

LATTICE-BASED ENCRYPTION

PHONG NGUYEN

<http://www.di.ens.fr/~pnguyen>

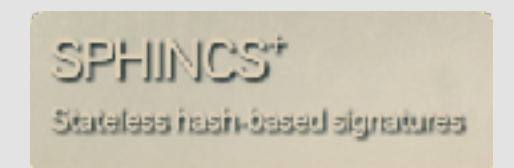
October 2026



4 NIST STANDARDS



- KEM/encryption (1):
 - **Crystals-Kyber**: El Gamal using MLWE lattices
- Signatures (3)
 - **Crystals-Dilithium**: Schnorr / Fiat-Shamir with aborts using MLWE
 - **Falcon**: Rabin / GGH-GPV using NTRU lattices
 - **Sphincs+**: Hash-based



Open**SSL**



TODAY

Lattice Analogues of:

- RSA
 - Encryption **with** Trapdoors
- Diffie-Hellman and El Gamal
 - Encryption **without** Trapdoors

LATTICE CRYPTO DESIGN

2 types of Lattice techniques:

- Cryptography **with** trapdoors
 - i.e. secret short basis of a lattice
 - Similarities with **RSA / Rabin** cryptography
- Cryptography **without** trapdoors
 - Similarities with **Discrete logarithm (DL)** cryptography
- Case study: Encryption

RSA

Encryption with Trapdoors



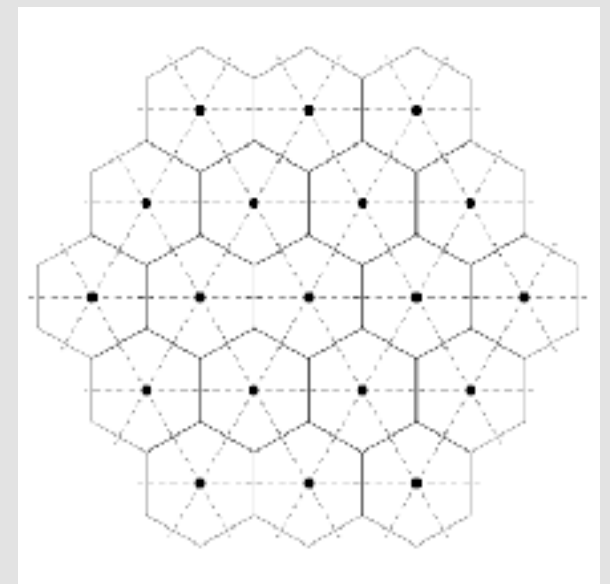
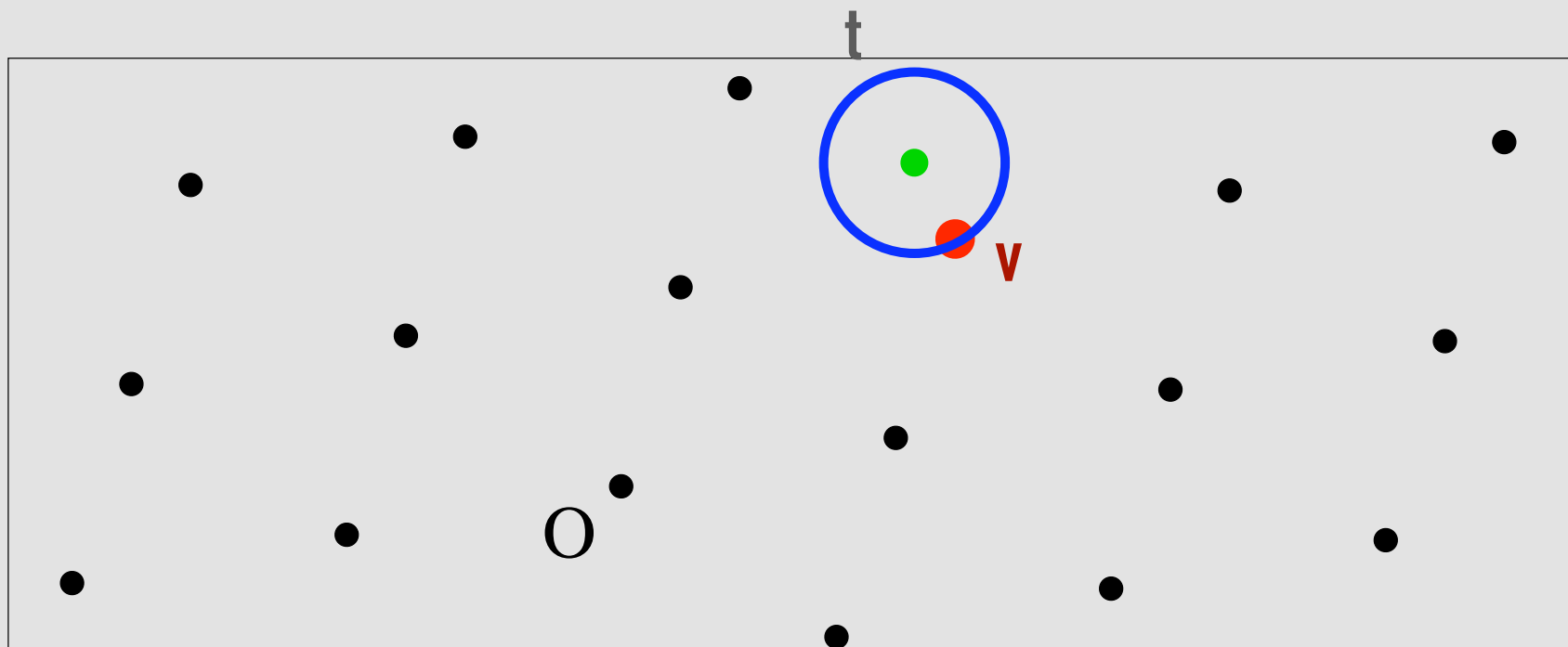
TRAPDOOR-BASED ENCRYPTION: GGH AND NTRU

REMEMBER

- $N = pq$ product of two large random primes.
- $ed \equiv 1 \pmod{\phi(N)}$ where $\phi(N) = (p-1)(q-1)$
 - e is the public exponent
 - d is the secret exponent

BOUNDED DISTANCE DECODING (BDD)

- Input: a basis of a lattice L of dim d , and a target vector t **very close to** L .
- Output: $v \in L$ minimizing $\|v - t\|$. Easy if one knows a nearly-orthogonal basis.



REDUCING MODULO A LATTICE

- If L is an integer lattice, the quotient $\mathbb{Z}^n / L$ is a **finite group**, with **many representations**: lattice crypto works **modulo a lattice**.
- We call **L-reduction** any efficiently computable map $f : \mathbb{Z}^n \rightarrow$ bounded subset of $\mathbb{Z}^n$ s.t. $f(x) \in x + L$ and $f(x) = f(y)$ iff $x - y \in L$.

ONE-WAY FUNCTIONS FROM BDD

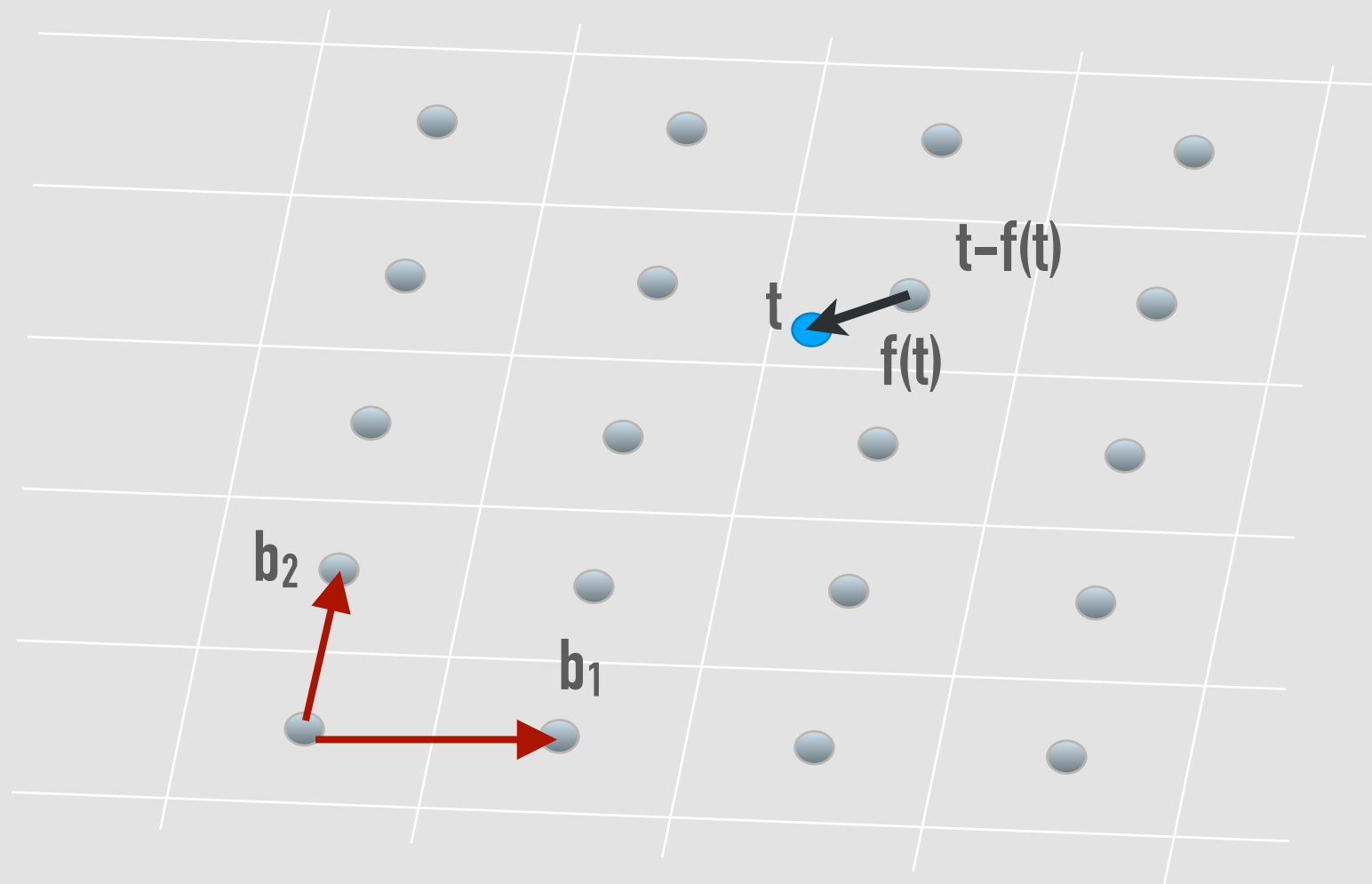
- If BDD is hard over a ball, any public L-reduction f is a one-way function over the same ball.
- Let (t, L) be a BDD instance: $t = v + e$ where $v \in L$ and e is very short.
- Then $f(t) = f(e)$ because $t - e = v \in L$: if f is not one-way, then given $f(e)$, one can recover small e' s.t. $f(e) = f(e')$, therefore $e - e' \in L$. If e and e' are sufficiently small, then $\|\vec{e} - \vec{e'}\| \leq \lambda_1(L)$ so $e = e'$. One recovers the BDD solution $v = t - e$.

BUILDING L-REDUCTIONS

- Any basis provides two L-reductions, thanks to Babai's nearest plane algorithm and rounding-off algorithm.
- NTRU encryption implicitly uses a **L-reduction**.

EXAMPLE: BABAI'S ROUNDING OFF

- Any basis provides two L-reductions, thanks to Babai's nearest plane algorithm and rounding-off algorithm.



- Choose $f(t)$ in the basis parallelepiped s.t. $t - f(t) \in L$

EXAMPLE: BABAI'S ROUNDING OFF

- Let t in $\mathbb{Z}^n$.
- Let B the lattice basis.
- Solve $t = \mathbf{u}B$ where $\mathbf{u}$ in $\mathbb{Q}^n$.
- Return $f(t) = (\mathbf{u} - \lfloor \mathbf{u} \rfloor)B$
- What happens if B is the SVP-challenge basis

$$\begin{pmatrix} p & 0 & & \dots & 0 \\ a_1 & 1 & \ddots & & \vdots \\ a_2 & 0 & 1 & \ddots & \\ \vdots & \vdots & \ddots & \ddots & 0 \\ a_{n-1} & 0 & \dots & 0 & 1 \end{pmatrix}$$

EXAMPLE: BABAI'S NEAREST PLANE ALGORITHM

- Let t in $\mathbb{Z}^n$.
- Let B the lattice basis and B^* its Gram-Schmidt orthogonalization.
- Find $v=uB$ where u in $\mathbb{Z}^n$ s.t. $t-v = xB^*$ where each coordinate of x is $\leq 1/2$ in absolute value

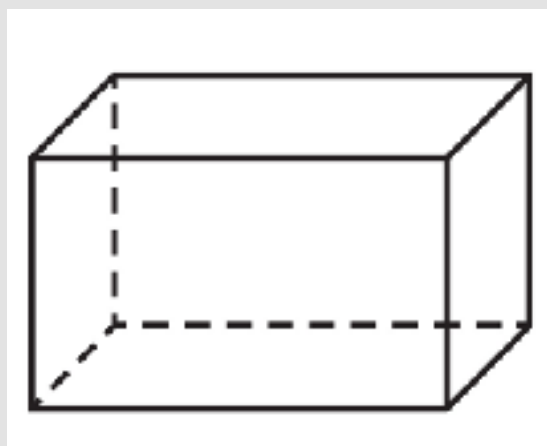
- For $i=n$ downto 1

- Compute $\mu_i = \frac{\langle \vec{t}, \vec{b}_i^* \rangle}{\|\vec{b}_i^*\|^2}$

- $\vec{t} \leftarrow \vec{t} - \lfloor \mu_i \rfloor \vec{b}_i$

- Return $\vec{v} = \sum_{i=1}^n \lfloor \mu_i \rfloor \vec{b}_i$ and $\vec{u} = (\mu_1, \dots, \mu_n)$

- Return $f(t)=t-v$.



- What happens if B is the SVP-challenge basis

$$\begin{pmatrix} p & 0 & \dots & 0 \\ a_1 & 1 & \ddots & \vdots \\ a_2 & 0 & 1 & \ddots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ a_{n-1} & 0 & \dots & 0 & 1 \end{pmatrix}$$

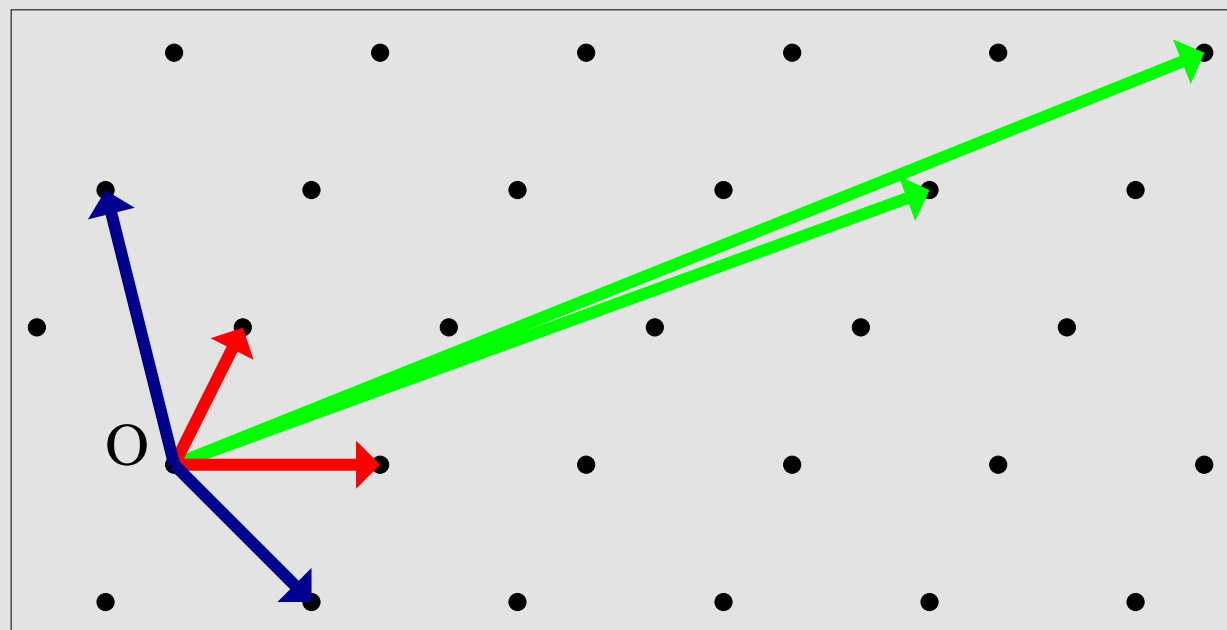
SOLVING BDD BY L-REDUCTION

- The L-reductions derived from Babai's algorithms leave some set invariant: there exists $D(B) \subseteq \mathbb{Z}^n$ s.t. $f(x)=x$ for all $x \in D(B)$.
- This allows to solve BDD when the error $\in D(B)$.
- The largest ball inside $D(B)$ depends on the quality of the basis.

DETERMINISTIC PUBLIC-KEY ENCRYPTION [GGH97-MICC01]

- **Secret key** = Good basis
- **Public key** = Bad basis
- **Message** = Short vector like $\{-1,0,1\}^n$.
- **Encryption** = L-reduction with the public key
- **Decryption** = L-reduction with the secret key

➤ Optimization:



CHOOSING A BAD BASIS

- Intuitively, most bases are « bad ».
- There is a canonical basis for integer lattices: the **Hermite normal form**.
- If L is full-rank, the HNF is a lower-triangular matrix with positive diagonal dominating each column.

ENCRYPTION WITH THE HARDEST LATTICES

SIS TRAPDOORS

- [Ajtai1999, AlwenPeikert2010, MicciancioPeikert2012] showed that it is possible to generate $g_1, \dots, g_m \in (\mathbb{Z}/q\mathbb{Z})^n$ with distribution statistically close to uniform, together with a short basis of the SIS lattice $L = \{\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_m) \in \mathbb{Z}^m \text{ s.t. } \sum_i \mathbf{x}_i g_i = 0\}$.

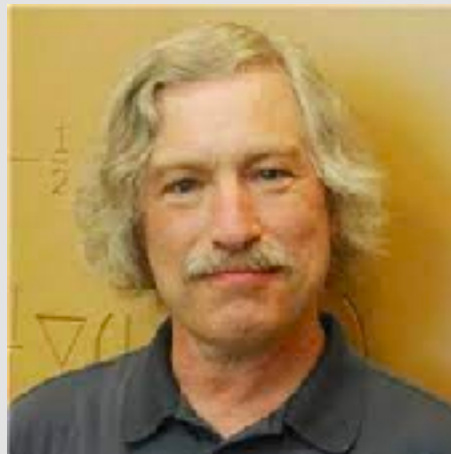
OPTIMIZING ENCRYPTION: NTRU

OPTIMIZATION: NTRU ENCRYPTION

- Ring $R = \mathbb{Z}[X] / (X^N - 1)$, secret key $(f, g) \in R^2$, public key $h = g / f \pmod{q}$.
- Encryption can be viewed [Mi01] as L-reducing a short vector with the Hermite normal form, where $L = \{(u, v) \in R^2 \text{ s.t. } u \equiv pv^*h \pmod{q}\}$.
- Decryption is a special BDD algorithm using the secret key (f, g) .

NTRU ENCRYPTION

- Invented by Hoffstein, Pipher and Silverman in 1996 (CRYPTO rump session):



- First published in 1998 (ANTS conference)
- First cryptanalysis (Coppersmith-Shamir) in 1997!
- One of the fastest public-key encryption known, and one of the most studied.
- Based on polynomials modulo a small integer.

REFERENCES

- Hoffstein, Pipher, Silverman: **NTRU: A Ring-Based Public Key Cryptosystem**. ANTS 1998: 267-288
- <https://ntru.org/>
 - Submission to NIST
 - Reference implementations in C

PARAMETERS

- Let N be a prime number, e.g. 251
- Consider the ring $R = \mathbb{Z}[X] / (X^N - 1)$
- Let p and q be **two small coprime integers**:

➤ In 1998:

N	p	q
107	3	64
167	3	128
503	3	256

➤ In 2020:

N	p	q
509	3	2048
677	3	2048
821	3	4096

KEY GENERATION

- The secret key is two polynomials f and g in R with **very small coefficients**:
 - f and g could be ternary $(0, 1, -1)$ or binary $(0, 1)$.
 - f must be invertible mod q and p . Let f_p and f_q be the inverse.
- The public key is $h = g * f_q \bmod q$ so $h * f = g \bmod q$.

CHOICE OF F AND G

- In 1998: f has d_f coeffs $+1$ and d_f-1 coeffs -1 ,
 g has d_g coeffs $+1$ and d_g coeffs -1

N	d_f	d_g
107	15	12
167	61	20
503	216	72

- In 2020: f is uniform $\{0, \pm 1\}$
 g has d_g coeffs $+1$ and d_g coeffs -1

N	d_g	q
509	127	2048
677	127	2048
821	255	4096

ENCRYPTION

- To encrypt a message $\mathbf{m}$ (a polynomial in R having small coefficients):
 - Choose at random a sparse polynomial $\mathbf{r}$ in R with very small coefficients.
 - The ciphertext is $c = \mathbf{m} + p \mathbf{r}^* h \bmod q$.
 - Encryption is probabilistic.

EFFICIENCY OF ENCRYPTION

- We can store $H = p * h \bmod q$ instead of h , so that encryption is simply $c = \mathbf{m} + \mathbf{r} * H \bmod q$.
- The product $\mathbf{r} * H$ is special cause has $0, \pm 1$ coefficients. Each coefficient of $\mathbf{r} * H$ can be computed with at most N additions and subtractions, rather than $O(N)$ multiplications mod q .
- So encryption costs at most N^2 additions mod q , even less if r is sparse.

DECRYPTION

- Multiplying by the secret key f , we can get:
 - $c * f = m * f + p * r * g \pmod{q}$
- If we could get the exact value of $m * f + p * r * g$ over the integers, we could easily recover $m \pmod{p}$.
- Note: both products $m * f$ and $r * g$ involve only polynomials with small coefficients, possibly sparse.

PRODUCTS OF SMALL POLYNOMIALS

- Let f and g be two polynomials in $\mathbb{R}$ such that:
 - f only has 0,1-coefficients.
 - g has small coefficients with identical distribution.
- Then any coefficient of $f * g$ is just a sum of coeffs of g : the distribution should approximately be Gaussian with small standard deviation.

HOW SMALL ARE THE COEFFICIENTS?

- In NTRU-2020
 - $\mathbf{m}$ has d_g coeffs $+1$ and d_g coeffs -1 , so each coefficient of $\mathbf{m}^*\mathbf{f}$ is in $\{-2d_g, \dots, 2d_g\}$
 - $\mathbf{g}$ has d_g coeffs $+1$ and d_g coeffs -1 , so each coefficient of $\mathbf{r}^*\mathbf{g}$ is in $\{-2d_g, \dots, 2d_g\}$
 - So each coefficient of $\mathbf{m}^*\mathbf{f} + p \mathbf{r}^*\mathbf{g}$ has absolute value $\leq 2d_g(1+p) = 8d_g < q/2$

N	d_g	q
509	127	2048
677	127	2048
821	255	4096

NTRU-1998

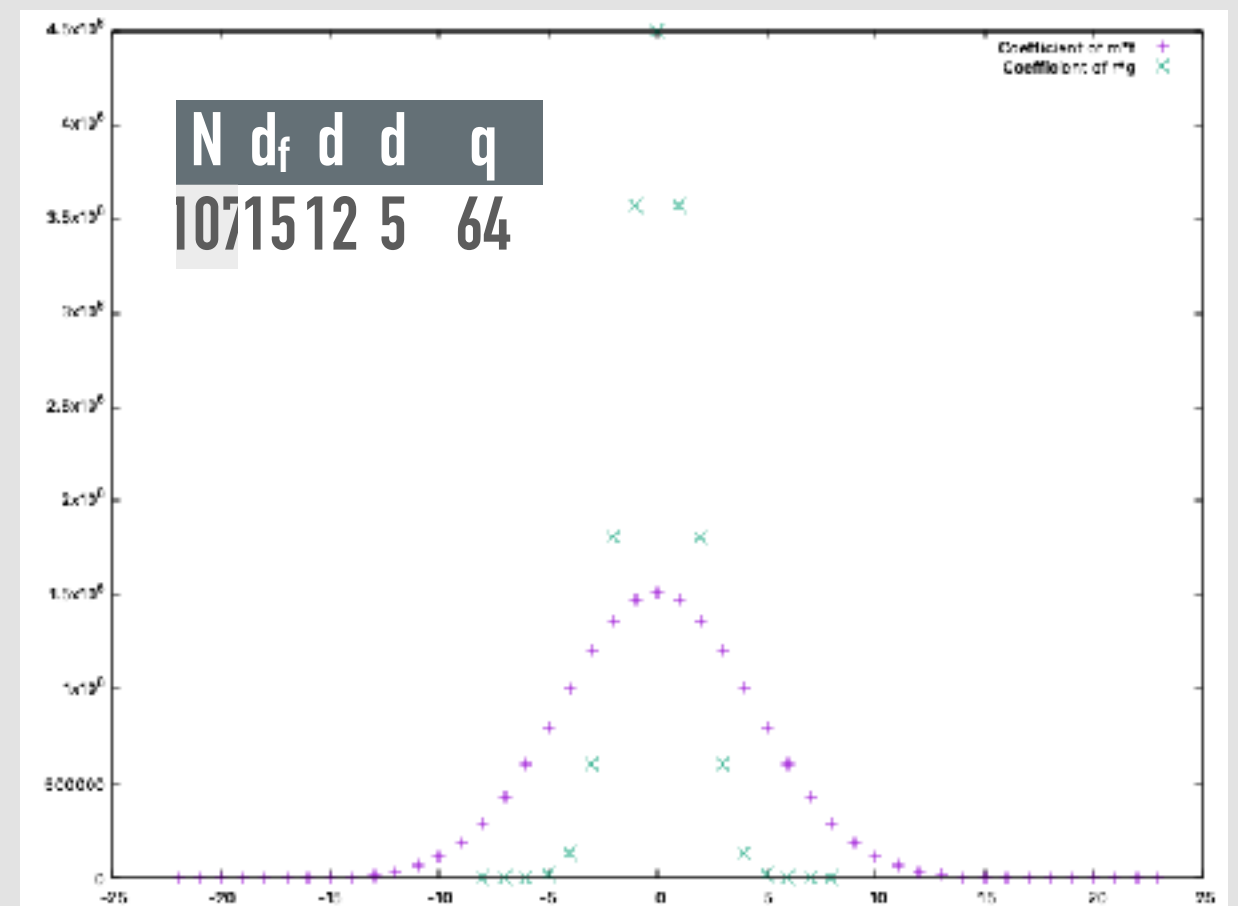
- f has d_f coeffs $+1$ and d_f-1 coeffs -1 , so each coefficient of m^*f is in $\{-2d_f+1, \dots, 2d_f-1\}$
- r has d_r coeffs $+1$ and d_r coeffs -1 , so each coefficient of r^*g is in $\{-2d_r, \dots, 2d_r\}$
- So each coefficient of $m^*f + p r^*g$ has absolute value $\leq 2d_f-1 + 2pd_r$ which is $\geq q/2$.

N	d_f	d_g	d_r	$2d_f-1+2pd_r$	$q/2$
107	15	12	5	59	32
167	61	20	18	229	64
503	216	72	55	761	128

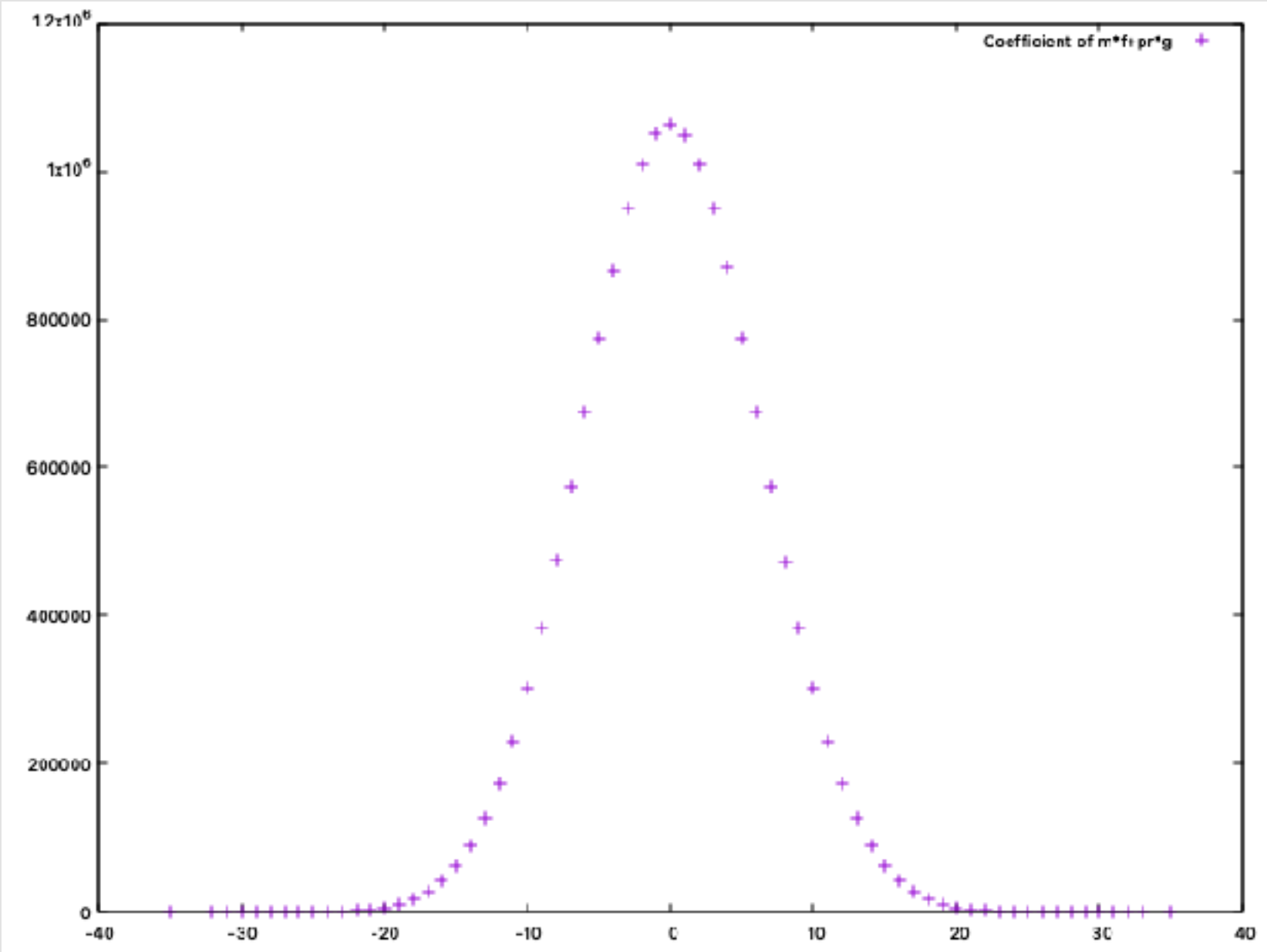
NTRU-1998 DISTRIBUTION

- $\mathbf{f}$ has d_f coeffs $+1$ and d_f-1 coeffs -1 , so each coefficient of $\mathbf{m}^*\mathbf{f}$ is a sum of $2d_f-1$ independent uniform variables in $\{0, \pm 1\}$, but the coeffs are not independent.
- $\mathbf{r}$ has d_r coeffs $+1$ and d_r coeffs -1 , so each coefficient of $\mathbf{r}^*\mathbf{g}$ is in $\{-2d_r, \dots, 2d_r\}$

N	d_f	d_g	d_r
107	15	12	5
167	61	20	18
503	216	72	55



NTRU-1998 DISTRIBUTION OF COEFFICIENTS



8 million samples

8 million samples	
N	q
107	64

IMPACT ON DECRYPTION

- This means that the coefficients of both $\mathbf{m}^*\mathbf{f}$ and $\mathbf{r}^*\mathbf{g}$ lie in a short interval, so that the coefficients of $\mathbf{m}^*\mathbf{f} + p\mathbf{r}^*\mathbf{g}$ lie in an interval of length possibly $\leq q$.
- Then, one could recover the exact value of $\mathbf{m}^*\mathbf{f} + p\mathbf{r}^*\mathbf{g}$ from its value mod q .

EFFICIENCY OF ENCRYPTION

- One needs to compute $p \mathbf{r}^* h \bmod q$, where h has mod q coefficients and $\mathbf{r}$ is sparse with coefficients $0, +1, -1$: each coefficient of $p \mathbf{r}^* h$ is just a sum / difference of coefficients of $p^* h$.
- Overall, this is $O(N^2)$ additions mod q , possibly less since $\mathbf{r}$ is “sparse”.

EFFICIENCY OF DECRYPTION

- The computation of $c * \mathbf{f} \bmod q$: again, $\mathbf{f}$ only has 0,+1,-1 coefficients. This is $O(N^2)$ additions mod q .
- Multiplication by the inverse of $\mathbf{f} \bmod p$.
 - If we choose a special form for $\mathbf{f}$, this can be negligible.
 - Otherwise, it is $O(N^2)$ mults mod p .

SECURITY

- The main security parameter is N , but other parameters are important.
- Key-recovery attacks
 - Brute force over f and g .
 - Square-root attack by Odlyzko.
 - Lattice attack by [CoppersmithShamir1997].

NTRU claims that this attack takes exponential time.

LATTICE ATTACK ON NTRU

- The equation $h^*f = g \bmod q$ can be interpreted in terms of lattice.
- The set L of all polynomials u and v in R such that $h^*u = v \bmod q$ is a lattice of $\mathbb{Z}^{2N}$, of dimension $2N$.
- The pair (f, g) belongs to the lattice L and it is very short because f and g have small coefficients: its norm is $O(N^{1/2})$.

LATTICE INTERPRETATION OF NTRU ENCRYPTION

- The encryption equation $c = m + p \mathbf{r}^* h \pmod{q}$ means that the vector $(0, c)$ in $\mathbf{Z}^{2N}$ is close to the lattice vector $(p\mathbf{r}, p\mathbf{r}^* h \pmod{q})$ in L , because the difference is $(p\mathbf{r}, m)$.
- This is a BDD problem like in GGH encryption.