An Introduction to Lattices, Lattice Reduction, and Lattice-Based Cryptography

Joseph H. Silverman

Brown University

PCMI Lecture Series

July 6–10, 2020

Lecture 4. Lattice-Based Public Key Cryptosystems

I am going to start with the final slide from a colloquium that I gave at Oklahoma State a few months ago:

Quantum Computers are Coming for You!!
Start Preparing Now!!

Taking this dire warning to heart, the remaining two lectures will be devoted to describing some representative cryptographic constructions based on hard lattice problems. These systems are secure against known quantum algorithms.

The Ajtai-Dwork Lattice Cryptosystem

- Ajtai and Dwork (1995) described a lattice-based public key cryptosystem whose security relies on the difficulty of solving CVP in a certain set of lattices $\mathcal{L}_{\mathrm{AD}}$.
- Breaking their system for a a random lattice of dimension m in $\mathcal{L}_{\mathrm{AD}}$ is as difficult as solving SVP for all lattices of dimension n, where n depends on m.
- This **average case-worst case equivalence** is a theoretical cryptographic milestone, but unfortunately the Ajtai-Dwork cryptosystem is impractical.
- Inspired by the work of Ajtai and Dwork, more practical lattice-based cryptosystems were proposed in 1996 independently by Goldreich–Goldwasser–Halevi and by Hoffstein–Pipher–Silverman.
- The original goal was speed, since lattice systems are ≈ 10 times faster than RSA and ECC. Now quantum security is a crucial attribute.

The GGH Public Key Cryptosystem

Here is the GGH lattice-based cryptosystem due to Goldreich–Goldwasser–Halevi.

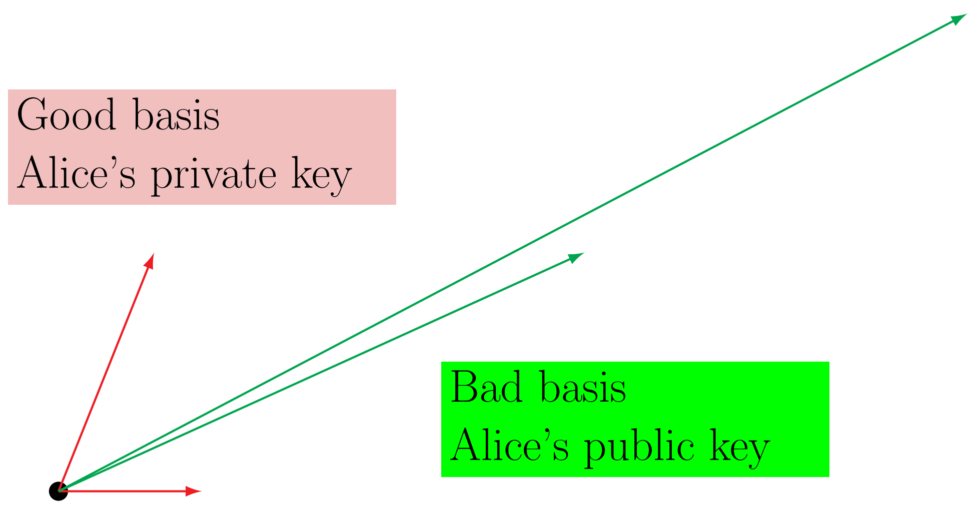
$$\begin{aligned}
\text{Private Key} &= \text{a good basis } \mathcal{B}^{\text{good}} \text{ for a lattice } \mathcal{L}\\
&= \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}.\\
\text{Public Key} &= \text{a bad basis } \mathcal{B}^{\text{bad}} \text{ for the lattice } \mathcal{L}.\\
&= \{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n\}.\\
\text{Plaintext} &= \text{a binary vector } (\epsilon_1, \ldots, \epsilon_n), \text{ i.e., } \epsilon_i \in \{0,1\}.\\
\text{Ciphertext} &= \epsilon_1 \boldsymbol{w}_1 + \cdots + \epsilon_n \boldsymbol{w}_n + r,\\
&\qquad\qquad \text{where } r \text{ is a small random vector.}
\end{aligned}$$

- Bob uses the bad public basis $\mathcal{B}^{\text{bad}}$ and a random small vector r to create the Ciphertext.
- Alice uses Babai with $\mathcal{B}^{\text{good}}$ to solve CVP. She finds a vector $\boldsymbol{v} \in \mathcal{L}$ close to the ciphertext. She writes $\boldsymbol{v}$ in terms of $\mathcal{B}^{\text{bad}}$ to recover

$$\boldsymbol{v} = \epsilon_1 \boldsymbol{w}_1 + \cdots + \epsilon_n \boldsymbol{w}_n.$$

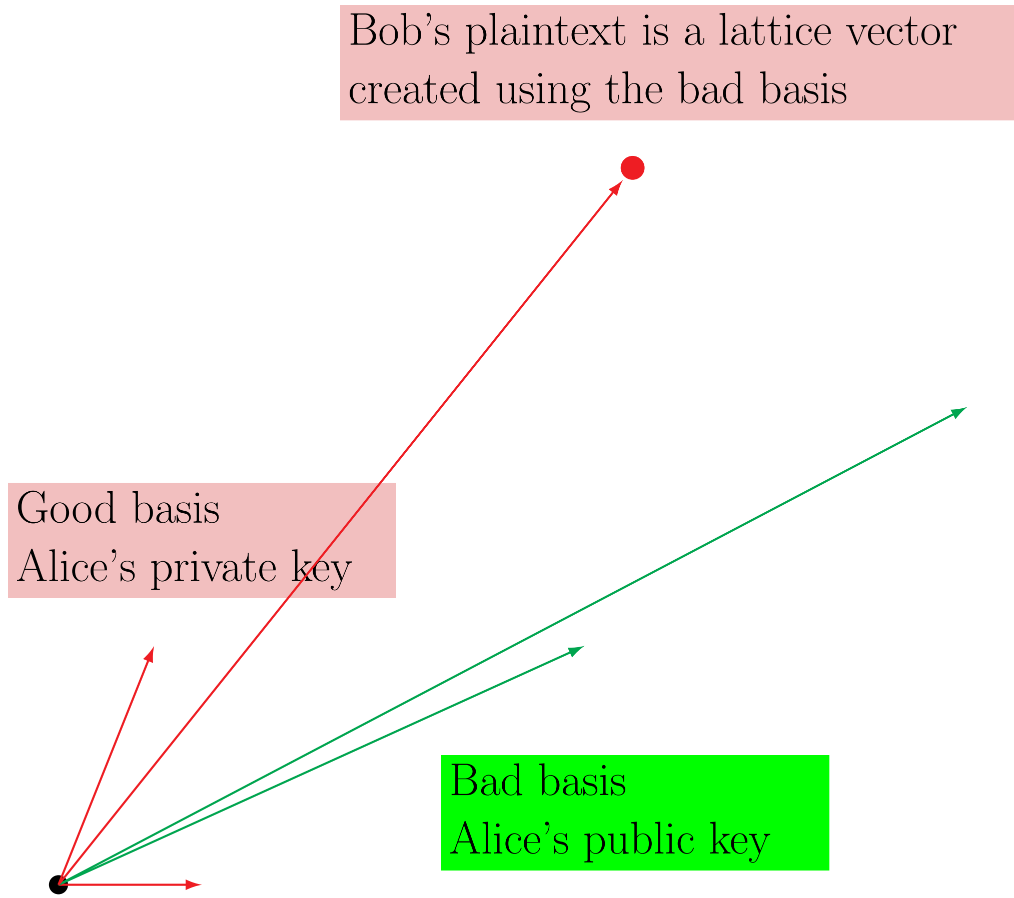
The GGH Cryptosystem In Pictures

We illustrate the GGH cryptosystem:



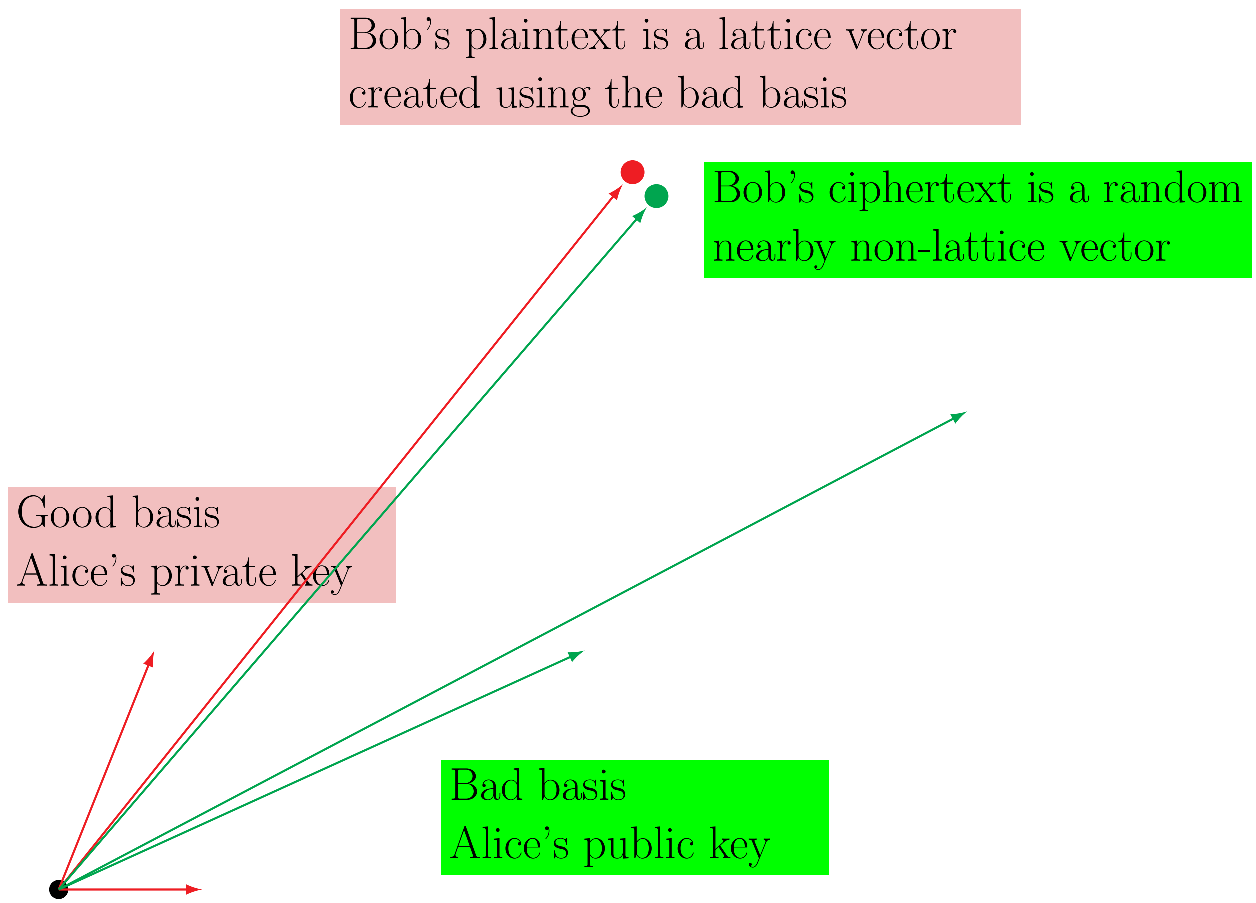
The GGH Cryptosystem In Pictures

We illustrate the GGH cryptosystem:



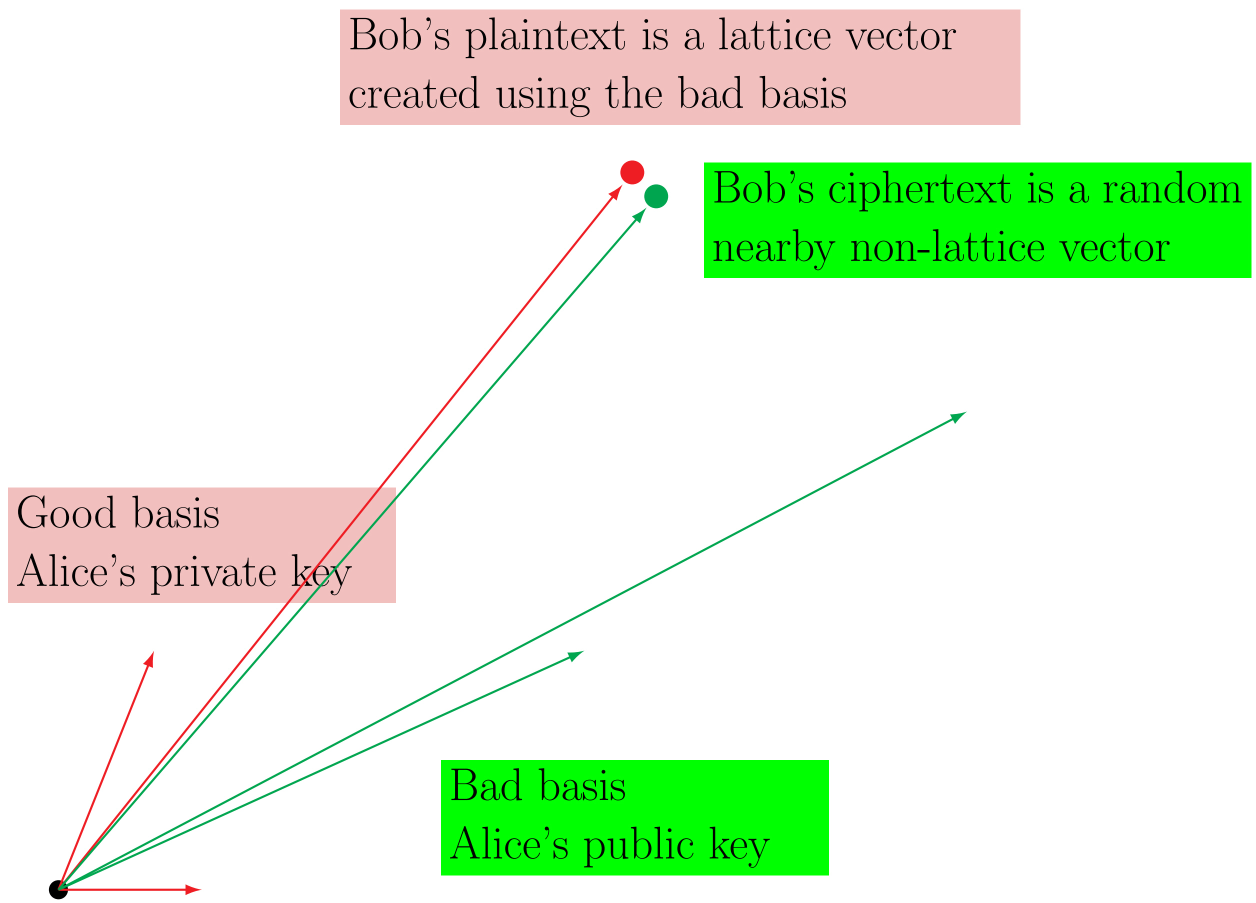
The GGH Cryptosystem In Pictures

We illustrate the GGH cryptosystem:



The GGH Cryptosystem In Pictures

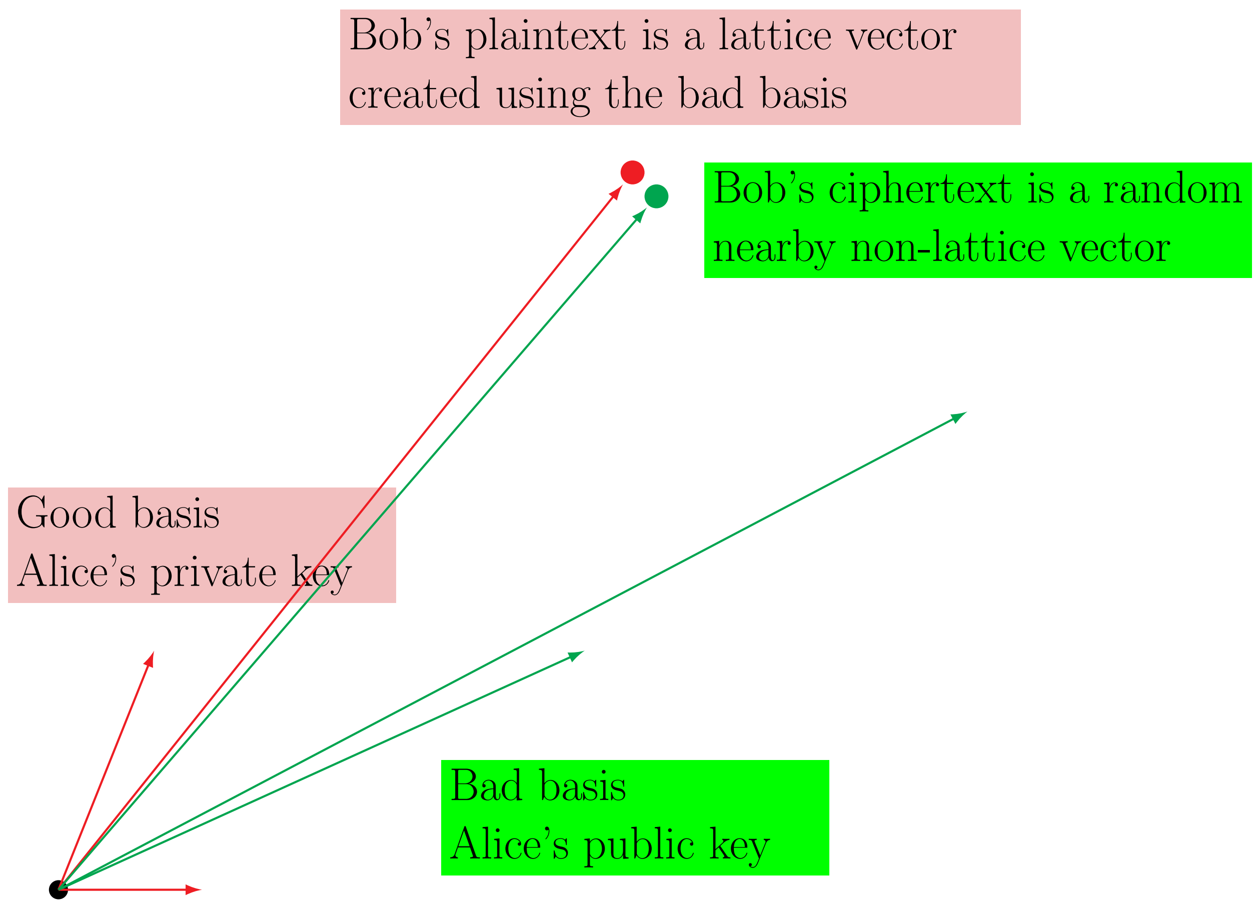
We illustrate the GGH cryptosystem:



Alice uses her good basis to find Bob's plaintext.

The GGH Cryptosystem In Pictures

We illustrate the GGH cryptosystem:



Alice uses her good basis to find Bob's plaintext.

Eve can't find Bob's plaintext using the bad basis.

GGH versus LLL: A Battle for Supremacy!

- The security of GGH rests on the difficulty of solving CVP using a highly nonorthogonal basis. The LLL lattice reduction algorithm finds a **moderately** orthogonal basis in polynomial time.
- The security of GGH thus comes down to the question of just how good LLL is at solving CVP.
- If $n = \dim(\mathcal{L}) < 100$, then LLL easily finds a basis that's good enough to break GGH. And even up to $n \approx 200$, variants of LLL will break GGH.

A GGH public key is a basis for $\mathcal{L} \subset \mathbb{R}^n$. That's n vectors, each with n coordinates, so

$$\text{Size of GGH Public Key} = O(n^2) \text{ bits.}$$

- GGH is probably(?) secure for $n = 500$ to 1000, but 2 megabit keys are impractical!

NTRUEncrypt: The NTRU Public Key Cryptosystem

- Independently of, and more-or-less simultaneously with GGH, Jeff Hoffstein, Jill Pipher, and I developed a public key cryptosystem that we called **NTRU**.
- The NTRU Public Key Cryptosystem solves the GGH issue of huge key size by using a special type of lattice having bases that can be described using roughly $\frac{1}{2}n\log_2(n)$ bits. This may be compared with GGH, whose keys require roughly n^2 bits.
- Actually, that's a bit of a lie. The NTRU lattice has dimension $2N$, and it has a subspace of dimension N that can be described using a single vector.
- Before describing the NTRU cryptosystem and its associated lattice, we need develop a bit more math.

Convolution Products

The **convolution product** of two vectors

$$\boldsymbol{a} = (a_0, a_1, \ldots, a_{N-1}) \quad \text{and} \quad \boldsymbol{b} = (b_0, b_1, \ldots, b_{N-1})$$

is the vector

$$\boldsymbol{c} = \boldsymbol{a} \star \boldsymbol{b} \quad \text{with} \quad c_k = \sum_{i+j \equiv k \pmod{N}} a_i b_j.$$

Vector addition and convolution product make the set of vectors into a ring, so for example

$$\begin{array}{rl} (\boldsymbol{a} \star \boldsymbol{b}) \star \boldsymbol{c} = \boldsymbol{a} \star (\boldsymbol{b} \star \boldsymbol{c}) & \text{Associative Law,} \\ \boldsymbol{a} \star (\boldsymbol{b} + \boldsymbol{c}) = \boldsymbol{a} \star \boldsymbol{b} + \boldsymbol{a} \star \boldsymbol{c} & \text{Distributive Law,} \\ \boldsymbol{a} \star \boldsymbol{b} = \boldsymbol{b} \star \boldsymbol{a} & \text{Commutative Law,} \\ \vdots & \vdots \end{array}$$

A Polynomial Description of the Convolution Ring

Alternatively, we identify vectors and polynomials by

$$\boldsymbol{a} \longleftrightarrow \boldsymbol{a}(X) = a_0 + a_1 X + \cdots + a_{N-1} X^{N-1},$$

and we multiply polynomials in the quotient ring

$$\mathcal{R} = \mathbb{Z}[X]/(X^N - 1)$$

using the multiplication rule $X^N = 1$. Then

$$\boldsymbol{c} = \boldsymbol{a} \star \boldsymbol{b} \quad \text{is the same as} \quad \boldsymbol{c}(X) \equiv \boldsymbol{a}(X)\boldsymbol{b}(X) \pmod{X^N - 1}.$$

Reducing the coefficients modulo q (or p), we can work in the ring

$$\mathcal{R}_q = (\mathbb{Z}/q\mathbb{Z})[X]/(X^N - 1).$$

It is then (generally) possible to find inverses, i.e.,

$$\boldsymbol{a}(X)\boldsymbol{a}(X)^{-1} \equiv 1 \pmod{q} \quad \text{for some } \boldsymbol{a}(X)^{-1} \in \mathcal{R}.$$

How NTRUEncrypt Works

Key Creation: Fix N, p, q, with N prime and with $\gcd(p,q) = 1$. Choose random polynomials $\boldsymbol{f}, \boldsymbol{g} \in \mathcal{R}$ with small coefficients. Compute inverses

$$\boldsymbol{F}_q \equiv \boldsymbol{f}^{-1} \pmod{q} \qquad \text{and} \qquad \boldsymbol{F}_p \equiv \boldsymbol{f}^{-1} \pmod{p}$$

and set

$$\boldsymbol{h} = \boldsymbol{g} \cdot \boldsymbol{F}_q \pmod{q}.$$

$$\boxed{\text{Public Key} = \boldsymbol{h} \quad \text{and} \quad \text{Private Key} = \boldsymbol{f} \ (\text{and } \boldsymbol{F}_p)}$$

Encryption: The plaintext $\boldsymbol{m}$ is a polynomial with mod p coefficients. Choose a random small polynomial $\boldsymbol{r}$. The ciphertext is

$$\boldsymbol{e} \equiv p \cdot \boldsymbol{r} \cdot \boldsymbol{h} + \boldsymbol{m} \pmod{q}.$$

Decryption: Compute

$$\boldsymbol{a} \equiv \boldsymbol{e} \cdot \boldsymbol{f} \pmod{q},$$

choosing the coefficients of $\boldsymbol{a}$ to satisfy $A \le a_i < A + q$. Then $\boldsymbol{F}_p \cdot \boldsymbol{a} \bmod p$ is equal to the plaintext $\boldsymbol{m}$.

Why NTRUEncrypt Works

The first decryption step gives the polynomial

Computation (mod q)	Reason
$\boldsymbol{a} \equiv \boldsymbol{e} \cdot \boldsymbol{f}$	
$\equiv (p \cdot \boldsymbol{r} \cdot \boldsymbol{h} + \boldsymbol{m}) \cdot \boldsymbol{f}$	$\boldsymbol{e} \equiv p \cdot \boldsymbol{r} \cdot \boldsymbol{h} + \boldsymbol{m}$
$\equiv p \cdot \boldsymbol{r} \cdot \boldsymbol{g} + \boldsymbol{m} \cdot \boldsymbol{f}$	$\boldsymbol{h} \cdot \boldsymbol{f} \equiv \boldsymbol{g} \cdot \boldsymbol{F}_q \cdot \boldsymbol{f} = \boldsymbol{g}$

The coefficients of $\boldsymbol{r}, \boldsymbol{g}, \boldsymbol{m}, \boldsymbol{f}$ are **small**, so the coefficients of

$$p \cdot \boldsymbol{r} \cdot \boldsymbol{g} + \boldsymbol{m} \cdot \boldsymbol{f}$$

will lie in an interval of length less than q. Choosing the appropriate interval, the polynomial

$$\boldsymbol{a} \quad \textbf{equals} \quad p \cdot \boldsymbol{r} \cdot \boldsymbol{g} + \boldsymbol{m} \cdot \boldsymbol{f} \quad \textbf{exactly},$$

and not merely modulo q. Now multiply by $\boldsymbol{F}_p$.

$$\begin{aligned} \boldsymbol{F}_p \cdot \boldsymbol{a} &= \boldsymbol{F}_p \cdot (p \cdot \boldsymbol{r} \cdot \boldsymbol{g} + \boldsymbol{m} \cdot \boldsymbol{f}) \\ &\equiv \boldsymbol{F}_p \cdot \boldsymbol{m} \cdot \boldsymbol{f} \pmod{p} \\ &\equiv \boldsymbol{m} \pmod{p} \qquad \text{since } \boldsymbol{F}_p \cdot \boldsymbol{f} \equiv 1 \pmod{p}. \end{aligned}$$

NTRU and SVP/CVP

The **Convolution Modular Lattice** $L_{\boldsymbol{h}}$ associated to the vector $\boldsymbol{h}$ and modulus q is the $2N$ dimensional lattice with basis given by the rows of the matrix:

$$L_{\boldsymbol{h}} = \operatorname{RowSpan} \left(\begin{array}{cccc|cccc} 1 & 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_{N-1} \\ 0 & 1 & \cdots & 0 & h_{N-1} & h_0 & \cdots & h_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & h_1 & h_2 & \cdots & h_0 \\ \hline 0 & 0 & \cdots & 0 & q & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & q \end{array} \right)$$

Another way to describe $L_{\boldsymbol{h}}$ is the set of vectors

$$L_{\boldsymbol{h}} = \left\{ (\boldsymbol{a}, \boldsymbol{b}) \in \mathbb{Z}^{2N} : \boldsymbol{a} \star \boldsymbol{h} \equiv \boldsymbol{b} \pmod{q} \right\}.$$

Finding an NTRU Private Key as an SVP Problem

An NTRU public/private key pair is given by

$$\boldsymbol{f} \star \boldsymbol{h} \equiv \boldsymbol{g} \pmod{q} \quad \text{with "small" } \boldsymbol{f} \text{ and } \boldsymbol{g}.$$

This formula implies that the lattice $L_{\boldsymbol{h}}$ contains the short (likely shortest non-zero) vector

$$[\boldsymbol{f}, \boldsymbol{g}] = [f_0, f_1, \ldots, f_{N-1}, g_0, g_1, \ldots, g_{N-1}].$$

To see that $[\boldsymbol{f}, \boldsymbol{g}]$ is in $L_{\boldsymbol{h}}$, let

$$\boldsymbol{u} = \frac{\boldsymbol{g} - \boldsymbol{f} \star \boldsymbol{h}}{q} \in \mathbb{Z}^N.$$

Then

$$[f_0, \ldots, f_{N-1}, u_0, \ldots, u_{N-1}] \begin{pmatrix} 1 & \cdots & 0 & h_0 & \cdots & h_{N-1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & h_1 & \cdots & h_0 \\ 0 & \cdots & 0 & q & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & q \end{pmatrix} = [f_0, \ldots, f_{N-1}, g_0, \ldots, g_{N-1}].$$

Finding an NTRU Plaintext as a CVP Problem

Recall that an NTRU ciphertext $\boldsymbol{e}$ has the form

$$\boldsymbol{e} = p\boldsymbol{r} \star \boldsymbol{h} + \boldsymbol{m} \pmod{q} \quad \text{with "small" } \boldsymbol{r} \text{ and } \boldsymbol{m}.$$

We can rewrite this relation in vector form as

$$\begin{aligned}[0, \boldsymbol{e}] &= [0, p\boldsymbol{r} \star \boldsymbol{h} + \boldsymbol{m} \mod q] \\ &\equiv [\boldsymbol{r}, \boldsymbol{r} \star (p\boldsymbol{h}) \mod q] + [-\boldsymbol{r}, \boldsymbol{m}].\end{aligned}$$

The vector $[\boldsymbol{r}, \boldsymbol{r} \star (p\boldsymbol{h}) \mod q]$ is in the convolution modular lattice $L_{p\boldsymbol{h}}$ obtained by using $p\boldsymbol{h}$ in place of $\boldsymbol{h}$. Further, the vector $[-\boldsymbol{r}, \boldsymbol{m}]$ is quite short.

Conclusion. Recovering the plaintext $\boldsymbol{m}$ from the ciphertext $\boldsymbol{e}$ is equivalent to finding the vector in $L_{p\boldsymbol{h}}$ that is closest to the vector $[0, \boldsymbol{e}]$.

It is then a question of estimating how hard it is to solve this CVP problem.

NTRU Notes

- The NTRU lattice has a sort of rotational symmetry, in the sense that
$$[X^i \boldsymbol{f}, X^i \boldsymbol{g}] \in L_{\boldsymbol{h}} \quad \text{for all } 0 \le i \le N-1.$$
Thus $L_{\boldsymbol{h}}$ is a $2N$-dimensional lattice that contains an N-dimensional subspace spanned by N independent short vectors.
- NTRU decryption may be formulated as solving CVP in this hidden N-dimensional subspace, more-or-less via Babai's method with the short partial basis.
- One should take N to be prime. Otherwise the NTRU key/message recovery problems may become lattice problems in lower dimension. This works if N is divisble by a small'ish prime ℓ. (If ℓ is big, the target vector gets larger and is lost in a sea of exponentially many similar length vectors.) But for example, it is a very bad idea to take N to be even!

NTRU Variants

Many variants of NTRU have been proposed. E.g.

Replace $X^N - 1$ with $X^N + 1$, where $N = 2^k$.

(This makes $X^N + 1$ is irreducible in $\mathbb{Z}[X]$.)

Replace $X^N - 1$ with an arbitrary (monic, irreducible, small coefficient) polynomial $\boldsymbol{\varphi}(X) \in \mathbb{Z}[X]$.

One can then look at sublattices of the **ideal lattice**

$$\left(\mathbb{Z}[X]/\boldsymbol{\varphi}(X)\mathbb{Z}[X]\right)^2.$$

The key to this construction lies in the fact that if $\boldsymbol{a}(X)$ and $\boldsymbol{b}(X)$ have small coefficients, then

$\boldsymbol{a}(X) \cdot \boldsymbol{b}(X) \bmod \boldsymbol{\varphi}(X)$ had small'ish coefficients.

The average coefficient size of the product depends on the size of the roots of $\boldsymbol{\varphi}(X)$. So taking $\boldsymbol{\varphi}(X)$ to be a cyclotomic polynomial is good. OTOH, non-cyclotomic polynomials kill symmetries in the cyclotomic lattice.

An Introduction to Lattices, Lattice Reduction, and Lattice-Based Cryptography

Joseph H. Silverman

Brown University

PCMI Lecture Series

July 6–10, 2020