Constant-time Lattice Reduction for SQIsign

O. Hanyecz, A. Karenin, E. Kirshanova, P. Kutas, **S. Schaeffler**

March 14, 2025
Cryptography Seminar, Rennes

SQIsign: A post-quantum signature

- Post-quantum signature

  De Feo, Kohel, Leroux, Petit and Wesolowski, 2020

Eprint 2020/1240
Logo from sqisign.org
LVL1 data from pqsort.tii.ae (24/02/2025)

SQIsign: A post-quantum signature

- Post-quantum signature

  De Feo, Kohel, Leroux, Petit and Wesolowski, 2020

- Round 2 NIST candidate

Eprint 2020/1240
Logo from sqisign.org
LVL1 data from pqsort.tii.ae (24/02/2025)

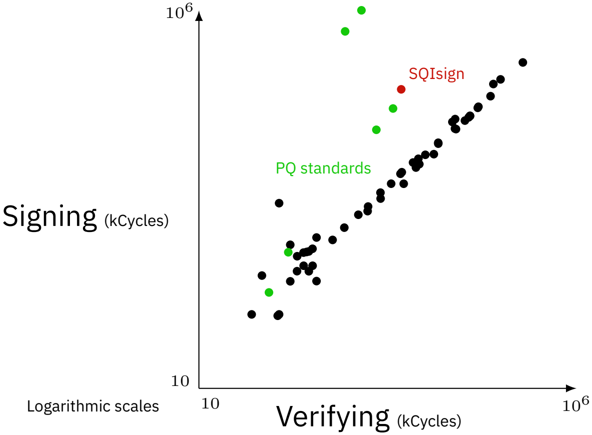
SQIsign: A post-quantum signature

+ Post-quantum signature

  De Feo, Kohel, Leroux, Petit and Wesolowski, 2020

+ Round 2 NIST candidate

≈ Not very fast



Eprint 2020/1240
Logo from sqisign.org
LVL1 data from pqsort.tii.ae (24/02/2025)

SQIsign: A post-quantum signature

+ Post-quantum signature

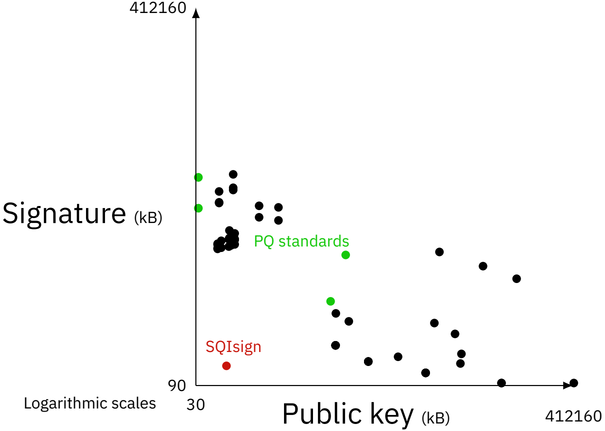
  De Feo, Kohel, Leroux, Petit and Wesolowski, 2020

+ Round 2 NIST candidate

≈ Not very fast

+ public key: 148 bytes

+ signatures: 65 bytes



Eprint 2020/1240
Logo from sqisign.org
LVL1 data from pqsort.tii.ae (24/02/2025)

SQIsign: A post-quantum signature

+ Post-quantum signature

  De Feo, Kohel, Leroux, Petit and Wesolowski, 2020

+ Round 2 NIST candidate

$\approx$ Not very fast

?

+ public key: 148 bytes
+ signatures: 65 bytes

– Complex

Eprint 2020/1240
Logo from sqisign.org
LVL1 data from pqsort.tii.ae (24/02/2025)

Short Quaternion and Isogeny Signature

Quaternions and Isogenies

	Secret World Quaternions	**Public World** Isogenies
Objects	Ideals	Isogenies
Lengths	Ideal norm	Isogeny degree

Quaternions and Isogenies

	Secret World Quaternions	**Public World** Isogenies
Objects	Ideals	Isogenies
Lengths	Ideal norm	Isogeny degree

short ideal $\rightarrow$ short isogeny

Quaternions and Isogenies

	Secret World Quaternions	**Public World** Isogenies
Objects	Ideals	Isogenies
Lengths	Ideal norm	Isogeny degree

short ideal $\rightarrow$ short isogeny

short element in ideal $\rightarrow$ short ideal

Quaternion ideals

Quaternion algebra:

- 4-dimensional vector space over $\mathbb{Q}$
+ ...
+ with quadratic form called norm N

Quaternion ideals

Quaternion algebra:

- 4-dimensional vector space over $\mathbb{Q}$
- $+$ ...
- $+$ with quadratic form called norm N

Ideal:

- rank 4 lattice
  - Lattice: set of $\mathbb{Z}$-linear combinations of a $\mathbb{Q}$-basis

Quaternion ideals

Quaternion algebra:

- 4-dimensional vector space over $\mathbb{Q}$
- \+ ...
- \+ with quadratic form called norm N

Ideal:

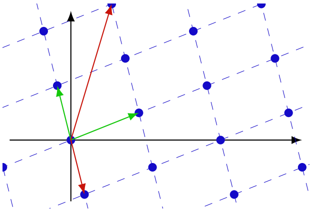
- rank 4 lattice
  - Lattice: set of $\mathbb{Z}$-linear combinations of a $\mathbb{Q}$-basis
- \+ ...
- \+ integer norm: $N(I) = \gcd_{x \in I} N(x)$

The lattice reduction problem

Lattice: Set of $\mathbb{Z}$-linear combinations of a $\mathbb{Q}$-basis of $\mathbb{Q}^4$

The lattice reduction problem

Lattice: Set of $\mathbb{Z}$-linear combinations of a $\mathbb{Q}$-basis of $\mathbb{Q}^4$

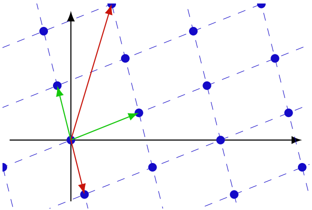


The lattice reduction problem

Lattice: Set of $\mathbb{Z}$-linear combinations of a $\mathbb{Q}$-basis of $\mathbb{Q}^4$

Given: A lattice basis B

Find: A *reduced* basis B' of the same lattice as B



Reduced: Containing only vectors which are

- of somewhat small norm
- and somewhat orthogonal

Several definitions exist

Lattice reduction like in lattice cryptography?

Lattice-crypto:

- Large dimension over $\mathbb{Q}$
- Often smaller integers

$\rightarrow$ Optimization for high dimension

- Lattice reduction in cryptanalysis

$\rightarrow$ Need fast reduction

SQIsign:

- Dimension 4 over $\mathbb{Q}$
- Large integers

$\rightarrow$ Optimization for large coefficients

- Lattice reduction used constructively

$\rightarrow$ Need secure reduction

Practical security in cryptography

Mathematical security:
Ensure that given only public information, an adversary cannot break the system

Implementation security:
Ensure observing the computation only gives public information to an adversary

Practical security in cryptography

Mathematical security:
Ensure that given only public information, an adversary cannot break the system

Implementation security:
Ensure observing the computation only gives public information to an adversary

Side channels:

- Runtime
- Power consumption
- Memory accesses
- Faults

Practical security in cryptography

Mathematical security:
Ensure that given only public information, an adversary cannot break the system

Implementation security:
Ensure observing the computation only gives public information to an adversary

Side channels:

- Runtime
- Power consumption
- Memory accesses

Constant-time algorithms

Constant-time: Secret-independent runtime

Constant-time algorithms

Constant-time: Secret-independent runtime

Assume basic arithmetic is constant-time

Constant-time algorithms

Constant-time: Secret-independent runtime

Assume basic arithmetic is constant-time

good	Naive matrix multiplication
bad	Euclid's gcd

Constant-time algorithms

Constant-time: Secret-independent runtime

Assume basic arithmetic is constant-time

good Naive matrix multiplication
bad Euclid's gcd

Simplified: No if, no while

Lattice reduction algorithm: LLL

Require: B basis of a lattice L of rank d and $c \in]1/4, 1[$
Ensure: B an c-LLL-reduced basis of L

$B^* :=$ Gram-Schmidt-Orthogonalize(B)
while B is not reduced **do**
    Size-reduce B , update B^*
    for i from 1 to $d-1$ **do**
        if not LLLcondition(c, i, B, B^*) **then**
            Swap b_i, b_{i+1} in B, update B^*, continue
        end if
    end for
end while
return B

From “Factoring polynomials with rational coefficients” by A. Lenstra, H. Lenstra, L. Lovász, 1982
Description adapted from H. Cohen’s “A Course in Computational Algebraic Number Theory”, 1993

Lattice reduction algorithm: Greedy

Require: B basis of a lattice L of rank $d \leq 4$, G its Gram matrix
Ensure: B a Minkowski-reduced basis of L, G its Gram matrix

1: done $:=$ False
2: **while** not done and $d > 1$ **do**
3: Sort $(b_1, ..., b_d)$ by norm, adapt B and G;
4: $b_1, ..., b_{d-1}, G' :=$ Greedy$(b_1, ..., b_{d-1})$ adapt B and G
5: $b_d := b_d - c$ where c is closest to b_d in the lattice of $b_1, ..., b_{d-1}$, adapt B and G;
6: done $:= (N(b_d) \geq N(b_{d-1}))$
7: **end while**
8: **return** B, G

From "Low-Dimensional Lattice Basis Reduction Revisited" by P. Q. Nguyen, D. Stehlé, 2004
(DOI 10.1007/978-3-540-2)

Lattice reduction algorithm: BKZ-2

Require: B basis of a lattice L of rank 4, parameter $\delta < 1$
Ensure: B a reduced basis of L

LLL-reduce(B)
while First tour or B has changed in previous tour **do**
    for i from 1 to 3 **do**
        b :=SVP(b_i, b_{i+1})
        if δ-condition(b, B) **then**
            Insert b in B
        end if
        LLL-reduce(B)
    end for
end while
return B

From "Lattice basis reduction: Improved practical algorithms and solving subset sum problems."
by C. P. Schnorr and M. Euchner, 1991 (DOI 10.1007/3-540-54458-5_51); Description from Eprint 2011/198

Lattice reduction algorithm: BKZ-2 for analysis

Require: B basis of a lattice L of rank 4, optional T_m max iteration number
Ensure: B a reduced basis of L if T_m large enough

while B has changed in previous tour and T_m not reached **do**
    for i from 1 to 3 **do**
        $b_1, b_{i+1} :=$ HKZ-reduce(b_i, b_{i+1})
        Size-reduce(B)
    end for
end while
return B

From "Terminating BKZ" by G. Hanrot, X. Pujol and D. Stehlé, 2011
(Eprint 2011/198)

Constant-time BKZ-2

Require: B basis of a lattice L of rank 4, iteration counts T_{Lagr}, T_{BKZ}
Ensure: B' a reduced basis of L if T_{Lagr}, T_{BKZ} are large enough

```
B', G, B := B, its Gram matrix, its orthogonalization
for j from 1 to T_BKZ do
    for i from 1 to 3 do
        Size-reduce b'_i, adapt G and B*;
        Constant-time Lagrange-reduce (b'_i, b'_{i+1}, T_Lagr), adapt B', B*, G
        Size-reduce b'_i then b'_{i+1} , adapt G and B*
    end for
end for
return B'
```

Lagrange reduction

Require: b_1, b_2 basis of a rank-2 lattice L
Ensure: c, d basis of L with c minimal in L

```
while first round or N(d) < N(c) do
    μ = ⌊(N(c+d) − N(c) − N(d)) / (2N(d))⌉
    c, d := d, c − μd
end while
return c, d
```

$\quad$ **while** first round or $N(d) < N(c)$ **do**
$\quad\quad \mu = \left\lfloor \frac{N(c+d)-N(c)-N(d)}{2N(d)} \right\rceil$
$\quad\quad c, d := d, c - \mu d$
$\quad$ **end while**
$\quad$ **return** c, d

N square of a norm (quadratic form)

Lagrange reduction

Require: b_1, b_2 basis of a rank-2 lattice L
Ensure: c, d basis of L with c minimal in L

1: **for** $1, \ldots, T_{\text{Lagr}}$ **do**
2: $\quad \mu = \left\lfloor \frac{N(c+d) - N(c) - N(d)}{2N(d)} \right\rceil$
3: $\quad c, d := d, c - \mu d$
4: **end for**
5: **return** c, d

N square of a norm (quadratic form)

Lagrange reduction

Require: b_1, b_2 basis of a rank-2 lattice L
Ensure: c, d basis of L with c minimal in L

1: **for** $1, \ldots, T_{\mathsf{Lagr}}$ **do**
2: $\quad \mu = \lfloor \frac{N(c+d)-N(c)-N(d)}{2N(d)} \rceil$
3: $\quad c, d := d, c - \mu d$
4: **end for**
5: CT-CONDITIONAL-SWAP$_{N(d)<N(c)}(c, d)$
6: **return** c, d

N square of a norm (quadratic form)

Iteration counts

Ensure:

$\|b_0\| \leqslant 2\left(\frac{4}{3}\right)^{3/2} D^{1/4}$

Require:

$$T_{\mathsf{BKZ}} \geq \frac{2}{\log_2(8/7)} \log_2\left(\log_2\left(\frac{B^*}{D^{1/4}}\right) + \sqrt{5}(\log(4/3))^{1/2}\right)$$

$$T_{\mathsf{Lagr}} \geq 2 + 2\lceil(\log_{\sqrt{3}} 2)\,(9 \log_2 B + 12)\rceil$$

B: Square root of largest diagonal coefficient of Gram matrix of the input
B^*: Square root of largest norm of a vector in the orthogonalization of the input
D: Lattice volume

Iteration counts

Ensure:
$\|b_0\| \leqslant 2\left(\frac{4}{3}\right)^{3/2} D^{1/4}$

Require:
$T_{\mathsf{BKZ}} \geq \frac{2}{\log_2(8/7)} \log_2\left(\log_2\left(\frac{B^*}{D^{1/4}}\right) + \sqrt{5}(\log(4/3))^{1/2}\right)$

$T_{\mathsf{Lagr}} \geq 2 + 2\lceil (\log_{\sqrt{3}} 2)\,(9\log_2 B + 12)\rceil$

In SQIsign:
$T_{\mathsf{BKZ}} \approx$ Tens to hundreds

$T_{\mathsf{Lagr}} \approx$ Thousands

B: Square root of largest diagonal coefficient of Gram matrix of the input
B^*: Square root of largest norm of a vector in the orthogonalization of the input
D: Lattice volume

Gap to practice

Example:

bitsize $N(I)$	T_{Lagr} theory	T_{Lagr} no failure observed	T_{BKZ} theory	T_{BKZ} no failure observed
260	2986	9	64	3

Possible reasons:

? LLL analysis not tight

? Worst case never met

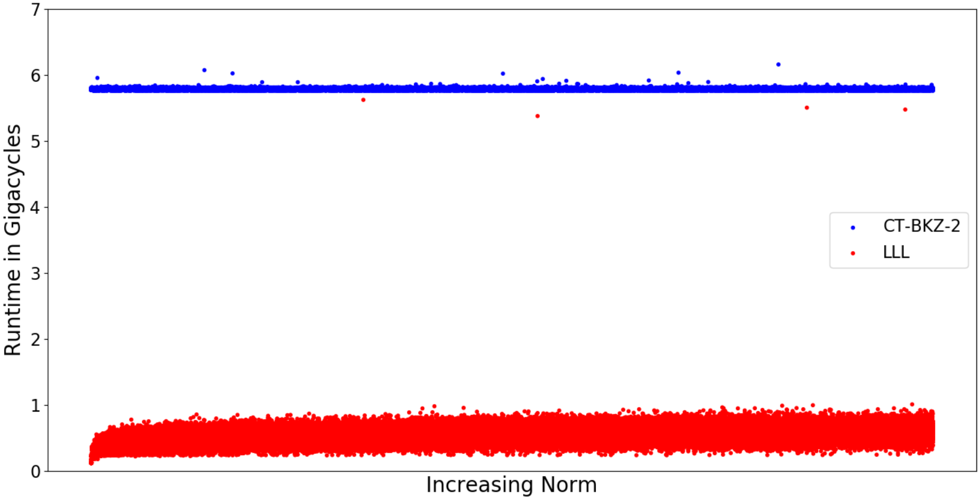
And if we did less iterations?

+ Guaranteed constant runtime
+ Output is lattice basis
– Output might not be sufficiently reduced

→ Running with reduced tours where risk is acceptable
→ Reasonable runtime

→ Experiments possible

CT-BKZ-2 is constant-time



Benchmarks and comparisons

Algorithm	old LLL	CT-BKZ-2		
Parameters			T_{BKZ}	T_{Lagr}
LVL1	7,39	35,9	4	10
LVL3	17,2	81,8	4	11
LVL5	26,4	133	4	13

All timings in gigacycles on an Intel i7-11850H processor

Benchmarks and comparisons

Algorithm	old LLL	CT-BKZ-2
Integers	CT	CT
LVL1	7,39 37 words	35,9 37 words
LVL3	17,2 55 words	81,8 55 words
LVL5	26,4 72 words	133 72 words

All timings in gigacycles on an Intel i7-11850H processor

Iterations for less than 1 failure on 100 random SQIsign-typical inputs. Such inputs are bases in HNF of random O_0-ideals of suitable norm.
For typical SQIsign inputs (from NIST round 1 or 2 implementation): $\log_2(N(I))$ about $260, 385, 508$ and $\log_2(p)$ equal $254, 378, 502$ respectively.

Benchmarks and comparisons

Algorithm	old LLL	CT-BKZ-2	
Integers	CT	CT	short CT
LVL1	7,39 37 words	35,9 37 words	9,59 20 words
LVL3	17,2 55 words	81,8 55 words	19,5 28 words
LVL5	26,4 72 words	133 72 words	38,9 37 words

All timings in gigacycles on an Intel i7-11850H processor

Iterations for less than 1 failure on 100 random SQIsign-typical inputs. Such inputs are bases in HNF of random O_0-ideals of suitable norm.
For typical SQIsign inputs (from NIST round 1 or 2 implementation): $\log_2(N(I))$ about $260, 385, 508$ and $\log_2(p)$ equal $254, 378, 502$ respectively.

Benchmarks and comparisons

Algorithm	old LLL		CT-BKZ-2	
Integers	CT	non-CT	CT	short CT
LVL1	7,39 37 words	0,0016	35,9 37 words	9,59 20 words
LVL3	17,2 55 words	0,0025	81,8 55 words	19,5 28 words
LVL5	26,4 72 words	0,0031	133 72 words	38,9 37 words

All timings in gigacycles on an Intel i7-11850H processor

Iterations for less than 1 failure on 100 random SQIsign-typical inputs. Such inputs are bases in HNF of random O_0-ideals of suitable norm.
For typical SQIsign inputs (from NIST round 1 or 2 implementation): $\log_2(N(I))$ about $260, 385, 508$ and $\log_2(p)$ equal $254, 378, 502$ respectively.

Benchmarks and comparisons

Algorithm	old LLL			CT-BKZ-2	
Integers	CT	non-CT	L2	CT	short CT
LVL1	7,39 37 words	0,0016	5×10^{-5}	35,9 37 words	9,59 20 words
LVL3	17,2 55 words	0,0025	6×10^{-5}	81,8 55 words	19,5 28 words
LVL5	26,4 72 words	0,0031	7×10^{-5}	133 72 words	38,9 37 words

All timings in gigacycles on an Intel i7-11850H processor

Iterations for less than 1 failure on 100 random SQIsign-typical inputs. Such inputs are bases in HNF of random O_0-ideals of suitable norm.
For typical SQIsign inputs (from NIST round 1 or 2 implementation): $\log_2(N(I))$ about $260, 385, 508$ and $\log_2(p)$ equal $254, 378, 502$ respectively.

Integers and other limitations

- Current bottleneck: constant-time GCD
  - Rationals
  - Constant-time GCD self-implemented
  - Very large numbers
- Other number types?

Possible improvements

Optimizations

$\longrightarrow$ Use compiler optimization

$\longrightarrow$ Other number types

Usability

$\longrightarrow$ Failure rate estimation

Possible improvements

Optimizations

→ Use compiler optimization

→ Other number types

Usability

→ Failure rate estimation

Questions?

eprint 2025/027