Ideal-SVP is Hard for Small-Norm Uniform Prime Ideals

Joël Felderhoff, Alice Pellet-Mary, Damien Stehlé and Benjamin Wesolowski

INRIA Lyon, ENS de Lyon

Contributions

- New reduction: $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP.
- Application: new distribution of NTRU instances with difficulty based on wc-ideal-SVP.

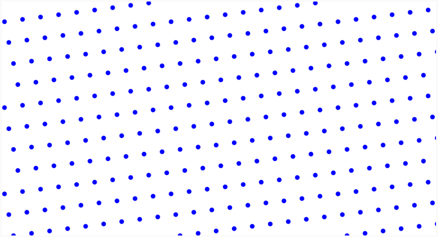
Contributions

- New reduction: $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP.
- Application: new distribution of NTRU instances with difficulty based on $\mathrm{wc\text{-}ideal\text{-}SVP}$.

To appear in the proceedings of **TCC 2023**. Available at:
`https://eprint.iacr.org/2023/1370`

Definitions

Lattices



A 2-dimensional lattice

Definition

For $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{Z}^n$ linearly independent, the lattice spanned by the basis $\mathbf{b}_1, \ldots, \mathbf{b}_n$ is $\mathcal{L} = \sum_i \mathbb{Z} \cdot \mathbf{b}_i \subset \mathbb{R}^n$.
It is discrete and has a shortest non-zero vector.

Finding any short non-zero vector in $\mathcal{L}$ given $(\mathbf{b}_i)_i$ is hard in general.

Number fields and ideals

$K = \mathbb{Q}[X]/(X^n + 1)$, $\mathcal{O}_K = \mathbb{Z}[X]/(X^n + 1)$ for $n = 2^r$
(K a number field, $\mathcal{O}_K$ its ring of integers).

The size of an element $a \in K$ is $\|a\| = \left(\sum_i |a_i|^2\right)^{1/2}$.

The size of an element is the ℓ_2-norm of its Minkowski embedding.

Number fields and ideals

$K = \mathbb{Q}[X]/(X^n + 1)$, $\mathcal{O}_K = \mathbb{Z}[X]/(X^n + 1)$ for $n = 2^r$
(K a number field, $\mathcal{O}_K$ its ring of integers).

The size of an element $a \in K$ is $\|a\| = \left(\sum_i |a_i|^2\right)^{1/2}$.

The size of an element is the ℓ_2-norm of its Minkowski embedding.

Definition (Ideal)

A set $\mathfrak{a} \subseteq K$ is an ideal if it is discrete, stable by addition and by multiplication by any element of $\mathcal{O}_K$. It is then a lattice.

Norm of an ideal: $\mathcal{N}(I) = \mathrm{Vol}(I)/\mathrm{Vol}(\mathcal{O}_K) \in \mathbb{Z}$.

Ideal arithmetic

Let $\mathfrak{a}, \mathfrak{b}$ ideals of K, and $a \in K$.

Principal ideal

$(a) = \{x \cdot a, x \in \mathcal{O}_K\}$.

Multiplication and inverse

$\mathfrak{a} \cdot \mathfrak{b} = \left\{\sum_i a_i \cdot b_i\right\}, \mathfrak{a}^{-1} = \{x \in K, x \cdot \mathfrak{a} \subseteq \mathcal{O}_K\}$.
We have that $\mathfrak{a} \cdot \mathfrak{a}^{-1} = \mathcal{O}_K$.

Prime ideals

An ideal $\mathfrak{p}$ is prime ($\mathfrak{p} \in \mathcal{P}$) if

$$\mathfrak{p} = \mathfrak{a} \cdot \mathfrak{b} \Rightarrow \mathfrak{a} = \mathcal{O}_K \text{ or } \mathfrak{b} = \mathcal{O}_K$$

The problem ideal-HSVP

Definition (ideal-HSVP$_\gamma$)

Given an ideal $\mathfrak{a} \subseteq K$, find $x \in \mathfrak{a} \setminus \{0\}$ with $\|x\| \leq \gamma \cdot \mathcal{N}(\mathfrak{a})^{1/d}$.

Ideal lattices are **not typical lattices**. E.g., they verify $\lambda_1(I) \approx \lambda_d(I)$.

[1] [CDPR16, CDW17, PHS19]

The problem ideal-HSVP

Definition (ideal-HSVP$_\gamma$)

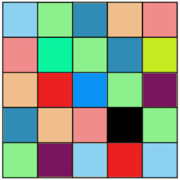
Given an ideal $\mathfrak{a} \subseteq K$, find $x \in \mathfrak{a} \setminus \{0\}$ with $\|x\| \leq \gamma \cdot \mathcal{N}(\mathfrak{a})^{1/d}$.

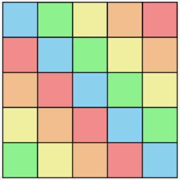
Ideal lattices are **not typical lattices**. E.g., they verify $\lambda_1(I) \approx \lambda_d(I)$.

- There are specifics attacks on ideal lattices[1].
- Ideals are the simplest examples of module lattices (KYBER, DILITHIUM).
- ideal-HSVP is related to other structured lattice problems (Module-SVP, NTRU, RingLWE).

[1] [CDPR16, CDW17, PHS19]

Why small ideal lattices?

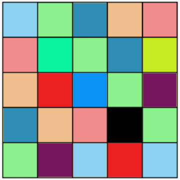


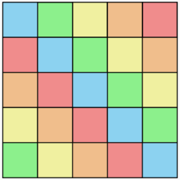


Typical lattice basis: $O(d^2)$ integers vs **ideal lattice basis**: $O(d)$ integers.[2]

[2] Images from [Qua14]

Why small ideal lattices?



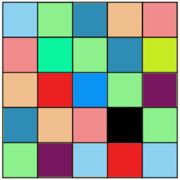


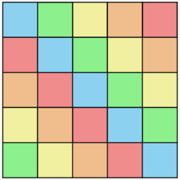
Typical lattice basis: $O(d^2)$ integers vs **ideal lattice basis**: $O(d)$ integers.[2]

Bitsize of a typical element of $\mathfrak{a}$ is $\log(\mathcal{N}(\mathfrak{a}))$.
$\rightarrow$ We want $\mathcal{N}(\mathfrak{a}) \approx \text{poly}(d)^d$ in order to have small keys.

[2] Images from [Qua14]

Why small ideal lattices?





Typical lattice basis: $O(d^2)$ integers vs **ideal lattice basis**: $O(d)$ integers.[2]

Bitsize of a typical element of $\mathfrak{a}$ is $\log(\mathcal{N}(\mathfrak{a}))$.
$\rightarrow$ We want $\mathcal{N}(\mathfrak{a}) \approx \text{poly}(d)^d$ in order to have small keys.

Also: faster algorithms.

[2] Images from [Qua14]

Average-case to average-case reduction

Worst-case: Solve $\mathcal{P}$ for **all** instance of $\mathcal{P}$ (for the worst instance).

Average-case for D: Solve $\mathcal{P}$ for $I \leftarrow D$ with non-negligible probability.

For cryptography, we are interested in **Average-case** hardness.

Average-case to average-case reduction

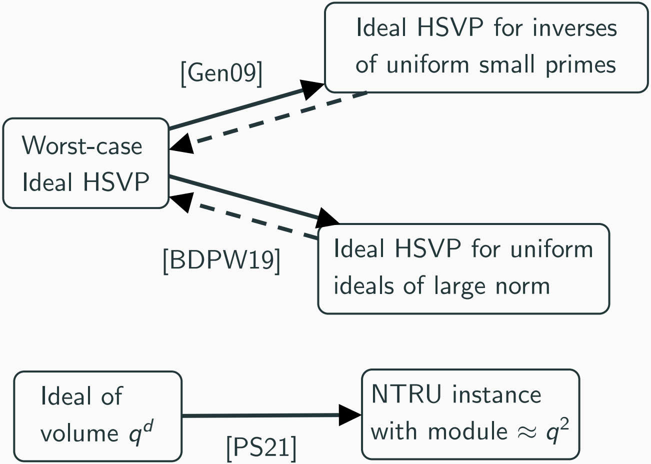
Worst-case: Solve $\mathcal{P}$ for **all** instance of $\mathcal{P}$ (for the worst instance).

Average-case for D: Solve $\mathcal{P}$ for $I \leftarrow D$ with non-negligible probability.

For cryptography, we are interested in **Average-case** hardness.

Here we show an Average-case to Average-case reduction.

Prior works on ideal-HSVP



Description of our work and motivation

Random version of $\mathrm{ideal\text{-}HSVP}$

$\mathcal{W}$-$\mathrm{ideal\text{-}HSVP}$: solving $\mathrm{ideal\text{-}HSVP}$ for a uniform element of $\mathcal{W}$.

Note: there are sets $\mathcal{W}$ such that $\mathcal{W}$-$\mathrm{ideal\text{-}HSVP}$ is easy [BGP22].

Description of our work and motivation

Random version of $\mathrm{ideal\text{-}HSVP}$

$\mathcal{W}$-$\mathrm{ideal\text{-}HSVP}$: solving $\mathrm{ideal\text{-}HSVP}$ for a uniform element of $\mathcal{W}$.

Note: there are sets $\mathcal{W}$ such that $\mathcal{W}$-$\mathrm{ideal\text{-}HSVP}$ is easy [BGP22].

We show that $\mathcal{P}^{-1}$-$\mathrm{ideal\text{-}HSVP}$ reduces to $\mathcal{P}$-$\mathrm{ideal\text{-}HSVP}$.

Two reasons

1. [Gen09]: $\mathrm{ideal\text{-}HSVP}$ (for all ideals) reduces to $\mathcal{P}^{-1}$-$\mathrm{ideal\text{-}HSVP}$.
2. The NTRU reduction from [PS21] works for integral ideals.

Sampling ideals

Arakelov ideal sampling [BDPW20, Boe22]

Algorithm 2.1 `ArakelovSampling` algorithm

Output: An ideal $\mathfrak{b}$

1: Let $\mathfrak{q}$ a uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

Arakelov ideal sampling [BDPW20, Boe22]

Algorithm 2.1 `ArakelovSampling` algorithm

Output: An ideal $\mathfrak{b}$

1: Let $\mathfrak{q}$ a uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

Arakelov ideal sampling [BDPW20, Boe22]

Algorithm 2.1 `ArakelovSampling` algorithm

Output: An ideal $\mathfrak{b}$

1: Let $\mathfrak{q}$ a uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

Arakelov ideal sampling [BDPW20, Boe22]

Algorithm 2.1 `ArakelovSampling` algorithm

Output: An ideal $\mathfrak{b}$

1: Let $\mathfrak{q}$ a uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

Arakelov ideal sampling [BDPW20, Boe22]

Algorithm 2.1 `ArakelovSampling` algorithm

Output: An ideal $\mathfrak{b}$

1: Let $\mathfrak{q}$ a uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

Arakelov ideal sampling [BDPW20, Boe22]

Algorithm 2.1 `ArakelovSampling` algorithm

Output: An ideal $\mathfrak{b}$

1: Let $\mathfrak{q}$ a uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

Arakelov ideal sampling [BDPW20, Boe22]

Algorithm 2.1 `ArakelovSampling` algorithm

Output: An ideal $\mathfrak{b}$

1: Let $\mathfrak{q}$ a uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

Outputs uniform integral ideals of norm $\approx r^d$ for $r = 2^{O(d)}$.
$\Rightarrow$ Too big for our use-cases!

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$.

Algorithm 2.2 `ArakelovSampling'` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $\mathfrak{q}$ an uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$.

Algorithm 2.2 `ArakelovSampling′` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $\mathfrak{q}$ an uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$.

Algorithm 2.2 `ArakelovSampling′` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow$ `SampleWithTrap(·)`. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$.

Algorithm 2.2 `ArakelovSampling'` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)

2: Sample a small continuous Gaussian ζ and a uniform rotation u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$

4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.

5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$.

Algorithm 2.2 `ArakelovSampling′` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$.

Algorithm 2.2 `ArakelovSampling`′ algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \in I$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$.

Algorithm 2.2 `ArakelovSampling′` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \in I$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$.

Algorithm 2.2 `ArakelovSampling'` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \in I$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$.

Algorithm 2.2 `ArakelovSampling'` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)

2: Sample a small continuous Gaussian ζ and a uniform rotation u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \in I$.

4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.

5: **Return** $\mathfrak{b} = x \cdot I^{-1}$ and $y = x^{-1} \cdot s_I$.

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$.

Algorithm 2.2 `ArakelovSampling`′ algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow$ `SampleWithTrap(·)`. (Quantum)

2: Sample a small continuous Gaussian ζ and a uniform rotation u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \in I$.

4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.

5: **Return** $\mathfrak{b} = x \cdot I^{-1}$ and $y = x^{-1} \cdot s_I$.

Drawback

The element $y = x^{-1} \cdot s_I$ can be very large compared to $\mathcal{N}(\mathfrak{b}^{-1})^{1/d}$.

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$.

Algorithm 2.2 `ArakelovSampling′` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

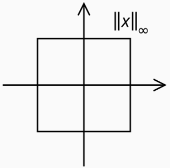
1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \in I$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$ and $y = x^{-1} \cdot s_I$.

Drawback

The element $y = x^{-1} \cdot s_I$ can be very large compared to $\mathcal{N}(\mathfrak{b}^{-1})^{1/d}$.

$\rightarrow$ This happens if x is **unbalanced**

Some details on `ArakelovSampling`



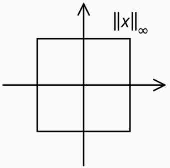
The set $\mathcal{B}_\infty(r)$

1. We pick $\mathfrak{q}$ uniform prime.
2. We sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap \mathfrak{q}\right)$.
3. We return $\mathfrak{b} = x \cdot \mathfrak{q}^{-1}$.

Sufficient conditions for uniform $\mathfrak{b}$

1. $\left|\mathcal{B}_\infty(r) \bigcap \mathfrak{q}\right|$ does not depend on $\mathfrak{q}$ (too much).
2. $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_\infty(r)) \bigcap \{\sum x_i = t\})$ is $\approx$ constant for $t \in [A, B]$.

Some details on `ArakelovSampling`



The set $\mathcal{B}_\infty(r)$

1. We pick $\mathfrak{q}$ uniform prime.
2. We sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap \mathfrak{q}\right)$.
3. We return $\mathfrak{b} = x \cdot \mathfrak{q}^{-1}$.

Sufficient conditions for uniform $\mathfrak{b}$

1. $\left|\mathcal{B}_\infty(r) \bigcap \mathfrak{q}\right|$ does not depend on $\mathfrak{q}$ (too much).
2. $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_\infty(r)) \bigcap \{\sum x_i = t\})$ is $\approx$ constant for $t \in [A, B]$.

Drawback

There are $x \in \mathcal{B}_\infty(r)$ with $\left\|x^{-1}\right\|$ very large.

Main contribution:

$\mathcal{P}^{-1}$-ideal-SVP **to** $\mathcal{P}$-ideal-SVP

First contribution: Generalized Arakelov ideal sampling

We generalize the approach of [BDPW20, Boe22]:

Algorithm 3.1 $\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}$ algorithm

Input: $\mathfrak{a}$ an ideal, $s_{\mathfrak{a}} \in \mathfrak{a}$ small, $\mathcal{B}_{A,B} \subset K_{\mathbb{R}}$ a **well chosen** set.

Output: $(\mathfrak{b}, y)$ such that $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)

2: Sample ζ and u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} \cdot \mathfrak{a}$

4: Let $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \cdot s_{\mathfrak{a}} \in I$.

5: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_{A,B} \bigcap I\right)$ using s_I .

6: **Return** $(\mathfrak{b} = x \cdot I^{-1}, y = x^{-1} \cdot s_I \cdot v_{\mathfrak{q}})$ $\triangleright$ $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$

First contribution: Generalized Arakelov ideal sampling

We generalize the approach of [BDPW20, Boe22]:

Algorithm 3.1 $\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}$ algorithm

Input: $\mathfrak{a}$ an ideal, $s_{\mathfrak{a}} \in \mathfrak{a}$ small, $\mathcal{B}_{A,B} \subset K_{\mathbb{R}}$ a **well chosen** set.

Output: $(\mathfrak{b}, y)$ such that $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)
2: Sample ζ and u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} \cdot \mathfrak{a}$
4: Let $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \cdot s_{\mathfrak{a}} \in I$.
5: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_{A,B} \bigcap I\right)$ using s_I .
6: **Return** $\left(\mathfrak{b} = x \cdot I^{-1}, y = x^{-1} \cdot s_I \cdot v_{\mathfrak{q}}\right)$ ▷ $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$

First contribution: Generalized Arakelov ideal sampling

We generalize the approach of [BDPW20, Boe22]:

Algorithm 3.1 $\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}$ algorithm

Input: $\mathfrak{a}$ an ideal, $s_{\mathfrak{a}} \in \mathfrak{a}$ small, $\mathcal{B}_{A,B} \subset K_{\mathbb{R}}$ a **well chosen** set.

Output: $(\mathfrak{b}, y)$ such that $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)

2: Sample ζ and u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} \cdot \mathfrak{a}$

4: Let $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \cdot s_{\mathfrak{a}} \in I$.

5: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_{A,B} \bigcap I\right)$ using s_I .

6: **Return** $\left(\mathfrak{b} = x \cdot I^{-1}, y = x^{-1} \cdot s_I \cdot v_{\mathfrak{q}}\right)$ $\triangleright$ $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$

First contribution: Generalized Arakelov ideal sampling

We generalize the approach of [BDPW20, Boe22]:

Algorithm 3.1 `SampleIdeal`$_{\mathcal{B}_{A,B}}$ algorithm

Input: $\mathfrak{a}$ an ideal, $s_{\mathfrak{a}} \in \mathfrak{a}$ small, $\mathcal{B}_{A,B} \subset K_{\mathbb{R}}$ a **well chosen** set.

Output: $(\mathfrak{b}, y)$ such that $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow$ `SampleWithTrap`$(\cdot)$. (Quantum)

2: Sample ζ and u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} \cdot \mathfrak{a}$

4: Let $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \cdot s_{\mathfrak{a}} \in I$.

5: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_{A,B} \bigcap I\right)$ using s_I .

6: **Return** $(\mathfrak{b} = x \cdot I^{-1}, y = x^{-1} \cdot s_I \cdot v_{\mathfrak{q}})$ $\triangleright$ $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$

First contribution: Generalized Arakelov ideal sampling

We generalize the approach of [BDPW20, Boe22]:

Algorithm 3.1 `SampleIdeal`$_{\mathcal{B}_{A,B}}$ algorithm

Input: $\mathfrak{a}$ an ideal, $s_{\mathfrak{a}} \in \mathfrak{a}$ small, $\mathcal{B}_{A,B} \subset K_{\mathbb{R}}$ a **well chosen** set.

Output: $(\mathfrak{b}, y)$ such that $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow$ `SampleWithTrap`$(\cdot)$. (Quantum)

2: Sample ζ and u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} \cdot \mathfrak{a}$

4: Let $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \cdot s_{\mathfrak{a}} \in I$.

5: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_{A,B} \bigcap I\right)$ using s_I .

6: **Return** $(\mathfrak{b} = x \cdot I^{-1}, y = x^{-1} \cdot s_I \cdot v_{\mathfrak{q}})$ ▷ $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$

First contribution: Generalized Arakelov ideal sampling

We generalize the approach of [BDPW20, Boe22]:

Algorithm 3.1 $\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}$ algorithm

Input: $\mathfrak{a}$ an ideal, $s_{\mathfrak{a}} \in \mathfrak{a}$ small, $\mathcal{B}_{A,B} \subset K_{\mathbb{R}}$ a **well chosen** set.

Output: $(\mathfrak{b}, y)$ such that $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)

2: Sample ζ and u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} \cdot \mathfrak{a}$

4: Let $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \cdot s_{\mathfrak{a}} \in I$.

5: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_{A,B} \bigcap I\right)$ using s_I .

6: **Return** $\left(\mathfrak{b} = x \cdot I^{-1}, y = x^{-1} \cdot s_I \cdot v_{\mathfrak{q}}\right)$ ▷ $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$

First contribution: Generalized Arakelov ideal sampling

We generalize the approach of [BDPW20, Boe22]:

Algorithm 3.1 $\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}$ algorithm

Input: $\mathfrak{a}$ an ideal, $s_{\mathfrak{a}} \in \mathfrak{a}$ small, $\mathcal{B}_{A,B} \subset K_{\mathbb{R}}$ a **well chosen** set.

Output: $(\mathfrak{b}, y)$ such that $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)

2: Sample ζ and u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} \cdot \mathfrak{a}$

4: Let $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \cdot s_{\mathfrak{a}} \in I$.

5: Sample $x \hookleftarrow \mathcal{U}(\mathcal{B}_{A,B} \bigcap I)$ using s_I .

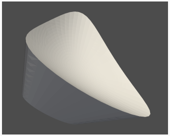
6: **Return** $(\mathfrak{b} = x \cdot I^{-1}, y = x^{-1} \cdot s_I \cdot v_{\mathfrak{q}})$ $\quad \triangleright\ y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$

Theorem

Let $(\mathfrak{b}, y) = \texttt{SampleIdeal}_{\mathcal{B}_{A,B}}(\mathfrak{a}, s_{\mathfrak{a}}, A, B)$.

If $\mathcal{B}_{A,B}$ is **well chosen** then $\mathfrak{b}$ is almost uniform in $\mathcal{I}_{A,B}$ and y is small.

What does "well chosen" mean?



1. $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ does not depend on $\mathfrak{a}$ (too much).
2. $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
3. Its elements must be balanced.

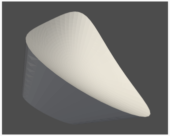
Balanced elements (for Minkowski embedding)

$x \in K$ is balanced if for all i,

$$\frac{1}{\eta} \leq \frac{x_i}{\prod_j x_j^{1/d}} \leq \eta.$$

This is the same as $x \approx \mathcal{N}(x)^{1/d} \cdot (1, \ldots, 1)$.

What does "well chosen" mean?



1. $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ does not depend on $\mathfrak{a}$ (too much).
2. $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
3. Its elements must be balanced.

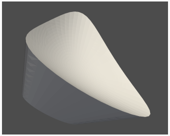
Balanced elements **(for Minkowski embedding)**

$x \in K$ is balanced if for all i,

$$\frac{1}{\eta} \leq \frac{x_i}{\prod_j x_j^{1/d}} \leq \eta.$$

This is the same as $x \approx \mathcal{N}(x)^{1/d} \cdot (1, \ldots, 1)$.

What does "well chosen" mean?



1. $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ does not depend on $\mathfrak{a}$ (too much).
2. $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
3. Its elements must be balanced.

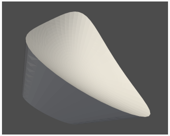
Balanced elements (for Minkowski embedding)

$x \in K$ is balanced if for all i,

$$\frac{1}{\eta} \leq \frac{x_i}{\prod_j x_j^{1/d}} \leq \eta.$$

This is the same as $x \approx \mathcal{N}(x)^{1/d} \cdot (1, \ldots, 1)$.

What does "well chosen" mean?



1. $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ does not depend on $\mathfrak{a}$ (too much).
2. $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
3. Its elements must be balanced.

Balanced elements (for Minkowski embedding)

$x \in K$ is balanced if for all i,

$$\frac{1}{\eta} \leq \frac{x_i}{\prod_j x_j^{1/d}} \leq \eta.$$

This is the same as $x \approx \mathcal{N}(x)^{1/d} \cdot (1, \ldots, 1)$.

In [BDPW20]: $\mathcal{B}_\infty(r)$: verifies items 1 and 2 but not 3!

Our shape

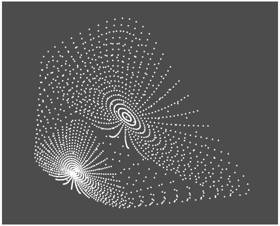
Reminder: conditions for being **well chosen**:

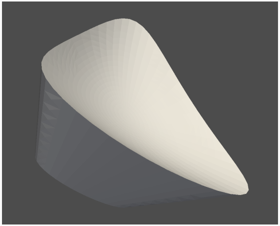
1. $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ does not depend on $\mathfrak{a}$ (too much).
2. $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
3. Its elements are balanced.

Our shape

Reminder: conditions for being **well chosen**:

1. $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ does not depend on $\mathfrak{a}$ (too much).
2. $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
3. Its elements are balanced.



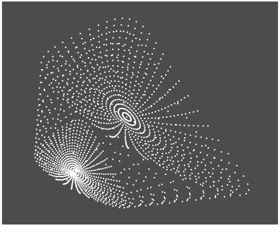


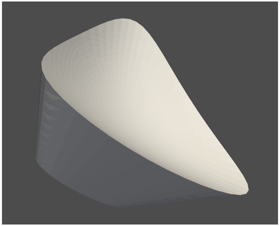
$$\mathcal{B}^{\eta}_{A,B} = \left\{ x \in K_{\mathbb{R}}, \quad |\mathcal{N}(x)| \in [A, B], \quad \left\| \mathrm{Log}\left(\frac{x}{\mathcal{N}(x)^{1/d}} \right) \right\|_2 \leq \log(\eta) \right\}$$

Our shape

Reminder: conditions for being **well chosen**:

1. $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ does not depend on $\mathfrak{a}$ (too much).
2. $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
3. Its elements are balanced.



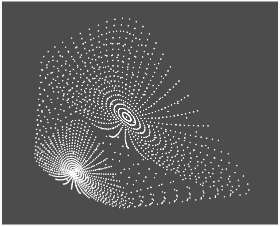


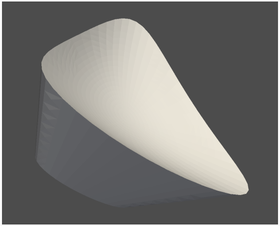
$$\mathcal{B}^{\eta}_{A,B} = \left\{ x \in K_{\mathbb{R}}, \quad |\mathcal{N}(x)| \in [A, B], \quad \left\| \mathrm{Log}\left(\frac{x}{\mathcal{N}(x)^{1/d}} \right) \right\|_2 \leq \log(\eta) \right\}$$

Our shape

Reminder: conditions for being **well chosen**:

1. $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ does not depend on $\mathfrak{a}$ (too much).
2. $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
3. Its elements are balanced.



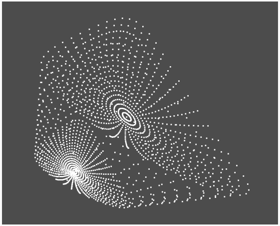


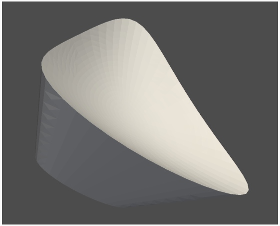
$$\mathcal{B}^{\eta}_{A,B} = \left\{ x \in K_{\mathbb{R}}, \quad |\mathcal{N}(x)| \in [A, B], \quad \left\| \mathrm{Log}\left(\frac{x}{\mathcal{N}(x)^{1/d}} \right) \right\|_2 \leq \log(\eta) \right\}$$

Our shape

Reminder: conditions for being **well chosen**:

1. $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ does not depend on $\mathfrak{a}$ (too much).
2. $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
3. Its elements are balanced.





$$\mathcal{B}^{\eta}_{A,B} = \left\{ x \in K_{\mathbb{R}}, \quad |\mathcal{N}(x)| \in [A, B], \quad \left\| \mathrm{Log}\left(\frac{x}{\mathcal{N}(x)^{1/d}} \right) \right\|_2 \leq \log(\eta) \right\}$$

Our $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

The oracle $\mathcal{O}$ solves ideal-HSVP for $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Input: An ideal $I = \mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform prime of norm in $[A, B]$.
Output: $x \in \mathfrak{p}^{-1} \setminus \{0\}$ small.

```
Let s_p = O(p).
Let (b, y) = SampleIdeal_{A,B}(p, s_p).                 ▷ ||y|| small
if b is not prime. then
    Fail.
Let s_b = O(b).                                         ▷ ||s_b|| small
Return s_b · y ∈ p^{-1}.                                ▷ ||y · s_b|| small
```

$$\underbrace{s_{\mathfrak{b}}}_{\in \mathfrak{b}} \cdot \underbrace{y}_{\in (\mathfrak{b}\cdot\mathfrak{p})^{-1}} \in \mathfrak{p}^{-1}$$

Our $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

The oracle $\mathcal{O}$ solves $\mathrm{ideal\text{-}HSVP}$ for $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Input: An ideal $I = \mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform prime of norm in $[A, B]$.
Output: $x \in \mathfrak{p}^{-1} \setminus \{0\}$ small.

1: Let $s_{\mathfrak{p}} = \mathcal{O}(\mathfrak{p})$.
2: Let $(\mathfrak{b}, y) = \mathtt{SampleIdeal}_{A,B}(\mathfrak{p}, s_{\mathfrak{p}})$. ▷ $\|y\|$ *small*
3: **if** $\mathfrak{b}$ is not prime. **then**
4: Fail.
5: Let $s_{\mathfrak{b}} = \mathcal{O}(\mathfrak{b})$. ▷ $\|s_{\mathfrak{b}}\|$ *small*
6: **Return** $\underbrace{s_{\mathfrak{b}}}_{\in \mathfrak{b}} \cdot \underbrace{y}_{\in (\mathfrak{b}\cdot\mathfrak{p})^{-1}} \in \mathfrak{p}^{-1}$. ▷ $\|y \cdot s_{\mathfrak{b}}\|$ *small*

Our $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

The oracle $\mathcal{O}$ solves ideal-HSVP for $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Input: An ideal $I = \mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform prime of norm in $[A, B]$.
Output: $x \in \mathfrak{p}^{-1} \setminus \{0\}$ small.

1: Let $s_{\mathfrak{p}} = \mathcal{O}(\mathfrak{p})$.
2: Let $(\mathfrak{b}, y) = \texttt{SampleIdeal}_{A,B}(\mathfrak{p}, s_{\mathfrak{p}})$. ▷ *$\|y\|$ small*
3: **if** $\mathfrak{b}$ is not prime. **then**
4: Fail.
5: Let $s_{\mathfrak{b}} = \mathcal{O}(\mathfrak{b})$. ▷ *$\|s_{\mathfrak{b}}\|$ small*
6: **Return** $\underbrace{s_{\mathfrak{b}}}_{\in \mathfrak{b}} \cdot \underbrace{y}_{\in (\mathfrak{b}\cdot\mathfrak{p})^{-1}} \in \mathfrak{p}^{-1}$. ▷ *$\|y \cdot s_{\mathfrak{b}}\|$ small*

Our $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

The oracle $\mathcal{O}$ solves ideal-HSVP for $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Input: An ideal $I = \mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Output: $x \in \mathfrak{p}^{-1} \setminus \{0\}$ small.

1: Let $s_{\mathfrak{p}} = \mathcal{O}(\mathfrak{p})$.

2: Let $(\mathfrak{b}, y) = \texttt{SampleIdeal}_{A,B}(\mathfrak{p}, s_{\mathfrak{p}})$. ▷ *$\|y\|$ small*

3: **if** $\mathfrak{b}$ is not prime. **then**

4: Fail.

5: Let $s_{\mathfrak{b}} = \mathcal{O}(\mathfrak{b})$. ▷ *$\|s_{\mathfrak{b}}\|$ small*

6: **Return** $\underbrace{s_{\mathfrak{b}}}_{\in \mathfrak{b}} \cdot \underbrace{y}_{\in (\mathfrak{b}\cdot\mathfrak{p})^{-1}} \in \mathfrak{p}^{-1}$. ▷ *$\|y \cdot s_{\mathfrak{b}}\|$ small*

Our $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

The oracle $\mathcal{O}$ solves $\mathrm{ideal\text{-}HSVP}$ for $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Input: An ideal $I = \mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform prime of norm in $[A, B]$.
Output: $x \in \mathfrak{p}^{-1} \setminus \{0\}$ small.

1: Let $s_\mathfrak{p} = \mathcal{O}(\mathfrak{p})$.
2: Let $(\mathfrak{b}, y) = \mathtt{SampleIdeal}_{A,B}(\mathfrak{p}, s_\mathfrak{p})$. ▷ *$\|y\|$ small*
3: **if** $\mathfrak{b}$ is not prime. **then**
4: Fail.
5: Let $s_\mathfrak{b} = \mathcal{O}(\mathfrak{b})$. ▷ *$\|s_\mathfrak{b}\|$ small*
6: **Return** $\underbrace{s_\mathfrak{b}}_{\in \mathfrak{b}} \cdot \underbrace{y}_{\in (\mathfrak{b}\cdot\mathfrak{p})^{-1}} \in \mathfrak{p}^{-1}$. ▷ *$\|y \cdot s_\mathfrak{b}\|$ small*

Our $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

The oracle $\mathcal{O}$ solves ideal-HSVP for $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Input: An ideal $I = \mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform prime of norm in $[A, B]$.
Output: $x \in \mathfrak{p}^{-1} \setminus \{0\}$ small.

Let $s_{\mathfrak{p}} = \mathcal{O}(\mathfrak{p})$.
Let $(\mathfrak{b}, y) = \texttt{SampleIdeal}_{A,B}(\mathfrak{p}, s_{\mathfrak{p}})$. ▷ *$\|y\|$ small*
if $\mathfrak{b}$ is not prime. **then**
    Fail.
Let $s_{\mathfrak{b}} = \mathcal{O}(\mathfrak{b})$. ▷ *$\|s_{\mathfrak{b}}\|$ small*
Return $\underbrace{s_{\mathfrak{b}}}_{\in \mathfrak{b}} \cdot \underbrace{y}_{\in (\mathfrak{b}\cdot\mathfrak{p})^{-1}} \in \mathfrak{p}^{-1}$. ▷ *$\|y \cdot s_{\mathfrak{b}}\|$ small*

Wrapping up

Contributions and open problems

Contributions:

- Solving $\mathrm{ideal\text{-}HSVP}$ on average over inverses of primes is at least as hard as solving $\mathrm{ideal\text{-}HSVP}$ on average over primes.
- This gives an NTRU instance distribution with hardness based on $\mathrm{ideal\text{-}HSVP}$ for all ideals.

Contributions and open problems

Contributions:

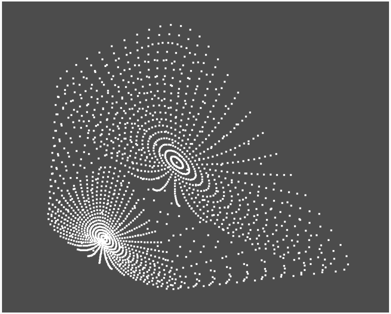
- Solving ideal-HSVP on average over inverses of primes is at least as hard as solving ideal-HSVP on average over primes.
- This gives an NTRU instance distribution with hardness based on ideal-HSVP for all ideals.

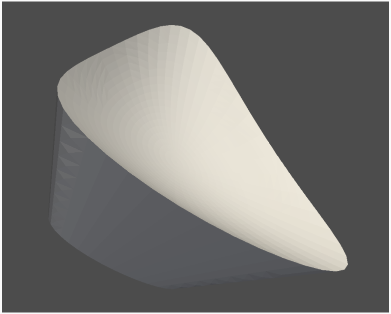
Open problems:

- Can we have such reduction without factoring?
- Can we get rid of the cost dependancy in ρ_K?
- Can we have more precise approximates for the running time?

Thank you for your attention

Any question?





References i

- K. de Boer, L. Ducas, A. Pellet-Mary, and B. Wesolowski, *Random self-reducibility of Ideal-SVP via Arakelov random walks*, CRYPTO, 2020.
- K. Boudgoust, E. Gachon, and A. Pellet-Mary, *Some easy instances of Ideal-SVP and implications on the partial Vandermonde knapsack problem*, CRYPTO, 2022.
- K. de Boer, *Random walks on arakelov class groups.*, Ph.D. thesis, Leiden University, 2022, Available on request from the author.
- R. Cramer, L. Ducas, C. Peikert, and O. Regev, *Recovering short generators of principal ideals in cyclotomic rings*, EUROCRYPT 2016, 2016.
- R. Cramer, L. Ducas, and B. Wesolowski, *Short Stickelberger class relations and application to Ideal-SVP*, EUROCRYPT, 2017.

References ii

- C. Gentry, *A fully homomorphic encryption scheme*, Ph.D. thesis, Stanford University, 2009.
- A. Pellet-Mary, G. Hanrot, and D. Stehlé, *Approx-SVP in ideal lattices with pre-processing*, EUROCRYPT, 2019.
- A. Pellet-Mary and D. Stehlé, *On the hardness of the NTRU problem*, ASIACRYPT, 2021.
- Quartl, *Matrix pattern qtl3*, 2014, File: `Matrix pattern qtl3.svg`.