Quantum Augmented Dual Attack

Martin R. Albrecht and Yixin Shen

Royal Holloway, Univesity of London

September 27, 2022

https://eprint.iacr.org/2022/656

Post-quantum cryptography

- isogeny-based
- multivariate
- code-based
- lattice-based: LWE

Learning with errors (LWE)

Let $n = 4$, $m = 6$ and $q = 17$.

secret

$$\underset{A \in \mathbb{Z}_q^{m \times n}}{\begin{bmatrix} 14 & 12 & 2 & 5 \\ 5 & 3 & 1 & 7 \\ 14 & 7 & 2 & 5 \\ 0 & 9 & 8 & 4 \\ 8 & 11 & 5 & 12 \\ 5 & 1 & 3 & 14 \end{bmatrix}} \times \underset{s \in \mathbb{Z}_q^{n}}{\begin{bmatrix} \\ \\ \\ \end{bmatrix}} = \underset{b \in \mathbb{Z}_q^{m}}{\begin{bmatrix} 11 \\ 5 \\ 14 \\ 6 \\ 12 \\ 13 \end{bmatrix}}$$

Given A and b, find s.

Learning with errors (LWE)

Let $n = 4$, $m = 6$ and $q = 17$.

secret

$$\underset{A \in \mathbb{Z}_q^{m \times n}}{\begin{bmatrix} 14 & 12 & 2 & 5 \\ 5 & 3 & 1 & 7 \\ 14 & 7 & 2 & 5 \\ 0 & 9 & 8 & 4 \\ 8 & 11 & 5 & 12 \\ 5 & 1 & 3 & 14 \end{bmatrix}} \times \underset{s \in \mathbb{Z}_q^{n}}{\begin{bmatrix} 1 \\ 2 \\ 1 \\ 5 \end{bmatrix}} = \underset{b \in \mathbb{Z}_q^{m}}{\begin{bmatrix} 11 \\ 5 \\ 14 \\ 6 \\ 12 \\ 13 \end{bmatrix}}$$

Given A and b, find s.

$\rightsquigarrow$ Very easy (e.g. Gaussian elimination) and in polynomial time

Learning with errors (LWE)

Let $n = 4$, $m = 6$ and $q = 17$.

$$
\begin{array}{ccccccc}
\textbf{random} & & \textbf{secret} & & \textbf{noise} & & \\
A \in \mathbb{Z}_q^{m \times n} & & s \in \mathbb{Z}_q^n & & e \in \mathbb{Z}_q^m & & b \in \mathbb{Z}_q^m \\
\begin{bmatrix} 14 & 12 & 2 & 5 \\ 5 & 3 & 1 & 7 \\ 14 & 7 & 2 & 5 \\ 0 & 9 & 8 & 4 \\ 8 & 11 & 5 & 12 \\ 5 & 1 & 3 & 14 \end{bmatrix} & \times & \begin{bmatrix} 1 \\ 2 \\ 1 \\ 5 \end{bmatrix} & + & \begin{bmatrix} -3 \\ -1 \\ 2 \\ -3 \\ 3 \\ -1 \end{bmatrix} & = & \begin{bmatrix} 11 \\ 5 \\ 14 \\ 6 \\ 12 \\ 13 \end{bmatrix}
\end{array}
$$

Learning with errors (LWE)

Let $n = 4$, $m = 6$ and $q = 17$.

random **secret** **noise**

$$\underset{A \in \mathbb{Z}_q^{m \times n}}{\begin{bmatrix} 14 & 12 & 2 & 5 \\ 5 & 3 & 1 & 7 \\ 14 & 7 & 2 & 5 \\ 0 & 9 & 8 & 4 \\ 8 & 11 & 5 & 12 \\ 5 & 1 & 3 & 14 \end{bmatrix}} \times \underset{s \in \mathbb{Z}_q^{n}}{\begin{bmatrix} \ \\ \ \\ \ \\ \ \end{bmatrix}} + \underset{e \in \mathbb{Z}_q^{m}}{\begin{bmatrix} \ \\ \ \\ \ \\ \ \\ \ \\ \ \end{bmatrix}} = \underset{b \in \mathbb{Z}_q^{m}}{\begin{bmatrix} 11 \\ 5 \\ 14 \\ 6 \\ 12 \\ 13 \end{bmatrix}}$$

Given A and b, find s.

$\leadsto$ Suspected hard problem, even for quantum algorithms

Learning with errors (LWE)

Let $n, m, q \in \mathbb{Z}$ and χ_e, χ_s two distributions over $\mathbb{Z}_q$.

LWE$(n, m, q, \chi_e, \chi_s)$: probability distribution on $\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$

- sample $A \leftarrow U(\mathbb{Z}_q^{m \times n})$
- sample $s \leftarrow \chi_s^n$
- sample $e \leftarrow \chi_e^m$
- output $(A, As + e)$.

Intuition: $As + e$ is very close to a uniform distribution.

Learning with errors (LWE)

Let $n, m, q \in \mathbb{Z}$ and χ_e, χ_s two distributions over $\mathbb{Z}_q$.

LWE$(n, m, q, \chi_e, \chi_s)$: probability distribution on $\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$

- sample $A \leftarrow U(\mathbb{Z}_q^{m \times n})$
- sample $s \leftarrow \chi_s^n$
- sample $e \leftarrow \chi_e^m$
- output $(A, As + e)$.

Intuition: $As + e$ is very close to a uniform distribution.

Search LWE problem: given $(A, b) \leftarrow$ LWE$(n, m, q, \chi_e, \chi_s)$, recover s.

Decision LWE problem:
distinguish LWE$(n, m, q, \chi_e, \chi_s)$ from $U(\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m)$.

Learning with errors (LWE)

Let $n, m, q \in \mathbb{Z}$ and χ_e, χ_s two distributions over $\mathbb{Z}_q$.

LWE$(n, m, q, \chi_e, \chi_s)$: probability distribution on $\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$

- sample $A \leftarrow U(\mathbb{Z}_q^{m \times n})$
- sample $s \leftarrow \chi_s^n$
- sample $e \leftarrow \chi_e^m$
- output $(A, As + e)$.

Intuition: $As + e$ is very close to a uniform distribution.

Search LWE problem: given $(A, b) \leftarrow$ LWE$(n, m, q, \chi_e, \chi_s)$, recover s.

Decision LWE problem:
distinguish LWE$(n, m, q, \chi_e, \chi_s)$ from $U(\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m)$.

Lemma: Search LWE is easy if and only if decision LWE is easy.

Learning with errors (LWE)

LWE$(n, m, q, \chi_e, \chi_s)$: probability distribution on $\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$

- sample $A \leftarrow U(\mathbb{Z}_q^{m\times n})$
- sample $s \leftarrow \chi_s^n$
- sample $e \leftarrow \chi_e^m$
- output $(A, As + e)$.

Learning with errors (LWE)

LWE(n, m, q, χ_e, χ_s): probability distribution on $\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$

- sample $A \leftarrow U(\mathbb{Z}_q^{m\times n})$
- sample $s \leftarrow \chi_s^n$
- sample $e \leftarrow \chi_e^m$
- output $(A, As + e)$.

Secret distributions χ_s:

- originally uniform in $\mathbb{Z}_q$
- now discrete Gaussian of small deviation σ_s (e.g. $\{-1, 0, 1\}$ whp)
- Fact: small secret is as hard as uniform secret
- small secret allows more efficient schemes

Learning with errors (LWE)

LWE$(n, m, q, \chi_e, \chi_s)$: probability distribution on $\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$

- sample $A \leftarrow U(\mathbb{Z}_q^{m\times n})$
- sample $s \leftarrow \chi_s^n$
- sample $e \leftarrow \chi_e^m$
- output $(A, As + e)$.

Noise distributions χ_e:

- usually discrete Gaussian of deviation σ_e
- encryption (Kyber/Saber): σ_e small (≈ 1)
- FHE: σ_e is larger

LWE: security and attacks

LWE is fundamental to lattice-based cryptography:

- several lattice-based NIST PQC candidates rely on LWE
- extensive literature
- all evidence points to resistance against quantum attacks

LWE: security and attacks

LWE is fundamental to lattice-based cryptography:

- several lattice-based NIST PQC candidates rely on LWE
- extensive literature
- all evidence points to resistance against quantum attacks

Two types of attacks:

- Primal attacks:
  - more efficient
  - no quantum speed-up known
- Dual attacks:
  - originally less efficient, now catching up
  - no quantum speed-up known up to now

Contribution: first significant quantum speed-up on dual attacks

Search to distinguish

Very naive attack:

$$\overset{A}{\begin{bmatrix} 8 & 9 & 10 & 12 \\ 5 & 3 & 11 & 3 \\ 0 & 15 & 10 & 4 \\ 15 & 9 & 16 & 15 \\ 1 & 2 & 10 & 8 \\ 11 & 16 & 13 & 9 \end{bmatrix}} \times \overset{s}{\begin{bmatrix} \ \\ \ \\ \ \\ \ \end{bmatrix}} + \overset{e}{\begin{bmatrix} \ \\ \ \\ \ \\ \ \\ \ \\ \ \end{bmatrix}} = \overset{b}{\begin{bmatrix} 8 \\ 3 \\ 11 \\ 15 \\ 2 \\ 15 \end{bmatrix}}$$

Attack:

- get (A, b)

Search to distinguish

Very naive attack: guess secret $\tilde{s}$

$$\overset{A}{\begin{bmatrix}8&9&10&12\\5&3&11&3\\0&15&10&4\\15&9&16&15\\1&2&10&8\\11&16&13&9\end{bmatrix}} \times \left(\overset{s}{\begin{bmatrix} \ \\ \ \\ \ \\ \ \end{bmatrix}} - \overset{\tilde{s}}{\begin{bmatrix}3\\11\\0\\14\end{bmatrix}} \right) + \overset{e}{\begin{bmatrix} \ \\ \ \\ \ \\ \ \\ \ \\ \ \end{bmatrix}} = \overset{b'}{\begin{bmatrix}7\\2\\3\\15\\11\\8\end{bmatrix}}$$

Attack:

- get (A, b)
- guess $\tilde{s}$
- output $b' = b - A\tilde{s}$

Search to distinguish

Very naive attack: guess secret $\tilde{s}$

$$\overset{A}{\begin{bmatrix} 8 & 9 & 10 & 12 \\ 5 & 3 & 11 & 3 \\ 0 & 15 & 10 & 4 \\ 15 & 9 & 16 & 15 \\ 1 & 2 & 10 & 8 \\ 11 & 16 & 13 & 9 \end{bmatrix}} \times \left(\overset{s}{\begin{bmatrix} \; \\ \; \\ \; \\ \; \end{bmatrix}} - \overset{\tilde{s}}{\begin{bmatrix} 3 \\ 11 \\ 0 \\ 14 \end{bmatrix}} \right) + \overset{e}{\begin{bmatrix} \; \\ \; \\ \; \\ \; \\ \; \\ \; \end{bmatrix}} = \overset{b'}{\begin{bmatrix} 7 \\ 2 \\ 3 \\ 15 \\ 11 \\ 8 \end{bmatrix}}$$

Attack:

- get (A, b)
- guess $\tilde{s}$
- output $b' = b - A\tilde{s}$

Good guess ($s = \tilde{s}$):

$$b' = e$$

follows a discrete Gaussian
of small deviation

Search to distinguish

Very naive attack: guess secret $\tilde{s}$

$$\overset{A}{\begin{bmatrix} 8 & 9 & 10 & 12 \\ 5 & 3 & 11 & 3 \\ 0 & 15 & 10 & 4 \\ 15 & 9 & 16 & 15 \\ 1 & 2 & 10 & 8 \\ 11 & 16 & 13 & 9 \end{bmatrix}} \times \left(\overset{s}{\begin{bmatrix} \ \\ \ \\ \ \\ \ \end{bmatrix}} - \overset{\tilde{s}}{\begin{bmatrix} 3 \\ 11 \\ 0 \\ 14 \end{bmatrix}} \right) + \overset{e}{\begin{bmatrix} \ \\ \ \\ \ \\ \ \\ \ \\ \ \end{bmatrix}} = \overset{b'}{\begin{bmatrix} 7 \\ 2 \\ 3 \\ 15 \\ 11 \\ 8 \end{bmatrix}}$$

Attack:

- get (A, b)
- guess $\tilde{s}$
- output $b' = b - A\tilde{s}$

Good guess ($s = \tilde{s}$):

$$b' = e$$

follows a discrete Gaussian of small deviation

Bad guess ($s \neq \tilde{s}$):

$$b' = e + A(s - \tilde{s})$$

follows a uniform[1] distribution (A uniform in $\mathbb{Z}_q^{m \times n}$)

[1] Technically only true for fixed s, random A and $\tilde{s}$

Uniform/Gaussian distinguisher

Given a sampler for χ, decide if $\chi = U(\mathbb{Z}_q)$ or $D_{\sigma,q}$ (discrete Gaussian)

Uniform/Gaussian distinguisher

Given a sampler for χ, decide if $\chi = U(\mathbb{Z}_q)$ or $D_{\sigma,q}$ (discrete Gaussian)

Essentially optimal distingusher: use FFT transform

$$\mathbb{E}_{x\leftarrow\chi}[e^{2i\pi x/q}], \mathrm{Var}_{x\leftarrow\chi}[e^{2i\pi x/q}] \approx \begin{cases} 0, 0 & \text{if } \chi = U(\mathbb{Z}_q) \\ e^{-2\left(\frac{\pi\sigma}{q}\right)^2}, e^{-8\left(\frac{\pi\sigma}{q}\right)^2} & \text{if } \chi = D_{\sigma,q} \end{cases}$$

Uniform/Gaussian distinguisher

Given a sampler for χ, decide if $\chi = U(\mathbb{Z}_q)$ or $D_{\sigma,q}$ (discrete Gaussian)

Essentially optimal distingusher: use FFT transform

$$\mathbb{E}_{x\leftarrow\chi}[e^{2i\pi x/q}], \mathrm{Var}_{x\leftarrow\chi}[e^{2i\pi x/q}] \approx \begin{cases} 0, 0 & \text{if } \chi = U(\mathbb{Z}_q) \\ e^{-2\left(\frac{\pi\sigma}{q}\right)^2}, e^{-8\left(\frac{\pi\sigma}{q}\right)^2} & \text{if } \chi = D_{\sigma,q} \end{cases}$$

Attack:

- sample $N = \Omega\left(1/\varepsilon^2\right)$ values $x_1, \ldots, x_N$ from χ
- compute
$$S = \frac{1}{N}\sum_{j=1}^{N} e^{2i\pi x_j/q}$$
- Check if $S > e^{-2\left(\frac{\pi\sigma}{q}\right)^2}$

The quantity $\varepsilon = e^{-2\left(\frac{\pi\sigma}{q}\right)^2}$ is called the advantage.

Very naive attack: summary

Very naive attack:

- guess $\tilde{s}$: deviation of s is σ_s so in $\{-\sigma_s, \dots, \sigma_s\}^n$ whp
- compute $1/\varepsilon^2$ samples to check guess

Very naive attack: summary

Very naive attack:

- guess $\tilde{s}$: deviation of s is σ_s so in $\{-\sigma_s, \ldots, \sigma_s\}^n$ whp
- compute $1/\varepsilon^2$ samples to check guess

Complexity estimate:

$$(2\sigma_s)^n \cdot e^{4\left(\frac{\pi\sigma_e}{q}\right)^2} = \text{too much}$$

Very naive attack: summary

Very naive attack:

- guess $\tilde{s}$: deviation of s is σ_s so in $\{-\sigma_s, \dots, \sigma_s\}^n$ whp
- compute $1/\varepsilon^2$ samples to check guess

Complexity estimate:

$$(2\sigma_s)^n \cdot e^{4\left(\frac{\pi\sigma_e}{q}\right)^2} = \text{too much}$$

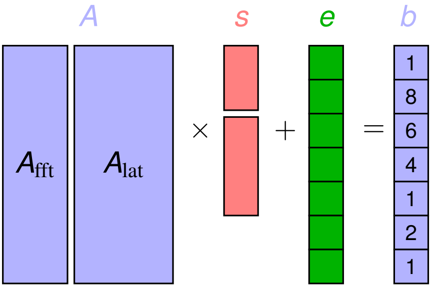
Dual attacks: provide an efficient way to only guess a part of the secret

Search to Decision LWE

$$\overset{A}{\begin{bmatrix} 3 & 7 & 2 & 3 & 6 \\ 4 & 1 & 5 & 8 & 4 \\ 1 & 8 & 1 & 8 & 1 \\ 5 & 2 & 5 & 6 & 0 \\ 2 & 1 & 6 & 3 & 0 \\ 8 & 2 & 7 & 3 & 6 \\ 5 & 5 & 6 & 6 & 2 \end{bmatrix}} \times \overset{s}{\begin{bmatrix} \ \\ \ \\ \ \\ \ \\ \ \end{bmatrix}} + \overset{e}{\begin{bmatrix} \ \\ \ \\ \ \\ \ \\ \ \\ \ \\ \ \end{bmatrix}} = \overset{b}{\begin{bmatrix} 1 \\ 8 \\ 6 \\ 4 \\ 1 \\ 2 \\ 1 \end{bmatrix}}$$

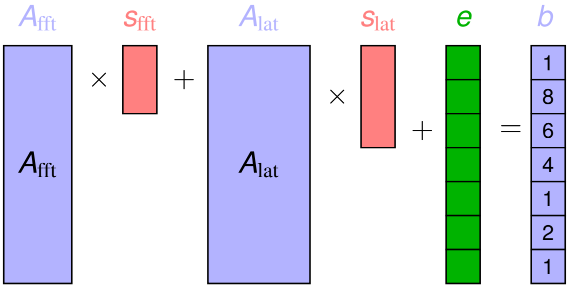
Search to Decision LWE

Split secret: $n = k_{\text{fft}} + k_{\text{lat}}$



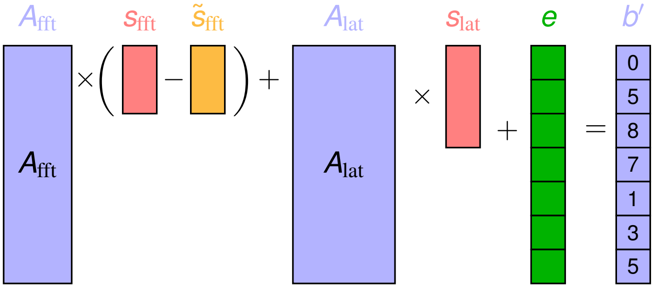
Search to Decision LWE

Split secret: $n = k_{\mathrm{fft}} + k_{\mathrm{lat}}$



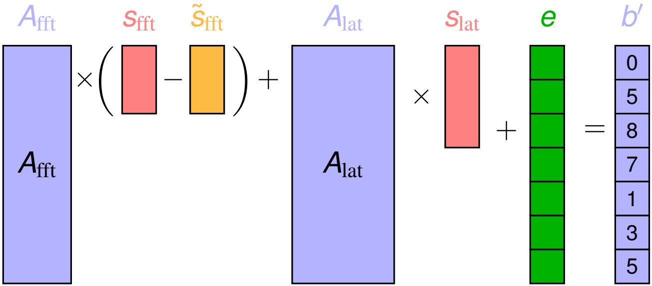
Search to Decision LWE

Split secret: $n = k_{\text{fft}} + k_{\text{lat}}$, guess $\tilde{s}_{\text{fft}}$, output $(A_{\text{lat}}, b' = b - A_{\text{fft}}\tilde{s}_{\text{fft}})$



Search to Decision LWE

Split secret: $n = k_{\text{fft}} + k_{\text{lat}}$, guess $\tilde{s}_{\text{fft}}$, output $(A_{\text{lat}}, b' = b - A_{\text{fft}}\tilde{s}_{\text{fft}})$



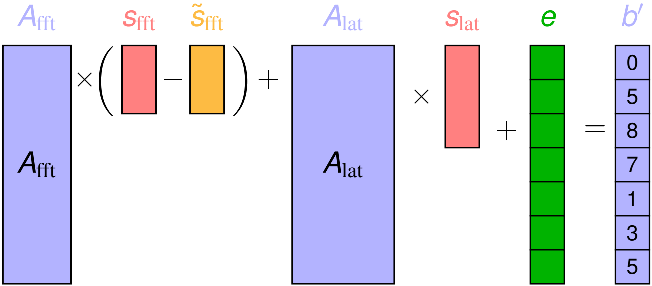
Good guess ($s_{\text{fft}} = \tilde{s}_{\text{fft}}$):

$$b' = A_{\text{lat}} s_{\text{lat}} + e$$

so (A_{lat}, b') follows an LWE distribution

Search to Decision LWE

Split secret: $n = k_{\text{fft}} + k_{\text{lat}}$, guess $\tilde{s}_{\text{fft}}$, output $(A_{\text{lat}}, b' = b - A_{\text{fft}}\tilde{s}_{\text{fft}})$



Good guess ($s_{\text{fft}} = \tilde{s}_{\text{fft}}$):

$$b' = A_{\text{lat}} s_{\text{lat}} + e$$

so (A_{lat}, b') follows an LWE distribution

Bad guess ($s_{\text{fft}} \neq \tilde{s}_{\text{fft}}$):

$$b' = A_{\text{fft}}(s_{\text{fft}} - \tilde{s}_{\text{fft}}) + \cdots$$

so (A_{lat}, b') follows a uniform distribution (A_{fft} uniform)

Uniform/LWE distinguisher

Given a sampler for χ, decide if $\chi =$ uniform or LWE.

Uniform/LWE distinguisher

Given a sampler for χ, decide if $\chi =$ uniform or LWE.

- sample (A, b) from χ
- compute $x \in \mathbb{Z}_q^m$ such that $x^T A = 0$
- output $x^T b$

$$\boxed{\quad x^T \quad} \times \boxed{b}$$

Uniform/LWE distinguisher

Given a sampler for χ, decide if $\chi =$ uniform or LWE.

- sample (A, b) from χ
- compute $x \in \mathbb{Z}_q^m$ such that $x^T A = 0$
- output $x^T b$

$$x^T \times b = x^T \times \left(A \times s + e \right) = x^T \times e$$

Uniform/LWE distinguisher

Given a sampler for χ, decide if $\chi =$ uniform or LWE.

- sample (A, b) from χ
- compute $x \in \mathbb{Z}_q^m$ such that $x^T A = 0$
- output $x^T b$

$$x^T \times b = x^T \times \left(A \times s + e \right) = x^T \times e$$

When $\chi =$ LWE:

$$x^T b = x^T e$$

follows an approximate
Gaussian distribution

Uniform/LWE distinguisher

Given a sampler for χ, decide if $\chi =$ uniform or LWE.

- sample (A, b) from χ
- compute $x \in \mathbb{Z}_q^m$ such that $x^T A = 0$
- output $x^T b$

$$x^T \times b = x^T \times \left(A \times s + e \right) = x^T \times e$$

When $\chi =$ LWE:

$$x^T b = x^T e$$

follows an approximate Gaussian distribution

When $\chi =$ Uniform:

$$x^T b$$

follows a uniform distribution (b uniform, independent from A)

Dual attack: summary

Basic dual attack:

- split secret $n = k_{\text{fft}} + k_{\text{lat}}$
- guess $\tilde{s}_{\text{fft}}$, subtract guess
- compute dual vector x and dot product $x^T b$
- compute $1/\varepsilon^2$ samples to check guess

Dual attack: summary

Basic dual attack:

- split secret $n = k_{\text{fft}} + k_{\text{lat}}$
- guess $\tilde{s}_{\text{fft}}$, subtract guess
- compute dual vector x and dot product $x^T b$
- compute $1/\varepsilon^2$ samples to check guess

What is ε ?

- e approx Gaussian deviation σ_e
- $x^T b = x^T e$ approx Gaussian deviation $\|x\|\sigma_e$

Dual attack: summary

Basic dual attack:

- split secret $n = k_{\mathrm{fft}} + k_{\mathrm{lat}}$
- guess $\tilde{s}_{\mathrm{fft}}$, subtract guess
- compute dual vector x and dot product $x^T b$
- compute $1/\varepsilon^2$ samples to check guess

What is ε ?

- e approx Gaussian deviation σ_e
- $x^T b = x^T e$ approx Gaussian deviation $\|x\|\sigma_e$

Complexity estimate:

$$(2\sigma_s)^{k_{\mathrm{fft}}} \cdot e^{4\left(\frac{\pi\|x\|\sigma_e}{q}\right)^2} \cdot (\text{time to compute } x)$$

Dual attack: summary

Basic dual attack:

- split secret $n = k_{\text{fft}} + k_{\text{lat}}$
- guess $\tilde{s}_{\text{fft}}$, subtract guess
- compute dual vector x and dot product $x^T b$
- compute $1/\varepsilon^2$ samples to check guess

What is ε ?

- e approx Gaussian deviation σ_e
- $x^T b = x^T e$ approx Gaussian deviation $\|x\|\sigma_e$

Complexity estimate:

$$(2\sigma_s)^{k_{\text{fft}}} \cdot e^{4\left(\frac{\pi\|x\|\sigma_e}{q}\right)^2} \cdot (\text{time to compute } x)$$

$\leadsto$ we want x to be short

Dual attack: summary

Basic dual attack:

- split secret $n = k_{\text{fft}} + k_{\text{lat}}$
- guess $\tilde{s}_{\text{fft}}$, subtract guess
- compute dual vector x and dot product $x^T b$
- compute $1/\varepsilon^2$ samples to check guess

What is ε ?

- e approx Gaussian deviation σ_e
- $x^T b = x^T e$ approx Gaussian deviation $\|x\|\sigma_e$

Complexity estimate:

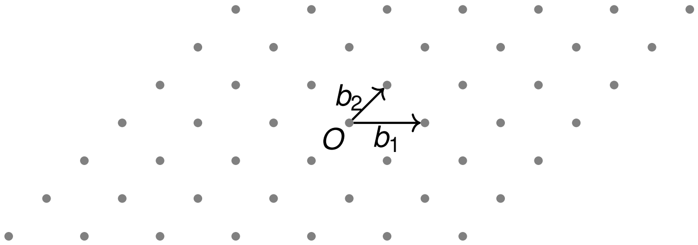
$$(2\sigma_s)^{k_{\text{fft}}} \cdot e^{4\left(\frac{\pi\|x\|\sigma_e}{q}\right)^2} \cdot (\text{time to compute } x)$$

$\leadsto$ we want x to be short $\leadsto$ lattice reduction

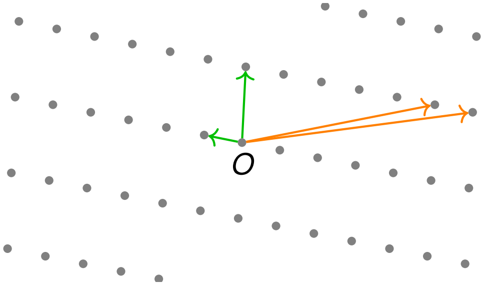
What is a (Euclidean) lattice?

Definition

$\mathcal{L}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n) = \left\{ \sum_{i=1}^{n} x_i \boldsymbol{b}_i : x_i \in \mathbb{Z} \right\}$ where $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ is a basis of $\mathbb{R}^n$.

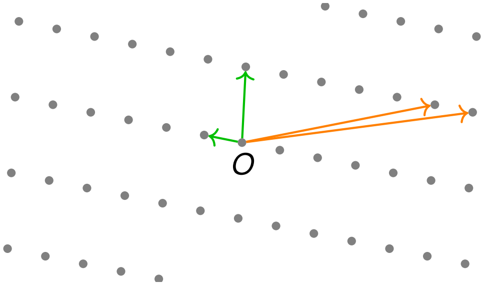


Lattice-based cryptography: fundamental idea



- good basis: private information, makes problem easy
- bad basis: public information, makes problem hard

Lattice-based cryptography: fundamental idea



- good basis: private information, makes problem easy
- bad basis: public information, makes problem hard

Basis reduction: transform a bad basis into a good one
Main tool: BKZ algorithm and its variants

Requires to solve the (approx-)SVP problem in smaller dimensions.

Dual attack: summary 2

Basic dual attack:

- split secret $n = k_{\text{fft}} + k_{\text{lat}}$
- guess $\tilde{s}_{\text{fft}}$, subtract guess
- compute dual vector x and dot product $x^T b$
- compute $1/\varepsilon^2$ samples to check guess

Dual attack: summary 2

Basic dual attack:

- split secret $n = k_{\mathrm{fft}} + k_{\mathrm{lat}}$
- guess $\tilde{s}_{\mathrm{fft}}$, subtract guess
- compute dual vector x and dot product $x^T b$
- compute $1/\varepsilon^2$ samples to check guess

Pick x short in lattice L using BKZ:

$$L = \left\{ x \in \mathbb{Z}^m : x^T A_{\mathrm{lat}} = 0 \bmod q \right\}$$

Complexity estimate:

$$(2\sigma_s)^{k_{\mathrm{fft}}} \cdot e^{4\left(\frac{\pi \|x\| \sigma_e}{q}\right)^2} \cdot T_{\mathrm{BKZ}}$$

Dual attack: summary 2

Basic dual attack:

- split secret $n = k_{\text{fft}} + k_{\text{lat}}$
- guess $\tilde{s}_{\text{fft}}$, subtract guess
- compute dual vector x and dot product $x^T b$
- compute $1/\varepsilon^2$ samples to check guess

Pick x short in lattice L using BKZ:

$$L = \left\{ x \in \mathbb{Z}^m : x^T A_{\text{lat}} = 0 \bmod q \right\}$$

Complexity estimate:

$$(2\sigma_s)^{k_{\text{fft}}} \cdot e^{4\left(\frac{\pi \|x\| \sigma_e}{q}\right)^2} \cdot T_{\text{BKZ}}$$

- BKZ trade-off short $x \rightsquigarrow$ more expensive algorithm
- best dual attack parameters ($k_{\text{fft}}, \ldots$) found by optimization

Advanced dual attacks

More details:

- modulo switching: only guess part of secret modulo p ($p \ll q$)
  - reduce guessing complexity
  - increase distinguishing cost due to modulo remainders
  - makes reduced secret dense

Advanced dual attacks

More details:

- modulo switching: only guess part of secret modulo p ($p \ll q$)
  - reduce guessing complexity
  - increase distinguishing cost due to modulo remainders
  - makes reduced secret dense
- split secret into three parts: brute force guess, FFT, lattice
  - leverage sparse secret before modulo reduction
  - decrease BKZ dimension and cost

Advanced dual attacks

More details:

- modulo switching: only guess part of secret modulo p ($p \ll q$)
  - reduce guessing complexity
  - increase distinguishing cost due to modulo remainders
  - makes reduced secret dense
- split secret into three parts: brute force guess, FFT, lattice
  - leverage sparse secret before modulo reduction
  - decrease BKZ dimension and cost
- BKZ with sieving to obtain many dual vectors at once

Real dual attack at the high-level

All you need to know for what follows: attack looks like

- enumerate $s_{\text{enum}} \in \mathbb{Z}_q^{k_{\text{enum}}}$ sampled from $\chi_s^{k_{\text{enum}}}$
  - enumerate all $s_{\text{fft}} \in \mathbb{Z}_q^{k_{\text{fft}}}$ uniform in $\mathbb{Z}_q^{k_{\text{fft}}}$
    - compute an FFT-like sum
    - check if it is above the threshold

Real dual attack at the high-level

All you need to know for what follows: attack looks like

- enumerate $s_{\text{enum}} \in \mathbb{Z}_q^{k_{\text{enum}}}$ — sampled from $\chi_s^{k_{\text{enum}}}$
  - enumerate all $s_{\text{fft}} \in \mathbb{Z}_q^{k_{\text{fft}}}$ — uniform in $\mathbb{Z}_q^{k_{\text{fft}}}$
    - compute an FFT-like sum
    - check if it is above the threshold

Classical complexity:

- guessing complexity: try s_{enum} in decreasing order of probability
- FFT: compute all FFT-sums in one go with a DFT

gives

$$G(\chi_s^{k_{\text{enum}}}) \cdot q^{k_{\text{fft}}}$$

Real dual attack at the high-level

All you need to know for what follows: attack looks like

- enumerate $s_{\text{enum}} \in \mathbb{Z}_q^{k_{\text{enum}}}$ sampled from $\chi_s^{k_{\text{enum}}}$
  - enumerate all $s_{\text{fft}} \in \mathbb{Z}_q^{k_{\text{fft}}}$ uniform in $\mathbb{Z}_q^{k_{\text{fft}}}$
    - compute an FFT-like sum
    - check if it is above the threshold

Classical complexity:

- guessing complexity: try s_{enum} in decreasing order of probability
- FFT: compute all FFT-sums in one go with a DFT

gives

$$G(\chi_s^{k_{\text{enum}}}) \cdot q^{k_{\text{fft}}}$$

Quantum complexity: can we hope for $\sqrt{G(\chi_s^{k_{\text{enum}}}) \cdot q^{k_{\text{fft}}}}$?

Real dual attack at the high-level

All you need to know for what follows: attack looks like

- enumerate $s_{\text{enum}} \in \mathbb{Z}_q^{k_{\text{enum}}}$ sampled from $\chi_s^{k_{\text{enum}}}$
  - enumerate all $s_{\text{fft}} \in \mathbb{Z}_q^{k_{\text{fft}}}$ uniform in $\mathbb{Z}_q^{k_{\text{fft}}}$
    - compute an FFT-like sum
    - check if it is above the threshold

Classical complexity:

- guessing complexity: try s_{enum} in decreasing order of probability
- FFT: compute all FFT-sums in one go with a DFT

gives

$$G(\chi_s^{k_{\text{enum}}}) \cdot q^{k_{\text{fft}}}$$

Quantum complexity: can we hope for $\sqrt{G(\chi_s^{k_{\text{enum}}}) \cdot q^{k_{\text{fft}}}}$? Probably not

Guessing complexity

D discrete distribution on $x_1, x_2, \ldots$, let p_i be the probability of x_i.

Guessing game: your friend samples $X \leftarrow D$, you must find i such that $X = x_i$ only by asking queries of the form "is $X = x_j$?" for some j.
Minimize (expected) number of queries.

Guessing complexity

D discrete distribution on $x_1, x_2, \ldots$, let p_i be the probability of x_i.

Guessing game: your friend samples $X \leftarrow D$, you must find i such that $X = x_i$ only by asking queries of the form "is $X = x_j$?" for some j. Minimize (expected) number of queries.

Example: $D = U(\mathbb{Z}_5)$

Friend samples $X = 3$

- is X equal to 1 ? No
- is X equal to 4 ? No
- is X equal to 5 ? No
- is X equal to 3 ? Yes

4 queries

For uniform the query order does not matter

Guessing complexity

D discrete distribution on $x_1, x_2, \ldots$, let p_i be the probability of x_i.

Guessing game: your friend samples $X \leftarrow D$, you must find i such that $X = x_i$ only by asking queries of the form "is $X = x_j$?" for some j.
Minimize (expected) number of queries.

Example: $D = U(\mathbb{Z}_5)$

Friend samples $X = 3$

- is X equal to 1 ? No
- is X equal to 4 ? No
- is X equal to 5 ? No
- is X equal to 3 ? Yes

4 queries

For uniform the query order does not matter

Example: $p_1 = 0.9$, $p_2 = 0.09$, $p_3 = 0.009$, $p_4 = 0.001$

Friend samples $X = 1$ (most likely)

- is X equal to 1 ? Yes

1 query

Guessing complexity

D discrete distribution on $x_1, x_2, \ldots$, let p_i be the probability of x_i.

Guessing game: your friend samples $X \leftarrow D$, you must find i such that $X = x_i$ only by asking queries of the form "is $X = x_j$?" for some j.
Minimize (expected) number of queries.

Example: $D = U(\mathbb{Z}_5)$

Friend samples $X = 3$

- is X equal to 1 ? No
- is X equal to 4 ? No
- is X equal to 5 ? No
- is X equal to 3 ? Yes

4 queries

For uniform the query order does not matter

Example: $p_1 = 0.9$, $p_2 = 0.09$, $p_3 = 0.009$, $p_4 = 0.001$

Friend samples $X = 4$ (unlikely)

- is X equal to 1 ? No
- is X equal to 2 ? No
- is X equal to 3 ? No
- is X equal to 4 ? Yes

4 queries

Guessing complexity

D discrete distribution on $x_1, x_2, \ldots$, let p_i be the probability of x_i.

Guessing game: your friend samples $X \leftarrow D$, you must find i such that $X = x_i$ only by asking queries of the form "is $X = x_j$?" for some j.
Minimize (expected) number of queries.

Example: $D = U(\mathbb{Z}_5)$

Friend samples $X = 3$

- is X equal to 1 ? No
- is X equal to 4 ? No
- is X equal to 5 ? No
- is X equal to 3 ? Yes

4 queries

For uniform the query order does not matter

Example: $p_1 = 0.9$, $p_2 = 0.09$, $p_3 = 0.009$, $p_4 = 0.001$

Friend samples $X = 4$ (unlikely)

- is X equal to 1 ? No
- is X equal to 2 ? No
- is X equal to 3 ? No
- is X equal to 4 ? Yes

4 queries

Ask most likely elements first

Guessing complexity

D discrete distribution on $x_1, x_2, \ldots$, let p_i be the probability of x_i.

Guessing game: your friend samples $X \leftarrow D$, you must find i such that $X = x_i$ only by asking queries of the form "is $X = x_j$?" for some j.
Minimize (expected) number of queries.

Optimal strategy: always guess elements by decreasing probability

Expected number of guesses ($p_1 \geqslant p_2 \geqslant \cdots \geqslant p_N$):

$$G(D) = \sum_{i=1}^{N} i \cdot p_i,$$

Guessing complexity

D discrete distribution on $x_1, x_2, \ldots$, let p_i be the probability of x_i.

Guessing game: your friend samples $X \leftarrow D$, you must find i such that $X = x_i$ only by asking queries of the form "is $X = x_j$?" for some j.
Minimize (expected) number of queries.

Optimal strategy: always guess elements by decreasing probability

Expected number of guesses ($p_1 \geqslant p_2 \geqslant \cdots \geqslant p_N$):

$$G(D) = \sum_{i=1}^{N} i \cdot p_i,$$

What about quantum guessing?

- Grover-like search
- can even handle faulty query oracles

Guessing complexity

D discrete distribution on $x_1, x_2, \ldots$, let p_i be the probability of x_i.

Guessing game: your friend samples $X \leftarrow D$, you must find i such that $X = x_i$ only by asking queries of the form "is $X = x_j$?" for some j.
Minimize (expected) number of queries.

Optimal strategy: always guess elements by decreasing probability

Expected number of guesses ($p_1 \geqslant p_2 \geqslant \cdots \geqslant p_N$):

$$G(D) = \sum_{i=1}^{N} i \cdot p_i, \qquad G^{qc}(D) = \sum_{i=1}^{N} \sqrt{i} \cdot p_i$$

What about quantum guessing?

- Grover-like search
- can even handle faulty query oracles

Guessing complexity (results)

Guessing complexity of the modular discrete Gaussian $D_{\sigma,q,n}$ on $\mathbb{Z}_q^n$:

$$D_{\sigma,q,n}(x) \propto \rho_\sigma(x + q\,\mathbb{Z}^n), \qquad \rho_\sigma(y) = e^{-\|y\|^2/2\sigma}, \quad y \in \mathbb{Z}^n\,.$$

Guessing complexity (results)

Guessing complexity of the modular discrete Gaussian $D_{\sigma,q,n}$ on $\mathbb{Z}_q^n$:

$$D_{\sigma,q,n}(x) \propto \rho_\sigma(x + q\,\mathbb{Z}^n), \qquad \rho_\sigma(y) = e^{-\|y\|^2/2\sigma}, \quad y \in \mathbb{Z}^n\,.$$

Theorem (Simplified)

$$G(D_{\sigma,q,n}) \lesssim 1.22^n \cdot 2^H, \qquad G^{qc}(D_{\sigma,q,n}) \lesssim 1.12^{n/2} \cdot 2^{H/2}$$

where $H \approx \frac{1/2+\log(\sigma\sqrt{2\pi})}{\log 2}$ *is the entropy of the discrete Gaussian.*

Observations:

- G exponentially times bigger than 2^H
- $G^{qc} \leqslant \sqrt{G}$ is true for any distributions
- G^{qc} seems exponentially smaller than $\sqrt{G}$ ...
- ... but we do not have matching lower bounds to confirm it yet

FFT search with threshold

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

FFT search with threshold

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\mathrm{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\mathrm{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

Naive complexity:

$$O(q^{k_{\mathrm{fft}}} \cdot N)$$

FFT search with threshold

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

Naive complexity:

$$O(q^{k_{\text{fft}}} \cdot N)$$

Classical algorithm with optimisation:

- $T \leftarrow k$-dimensional array set to zero
- $T[x_j] \leftarrow 1$ for all j
- compute FFT $\widehat{T}$ of T (Fact: $\widehat{T}[s] = F_s$)
- check all $\widehat{T}[s]$ against threshold

FFT search with threshold

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

Naive complexity:

$$O(q^{k_{\text{fft}}} \cdot N)$$

Classical algorithm with optimisation:

- $T \leftarrow k$-dimensional array set to zero
- $T[x_j] \leftarrow 1$ for all j
- compute FFT $\widehat{T}$ of T (Fact: $\widehat{T}[s] = F_s$)
- check all $\widehat{T}[s]$ against threshold

Complexity:

array filling time $+$ FFT time $+$ search time $= O(N + q^{k_{\text{fft}}}) = O(q^{k_{\text{fft}}})$

FFT search with threshold

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

What about quantum? initial idea: use the QFT

FFT search with threshold

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\mathrm{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\mathrm{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

What about quantum? initial idea: use the QFT

- create superposition

$$\psi = \frac{1}{\sqrt{N}} \sum_{j=1}^{N} |x_j\rangle$$

FFT search with threshold

Fundamental operation: given samples $x_1, \dots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

What about quantum? initial idea: use the QFT

- create superposition

$$\psi = \frac{1}{\sqrt{N}} \sum_{j=1}^{N} |x_j\rangle$$

- apply QFT to get

$$\hat{\psi} = \frac{1}{\sqrt{N}} \sum_{s \in \mathbb{Z}_q^k}^{N} F_s \, |s\rangle$$

FFT search with threshold

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

What about quantum? initial idea: use the QFT

- create superposition

$$\psi = \frac{1}{\sqrt{N}} \sum_{j=1}^{N} |x_j\rangle$$

- apply QFT to get

$$\hat{\psi} = \frac{1}{\sqrt{N}} \sum_{s \in \mathbb{Z}_q^k}^{N} F_s \, |s\rangle$$

- check if any amplitude in the superposition is above the threshold

FFT search with threshold

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

What about quantum? initial idea: use the QFT

- create superposition ▶ impossible without QRAM?

$$\psi = \frac{1}{\sqrt{N}} \sum_{j=1}^{N} |x_j\rangle$$

- apply QFT to get

$$\hat{\psi} = \frac{1}{\sqrt{N}} \sum_{s \in \mathbb{Z}_q^k}^{N} F_s \, |s\rangle$$

- check if any amplitude in the superposition is above the threshold

FFT search with threshold

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

What about quantum? initial idea: use the QFT

- create superposition ► impossible without QRAM?

$$\psi = \frac{1}{\sqrt{N}} \sum_{j=1}^{N} |x_j\rangle$$

- apply QFT to get ► polynomial time

$$\hat{\psi} = \frac{1}{\sqrt{N}} \sum_{s \in \mathbb{Z}_q^k}^{N} F_s |s\rangle$$

- check if any amplitude in the superposition is above the threshold

FFT search with threshold

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

What about quantum? initial idea: use the QFT

- create superposition ▶ impossible without QRAM?

$$\psi = \frac{1}{\sqrt{N}} \sum_{j=1}^{N} |x_j\rangle$$

- apply QFT to get ▶ polynomial time

$$\hat{\psi} = \frac{1}{\sqrt{N}} \sum_{s \in \mathbb{Z}_q^k}^{N} F_s \, |s\rangle$$

- check if any amplitude in the superposition is above the threshold ▶ extremely expensive?

Open question: can this approach be made efficient?

FFT search with threshold (quantum)

Fundamental operation: given samples $x_1, \dots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

Alternative quantum algorithm:

FFT search with threshold (quantum)

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

Alternative quantum algorithm:

- search over $s \in \mathbb{Z}_q^{k_{\text{fft}}}$ with Grover for...
  - compute F_s and check against threshold

Complexity: $O(\sqrt{q^{k_{\text{fft}}}} \cdot N)$

FFT search with threshold (quantum)

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

Alternative quantum algorithm:

- search over $s \in \mathbb{Z}_q^{k_{\text{fft}}}$ with Grover for...
  - compute F_s and check against threshold

Complexity: $O(\sqrt{q^{k_{\text{fft}}}} \cdot N)$ ▶ worse than classical unless N is very small!

FFT search with threshold (quantum)

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

Alternative quantum algorithm:

- search over $s \in \mathbb{Z}_q^{k_{\text{fft}}}$ with Grover for...
  - compute F_s and check against threshold

Complexity: $O(\sqrt{q^{k_{\text{fft}}}} \cdot N)$ ► worse than classical unless N is very small!

Theorem (Simplified)

There is a quantum algorithm that computes $F_s \pm \varepsilon$ given oracle access by making $O(1/\varepsilon)$ queries to $\mathcal{O}_X$:

$$\mathcal{O}_X : |j\rangle\,|0\rangle \to |j\rangle\,\left|x_j\right\rangle.$$

FFT search with threshold (quantum)

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

Alternative quantum algorithm:

- search over $s \in \mathbb{Z}_q^{k_{\text{fft}}}$ with Grover for...
  - compute F_s and check against threshold

Complexity: $O(\sqrt{q^{k_{\text{fft}}}} \cdot N)$ ► worse than classical unless N is very small!

Theorem (Simplified)

There is a quantum algorithm that computes $F_s \pm \varepsilon$ given oracle access by making $O(1/\varepsilon)$ queries to $\mathcal{O}_X$:

$$\mathcal{O}_X : |j\rangle\,|0\rangle \to |j\rangle\,\left|x_j\right\rangle .$$

How can we build such an oracle?

FFT search with threshold (quantum)

Fundamental operation: given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^{k_{\text{fft}}}$ (N large)

- enumerate all $s \in \mathbb{Z}_q^{k_{\text{fft}}}$
  - compute an FFT sum $F_s = \sum_{j=1}^{N} e^{2i\pi s^T x_j / q}$
  - check if $F_s \geqslant$ threshold

Alternative quantum algorithm:

- search over $s \in \mathbb{Z}_q^{k_{\text{fft}}}$ with Grover for...
  - compute F_s and check against threshold

Complexity: $O(\sqrt{q^{k_{\text{fft}}}} \cdot N)$ ► worse than classical unless N is very small!

Theorem (Simplified)

There is a quantum algorithm that computes $F_s \pm \varepsilon$ given oracle access by making $O(1/\varepsilon)$ queries to $\mathcal{O}_X$:

$$\mathcal{O}_X : |j\rangle\,|0\rangle \to |j\rangle\,\left|x_j\right\rangle .$$

How can we build such an oracle? $\rightsquigarrow$ QRAM

FFT search with threshold (quantum cont.)

Given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^k$

- put samples in a QRAM $\mathcal{O}_X$
- search over $s \in \mathbb{Z}_q^k$ with Grover for...
  - compute F_s using theorem with $\mathcal{O}_X$ and check against threshold

Theorem (Simplified)

There is a quantum algorithm that computes $F_s \pm \varepsilon$ given oracle access by making $O(1/\varepsilon)$ queries to $\mathcal{O}_X$.

FFT search with threshold (quantum cont.)

Given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^k$

- put samples in a QRAM $\mathcal{O}_X$
- search over $s \in \mathbb{Z}_q^k$ with Grover for...
  - compute F_s using theorem with $\mathcal{O}_X$ and check against threshold

Theorem (Simplified)

There is a quantum algorithm that computes $F_s \pm \varepsilon$ given oracle access by making $O(1/\varepsilon)$ queries to $\mathcal{O}_X$.

What about ε?

FFT search with threshold (quantum cont.)

Given samples $x_1, \dots, x_N \in \mathbb{Z}_q^k$

- put samples in a QRAM $\mathcal{O}_X$
- search over $s \in \mathbb{Z}_q^k$ with Grover for...
  - compute F_s using theorem with $\mathcal{O}_X$ and check against threshold

Theorem (Simplified)

There is a quantum algorithm that computes $F_s \pm \varepsilon$ given oracle access by making $O(1/\varepsilon)$ queries to $\mathcal{O}_X$.

What about ε? For dual attacks: $\varepsilon = \Omega(1/\sqrt{N})$

Quantum complexity

$$O(\sqrt{q^{k_{\text{fft}}} \cdot N})$$

FFT search with threshold (quantum cont.)

Given samples $x_1, \ldots, x_N \in \mathbb{Z}_q^k$

- put samples in a QRAM $\mathcal{O}_X$
- search over $s \in \mathbb{Z}_q^k$ with Grover for...
  - compute F_s using theorem with $\mathcal{O}_X$ and check against threshold

Theorem (Simplified)

There is a quantum algorithm that computes $F_s \pm \varepsilon$ given oracle access by making $O(1/\varepsilon)$ queries to $\mathcal{O}_X$.

What about ε? For dual attacks: $\varepsilon = \Omega(1/\sqrt{N})$

Quantum complexity	Classical complexity
$O(\sqrt{q^{k_{\text{fft}}} \cdot N})$	$O(q^{k_{\text{fft}}} + N)$

- quantum never worse than classical
- significant gain when $N \ll q^{k_{\text{fft}}}$: like in dual attacks

Quantum Algorithm using QRAM

Prepare the state

$$\frac{1}{\sqrt{N}} \sum_{j=1}^{N} |j\rangle |\mathbf{0}\rangle |\boldsymbol{s}\rangle |0\rangle |0\rangle ,$$

apply $O_X : |j\rangle |0\rangle \to |j\rangle \left|\boldsymbol{x}_j\right\rangle$ on the first and second registers to get

Quantum Algorithm using QRAM

Prepare the state

$$\frac{1}{\sqrt{N}} \sum_{j=1}^{N} |j\rangle \, |\mathbf{0}\rangle \, |\boldsymbol{s}\rangle \, |0\rangle \, |0\rangle \, ,$$

apply $O_X : |j\rangle \, |0\rangle \to |j\rangle \left| \boldsymbol{x}_j \right\rangle$ on the first and second registers to get

$$\frac{1}{\sqrt{N}} \sum_{j=1}^{N} |j\rangle \left| \boldsymbol{x}_j \right\rangle |\boldsymbol{s}\rangle \, |0\rangle \, |0\rangle \, .$$

Quantum Algorithm using QRAM

Prepare the state

$$\frac{1}{\sqrt{N}} \sum_{j=1}^{N} |j\rangle \, |\mathbf{0}\rangle \, |\boldsymbol{s}\rangle \, |0\rangle \, |0\rangle \, ,$$

apply $O_X : |j\rangle \, |0\rangle \to |j\rangle \, \left|\boldsymbol{x}_j\right\rangle$ on the first and second registers to get

$$\frac{1}{\sqrt{N}} \sum_{j=1}^{N} |j\rangle \, \left|\boldsymbol{x}_j\right\rangle \, |\boldsymbol{s}\rangle \, |0\rangle \, |0\rangle \, .$$

Then apply

$$O_{\cos} : |\boldsymbol{x}\rangle \, |\boldsymbol{s}\rangle \, |0\rangle \to |\boldsymbol{x}\rangle \, |\boldsymbol{s}\rangle \, |\cos(2\pi\langle \boldsymbol{x}, \boldsymbol{s}\rangle / q)\rangle \, ,$$

on the second, third, fourth registers to get

Quantum Algorithm using QRAM

Prepare the state

$$\frac{1}{\sqrt{N}}\sum_{j=1}^{N}|j\rangle\,|\mathbf{0}\rangle\,|\boldsymbol{s}\rangle\,|0\rangle\,|0\rangle\,,$$

apply $O_X : |j\rangle\,|0\rangle \to |j\rangle\,\left|\boldsymbol{x}_j\right\rangle$ on the first and second registers to get

$$\frac{1}{\sqrt{N}}\sum_{j=1}^{N}|j\rangle\,\left|\boldsymbol{x}_j\right\rangle\,|\boldsymbol{s}\rangle\,|0\rangle\,|0\rangle\,.$$

Then apply

$$O_{\cos} : |\boldsymbol{x}\rangle\,|\boldsymbol{s}\rangle\,|0\rangle \to |\boldsymbol{x}\rangle\,|\boldsymbol{s}\rangle\,|\cos(2\pi\langle\boldsymbol{x},\boldsymbol{s}\rangle/q)\rangle\,,$$

on the second, third, fourth registers to get

$$\frac{1}{\sqrt{N}}\sum_{j=1}^{N}|j\rangle\,\left|\boldsymbol{x}_j\right\rangle\,|\boldsymbol{s}\rangle\,\left|\cos(2\pi\langle\boldsymbol{x}_j,\boldsymbol{s}\rangle)/q\right\rangle\,|0\rangle\,.$$

Quantum Algorithm using QRAM

$$\frac{1}{\sqrt{N}} \sum_{j=1}^{N} |j\rangle \left|\boldsymbol{x}_j\right\rangle |\boldsymbol{s}\rangle \left|\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle / q)\right\rangle |0\rangle \,.$$

Quantum Algorithm using QRAM

$$\frac{1}{\sqrt{N}} \sum_{j=1}^{N} |j\rangle \left|\boldsymbol{x}_j\right\rangle |\boldsymbol{s}\rangle \left|\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)\right\rangle |0\rangle \,.$$

Apply

$$O_{CR^+} : |a\rangle\,|0\rangle \to \begin{cases} |a\rangle\,(\sqrt{a}\,|1\rangle + \sqrt{1-a}\,|0\rangle), & \text{if } a \geq 0 \\ |a\rangle\,|0\rangle\,, & \text{otherwise,} \end{cases}$$

on the fourth and fifth registers to get

Quantum Algorithm using QRAM

$$\frac{1}{\sqrt{N}} \sum_{j=1}^{N} |j\rangle \left|\boldsymbol{x}_j\right\rangle |\boldsymbol{s}\rangle \left|\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)\right\rangle |0\rangle .$$

Apply

$$O_{CR^+} : |a\rangle\,|0\rangle \to \begin{cases} |a\rangle\,(\sqrt{a}\,|1\rangle + \sqrt{1-a}\,|0\rangle), & \text{if } a \geq 0 \\ |a\rangle\,|0\rangle\,, & \text{otherwise,} \end{cases}$$

on the fourth and fifth registers to get

$$\frac{1}{\sqrt{N}} \sum_{\substack{j\in[N] \text{ and} \\ \cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{t}\rangle/q)\geq 0}} |j\rangle \left|\boldsymbol{x}_j\right\rangle |\boldsymbol{s}\rangle \left|\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)\right\rangle \begin{pmatrix} \sqrt{1-\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)}\,|0\rangle \\ +\sqrt{\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle}/q)\,|1\rangle \end{pmatrix}$$

$$+\frac{1}{\sqrt{N}} \sum_{\substack{j\in[N] \text{ and} \\ \cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)<0}} |j\rangle \left|\boldsymbol{x}_j\right\rangle |\boldsymbol{s}\rangle \left|\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q\right\rangle |0\rangle .$$

Using QRACM to construct U

$$\frac{1}{\sqrt{N}} \sum_{\substack{j \in [N] \text{ and} \\ \cos(2\pi \langle \boldsymbol{x}_j, \boldsymbol{s} \rangle) \geq 0}} |j\rangle \left|\boldsymbol{x}_j\right\rangle |\boldsymbol{s}\rangle \left|\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)\right\rangle \begin{pmatrix} \sqrt{1-\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)}\,|0\rangle \\ +\sqrt{\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)}\,|1\rangle \end{pmatrix}$$

$$+ \frac{1}{\sqrt{N}} \sum_{\substack{j \in [N] \text{ and} \\ \cos(2\pi \langle \boldsymbol{x}_j, \boldsymbol{s} \rangle/q) < 0}} |j\rangle \left|\boldsymbol{x}_j\right\rangle |\boldsymbol{s}\rangle \left|\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)\right\rangle |0\rangle$$

Using QRACM to construct U

$$\frac{1}{\sqrt{N}} \sum_{\substack{j \in [N] \text{ and} \\ \cos(2\pi \langle \boldsymbol{x}_j, \boldsymbol{s} \rangle) \geq 0}} |j\rangle \left|\boldsymbol{x}_j\right\rangle |\boldsymbol{s}\rangle \left|\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle / q)\right\rangle \begin{pmatrix} \sqrt{1 - \cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)}\,|0\rangle \\ +\sqrt{\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)}\,|1\rangle \end{pmatrix}$$

$$+ \frac{1}{\sqrt{N}} \sum_{\substack{j \in [N] \text{ and} \\ \cos(2\pi \langle \boldsymbol{x}_j, \boldsymbol{s} \rangle / q) < 0}} |j\rangle \left|\boldsymbol{x}_j\right\rangle |\boldsymbol{s}\rangle \left|\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle / q)\right\rangle |0\rangle$$

$$= \sqrt{a^+}\,|\phi_1\rangle\,|1\rangle + \sqrt{1 - a^+}\,|\phi_0\rangle\,|0\rangle\,,$$

where

$$a^+ = \sum_{\substack{j \in [N] \text{ and} \\ \cos(2\pi \langle \boldsymbol{x}_j, \boldsymbol{s} \rangle / q) \geq 0}} \frac{\cos\left(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle / q\right)}{N}.$$

Using QRACM to construct U

$$\frac{1}{\sqrt{N}} \sum_{\substack{j \in [N] \text{ and} \\ \cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle) \geq 0}} |j\rangle \left|\boldsymbol{x}_j\right\rangle |\boldsymbol{s}\rangle \left|\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)\right\rangle \begin{pmatrix} \sqrt{1-\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)}\,|0\rangle \\ +\sqrt{\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)}\,|1\rangle \end{pmatrix}$$

$$+ \frac{1}{\sqrt{N}} \sum_{\substack{j \in [N] \text{ and} \\ \cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q) < 0}} |j\rangle \left|\boldsymbol{x}_j\right\rangle |\boldsymbol{s}\rangle \left|\cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q)\right\rangle |0\rangle$$

$$= \sqrt{a^+}\,|\phi_1\rangle\,|1\rangle + \sqrt{1-a^+}\,|\phi_0\rangle\,|0\rangle\,,$$

where

$$a^+ = \sum_{\substack{j \in [N] \text{ and} \\ \cos(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q) \geq 0}} \frac{\cos\left(2\pi\langle \boldsymbol{x}_j, \boldsymbol{s}\rangle/q\right)}{N}.$$

$\rightsquigarrow$ Amplitude Estimation

Dual attack cost estimates (logarithms to base two)

Scheme	CC	CN	C0	GE19	QN	Q0	This work (QN)	This work (Q0)
Kyber 512	139	134	115	139	124	103	113	95
Kyber 768	196	191	174	192	175	155	159	142
Kyber 1024	262	256	242	252	235	215	212	196
LightSaber	139	133	114	138	123	101	113	94
Saber	201	196	179	196	180	159	165	147
FireSaber	264	258	244	253	236	217	215	199
TFHE630	118	113	93	120	105	83	95	77
TFHE630	122	117	95	124	109	85	101	80