I) Introduction:

- Lattice, SVP, CVP \(\Rightarrow\) norm = \(\|\cdot\|_2\) and sometimes \(\|\cdot\|_\infty\)

Schrijver 87, Schnorr-Euchner 84

- Lattice reduction: BKZ, LLL Approx

(- compute a short basis \(\Rightarrow\) helps with SVP and CVP)

* Module Lattices:
  - Lattice: storage \(n^2\) (cf. codes)

  L-structured lattices

  \(\mathbf{a}\) (dense) \(\rightarrow\)

  \[
  \mathbf{H}_a = \begin{bmatrix}
  a_0 & a_1 & \cdots & a_{n-1} \\
  \vdots & \ddots & \ddots & \vdots \\
  a_{n-1} & a_0 & \cdots & a_{n-2}
  \end{bmatrix}
  \]

  Ls mult by \(\alpha x + \alpha x \cdots + \alpha x^{n-1} \mod x^n 1\)

  \(\mathbf{R} = \mathbb{Z}[x]/x^n 1\)

  \(\sigma: \mathbf{R} \rightarrow \mathbb{R}^n\) (or even \(\mathbb{Z}^n\))

  \(\alpha \rightarrow (\alpha_0, \ldots, \alpha_{n-1})\)

  \[
  \mathbf{M}_a = \begin{bmatrix}
  \sigma(\alpha) \sigma(\alpha) - \sigma(\alpha)^2 \\
  \vdots & \ddots & \ddots & \vdots \\
  \sigma(\alpha)^{n-1} & \cdots & \sigma(\alpha) & \sigma(\alpha)^{n-1}
  \end{bmatrix}
  \]

  \(L(H_a) = \sigma(\langle \alpha \rangle)\)

  Ls the lattice is an ideal in \(\mathbf{R}\)

  (via \(\sigma\))

  called "ideal lattice"
In more generality: ideal lattices $L = \sigma(I)$

* I ideal maybe not principal
  $L$ but in this talk always principal (for simplicity)

* $R$ can be another ring
  $
  \Rightarrow \text{for this work: } R = \text{ring of integers of } K
  
  \text{for } K = \mathbb{Q} \langle x \rangle / p(x)
  
  p \text{ irreducible monic degree } d

  \text{for simplicity: can think of } R = \mathbb{Z} \langle x \rangle / p(x)

  \text{Eg: } R = \mathbb{Z} \langle x \rangle / x^{d+1} \quad d = 2^k \quad \text{(power of 2 cycle)}

  R = \mathbb{Z} \langle x \rangle / x^d - x - 1 \quad d \text{ prime (NTRU prime)}

\underline{Module lattice}: fixing some $K$ and $R = \mathbb{Z} \langle x \rangle / p$

\[ B = \begin{array}{cccc}
H_{0,0} & H_{0,1} & \ldots & H_{0,d} \\
H_{1,0} & H_{1,1} & \ldots & H_{1,d} \\
\vdots & \vdots & \ddots & \vdots \\
H_{d,0} & H_{d,1} & \ldots & H_{d,d}
\end{array} \]

\[ \text{recall } M_{\alpha} = \{ (\xi) \} \quad \text{for } \alpha \in \mathbb{Z}_{\geq 0} \]

$L(B)$ is a (free) module lattice

Why the name? $b_i = (a_i, x_i, \ldots, x_i) \in R^k \rightarrow K$-linearly independent

$M = \{ \sum x_i b_i: x_i \in R \}$

is an $R$-module in $K^r$

and $L(B) = \bigoplus_{\alpha \geq 0} M_{\alpha}$

$(\sigma(b_i))$ is the concatenated vector

$(\sigma(a_i), x_i, \ldots, x_i)$
More generally: A module lattice is \( \Sigma(M) \) for \( M \in K^r \) an \( R \)-module.

\( L \)\( \Rightarrow \) pseudo-basis instead of basis.

For us today: a module lattice is \( \Sigma(M) \)

where \( M = \left\{ \sum_{i=1}^{r} x_i b_i ; x_i \in E \right\} \)

with \( b_i \in K^r \) linearly independent.

* \( (b_1, \ldots, b_r) \) basis of \( M \)

* \( r \) rank of \( M \)

\( L \)\( \Rightarrow \) basis of the module:

\[
\begin{pmatrix}
\sigma(a_{11}) & \sigma(a_{12}) & \cdots & \sigma(a_{1r}) \\
\sigma(a_{21}) & \sigma(a_{22}) & \cdots & \sigma(a_{2r}) \\
\vdots & \vdots & \ddots & \vdots \\
\sigma(a_{r1}) & \sigma(a_{r2}) & \cdots & \sigma(a_{rr})
\end{pmatrix}
\]

Embeddings \( \text{[Here!] (next page)} \)

Why do we care about module lattices?

RLWE, RSIS, Module-LWE, Module-SIS are equivalent to SIVP in module lattices.

(Which is no harder than SVP)

(And \( \ast \) NTRU is no harder than SVP in module lattices)

NIST: 11 of the 12 lattice submissions use NTRU, RLWE, RSIS, ...,

\( L \)\( \Rightarrow \) the modules involved have rank \( \leq 3/4 \), always \( \leq 10 \)

\( \Rightarrow \) solving SVP in modules of small rank (\( \leq 10 \)) has an impact on these constructions.
Embeddings: \[ \sigma : K \to \mathbb{R}^d \]

Coefficient embedding:
\[ a_0 + a_1 x + \cdots + a_d x^d \mapsto (a_0, \ldots, a_d) \]

Canonical embedding:
\[ \sigma : K \cong \mathbb{C}^d \]
\[ a(x) \mapsto (a(a_1), \ldots, a(a_d)) \]
with \( a_1, \ldots, a_d \) the complex roots of \( P \) \( (K = \mathbb{Q}[x]/P) \)

A module lattice is \( \sigma(M) \) with the same basis as before, but we can change the embedding \( \sigma \).

* Constructions usually use \( \sigma_{\text{coeff}} \)
  (easier to handle: elements in \( \mathbb{Q} \) or even in \( \mathbb{Z} \))

* Cryptanalysis usually use \( \sigma_{\text{Canonical}} \)
  (nice algebraic properties: eg. multi is coordinate-wise)

\( \sigma_{\text{coeff}}(M) \) is usually not the same lattice as \( \sigma_{\text{Canonical}} \)

But: * for power-of-2 cyclotomics -> same lattice (up to rotation and scaling)

* NTRU Prime and other "nice" fields (that we want to use) \( \to \) similar geometry
  (not exactly the same, but \( \delta \)).

From now on: \( \sigma = \sigma_{\text{Canonical}} \)
A module is a lattice over $\mathbb{R}$

module: $M = \{ \sum x_i b_i^k, x_i \in \mathbb{R} \}$
$\{b_i^k\}$ linearly indep

$= \begin{vmatrix} b_1^k \\ \vdots \\ b_r^k \end{vmatrix} \times \mathbb{R}^r$

lattice: $L = \{ \sum x_i b_i^k, x_i \in \mathbb{Z} \}$
$\{b_i^k\} \subseteq \mathbb{R}^r$
linearity indep

$= \begin{vmatrix} b_1^k \\ \vdots \\ b_r^k \end{vmatrix} \mathbb{Z}^r$

A module is both a "lattice" of rank $\frac{r}{2}$ over $\mathbb{R}$

$\begin{bmatrix} b_1^k & \vdots & b_r^k \end{bmatrix} \mathbb{R}^r$

a lattice of rank $\frac{r}{2}$ over $\mathbb{Z}$

in practice (NIST): $r = 3, 4$ ($\leq 10$)

d $= 512, 1024, 256$
s.t. $s.d. \leq 1000$

Remember lattice reduction

LHK works for lattices over $\mathbb{Z}$. What about lattices over $\mathbb{R}$?
**Objective**: Adapt LLL to lattices over $\mathbb{R}$

Why LLL not BKZ?  
- easier $\rightarrow$ this is the true reason  
- sufficient for modules of rank 3, 4...

**History**: Npras '96: specific number fields  
No bound

<table>
<thead>
<tr>
<th></th>
<th>bound on quality</th>
<th>bound on run time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Npras '96</td>
<td>$\times$</td>
<td>$\times$</td>
</tr>
<tr>
<td>specific number fields</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fieker-Behn '96</td>
<td>$\times$</td>
<td>$\checkmark$</td>
</tr>
<tr>
<td>All number fields</td>
<td>$\times$</td>
<td>$\checkmark$</td>
</tr>
<tr>
<td>totally real fields</td>
<td>$\checkmark$</td>
<td>$\checkmark$</td>
</tr>
<tr>
<td>Kim - Lee '17</td>
<td>$\times$</td>
<td>$\checkmark$</td>
</tr>
<tr>
<td>norm Euclidean fields</td>
<td>$\checkmark$</td>
<td>$\checkmark$</td>
</tr>
<tr>
<td>biquadratic fields</td>
<td>$\checkmark$</td>
<td>$\checkmark$</td>
</tr>
<tr>
<td>This work</td>
<td>$\checkmark$</td>
<td>$\approx$ $\checkmark$</td>
</tr>
<tr>
<td>any field</td>
<td>$\checkmark$</td>
<td>$\checkmark$</td>
</tr>
</tbody>
</table>

$\approx$ (if we have an oracle solving CVP in a fixed lattice)
4) $\textbf{LLL in dimension } \mathbb{Z} = \text{Lagrange-Gauss algorithm}$

Over $\mathbb{Z}$: animation on slides

1) What is $\| \cdot \|$ over $\mathbb{R}$?

- $\alpha \in \mathbb{R}$ def: $\| \alpha \|_2 = \| \sigma(\alpha) \|_2$
- $\mathbf{b} \in \mathbb{R}^r$
  \[ \| \mathbf{b} \|_2 = \left( \sum \| \sigma(b_i) \|_2 \right)^{1/2} \text{ (Pythagorean)} \]

This is what we want: small in $\mathbb{R}^r (=)$ small in $\mathbb{Z}^d$

Two notions of number theory:

- $\text{Tr}(\alpha) = \sum \sigma(\alpha)_i$
- $\mathbb{N}(\alpha) = \prod \sigma(\alpha)_i$

*Assume $\mathbb{Z}$ is well-defined (e.g. $\mathbb{R} = \mathbb{E}^{\mathbb{R}^d}$)

Define $\overline{\alpha} = (\overline{\alpha}_1, \ldots, \overline{\alpha}_d)$ and assume $\overline{\alpha} \in \mathbb{R}$ if $\alpha \in \mathbb{R}$

(for example $\mathbb{R} = \mathbb{E}(\mathbb{Z}[x]/x^n+1)$

Then

- $\| \alpha \|_2 = \sqrt{\sum \| \sigma(\alpha)_i \|^2} = \sqrt{\text{Tr}(\overline{\alpha})}$

  and

- $\| \mathbf{b} \|_2 = \sqrt{\sum \text{Tr}(\mathbf{b}_i \overline{\mathbf{b}}_i)} = \sqrt{\text{Tr}(\mathbf{Z} \mathbf{b} \overline{\mathbf{b}})}$

$L)$ define $\alpha$ as "scalar product"

\[ \langle \mathbf{b}, \mathbf{c} \rangle = \sum \mathbf{b}_i \overline{\mathbf{c}}_i \in \mathbb{R}(nK) \]
2) QR factorisation over $\mathbb{R}^2$

For $x_3$

$b_1^* = b_1$

\[ b_2^* = b_2 - \frac{\langle b_2, b_1^* \rangle}{\langle b_1^*, b_1^* \rangle} b_1^* \in \mathbb{R}^2 \]

\[ L \Rightarrow \langle b_2^*, b_1^* \rangle = 0 \]

\[ Q = \begin{pmatrix} b_1^* \\ b_2^* \end{pmatrix} \quad R = \begin{pmatrix} \| b_1^* \| & \langle b_2, b_1^* \rangle \\ 0 & \| b_2^* \| \end{pmatrix} \]

\[ \| b_i^* \| = \sqrt{\langle b_i^*, b_i^* \rangle} \in \mathbb{R} \]

**Question:** $\sqrt{\_}$ in $\mathbb{R}$?  

- can avoid $\sqrt{\_}$ by using "Gram-Schmidt"

  - div of $\langle b_2, b_1^* \rangle$ by $\| b_1^* \|

  - div of $\langle b_2, b_1^* \rangle$ by $\| b_2^* \|^2$

  - or, we can define $K_{\mathbb{R}}$ and $\sqrt{\_}$ is well defined

**Properties:**

- $QR = \begin{pmatrix} b_1 & b_2 \end{pmatrix}$

- $R$ triangular

- $Q Q^T = I_2$

- $R(v)$ short $\leq$ $B(u)$ short

  (preserves geometry)

\[ L \Rightarrow \langle B(v), B(v) \rangle = \langle (u \cdot v) B(v), B(v) \rangle = (\langle v, v \rangle B(v))^T B(v) \]

\[ = (\langle v, v \rangle)^T \tilde{Q}^T \tilde{Q} R(v) = \langle v, v \rangle \tilde{I}_2 \]
and $\| \| \text{ } \text{ } \text{ because } \sqrt{\text{Tr}(A, B)}/2$

$\rightarrow$ QR factorisation is ok, we can assume that our basis is triangular

3) Euclidean division

$a, b \in \mathbb{R}$ (or $\mathbb{K}$)

we would like $r \in \mathbb{R}$ s.t. $\|a + rb\| \leq \frac{1}{2} \|b\|$

Pr: Most of the number fields we are interested in are not euclidean $\rightarrow$ no such $r$.

But: There should exist (counting argument) $a, v \in \mathbb{R}$ s.t.: $\|av + bv\| \leq \frac{1}{2} \|b\|$

* $\|w\| \leq \text{poly}(d)$ (here, $R$ is "nice")

$\Delta$ and in general

or even $\lambda_n(R)$

$L$: we can use it to reduce a small multiple of $\hat{b}_2$ by $\hat{b}_1$.

Re: over $\mathbb{Z}$: when we have $\begin{pmatrix} r_{11} & r_{12} \\ 0 & r_{22} \end{pmatrix}$

and $r_{11} + r_{22}$, we are roughly done because any vector is $\geq \min(r_{11}, r_{22}) \approx r_{11}$

$L$: so $\begin{pmatrix} r_{11} \\ 0 \end{pmatrix}$ is a small vector.

$\implies$ same is true over $\mathbb{R}$. 
The case where we make progress is $r_{22} \ll r_{11}$. And then, even if we compute
\[ \| u (r_{22}) + v (r_{11}) \|^2 \text{ with } u \text{ not too large} \]
\[ \leq (\frac{r_{22}}{4})^2 \leq (\frac{r_{11}}{4})^2 \]
L5: again, same is true over R.

3.2) How to compute it?

\[
\begin{aligned}
\text{input: } & a, b \in \mathbb{R} \\
\text{output: } & u, v \in \mathbb{R} \text{ s.t. } \| au + bv \| \leq \frac{1}{2} \| bv \| \\
\text{such that } & \| u \| \leq \text{poly}(d) \\
\end{aligned}
\]

1. $\| au + bv \| \leq \frac{1}{2} \| bv \|$ means $au$ should be close to $bv$ (or to $bv$ if we change the sign of $v$).

2. $au$ and $bv$ : products $\rightarrow$ take the log to have

\[ \text{def Log: } \mathbb{R} \rightarrow \mathbb{R}^d \]
\[ x \mapsto (\log(x, y)) \]
Typical in Number Theory

\[ \text{pl: } \log(1 \| x \|) = \log(1 \| y \|) \text{ but } x \text{ not close to 1} \]
\[ \Rightarrow \| \log(au) - \log(bv) \| \text{ small} \]
\[ \not\Rightarrow \| au - bv \| \text{ small} \]
\[ \log : \mathbb{R} \rightarrow (\mathbb{R} / 2\pi)^d \]
\[ x \mapsto (\log \| x \|_2 / \log 2, \Theta(x)) \]
\[ \Theta( re^{i\theta}) = 1 \]

Now:
\[ \| \log(x) - \log(y) \|_2 \leq \frac{1}{2} + 3 \leq \frac{1}{2} \]

Then:
\[ \| x - y \|_2 \leq 4 \min(\| x \|_0, \| y \|_0) \]
\[ \| x - y \|_2 \leq \min(\| x \|_1, \| y \|_1) \]

**Picture (with only Log):** just for the idea (maths after)

\[ \log(a) \log(b) = \log(a^b) \]
\[ \| \log(a) - \log(b) \|_2 \leq \frac{1}{2} \| \log(a) \|_2 + \frac{1}{2} \| \log(b) \|_2 \leq O(\| \log(a) \|_2) \]

This is a CVP over a set which is not a lattice.

To make it a lattice: consider a factor basis
\[ g_1, \ldots, g_k \]
and look for \( u, v \) as a product of \( g_i \)’s.

\[ B = \begin{bmatrix} \log(1) & \log(2) \\ 1 & 2 \end{bmatrix} \]
\[ B = B(B) \]
\[ L = \log(b/a) \]
\[ \Rightarrow \text{add a block of } 2\pi \]
\[ \text{of a block of \( u, v \)} \]
Algorithm: compute $h$ and $k$

* solve CVP in $L$ with $k \rightarrow$ output $s$

* write \( s = \sum \log(w_i) \)

* output $w_i$

Why does it work? How close is $s$ to $k$?

(\text{correctness of Alg)} \quad (i.e. \text{How close } \log(w_i) \text{ to } \log(vb))

$L$ depends on the density of $L$.

\( \text{Vol(L)} \) is fixed $\rightarrow$ increasing dim shorten vectors

$\rightarrow$ with a heuristic counting argument, we believe that $k=O(d^2)$ is enough for $L$ to be dense enough.

Run time of Alg: everything poly except "solve CVP in $L$"

$L$ assume oracle and everything becomes poly-time.

Now, we have the 2 key ingredients: OR and Euclidean div $\rightarrow$ we can do LLL in $R^2$.

$L$ to prove that LLL terminates in poly time + output is small we need an extra ingredient: $N^{\text{s}}$.
LLL in dim 1

Very similar to dim 2:

\[ QR \begin{bmatrix} a & \gamma_j \\ o & c_{ij} \end{bmatrix} \rightarrow \text{choose } i \text{ st. } c_{i, \text{min}} \text{ can be improved and do the reduce and swap.} \]

Result: Approx factor \( \text{quasi-poly}(d)^{O(1)} \)

Time \( \text{poly}(d, k) \) if oracle

What if we actually want to run it?

need to instantiate the oracle \( \Rightarrow \) with generic algorithms

\[ \dim(\mathbb{H}) = d^2 \rightarrow \text{CVP in } \mathbb{H} \text{ can be done in } 2^d \]

\[ \text{Time} \begin{bmatrix} 2^r \\ 2^{dr} \end{bmatrix} \rightarrow \text{Approx} \]

Don't use it in practice
Open problems:

- Improve SVP in L?
  - decrease its dim?
  - use its structure?

* Generalize LLL to BKZ?

LLL: Lattice
SVP in dim 2 → $2^n$-SVP in dim $n$

BKZ: SVP in dim $\beta$ → $2^{n/\beta}$-SVP in dim $n$

The reduction should be easy.
The hard part is how to solve SVP in dim $\beta$.
(That was the hard part in LLL too.)