Factoring Univariates over $\mathbb{Q}$

- Suppose $f \in \mathbb{Q}[x]$ is a polynomial to be factored. By multiplying with a positive integer we could clear away the denominators.

$\Rightarrow$ So, wlog $f \in \mathbb{Z}[x]$. Let n be its degree & the coefficients a_i be of ℓ-bits.

Qn: How do we factor, or test irreducibility of, the integral polynomial f ?

$\to$ Over $\mathbb{F}_p$, we had used $(x^p - x)$. What do we do now?

- Starting Idea: Factor f mod prime p; do Hensel lifting to get to mod p^k; solve a linear system; take gcd to factor f!
- Let us first see the algorithm & then a new analysis. It was discovered by (Lenstra, Lenstra, Lovász) in 1982, igniting a new field.

<u>Input</u>: $f = \sum_{0\le i\le n} a_i x^i \in \mathbb{Z}[x]$, $|a_i| < 2^{\ell-1}$ $(0 \le i \le n)$.

<u>Output</u>: Nontrivial integral factor (if one exists).

L^3-algorithm: 1) Preprocess: Assure that f is square-free. Find the smallest prime p s.t.

$p \nmid a_n$ & $f \bmod p$ is square-free.

2) Factor mod p: Using Berlekamp's algorithm compute a factorization $f \equiv \underline{g_0} \cdot h_0 \pmod p$, where $g_0 \bmod p$ is monic, irreducible & coprime to h_0.

3) Hensel lift: Compute $f \equiv \underline{g_k} \cdot h_k \bmod p^{2^k}$, for $k = \lceil \lg 2n^3 \ell \rceil$.

4) Linear system: Find $(\tilde{g}, \ell_k)$ s.t. $\tilde{g} \equiv g_k \cdot \ell_k \bmod p^{2^k}$, with $\deg \tilde{g} < n$; coefficients of $\tilde{g}$ have bit-size $\leq n \cdot (\ell + \lg n)$.

5) OUTPUT $\gcd(f, \tilde{g})$.

Analyzing the steps of L^3

Step 1: $\triangleright$ f is square-full $\Rightarrow \overset{gcd}{}(f, f')$ factors f.

$\triangleright$ $f \bmod p$ is " " $\Rightarrow (f \bmod p, f' \bmod p) \neq 1$

$\Rightarrow r := \mathrm{res}_x(f, f') \equiv 0 \bmod p$

• Ensure that $p \nmid a_n \cdot r$ (Note: $a_n \cdot r \neq 0$).

$\triangleright |a_n \cdot r| < 2^{\ell} \cdot (2n+1)! \cdot (2^{\ell})^{n+1} (n 2^{\ell})^n$

$\Rightarrow$ # primes dividing $a_n \cdot r$ are at most

$\leq 2\ell(n+1) + 3n \lg n < 3n \cdot (\ell + \lg n)$.

$\Rightarrow p = \tilde{O}(\ell n)$ exists! $\square$

<u>Step 2:</u> Since $p = \tilde{O}(\ell n)$, factoring ($f \bmod p$) to find g_0, is doable in $\mathrm{poly}(p,n) = \mathrm{poly}(n\ell)$ time. $\square$

<u>Step 3:</u> By Hensel lifting, in $\mathrm{poly}(n\ell)$-time. $\square$

<u>Step 4:</u> This requires a "small" root of a linear system. Let us first estimate the bit-size of the factors of f:

<u>Lemma 1</u>: (Mignotte's bound) Any root $\alpha \in \mathbb{C}$ of a polynomial $f(x) = \sum_{i=0}^{n} a_i \cdot x^i \in \mathbb{Z}[x]$ satisfies $|\alpha| \leq n \cdot \max_i |a_i|$.

<u>Proof</u>:

- If $|\alpha| < 1$ then the claim holds.
- Else $0 = f(\alpha) = \sum_{i=0}^{n} a_i \cdot \alpha^i \geq |a_n \alpha^n| - \sum_{i=0}^{n-1} |a_i \cdot \alpha^i|$

$\geq |\alpha|^n - n \cdot \max_i |a_i| \cdot |\alpha|^{n-1}$

$\Rightarrow \quad |\alpha| \leq n \cdot \max_i |a_i|$. $\square$

Lemma 2: Any factor g of f has coefficients of magnitude at most $2^{(\ell+\lg n-1)n}$.

Proof:

- Let $g(x) = \prod_{i=1}^{m}(x-\alpha_i)$, $\alpha_i \in \mathbb{C}$

- $\mathrm{Coeff}(x^{m-j})(g) = \sum_{S \in \binom{[m]}{j}} \prod_{i \in S} (-\alpha_i)$

with magnitude $\leq \sum_{S} \prod_{i \in S} |\alpha_i|$

[Lemma 1] $< \binom{m}{j} \cdot (n2^{\ell-1})^j < (1+n2^{\ell-1})^m < 2^{(\ell+\lg n-1)n}$. $\square$

Step 5: • If $\tilde{g}$ exists in Step 4, and $(f,\tilde{g})=1$,
$\exists\, u, v \in \mathbb{Z}[x]$: $u \cdot f + v \cdot \tilde{g} = \mathrm{res}(f,\tilde{g}) \neq 0$

$\Longrightarrow$ $u \cdot g_k \cdot h_k + v \cdot g_k \cdot l_k \equiv \mathrm{res}(f,\tilde{g}) \mod p^{2^k}$.

$\Rightarrow$ $g_k \cdot (u h_k + v l_k) \equiv$ " " ---(i)

• Note: $|\mathrm{res}(f,\tilde{g})| < (2n+1)! \cdot (2^{l-1})^{n+1} \cdot (2^{(l+6n)n})^n$
$\ll 2^{2n^3 l} < p^{2^k}$.

$\Rightarrow$ RHS in eqn.(i) is a nonzero constant,
while LHS " " " " multiple of $g_k(x)$. $\lightning$

$\Longrightarrow$ The contradiction implies that Step-5 factors f, if $\tilde{g}$ exists. $\square$

How do we compute $\tilde{g}$ (with "small" coeffs)?

- Let g_k be of $\deg = n' < n$. Unknown polynomials are:

$$\tilde{g} =: \sum_{i=0}^{n-1} c_i \cdot x^i \quad \& \quad \ell_k =: \sum_{i=0}^{n-1-n'} \alpha_i \cdot x^i \text{ s.t.}$$

$$\tilde{g} \equiv g_k \cdot \ell_k \mod p^{2^k}.$$

$$\Rightarrow \sum_{i=0}^{n-1} c_i \cdot x^i = \sum_{i=0}^{n-1-n'} \alpha_i \cdot (x^i g_k) + \sum_{i=0}^{n-1} \beta_i \cdot (p^{2^k} x^i) \quad \text{(ii)}$$

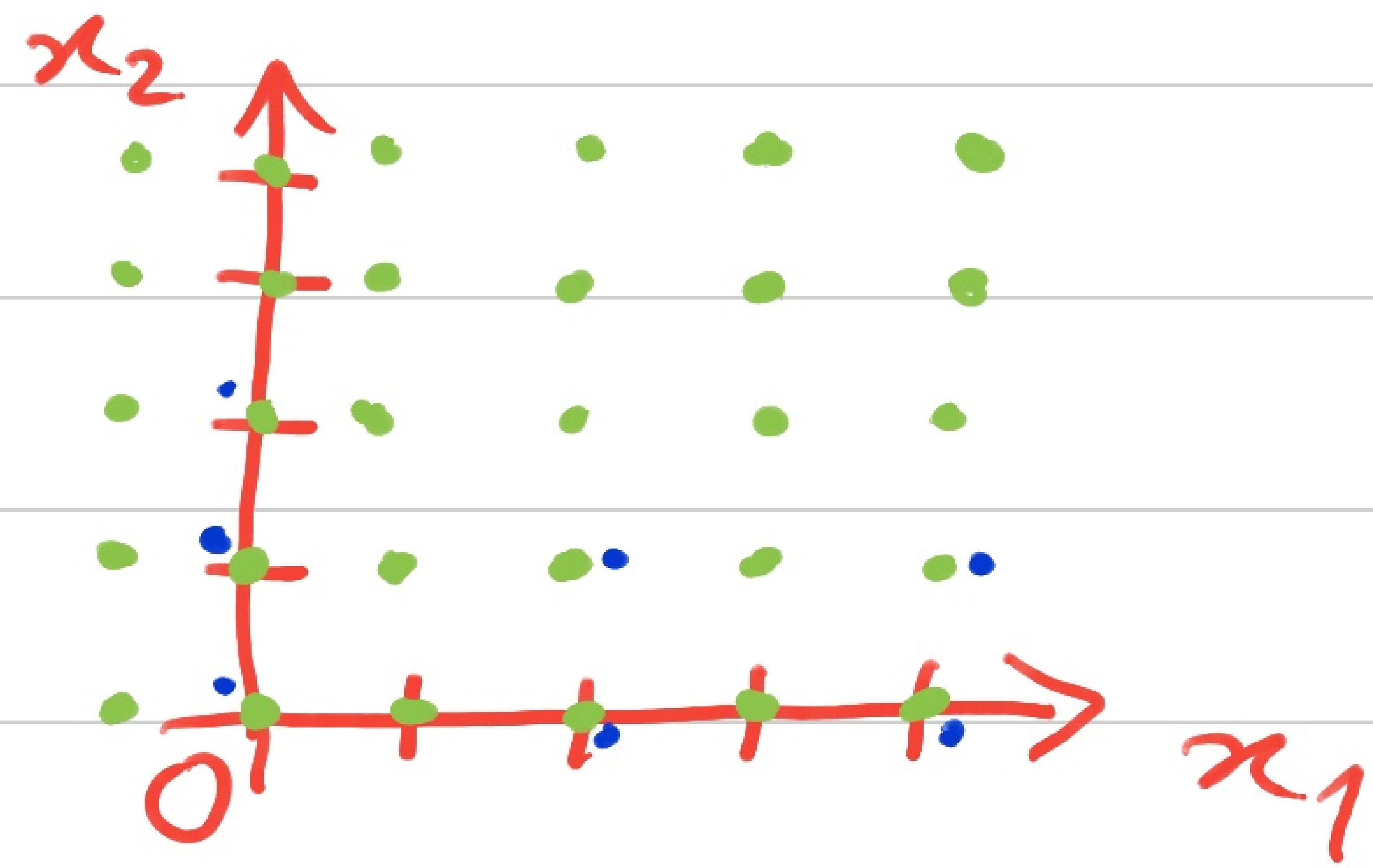
$\triangleright$ Find integral $\bar{c}, \bar{\alpha}, \bar{\beta}$'s in eqn. (ii) s.t. $\|\bar{c}\| = \sqrt{\sum_i c_i^2}$ is "small" $< 2^{(\ell+\lg n)\cdot n}$.

- So the fundamental problem to solve is:

Given $b_1, \dots, b_m \in \mathbb{Z}^n$, find $r_1, \dots, r_m \in \mathbb{Z}$ s.t. $\| \sum_{i=1}^{n} r_i b_i \|$ is "small".

<u>Defn</u>: The $\mathbb{Z}$-linear-combinations of $\{b_i \mid i \in [m]\}$ define a lattice $\mathcal{L}(b_1, \dots, b_m) := \{\sum_{i=1}^{m} r_i b_i \mid r_i \in \mathbb{Z}\}$.

- eg. $\mathcal{L}\left(\begin{pmatrix}1\\0\end{pmatrix}, \begin{pmatrix}0\\1\end{pmatrix}\right)$ is:



▷ $\mathcal{L}\left(\begin{pmatrix}1\\0\end{pmatrix}, \begin{pmatrix}1\\1\end{pmatrix}\right) = \mathcal{L}\left(\begin{pmatrix}1\\0\end{pmatrix}, \begin{pmatrix}0\\1\end{pmatrix}\right)$.

$\supsetneq \mathcal{L}\left(\begin{pmatrix}2\\0\end{pmatrix}, \begin{pmatrix}0\\1\end{pmatrix}\right) \not\ni \begin{pmatrix}1\\0\end{pmatrix}$.

- Shortest vector problem (SVP) : Find a vector $\bar{v} \in \mathcal{L}(b_1, \ldots, b_m)$ s.t. $\|\bar{v}\| = \min_{0 \neq \bar{u} \in \mathcal{L}} \|\bar{u}\|$.

▷ [Ajtai'98] SVP is NP-hard.
[Micciancio'98] constant-approx. of SVP is NP-hard.

- But, we need merely a 2^n-approximation for Step 4 !

⟶ We'll develop this approximation algorithm. (L^3 = Lenstra-Lenstra-Lovász)

- First, we do a useful preprocessing:

<u>Lemma 1</u>: For SVP, wlog we assume that $B := \{b_1, \ldots, b_m\} \in \mathbb{Z}^n$ are $\mathbb{R}$-linearly independent.

<u>Proof</u>: • Consider their coordinates & the matrix

$$B := \begin{pmatrix} b_{11} & b_{21} & & b_{m1} \\ b_{12} & b_{22} & \cdots & b_{m2} \\ \vdots & \vdots & & \vdots \\ b_{1n} & b_{2n} & & b_{mn} \end{pmatrix} \in \mathbb{Z}^{n\times m}.$$

• Let $\sum_{i=1}^{m} a_i \cdot b_{i1} = g := \gcd(\text{1st row})$.

• Apply the extended-Euclid-gcd transformations on the columns of B.

- Say the new columns are $b_1', \ldots, b_m'$.
- Ensure: $\begin{cases} (1,1)\text{-th entry becomes } g. \\ \text{Remaining entries in 1st-row become } \underline{\text{zero}}! \end{cases}$

$$\Rightarrow B \mapsto B' := \begin{pmatrix} g & 0 & \cdots & 0 \\ * & & & \\ \vdots & & & \\ * & & & \end{pmatrix}_{n\times m}$$

- The transformation is $B' = B \cdot U$, where U follows each step of Euclid-gcd-algorithm.

$\triangleright$ U is <u>unimodular</u>, i.e. $|U| = \pm 1 \Rightarrow U^{-1}$ integral.

$$\Rightarrow \mathcal{L}(B') = \mathcal{L}(B).$$

<u>Pf</u>: $\begin{pmatrix} a \\ b \end{pmatrix} \to \begin{pmatrix} a \\ b-qa \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -q & 1 \end{pmatrix} \cdot \begin{pmatrix} a \\ b \end{pmatrix}$; $\left| \begin{pmatrix} 1 & 0 \\ -q & 1 \end{pmatrix} \right| = 1$. $\square$

- Repeatedly apply this Gauss-Euclid trick, to get a matrix

$$\tilde{B} = \left(\begin{array}{c|c} A_{m' \times m'} & \\ \hline C_{(n-m') \times m'} & 0 \end{array} \right)_{n \times m}$$

where A is lower-triangular & invertible.

▷ $\mathcal{L}(\tilde{B}) = \mathcal{L}(B)$.

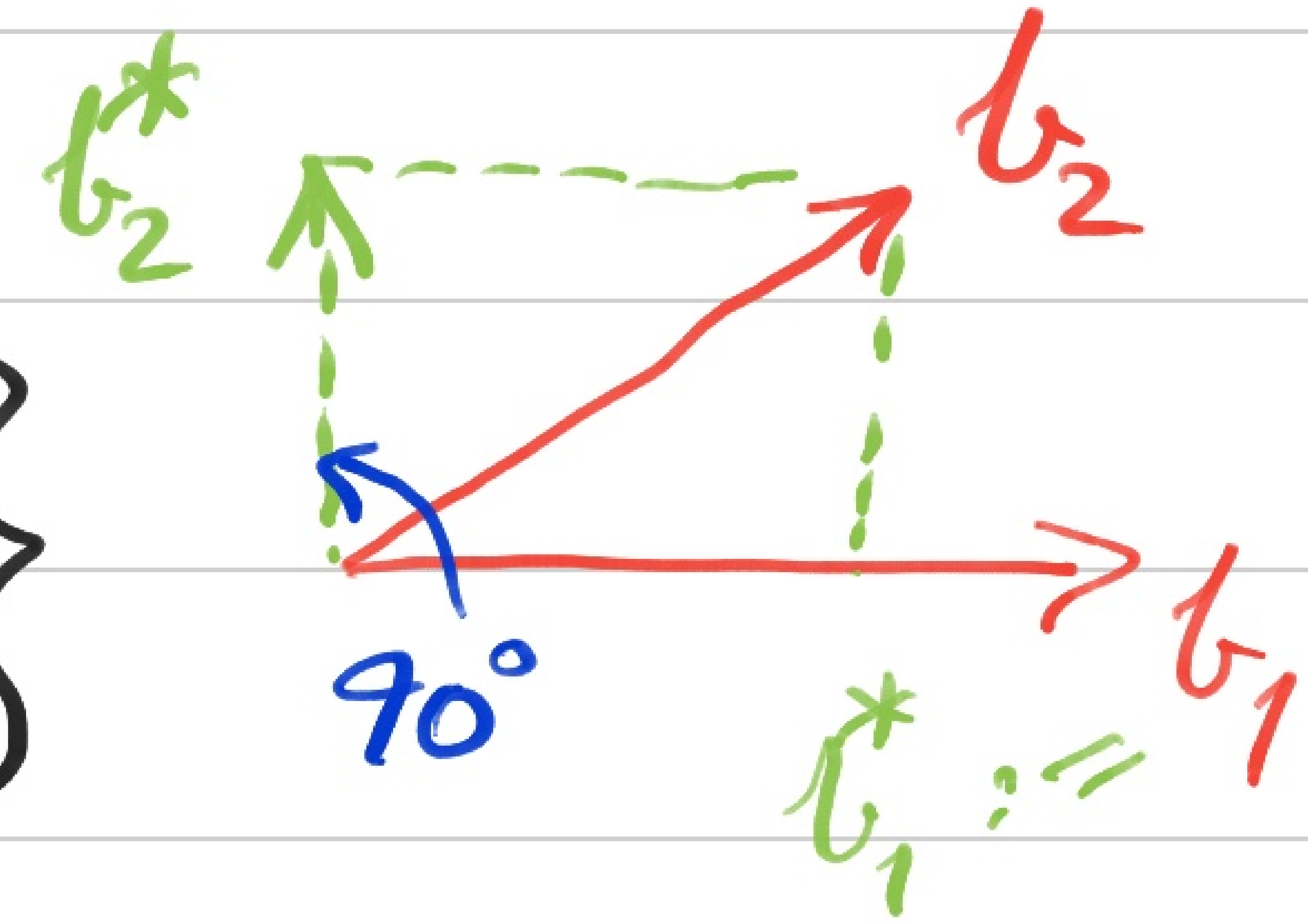
▷ First m' columns of $\tilde{B}$ form an $\mathbb{R}$-basis of size $m' \leq \min(n, m)$.

$\square$

– So, we work with $\mathbb{R}$-l.i. $b_1, \ldots, b_m \in \mathbb{Z}^n$.

- In the vector space $V(B) := \langle b_1, \dots, b_m \rangle_{\mathbb{R}}$ there is an orthogonal basis.
- Idea: • Orthogonalize $\{b_1, b_2\}$ to

$$\left\{ b_1^* := b_1,\ b_2^* := b_2 - \left\langle b_2, \frac{b_1}{\|b_1\|} \right\rangle \cdot \frac{b_1}{\|b_1\|} \right\}$$



$$\Rightarrow \langle b_1^*, b_2^* \rangle = 0$$

▷ The shorter of $\{b_1^*, b_2^*\}$ is the shortest vector in $\mathcal{L}(b_1^*, b_2^*)$.

Pf: $\|\alpha_1 b_1^* + \alpha_2 b_2^*\|^2 = \|\alpha_1 b_1^*\|^2 + \|\alpha_2 b_2^*\|^2$.

$$\geq \min(\|b_1^*\|^2, \|b_2^*\|^2). \quad \square$$

→ $\mathcal{L}(B)$ may not have an orthogonal basis!

Gram-Schmidt Orthogonalization (GSO):

1) Let $b_1^* := b_1$.

2) For $2 \le i \le m$, do

$$b_i^* := b_i - \sum_{j=1}^{i-1} \overbrace{\left(\frac{\langle b_i, b_j^* \rangle}{\| b_j^* \|^2} \right)}^{\mu_{ij}} \cdot b_j^* .$$

$\triangleright$ GSO gives an orthogonal basis.

Lemma: For $0 \neq b \in \mathcal{L}(b_1, \ldots, b_m)$, $\|b\| \geq \min_i \|b_i^*\|$.

Pf: • Let $b =: \sum_{i=1}^{m} \lambda_i b_i$ for λ's $\in \mathbb{Z}$ & $\lambda_m \neq 0$.

$$\Rightarrow b = \lambda_1 b_1^* + \lambda_2 \cdot (b_2^* + \mu_{21} b_1^*) + \cdots + \lambda_m \cdot (b_m^* + \mu_{m,m-1} b_{m-1}^* + \cdots + \mu_{m,1} b_1^*)$$

$$\Rightarrow \|b\|^2 = (\ldots)^2 \cdot \|b_1^*\|^2 + \cdots \quad \lambda_m^2 \cdot \|b_m^*\|^2 \geq \lambda_m^2 \cdot \|b_m^*\|^2$$

$$\Rightarrow \quad \|b\| \geq |\lambda_m| \cdot \|b_m^*\| \geq \|b_m^*\| \geq \min_i \|b_i^*\| .$$

$\square$

- So, L^3-algo. tries to make the <u>angles</u>, in a basis of $\mathcal{L}(B)$, close to $60°$. ← pseudo-orthogonal.

→ In that basis it'll pick the <u>first</u>!

- <u>Defn</u>: • L^3 finds a <u>reduced basis</u> of $\mathcal{L}(B)$. These are lattice elements $\{c_1, \ldots, c_m\} \subset \mathcal{L}(B)$ s.t.

c_{i+1} not much smaller than c_i

(i) $\forall i, \ \|c_i^*\|^2 \leq \frac{4}{3} \cdot \|c_{i+1}^* + \mu_{i+1,i}\, c_i^*\|^2$

angles $\approx 60°$

(ii) $\forall i \geq j, \ |\mu_{ij}| \leq 1/2$, where $\mu_{ij} := \frac{\langle c_i, c_j^* \rangle}{\|c_j^*\|^2}$.

$\cdot$ (i), (ii) $\Rightarrow \|c_i^*\|^2 \leq \frac{4}{3}\|c_{i+1}^*\|^2 + \frac{1}{3}\|c_i^*\|^2$

$$\Rightarrow \|c_i^*\| \leq \sqrt{2}\cdot \|c_{i+1}^*\|$$

(i)

$$\Rightarrow \|c_1^*\| \leq \min_i \left\{ \sqrt{2}^{i-1} \cdot \|c_i^*\| \right\} \leq \sqrt{2}^{m-1} \cdot \|c_i^*\| \quad (\forall i)$$

shortest-length in lattice $:= \lambda(\mathcal{L}(c_1,\dots,c_m)) \geq \|c_1^*\| \quad (\because c_1^* = c_1 \in \mathcal{L}(\cdot))$

& $\|c_1^*\| \leq 2^{\frac{m-1}{2}} \cdot \lambda(\mathcal{L}(c_1,\dots,c_m))$ [by (i)]

$\triangleright$ c_1 estimates $\lambda(\mathcal{L})$ by a factor of $2^{(m-1)/2}$.

L^3-reduced basis algorithm

1) Compute GSO of $B = \{b_1, \ldots, b_m\}$. gives μ_{ij} & b_i^*
2) For $i = 2$ to m

   For $j = i-1$ to 1

   $b_i \leftarrow b_i - \lfloor \mu_{ij} \rceil \cdot b_j$ round-off to nearest integer
3) If $\exists i$, $\|b_i^*\|^2 > \frac{4}{3} \cdot \|b_{i+1}^* + \mu_{i+1,i}\, b_i^*\|^2$

   then swap $\{b_i, b_{i+1}\}$ & GOTO (1).
4) Output $\{b_1, \ldots, b_m\}$.

Analysis

Step 2: The new $b_2 \leftarrow b_2 - \left\lfloor \frac{\langle b_2, b_1 \rangle}{\|b_1\|^2} \right\rceil \cdot b_1$

▷ So, $\frac{\langle b_2, b_1 \rangle}{\|b_1\|^2} \leftarrow \frac{\langle b_2, b_1 \rangle}{\|b_1\|^2} - \left\lfloor \frac{\langle b_2, b_1 \rangle}{\|b_1\|^2} \right\rceil \cdot \frac{\langle b_1, b_1 \rangle}{\|b_1\|^2} =: \underline{\mu_{2,1}}$

$\Rightarrow |\mu_{2,1}| \leq 1/2$.

▷ The same holds for $|\mu_{i,i-1}|$, $i \in [m]$.

▷ Also, the transformation is unimodular & lattice remains unchanged.

Step 3: • To show that it's repeated only few times, we need a potential function:

$$D(b_1, \dots, b_m) := \prod_{i \in [m]} \|b_i^*\|^{2(m-i)}$$

• Step 2 has no effect on this. [$\because$ b_i^*'s do not change]
While each repetition of Step 3 - swap reduces D by a factor of $\frac{\|b_{i+1}^*\|^2}{\|b_i^*\|^2} < \frac{3}{4} - \mu_{i+1,i}^2 < \frac{3}{4}$.

Lemma 3: $|D(b_1, \dots, b_m)|$ is a positive integer of value less than $2^{\tilde{O}(n^5 \ell)}$.

Proof: • Write D as $\prod_{j \in [m-1]} D_j$, were $D_j = \prod_{i \in [j]} \|b_i^*\|^2$

• D_j relates to vol$(b_1, \ldots, b_j)$:

• D_j is the determinant of $(b_1^*, \ldots, b_j^*)^T \cdot (b_1^*, \ldots, b_j^*)$, which is diagonal & equals $((b_1, \ldots, b_j) \cdot C)^T \cdot ((b_1, \ldots, b_j) C)$, for a unimodular transformation C.

$\Longrightarrow \quad D_j = |(b_1, \ldots, b_j)^T \cdot (b_1, \ldots, b_j)| \in \mathbb{Z}_{>0}$.

$\Rightarrow |D_j| < \left(2^{\tilde{O}(n^3 \ell)}\right)^j \Rightarrow |D| < 2^{\tilde{O}(n^3 \ell) \cdot n^2}$. $\square$

▷ Thus, Step-3 repeats at most $\tilde{O}(n^5\ell)$-times in L^3-algorithm, and gives a reduced-basis.
[$\Rightarrow b_1$ is an $2^{n/2}$-approx. of $\lambda(\mathcal{L}(B))$.]

<u>Time:</u>

▷ A crude time estimate of the polynomial factoring algorithm is $(n^5\ell)\cdot n^3\cdot \tilde{O}(n^3\ell) = \tilde{O}(n^{11}\cdot\ell^2)$.

\# Step-3   Step-2   \# bitsize of integers

▷ Assume $L :=$ max bitsize in b_i's, L^3-algo. approx. SVP in time $\le \tilde{O}(L\cdot m\cdot m^2)^2 = \tilde{O}(L^2\cdot m^6)$.

preprocessing   growth on repetition of Step-3

- L^3-algo., & reduced basis, is used in many places.

  Egs. computational algebraic number theory, faster arithmetic in number fields, Knapsack problem, ...

- The main properties exploited are:

<u>Theorem:</u> Let $b_1, \ldots, b_n$ be a reduced basis of the lattice $L \triangleleft \mathbb{Z}^n$ & $b_1^*, \ldots, b_n^*$ be its GSO-basis. Then,

(i) $\|b_j\| \leq 2^{\frac{i-1}{2}} \cdot \|b_i^*\|$, for $1 \leq j \leq i \leq n$.

(ii) $d(L) \leqslant \prod_{i \in [n]} \|b_i\| \leqslant 2^{n(n-1)/4} \cdot d(L)$.

vol. or det. of lattice

(iii) $\|b_1\| \leqslant 2^{\frac{n-1}{4}} \cdot d(L)^{1/n}$.

[<u>Note</u>: shortest-vector in L has length $\approx \sqrt{n} \cdot d(L)^{1/n}$.]

Pf. sketch: (i) Use condition-(i) of reduced-basis definition.

(ii) Use (i) & $d(L) = \det(b_1^*, \ldots, b_n^*) = \prod_{i \in [n]} \|b_i^*\|$.

vol(L) ⇐

(iii) By (i): $\|b_1\| \leq 2^{\frac{i-1}{2}} \cdot \|b_i^*\|$, for $2 \leq i \leq n$.

& $\prod_i \|b_i^*\| = d(L)$.

$\square$

Application: Given rationals $\alpha_1, \ldots, \alpha_n, \varepsilon$. Find integers $p_1, \ldots, p_n, q$ s.t. $\forall i$, $\left|\frac{p_i}{q} - \alpha_i\right| \leq \varepsilon$ & minimize q.

[Simultaneous Diophantine Approximation]

Exercise: Solve it by L^3-reduced-basis algo.

Public-Key Cryptosystem (Lattice-based)

- Example is secure communication between Bank & Client.
- RSA is used commonly.
  - ↳ it is insecure in quantum computers.
- Lattice-based methods are "quantum secure".
- NTRU cryptosystem was proposed by (Hoffstein, Pipher & Silverman) in Crypto'96.
- NTRU = N-th degree Truncated Polynomial Ring

$$R := \mathbb{Z}[X]/\langle X^N - 1\rangle$$

▷ R is a lattice.

- NTRU's security relies on the fact that L^3-algo. cannot approximate SVP fast enough, for <u>large</u> $2N$. (eg. $N = 251$)

↳ It allow smaller key-sizes than RSA!

<u>Public parameters</u>: • N & prime-powers p, q.
(eg. $N = 251$, $p = 3$, $q = 2^7 = 128$)

• <u>Support-bounds</u> d_f, d_g, d_m, d_r (eg. ≈ 80).

<u>Private-Key of Bank</u>: Pick random polynomials in R

$$\left\{\begin{array}{l} f, \text{ with } d_f \text{ 1's \& } (d_f - 1) \text{ (-1)'s.} \quad \triangleright f(1) = 1. \\ g, \quad " \quad d_g \quad " \;\&\; d_g \quad " \qquad . \quad \triangleright g(1) = 0. \end{array}\right.$$

} Ternary polynomials

- **Public-Key (of Bank):** Compute $h := g/f \mod q$. Publish $h(x)$.

reduce numbers to $[-\frac{q}{2}, \frac{q}{2}]$

Encryption (on Client side): • Let the message be $m \in R$, which is ternary with d_m 1's & d_m (-1)'s.

• Pick random $r \in R$ with d_r " " d_r ".

• Compute ciphertext $e := m + p \cdot r \cdot h \pmod q$.

• Send it to Bank (over an insecure channel!)

Decryption (by Bank): (i) $f \cdot e \pmod q$ $[= \underbrace{f.m}_{} + \underbrace{p.r.g}_{} \bmod q]$

[have coeffs. $< q/2$ in magnitude]

(ii) Go mod p to get $f \cdot m$.

(iii) $f^{-1} \cdot (f.m) \equiv m \pmod p$. $\Rightarrow$ Get m $[\because p \geq 3]$

- Security: • Adversary can try to find (f, g) from (h, q) using the eqn: $\underline{f} \cdot h = \underline{g} + q \cdot \underline{k}$

• Consider matrix $M :=$

$$\left(\begin{array}{c|c} I_N & \begin{matrix} x^0 \cdot h \\ x^1 \cdot h \\ \vdots \\ x^{N-1} \cdot h \end{matrix} \\ \hline O_N & q \cdot I_N \end{array}\right)$$

$\leftarrow$ cyclic permuted rows

$2N \times 2N$

• Note: Using coefficient-vectors $\bar{f}, \bar{k}$ (of f, k resp.) we have: $(\bar{f}, -\bar{k}) \cdot M = (\bar{f}, \bar{g})$

$\leftarrow$ coeff-vec. of $g(x)$

$\Rightarrow$

▷ $\mathcal{L}$(rows of M) contains the short-vector $(\bar{f}, \bar{g})$ of length $= \sqrt{2d_f - 1 + 2d_g} \ll \sqrt{N}$.

- But, using L^3-algo. to find it takes time >
$(2N)^6 \cdot (\lg q)^2 \approx 500^6 \cdot 7^2 \approx 10^{18}$.

$\to$ NTRU is "secure" for the practical settings:
- $N/3 \leq q \leq 2N/3$; $N \geq 251$.
- $d_f, d_g, d_r, d_m \geq 80$.

▷ Guessing f directly takes $\approx 2^{2 \times 80}$ steps!