Lattice-based Cryptography

Léo Ducas

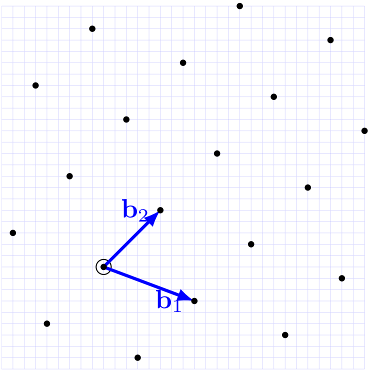
Centrum Wiskunde & Informatica, Amsterdam
Leiden University, Mathematical Institute

The Mathematics of Post-Quantum Cryptography,
, Bonn, Dec 2024

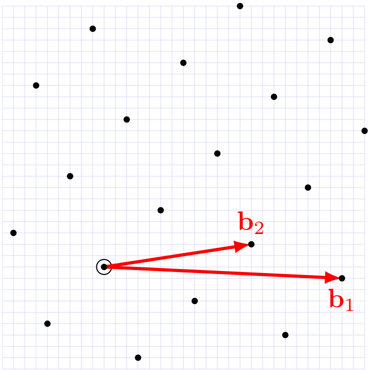
- Lattices
- Public Key Encryption with Lattices
- Digital Signatures with Lattices
- Cryptanalysis: Lattice Reduction

Lattices and their Bases

Lattices are (infinite) regular grids of point in (euclidean) space.
They can be finitely described thanks to their bases.
Example in Dimension 2:



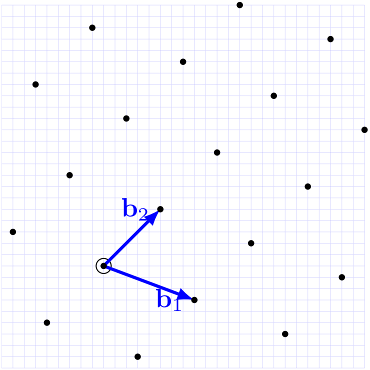
Good Basis $\mathbf{G}$ of L



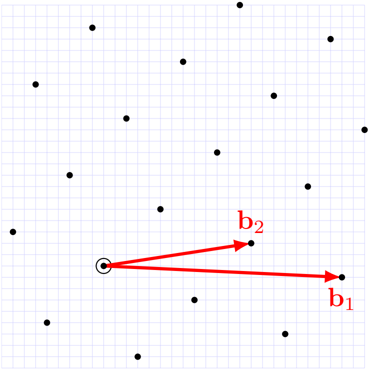
Bad Basis $\mathbf{B}$ of L

Lattices and their Bases

Lattices are (infinite) regular grids of point in (euclidean) space.
They can be finitely described thanks to their bases.
Example in Dimension 2:



Good Basis $\mathbf{G}$ of L

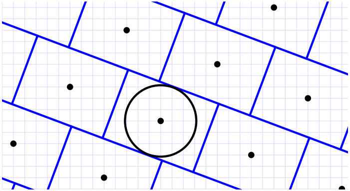


Bad Basis $\mathbf{B}$ of L

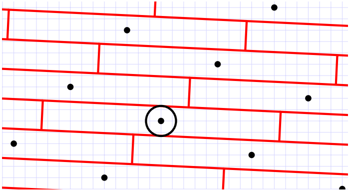
G $\rightarrow$ **B** : easy (randomization);
B $\rightarrow$ **G** : hard (LLL, BKZ, Lattice Sieve...).

Using Lattices in Cryptography

Bases allow to 'tile' the space, and perform error correction.



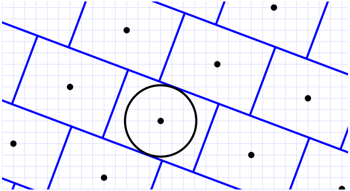
Decoding radius with $\mathbf{G}^\star$



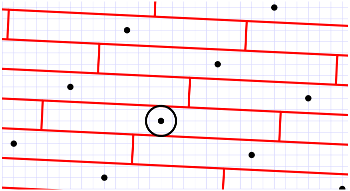
Decoding radius with $\mathbf{B}^\star$

Using Lattices in Cryptography

Bases allow to 'tile' the space, and perform error correction.



Decoding radius with $\mathbf{G}^\star$



Decoding radius with $\mathbf{B}^\star$

As dimension grows > 2, the error tolerance gap between **G** and **B** grows exponentially.

Lattice-Based Asymetric Cryptography

- secret key $=$ good basis **G**
- public key $=$ bad basis **B**

Public Key Encryption with Lattices

Public Key Encryption with Lattices

Encryption Procedure

- View the message as a lattice point $m \in L$ (can do with **B**)
- Choose a random small error vector e (e.g. binary)
- Return ciphertext $c = m + e$

Public Key Encryption with Lattices

Encryption Procedure

- View the message as a lattice point $m \in L$ (can do with **B**)
- Choose a random small error vector e (e.g. binary)
- Return ciphertext $c = m + e$

Decryption Procedure

- Tile to recover the center m of the tile (should do with **G**)
- Return decrypted message m

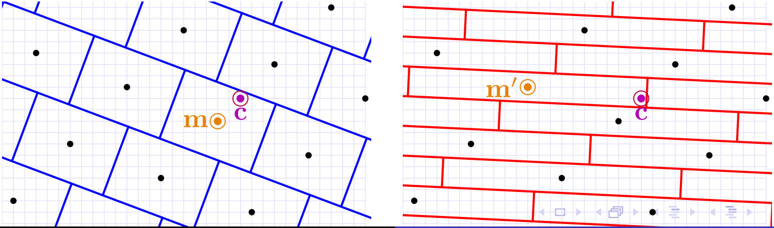
Public Key Encryption with Lattices

Encryption Procedure

- View the message as a lattice point $m \in L$ (can do with **B**)
- Choose a random small error vector e (e.g. binary)
- Return ciphertext $c = m + e$

Decryption Procedure

- Tile to recover the center m of the tile (should do with **G**)
- Return decrypted message m

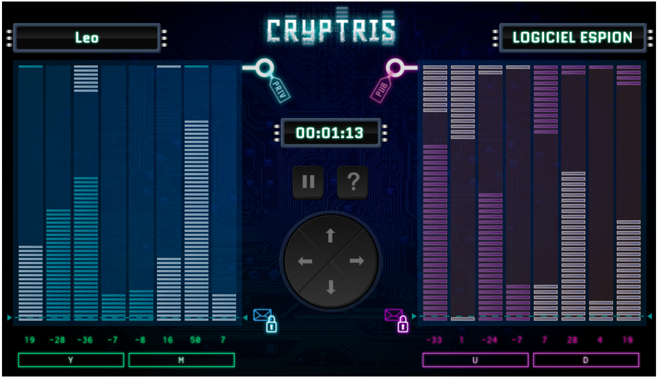


Lattice-based Encryption is as simple as Tetris

It might be hard to get intuition for lattice in dimension > 2...

Cryptris:

A serious game to understand how it works, and why it is secure.



Developed with **Inria** (FR), translated to EN and NL at **CWI**

`https://cryptris.nl/`

Simple to Implement

- Encryption involve a Matrix-Vector product
- Tiling is a more involved, but Decryption can be simplified
- We can choose q-ary lattices, to make all computation mod q

Structured Lattices

- Use circulant blocks in the matrix to improve compactness

$$\begin{bmatrix} c_0 & c_{n-1} & \cdots & c_2 & c_1 \\ c_1 & c_0 & c_{n-1} & & c_2 \\ \vdots & c_1 & c_0 & \ddots & \vdots \\ c_{n-2} & & \ddots & \ddots & c_{n-1} \\ c_{n-1} & c_{n-2} & \cdots & c_1 & c_0 \end{bmatrix}$$

- Speed benefits as well thanks to Fast Fourier Transform

Digital Signatures with Lattices

And why they are a bit more painful

A Naive Approach

RSA "Hash-then-Sign" Signatures

- Signature : Set sig := RSA-decrypt(Hash(message))
- Encryption : Check RSA-encrypt(sig) = Hash(message)

Could we just do the same with lattices ?

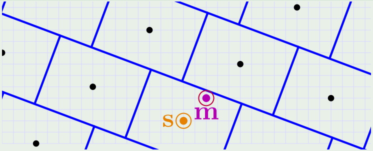
A Naive Approach

RSA "Hash-then-Sign" Signatures

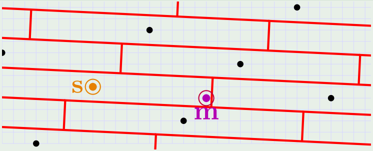
- Signature : Set sig := RSA-decrypt(Hash(message))
- Encryption : Check RSA-encrypt(sig) = Hash(message)

Could we just do the same with lattices ?

Yes !



Correct signature (close)



Incorrect signature (far)

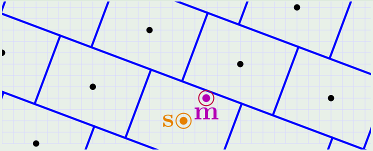
A Naive Approach

RSA "Hash-then-Sign" Signatures

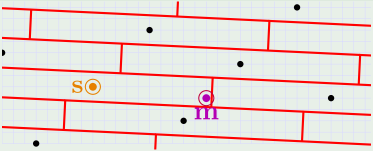
- Signature : Set sig := RSA-decrypt(Hash(message))
- Encryption : Check RSA-encrypt(sig) = Hash(message)

Could we just do the same with lattices ?

Yes !



Correct signature (close)



Incorrect signature (far)

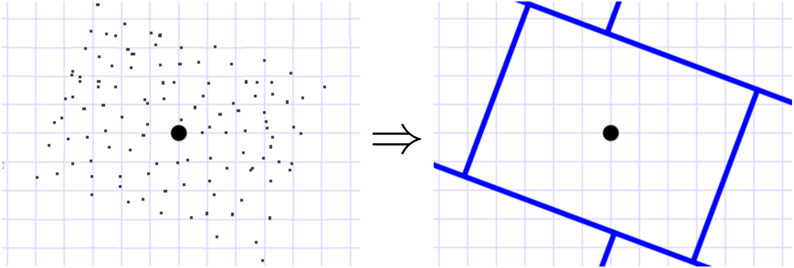
but ...

It's only secure if you don't use it much...

A Learning attack

Nguyen-Regev 2006

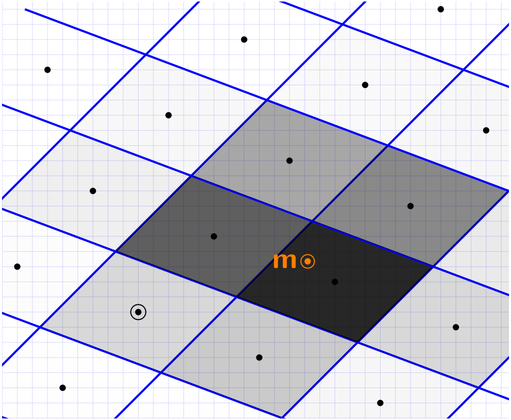
The distribution of signatures leaks the secret key !



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

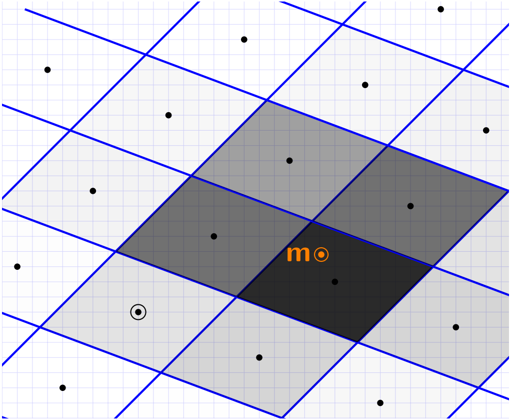
Idea: Hide the tile by randomized rounding



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

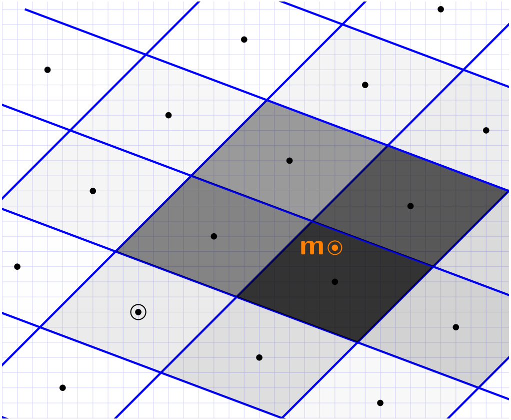
Idea: Hide the tile by randomized rounding



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

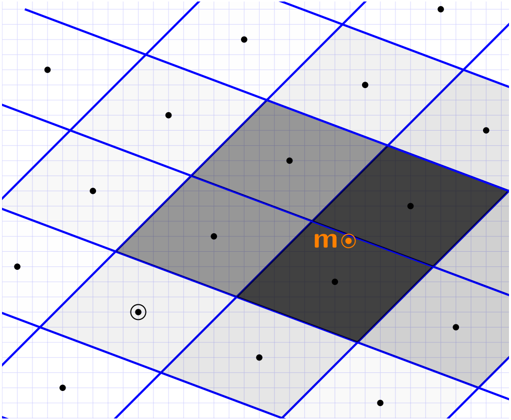
Idea: Hide the tile by randomized rounding



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

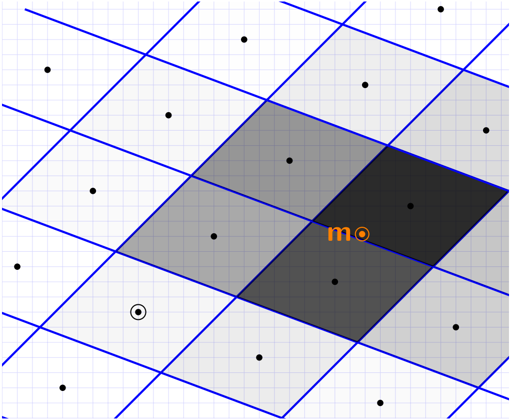
Idea: Hide the tile by randomized rounding



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

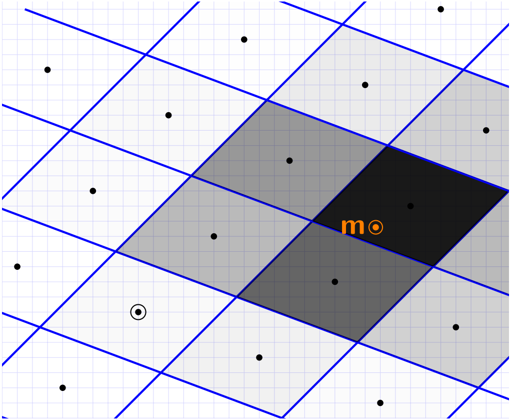
Idea: Hide the tile by randomized rounding



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

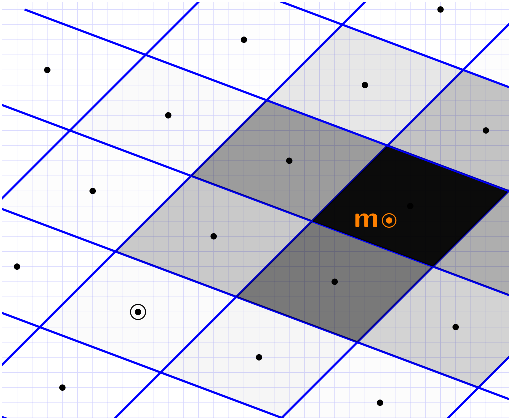
Idea: Hide the tile by randomized rounding



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

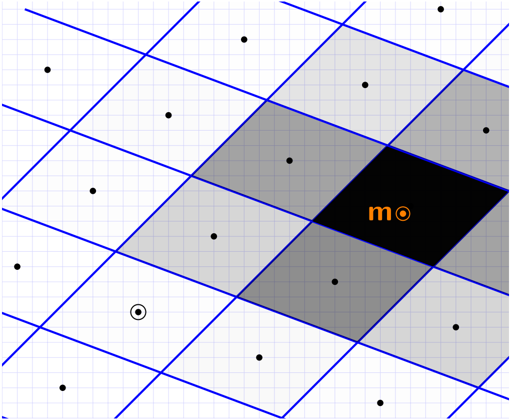
Idea: Hide the tile by randomized rounding



A Provably Secure Randomisation: Discrete Gaussian

Gentry-Peikert-Vaikutanathan 2008

Idea: Hide the tile by randomized rounding



Implementation Details

- Linear algebra mod q (as for Encryption)
- Linear algebra over the real numbers
- Sampling from very specific distribution

Requires Floating Point Arithmetic

Something never done in crypto before !

- Numerical precision issues
- Determinism issues
- Timing side-channel issues

Cryptanalysis: Lattice Reduction

Reduction

Find a $\left\{ \begin{array}{l} \text{unique} \\ \text{canonical} \\ \text{good} \end{array} \right\}$ representative $x \in X$

of a given class $c \in X/_\sim$.

Reduction

Find a $\left\{\begin{array}{l}\text{unique}\\ \text{canonical}\\ \text{good}\end{array}\right\}$ representative $x \in X$

of a given class $c \in X/_{\sim}$.

Lattice Reduction

Find a good basis $B \in GL_n(\mathbb{R})$

of a lattice $\mathcal{L} \in GL_n(\mathbb{R}) / GL_n(\mathbb{Z})$.

Invariants

B and B' generate the same lattice iff:

$$\exists\, U \in GL_n(\mathbb{Z}) \quad \text{st} \quad B' = B \cdot U.$$

$\Rightarrow$ $\det(\mathcal{L}) := \det(B)$ is an invariant of $\mathcal{L}$.

Invariants

B and B' generates the same lattice iff :

$\exists U \in GL_n(\mathbb{Z})$ st $B' = B \cdot U$.

$\Rightarrow$ $\det(\mathcal{L}) := \det(B)$ is an invariant of $\mathcal{L}$.

Gram-Schmidt Orthogonalisation

$$b_i^* := \pi^{\perp}_{(b_1 \ldots b_{i-1})}(b_i)$$

$$= b_i - \sum_{j<i} \frac{\langle b_i , b_j^* \rangle}{\langle b_i^* , b_i^* \rangle} \cdot b_j^*$$

Invariants

B and B′ generates the same lattice iff :

$$\exists\, U \in GL_n(\mathbb{Z}) \quad \text{st} \quad B' = B \cdot U .$$

$\Rightarrow \quad \det(\mathcal{L}) := \det(B)$ is an invariant of $\mathcal{L}$.

Gram-Schmidt Orthogonalisation

$$b_i^* := \pi^{\perp}_{(b_1, \dots, b_{i-1})}(b_i)$$

$$= b_i - \sum_{j<i} \frac{\langle b_i , b_j^* \rangle}{\langle b_j^* , b_j^* \rangle} \cdot b_j^*$$

Invariant

$$\det(\mathcal{L}) = \prod_i \| b_i^* \|$$

Invariants

B and B' generates the same lattice iff:

$$\exists\, U \in GL_n(\mathbb{Z}) \quad \text{st} \quad B' = B \cdot U.$$

$\Rightarrow \quad \det(\mathcal{L}) := \det(B)$ is an invariant of $\mathcal{L}$.

Gram-Schmidt Orthogonalisation

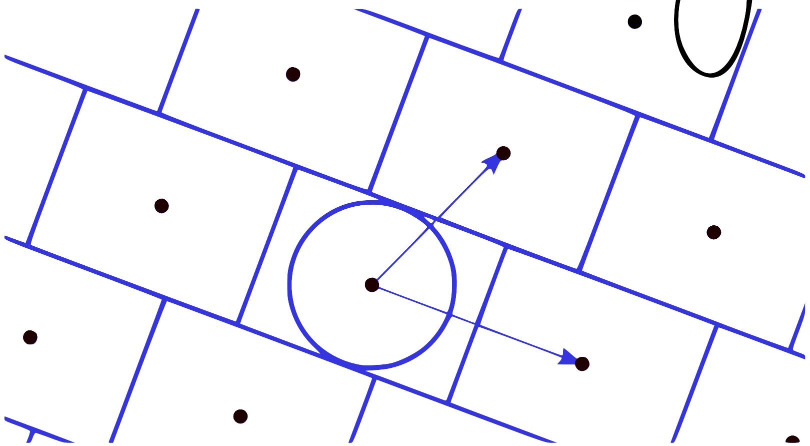
$$b_i^* := \underbrace{\pi^{\perp}_{(b_1, \dots, b_{i-1})}}_{=:\, \pi_i}(b_i)$$

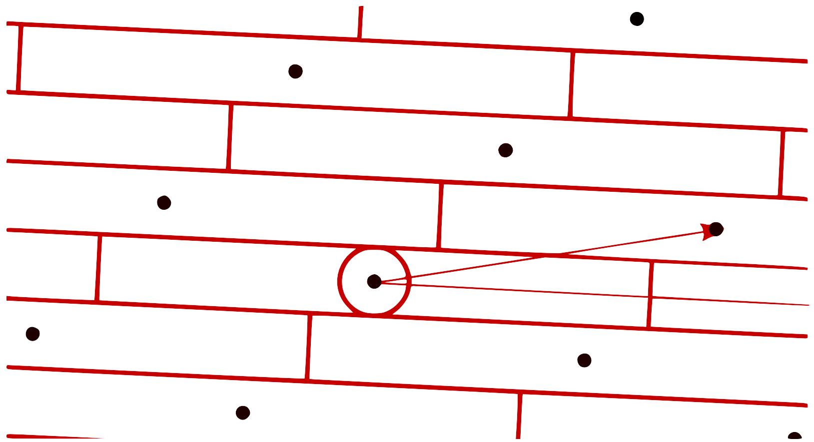
$$= b_i - \sum_{j<i} \frac{\langle b_i, b_j^* \rangle}{\langle b_i^*, b_i^* \rangle} \cdot b_j^*$$

Invariant

$$\det(\mathcal{L}) = \prod_i \| b_i^* \|$$

Good basis

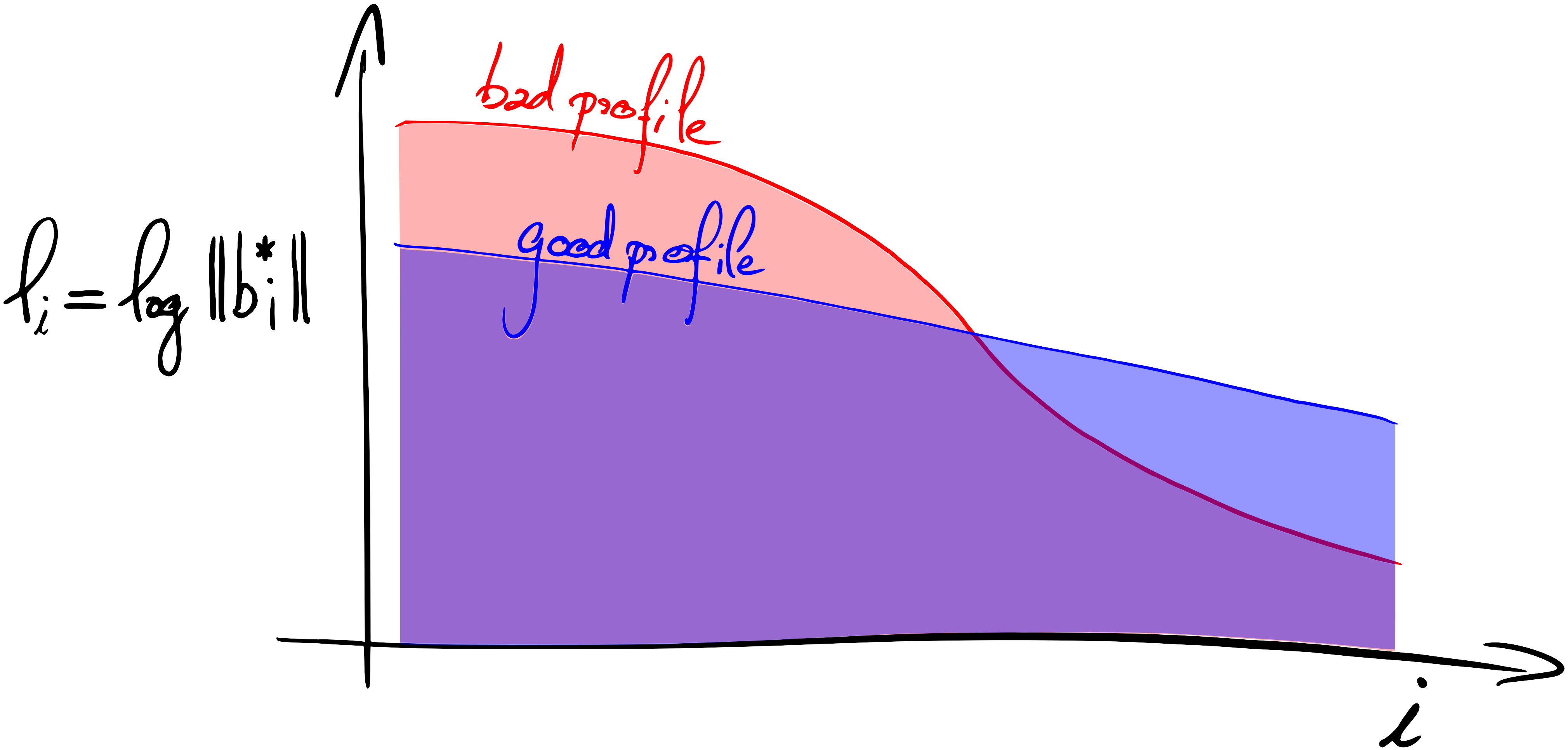




"Good basis" $\iff$ Fundamental Parallelliped $P(B^*)$ is "close" to a hypercube

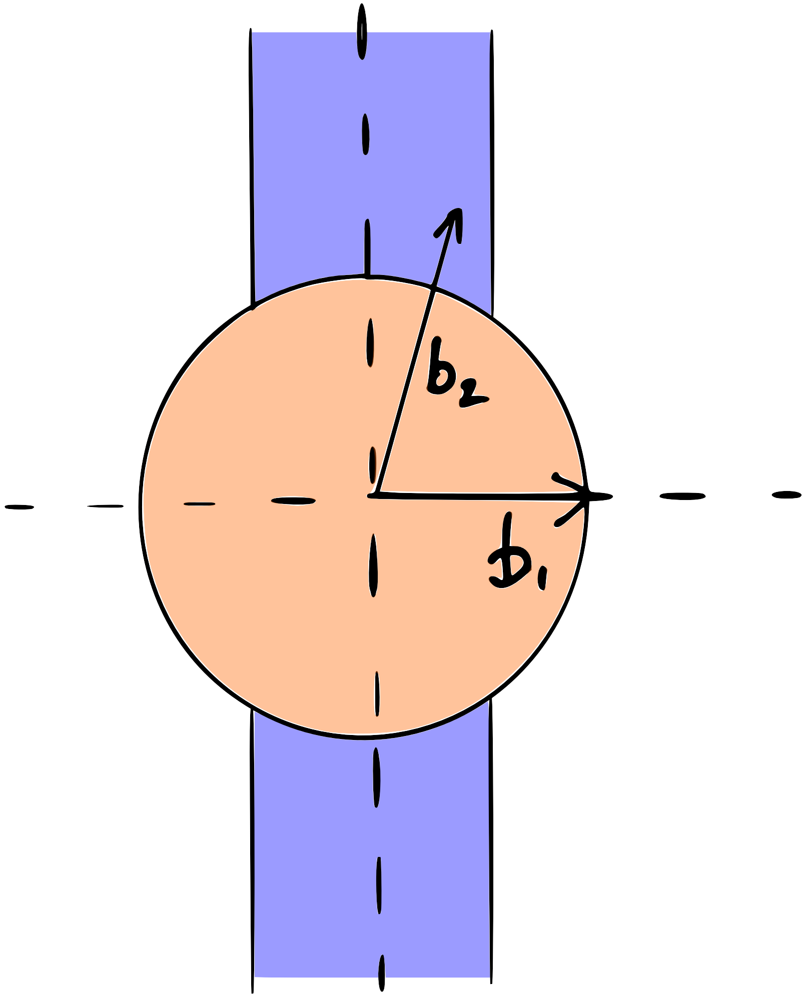
$$\iff \|b_1^*\| \approx \|b_2^*\| \approx \dots \approx \|b_n^*\|.$$

Profile



Area = Area = $\log \det(\mathcal{L})$, invariant.

$n=2$: Lagrange Reduction



Wristwatch Lemma

For any lattice $\mathcal{L}$ of dim 2

$\exists (b_1, b_2)$ a basis s.t.

$$\|b_1\| \leq \|b_2\|$$

$$|\langle b_1, b_2 \rangle| \leq 1/2 \cdot \|b_1\|$$

In particular

$$\|b_1\| \leq \sqrt{4/3} \cdot \|b_2^*\|$$

LLL Reduction

Definition

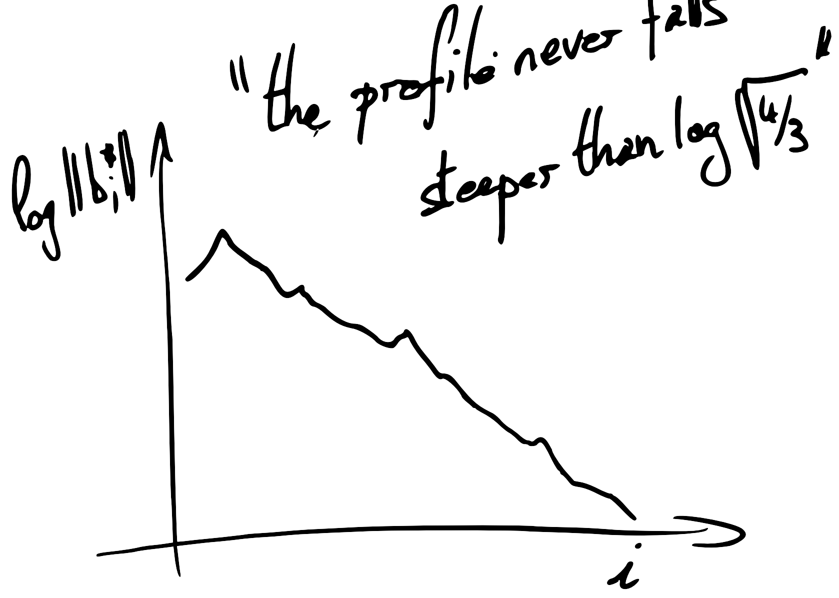
A basis B of $\mathcal{L}$ is LLL-reduced if $(\pi_i(b_i), \pi_i(b_{i+1}))$ is Lagrange-reduced for all $i < n$.

LLL Reduction

Definition

A basis B of $\mathcal{L}$ is LLL-reduced if $(\pi_i(b_i), \pi_i(b_{i+1}))$ is Lagrange-reduced for all $i<n$.

$$\Longrightarrow \forall i<n, \|b_i^*\| \leq \sqrt{4/3} \cdot \|b_{i+1}^*\|$$



LLL Reduction

Definition

A basis B of $\mathcal{L}$ is LLL-reduced if $(\pi_i(b_i), \pi_i(b_{i+1}))$ is Lagrange-reduced for all $i<n$.

"the profile never falls steeper than $\log\sqrt{4/3}$"

$$\Longrightarrow \quad \forall i<n, \; \|b_i^*\| \leq \sqrt{4/3} \cdot \|b_{i+1}^*\|$$

Chain & collect $\Longrightarrow$

$$\|b_1\| \leq \sqrt{4/3}^{\frac{n-1}{2}} \cdot \det(\mathcal{L})^{1/n}.$$

LLL Algorithm

While $\exists i$ **s.t.** $(\pi_i(b_i), \pi_i(b_{i+1}))$ **is not** *Lagrange-reduced*

Lagrange-reduce it ...

Correctness: Trivial

LLL Algorithm

While $\exists i$ **s.t.** $(\pi_i(b_i), \pi_i(b_{i+1}))$ **is not** *Lagrange-reduced*

Lagrange-reduce it ...

Correctness : Trivial

Termination in poly-time :

- Requires a slight relaxation (ε-Lagrange-Reduced)
- Proved using a potential argument :

$$P = \sum_{i \leq n} \sum_{j \leq i} \log(\|b_j^*\|)$$

decreases by ε at each step and is lower-bounded.

Principal Ideal Lattice Reduction

The Problem

Short generator recovery

Given $h \in R$, find a small generator g of the ideal (h).

Note that $g \in (h)$ is a generator iff $g = u \cdot h$ for some unit $u \in \mathbb{R}^\times$.
We need to explore the (multiplicative) unit group $R^\times$.

Short generator recovery

Given $h \in R$, find a small generator g of the ideal (h).

Note that $g \in (h)$ is a generator iff $g = u \cdot h$ for some unit $u \in \mathbb{R}^{\times}$.
We need to explore the (multiplicative) unit group $R^{\times}$.

Translation an to additive problem

Take logarithms:

$$\mathrm{Log} : g \mapsto (\log|\sigma_1(g)|, \ldots, \log|\sigma_n(g)|) \in \mathbb{R}^n$$

where the σ_i's are the canonical embeddings $\mathbb{K} \to \mathbb{C}$.

The Unit Group and the log-unit lattice

Let $R^\times$ denotes the multiplicative group of units of R. Let

$$\Lambda = \operatorname{Log} R^\times.$$

Theorem (Dirichlet unit Theorem)

$\Lambda \subset \mathbb{R}^n$ *is a lattice (of a given rank).*

The Unit Group and the log-unit lattice

Let $R^\times$ denotes the multiplicative group of units of R. Let

$$\Lambda = \operatorname{Log} R^\times.$$

Theorem (Dirichlet unit Theorem)

$\Lambda \subset \mathbb{R}^n$ is a lattice (of a given rank).

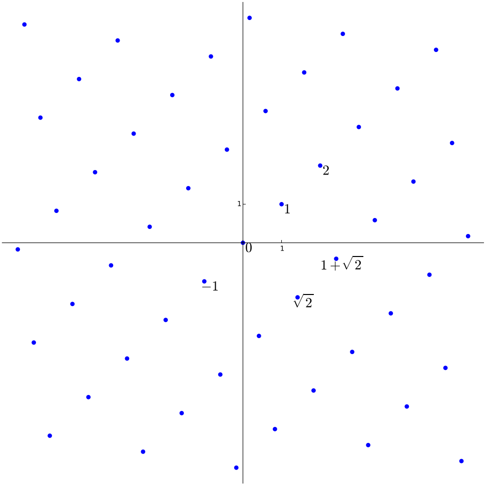
Reduction to a Close Vector Problem

Elements g is a generator of (h) if and only if

$$\operatorname{Log} g \in \operatorname{Log} h + \Lambda.$$

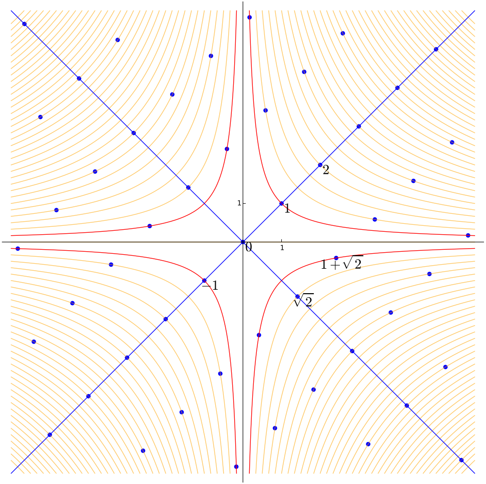
Moreover the map Log preserves some geometric information:
g is the "smallest" generator iff $\operatorname{Log} g$ is the "smallest" in $\operatorname{Log} h + \Lambda$.

Example: Embedding $\mathbb{Z}[\sqrt{2}] \hookrightarrow \mathbb{R}^2$



- x-axis: $\sigma_1(a + b\sqrt{2}) = a + b\sqrt{2}$
- y-axis: $\sigma_2(a + b\sqrt{2}) = a - b\sqrt{2}$
- component-wise additions and multiplications

Example: Embedding $\mathbb{Z}[\sqrt{2}] \hookrightarrow \mathbb{R}^2$

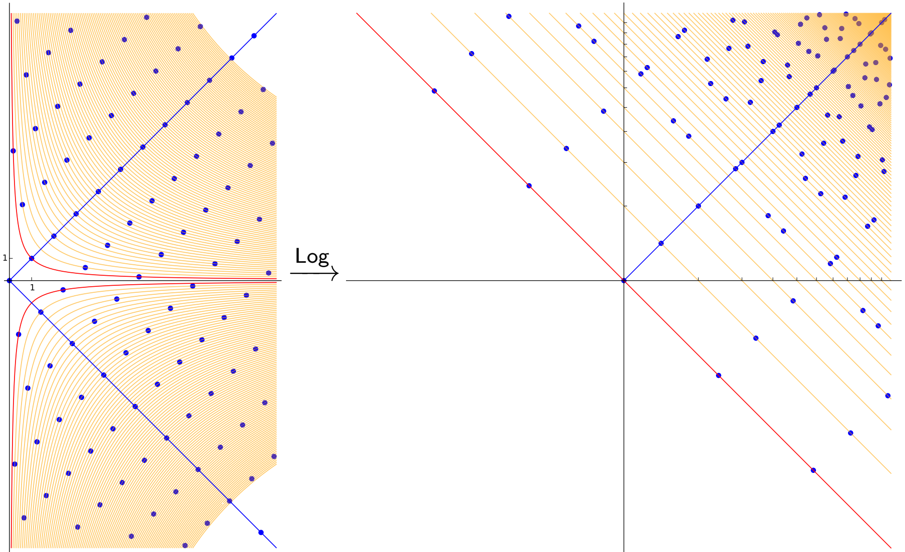


- x-axis: $\sigma_1(a + b\sqrt{2}) = a + b\sqrt{2}$
- y-axis: $\sigma_2(a + b\sqrt{2}) = a - b\sqrt{2}$
- component-wise additions and multiplications

- "Orthogonal" elements (blue)
- Units (algebraic norm 1) (red)
- "Isonorms" curves (orange)

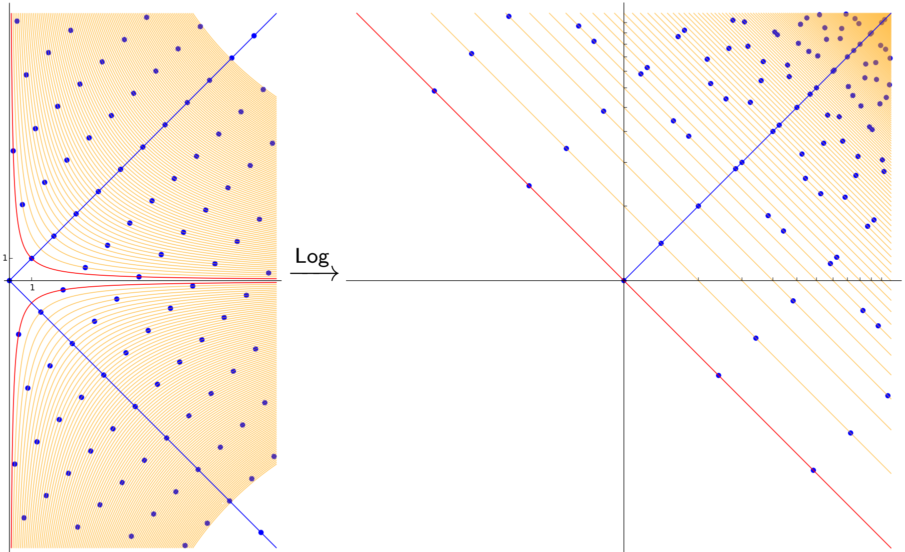
Example: Logarithmic Embedding Log $\mathbb{Z}[\sqrt{2}]$

$(\{\bullet\}, +)$ is a sub-monoid of $\mathbb{R}^2$



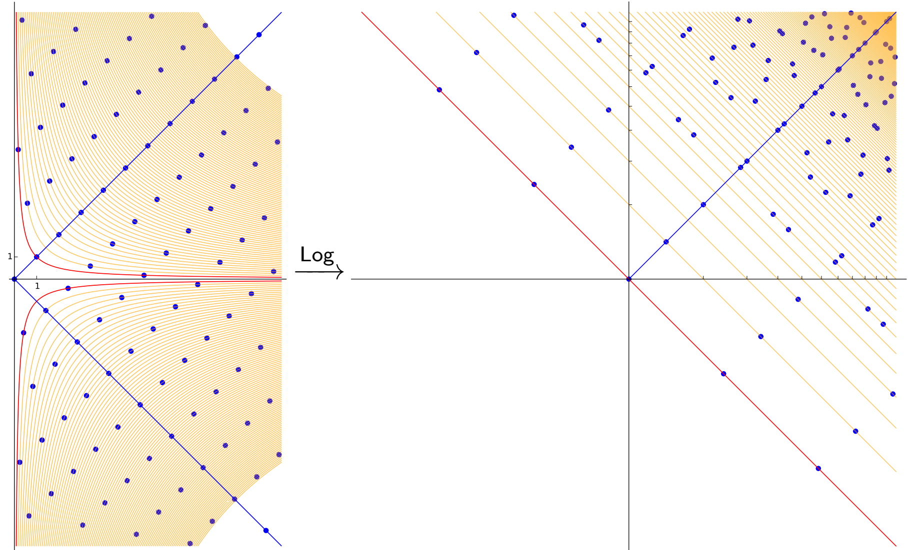
Example: Logarithmic Embedding Log $\mathbb{Z}[\sqrt{2}]$

$\Lambda = (\{\bullet\}, +) \cap \diagdown$ is a lattice of $\mathbb{R}^2$, orthogonal to $(1, 1)$



Example: Logarithmic Embedding Log $\mathbb{Z}[\sqrt{2}]$

$\{\bullet\} \cap \diagdown$ are shifted finite copies of Λ



Reduction modulo $\Lambda = \mathrm{Log}\,\mathbb{Z}[\sqrt{2}]^\times$

The reduction modΛ for various fundamental domains.