,i} \notin \mathbb{Z}$ **then**
    return $\mathbf{a} \notin L$
  else
    $x_i \longleftarrow \mu_{d+1,i}$
    $\mathbf{a} \longleftarrow \mathbf{a} - x_i \mathbf{b}_i$
    for $k = 1$ to $i - 1$ **do**
      $\mu_{d+1,k} \longleftarrow \mu_{d+1,k} - x_i \mu_{i,k}$ // Update the GSO
    end for
  end if
end for
if $\mathbf{a} = 0$ **then**
  return $\mathbf{a} \in L$ and $(x_1, \ldots, x_d)$.
else
  return $\mathbf{a} \notin L$
end if

4 Computing a basis

We present two algorithms to compute a lattice basis from generators.

4.1 Computing a basis with swaps

The first algorithm is Alg. 7, which iteratively modifies the sequence $(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ of generators given as input. It uses two indices j and z such that: $\mathbf{b}_1, \ldots, \mathbf{b}_z$ are zero vectors, but $\mathbf{b}_{z+1}, \ldots, \mathbf{b}_{j-1}$ are linearly independent. At the end of the algorithm, $j-1 = n$, which implies that $(\mathbf{b}_{z+1}, \ldots, \mathbf{b}_n)$ is a basis. It can be checked that if the input matrix is actually a basis, then the algorithm returns the same basis. The main result on Alg. 7 is the following:

Theorem 4.1 *Given as input a row integral matrix B_0, Alg. 7 runs in time polynomial in the size of B_0, and outputs a basis B of the lattice L spanned by the rows of B_0 such that $\|B\| \leq \sqrt{\text{rank}(L)}\|B_0\|$. where $\|.\|$ denotes the maximal row norm.*

Proof. Let L be the lattice spanned by the rows of the input matrix B_0. It can easily be checked that during the algorithm, we always have $L = L(\mathbf{b}_1, \ldots, \mathbf{b}_n)$, *i.e.* the $\mathbf{b}_i$'s generate L.

We first prove the correctness of the algorithm. To do so, note that the following invariants hold at Steps 3 and 22:

1. $j \geq z + 2$
2. If $1 \leq i \leq z$, then $\mathbf{b}_i = 0$.
3. If $z < i < j$, then $\mathbf{b}_i^\star \neq 0$.

Property 1 is obvious. Property 2 holds initially when $z = 0$, and the only operation which increases z is Step 17, where a zero vector is inserted at index $z + 1$. Property 3 also holds initially when $j = 2$ because the input $\mathbf{b}_1 \neq 0$: the only operations which can change j are Steps 4 and 20. Step 4 preserves 3 by definition. For Step 20, $\mathbf{b}_{j-1}^\star$ and $\mathbf{b}_j^\star$ have been changed: either j decreases which preserves 3 no matter, or $j = z + 2$, which implies that $\mathbf{b}_{z+1}$ has been replaced by $\mathbf{b}_j \neq 0$, and therefore $\mathbf{b}_{z+1}^\star = \mathbf{b}_j \neq 0$ which proves 3. Now if the algorithm terminates, then $j = n + 1$: 3 implies that $\mathbf{b}_{z+1}, \ldots, \mathbf{b}_n$ are linearly independent, and together with 2, it implies that they form a basis of L: hence, the output returned by Step 24 is indeed a basis of L.

Next, we study the running time of the algorithm. We consider the potential $D = D_L \times D_R$ where:

$$D_L = \prod_{i=1}^{\dim(L)} d_i \quad \text{and} \quad D_R = \prod_{\|Starb_i\|=0} 2^i.$$

with d_i the product of the i first non-zero $\|\mathbf{b}_i^\star\|^2$'s. It is known that the d_i's are strictly positive integers, and so are therefore D_L, D_R and D.

We use D to upper bound the number of iterations of the `while` loop, and we study the evolution of $\|B^\star\|$. Initially, we have:

$$D \leq \|B_0\|^{\dim(L)^2} 2^{n^2} \quad \text{and} \quad \|B^\star\| \leq \|B_0\|.$$

Since size-reduction does not change the Gram-Schmidt vectors, the only operations which can change D or $\|B^\star\|$ are Steps 14-17 (`then`) and Step 19.

Steps 14-17 do not change D_L, but they decrease D_R by a multiplicative factor ≥ 2: indeed, 1 implies that at least one non-zero vector is moved, and a zero vector is inserted at index $z+2$. Thus, Steps 14-17 decrease D by a multiplicative factor ≥ 2, and they do not change $\|B^*\|$.

Now, consider the effect of Step 19 on the Gram-Schmidt vectors:

- The new $\mathbf{b}_{j-1}^\star$ is $\mu_{j,j-1}\mathbf{b}_{j-1}^\star$ where $|\mu_{j,j-1}| \leq 1/2$ by size-reduction.
- The new $\mathbf{b}_j^\star$ is either zero if $\mu_{j,j-1} \neq 0$, or $\mathbf{b}_{j-1}^\star$ otherwise.
- All the other Gram-Schmidt vectors are preserved.

This implies that Step 19 cannot increase $\|B^\star\|$, and there are two cases:

- If $\mu_{j,j-1} \neq 0$, then D_L decreases by a multiplicative factor ≥ 4 (because $0 < \mu_{j,j-1}^2 \leq 1/4$) and D_R increases by a factor 2.
- Otherwise $\mu_{j,j-1} = 0$, which preserves D_L, but decreases D_R decreases by a multiplicative factor 2.

To summarize, $\|B^\star\|$ never increases, and D decreases by a multiplicative factor ≥ 2 at each Step. 5. This proves that the number of loop iterations is upper bounded by $O(\log \|B_0\|^{\dim(L)^2} 2^{n^2})$, which is polynomial in $size(B_0)$. It remains to bound the cost of each iteration.

The only operation which can change $\|B\|$ is the size-reduction (Step. 6): at this point, we know that $\mathbf{b}_j^\star = 0$, therefore $\mathbf{b}_j = \sum_{i=1}^{j-1} \mu_{j,i}\mathbf{b}_i^\star$ where $|\mu_{j,i}| \leq 1/2$ and the $\mathbf{b}_i^\star$'s are pairwise orthogonal, thus $\|\mathbf{b}_j\| \leq \|B^\star\|\sqrt{\dim(L)}/2$ because at most $\dim(L)$ coefficients $\mu_{j,i}$'s are non-zero. Hence, we always have:

$$\|B\| \leq \sqrt{\dim(L)} \times \|B_0\|.$$

This shows that the size of the $\mathbf{b}_i$'s is always polynomial in $size(B_0)$, and it follows that each loop iteration runs in time polynomial in $size(B_0)$. □

Alg. 7 has many applications: for instance, it can be used to extend a primitive set to a lattice basis: if E is a subspace and L is a lattice, then $E \cap L$ is a sublattice such that any $\mathbb{Z}$-basis of $E \cap L$ is a primitive set, which means that it can be extended into a $\mathbb{Z}$-basis of L.

4.2 Computing a basis with XGCDs

In the lecture, we presented a faster variant of Alg. 7, where we collect several operations together: instead of doing a series of size reductions and swaps, we perform a series of transformations (based on XGCD) and size-reduction. This is very similar to the Incremental HNF approach of Alg. 1. In Alg. 8, we are given a basis $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ of an integer lattice $L \subseteq \mathbb{Z}^n$ and a vector $\mathbf{b}_{d+1} \in \mathbb{Z}^n$, and we want to obtain a basis of the augmented lattice $L + \mathbb{Z}\mathbf{b}_{d+1}$. We may assume that $\mathbf{b}^\star_{d+1} = 0$, otherwise we already know a basis. In Alg. 1, we canceled the last coordinate by applying a well-chosen unimodular transform over the last two vectors, based on the XGCD algorithm: here, we want to cancel the projection π over $\mathrm{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{d-1})^\perp$. Initially, we have $\pi(\mathbf{b}_d) = \mathbf{b}^\star_d$ and $\pi(\mathbf{b}_{d+1}) = \mu_{d+1,d}\mathbf{b}^\star_d = \frac{\lambda_{d+1,d}}{\lambda_{d,d}}\mathbf{b}^\star_d$. After transforming $\mathbf{b}_d$ and $\mathbf{b}_{d+1}$, we want $\pi(\mathbf{b}_d) = 0$ and $\pi(\mathbf{b}_{d+1}) \neq 0$. This gives rise to Alg. 8, whose similarity with Alg. 1 is direct. We immediately obtain a basis algorithm by iterating the algorithm:

Theorem 4.2 *Given as input a row integral matrix A, Alg. 9 runs in time polynomial in the size of A, and outputs a basis B of the lattice L spanned by the rows of A such that $\|B\| \leq \sqrt{\mathrm{rank}(L)}\|A\|$. where $\|.\|$ denotes the maximal row norm.*

Algorithm 7 Computing a basis of a lattice given by generators

Input: A row matrix $B = (\mathbf{b}_1, \ldots, \mathbf{b}_n) \in \mathbb{Z}^{n \times m}$ such that $\mathbf{b}_1 \neq \mathbf{0}$.

Output: A basis of the lattice $L = L(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ spanned by the rows of B.

$z \leftarrow 0$, $j \leftarrow 2$
while $j \leq n$ **do**
  if $\mathbf{b}_j^* \neq \mathbf{0}$ **then**
    $j \leftarrow j + 1$
  else
    Size-reduce $\mathbf{b}_j$ {with respect to the previous vectors $(\mathbf{b}_{z+1}, \ldots, \mathbf{b}_{j-1})$ as follows}.
    for $i = j - 1$ downto $z + 1$ **do**
      $\mathbf{b}_j \longleftarrow \mathbf{b}_j - \lceil \mu_{j,i} \rfloor \mathbf{b}_i$
      for $k = 1$ to $j - 1$ **do**
        $\mu_{j,k} \longleftarrow \mu_{j,k} - \lceil \mu_{j,i} \rfloor \mu_{j,k}$
      end for
    end for
    if $\mathbf{b}_j = 0$ **then**
      for $i = j$ downto $z + 2$ {move $\mathbf{b}_j$ to the front, and shift the rest} **do**
        $\mathbf{b}_i \leftarrow \mathbf{b}_{i-1}$
      end for
      $\mathbf{b}_{z+1} \leftarrow 0$, $j \leftarrow j + 1$; $z \leftarrow z + 1$ {We have found one more zero-vector}
    else
      Swap $\mathbf{b}_{j-1}$ and $\mathbf{b}_j$
      $j \leftarrow \max\{z + 2, j - 1\}$
    end if
  end if
end while
return $(\mathbf{b}_{z+1}, \ldots, \mathbf{b}_n)$

Algorithm 8 Incremental Basis.

Input: A basis $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ of an integer lattice $L \subseteq \mathbb{Z}^n$ and a vector $\mathbf{b}_{d+1} \in \mathbb{Z}^n$.

Output: A basis of the augmented lattice $L + \mathbb{Z}\mathbf{b}_{d+1}$.

Compute the λ-GSO of $(\mathbf{b}_1, \ldots, \mathbf{b}_{d+1})$
if $\mathbf{b}^\star_{d+1} \neq 0$, *i.e.* $\lambda_{d+1,d} \neq 0$ **then**
  return $(\mathbf{b}_1, \ldots, \mathbf{b}_{d+1})$
else
  Compute the transformation U from Lemma 1.2 such that $U \begin{pmatrix} \lambda_{d,d} \\ \lambda_{d+1,d} \end{pmatrix} = \begin{pmatrix} 0 \\ \gcd(\lambda_{d,d}, \lambda_{d+1,d}) \end{pmatrix}$.
  $\begin{pmatrix} \mathbf{b}_d \\ \mathbf{b}_{d+1} \end{pmatrix} \longleftarrow U \begin{pmatrix} \mathbf{b}_d \\ \mathbf{b}_{d+1} \end{pmatrix}$
  if $d = 1$ **then**
    return $\mathbf{b}_{d+1}$
  else
    // Size-reduce $\mathbf{b}_d$ and $\mathbf{b}_{d+1}$
    for $i = d - 1$ downto 1 **do**
      $\mathbf{b}_d \longleftarrow \mathbf{b}_d - \lceil \mu_{d,i} \rfloor \mathbf{b}_i$
      for $k = 1$ to $i - 1$ **do**
        $\mu_{d,k} \longleftarrow \mu_{d,k} - \lceil \mu_{d,i} \rfloor \mu_{i,k}$
      end for
    end for
    for $i = d$ downto 1 **do**
      $\mathbf{b}_{d+1} \longleftarrow \mathbf{b}_{d+1} - \lceil \mu_{d+1,i} \rfloor \mathbf{b}_i$
      for $k = 1$ to $i - 1$ **do**
        $\mu_{d+1,k} \longleftarrow \mu_{d+1,k} - \lceil \mu_{d+1,i} \rfloor \mu_{i,k}$
      end for
    end for
    Call recursively Incremental Basis on $(\mathbf{b}_1, \ldots, \mathbf{b}_{d-1})$ and $\mathbf{b}_d$
    return The row concatenation of the output with $\mathbf{b}_{d+1}$.
  end if
end if

Algorithm 9 Computing a basis incrementally.

Input: Integral vectors $(\mathbf{a}_1, \ldots, \mathbf{a}_m)$ in $\mathbb{Z}^n$.
Output: A $\mathbb{Z}$-basis of the lattice L spanned by the $\mathbf{a}_i$'s.

1: Make $\mathbf{a}_1 \neq 0$ by removing all the front zero vectors.
2: Let $B = \mathbf{a}_1$ be an $1 \times n$ matrix.
3: **for** $i = 2$ to m **do**
4: Call Alg. 8 on $(B, \mathbf{a}_i)$: the output is the new B.
5: **end for**
6: Return B