Improved Decryption Bounds and Key Generation for Matrix NTRU over Integral Domain

Thiago do Rêgo Sousa [1], **Tertuliano Souza Neto**[1]

[1] Centro de Pesquisa e Desenvolvimento para a Segurança das Comunicações (Cepesc)

thiagodoregosousa@gmail.com, tsouzaneto@gmail.com

Abstract. *Shor's algorithm [Shor 1994] is the main threat to classical public-key cryptography. Since its introduction in 1996, NTRU and its variants aim to develop cryptographic algorithms that are secure even against quantum computers. In this work, we study the matrix NTRU over integral domains proposed in 2023. We found that there is an error on the condition to avoid decryption failures and the key generation process is not practical due to severe limitations on matrix inversion. We propose a corrected statement for the decryption failure theorem and an expansion of the set of solutions when dealing with the problem of inverting matrix in* $M_n(\mathbb{Z}[\sqrt{-3}])$ *that makes the key generation significantly faster.*

1. Introduction

Shor's algorithm [Shor 1994] is the main threat to classical public-key cryptography. Since it can efficiently factor large integers and compute discrete logarithms, Shor's algorithm leads directly to attacks against well known and widely used cryptosystems such as RSA [Rivest et al. 1978] and Eliptic Curve Cryptography ([Koblitz 1987], [Miller 1985]).

The combination of Shor's algorithm and quantum computers, as well as the application of Grover's algorithm [Grover 1996], led the cryptographic community to develop what is called today's post-quantum cryptography (PQC). Since then, designing robust cryptographic schemes underpinned on mathematical problems that are intractable by both quantum and conventional computers is a fast growing research topic and the development of PQC has taken off in the last decade.

The National Institute of Standards and Technology (NIST) launched a PQC standardization project in 2016 [NIST]. The competition aimed to select and standardize cryptographic algorithms that are secure even against quantum computers. After four rounds and six years, NIST selected one algorithm for public-key encryption/KEM and three algorithms for digital signature. Two lattice-based systems have progressed significantly in the NIST Post-Quantum Cryptography standardization process [NIST]: the NTRU [Chen et al. 2020] and the NTRU Prime systems [Daniel J. Bernstein et al. 2024], both based on the original NTRU system from [Hoffstein et al. 1998].

Lattice-Based Cryptography relies on the difficulty of solving lattice-related problems, which are believed to be hard to solve even for a quantum computer. Among those problems, we have the NTRU-related ones like SVP (Shortest Vector Problem) and CVP (Closest Vector Problem), both believed to be quantum-resistant.

Introduced at CRYPTO'96, the NTRU cryptosystem has since undergone extensive analysis and numerous extensions. These include variations in polynomial coefficients and the adoption of matrix-based structures. Comprehensive reviews can be found in works such as [Singh and Padhye 2016, Salleh and Kamarulhaili 2020, Mittal and Ramkumar 2022]

Regarding NTRU and the NIST PQC standardization process, a variant called NTRUEncrypt and its relatives were submitted and NTRUEncrypt was selected as an alternate candidate, which means that it wasn't standardized at first, but it is considered valuable and can still be used or standardized later. The growing interest on NTRU algorithm is because it has interesting properties, such as speed and small key sizes compared to some other PQC algorithms, resistance to both classical and quantum attacks, and easy adaptation into newer variants.

Since the introduction of NTRU, many variants have been proposed in the literature [Salleh and Kamarulhaili 2020]. The first NTRU generalization using matrices was introduced in [Coglianese and Goi 2005], where key generation, encryption and decryption operate over matrices whose entries are polynomials. In 2008, [Nayak et al. 2008] proposed a matrix-based system as a variant of NTRU using only matrices with integer entries, replacing the arithmetic of truncated polynomials with a simple modular arithmetic on matrices. This new system, called matrix NTRU, was subject to some further analysis and even suggested for real data applications. However, [Sousa and Neto 2024] showed that Matrix NTRU is completely insecure and should be avoided in practical applications.

More recently, [Wijayanti et al. 2023] developed a variant of NTRU which is based on matrix arithmetic defined over $\mathbb{Z}[\sqrt{-3}]$ called Matrix NTRU over Integral Domains. We can say that this system is a Matrix NTRU variant, where the authors replaced the underlined ring of the entries of the matrices. Of course they had to make some changes in the key generation process and the approach of decryption failures.

When diving into mathematical details and in implementing [Wijayanti et al. 2023]'s cryptosystem, we found out that there is an error on the condition to avoid decryption failures and the key generation process is not practical due to severe limitations on matrix inversion.

Our main contribution to matrix NTRU over integral domain is twofold. First we made a more intensive simulation study and found a counterexample to [Wijayanti et al. 2023, Theorem 2] and these results, together with [Sousa and Neto 2024, Proposition 1], led us to propose a corrected statement for the decryption failure theorem. Second, we found that the conditions from the proof of [Wijayanti et al. 2023, Theorem 1] used in their paper for the key generation process are overly restrictive making it very inefficient (around 14 minutes to generate keys for matrix of dimension 7×7). To solve this problem we propose the expansion of the set of solutions when dealing with the problem of inverting matrix in $M_n(\mathbb{Z}[\sqrt{-3}])$ lowering the time for key generation of 7×7 matrices to less than one second.

The rest of the paper is organized as follows. In Section 2, we give some necessary background on Algebraic Number Theory with respect to modular algebra over matrices as used in [Wijayanti et al. 2023]. In Section 3 we show how the algorithm works in details and discuss their decryption failure theorem and the limitations of the key generation process. In Section 4 we give a detailed example that abides to the conditions of [Wijayanti et al. 2023, Theorem 2] to avoid decryption failure, but the ciphertext does not correctly decrypts to the original message. Then we correct the decryption failure condition and make an intensive simulation study on critical parameters to support the result. In Section 5 we dive deeper into the conditions for key generation, propose an extension of the solution space for finding matrix inverses and conclude with an experimental study showing that this extended process not only works but is significantly faster than [Wijayanti et al. 2023]'s original algorithm. Finally, we conclude in Section 6 and give further research directions.

2. Background

Let's consider the ring

$$\mathbb{Z}[\sqrt{-3}] = \{a + b\sqrt{-3}, a, b \in \mathbb{Z}\}.$$

From Algebraic Number Theory, it is well known that $\mathbb{Z}\sqrt{-3}]$ is an integral domain. For any $\alpha = a + b\sqrt{-3} \in \mathbb{Z}[\sqrt{-3}]$ and p a prime number, we define

$$\alpha \mod p = (a \mod p) + (b \mod p)\sqrt{-3}.$$

We define $M_n(\mathbb{Z}[\sqrt{-3}])$ as the set of $n \times n$ matrices whose coefficients belong to $\mathbb{Z}[\sqrt{-3}]$.

As is common in cryptosystems like NTRU and its variants, we are going to work with ternary elements. In the case of Matrix NTRU over Integral Domain being ternary simply means that the real and imaginary parts of its entries are in $\in \{-1, 0, 1\}$.

Let p be a prime number and let

$$A = \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{pmatrix}$$

be a matrix in $M_n(\mathbb{Z}[\sqrt{-3}])$. Reduction modulo p in $M_n(\mathbb{Z}[\sqrt{-3}])$ is quite straightforward. We say that A is reduced modulo p, denoted as $A \mod p$, if every entry in A is reduced modulo p. Therefore

$$A \mod p = \begin{pmatrix} a_{11} \mod p & \dots & a_{1n} \mod p \\ \vdots & \ddots & \vdots \\ a_{n1} \mod p & \dots & a_{nn} \mod p \end{pmatrix}. \tag{1}$$

We can also define sum and multiplication of matrices modulo p. Let A and B be matrices in $M_n(\mathbb{Z}[\sqrt{-3}])$. Operation $A + B \mod p$ is done component-wise, so we can define

$$(A + B) \mod p = (A \mod p + B \mod p) \mod p.$$

In the same way, multiplication is defined as

$$(A * B) \mod p = (A \mod p * B \mod p) \mod p.$$

A careful reader will notice that the modulo p operation is not always canonical. That is common in NTRU-like cryptosystems, because of what is called center lift operation, which is necessary for the decryption process work properly. In that case, it can be useful to consider a non-canonical representation of the elements on the rings on what the coefficients of the matrices belong to. Therefore, when we are dealing with a prime modulus, we have

$$\mathbb{F}_p = \left\{ -\frac{p-1}{2}, \dots, \frac{p-1}{2} \right\}$$

and for a composite number, we have

$$\mathbb{Z}_q = \left\{ -\frac{q}{2} + 1, \dots, \frac{q}{2} \right\}.$$

Since modular arithmetic over the rings $M_n(\mathbb{Z}[\sqrt{-3}])$ is crucial for the Matrix NTRU cryptosystem over Integral Domain work properly, we need to address the question of how to invert matrices modulo p in that ring. First of all, a matrix A in this ring is invertible modulo p if and only if p and the determinant of A are relatively prime [Wijayanti et al. 2023]. If that is the case, then there exists a (unique) matrix B such that $AB = BA = I_n$ in $M_n(\mathbb{Z}[\sqrt{-3}])$, where I_n is the $n \times n$ identity matrix. The matrix B is called the inverse of A modulo p and denoted $A^{-1} \mod p$.

3. Matrix NTRU Cryptosystem over Integral Domain

3.1. How the algorithm works

As we already know, Matrix NTRU over Integral Domain cryptosystem is defined over matrices whose coefficients are in the ring $\mathbb{Z}[\sqrt{-3}]$. We have also noticed that we have to deal with the question on how to invert matrices in a correct and efficient way.

Let $A \in M_n(\mathbb{Z}[\sqrt{-3}])$ and suppose $\det(A) \neq 0$. If $\gcd(\det(A), p) = 1$, then A is invertible modulo p and we can write

$$A^{-1} \mod p = (\det(A))^{-1} \mathrm{Adj}(A),$$

where $\mathrm{Adj}(A)$ represents the adjoint matrix of A. The previous result is a classical one regarding matrix theory and the interested reader can see [Hungerford 2012] for a proof.

The method that can be used to invert the matrices is really important if one wants to develop a secure and efficient key generation process. The authors in [Wijayanti et al. 2023] modify the algorithm in the original development of Matrix NTRU cryptosystem [Nayak et al. 2008] to replace the condition for matrix inversion. Although their choice brought interesting properties, we show that it has serious limitations with a huge impact on the speed of key generation, as we will show later in this work.

In what follows, let $\mathcal{L}(d)$ be the set of all matrices in $M_n(\mathbb{Z}[\sqrt{-3}])$ with d coefficients equal to 1, d coefficients equal to -1 and $n^2 - 2d$ entries equal to zero. In [Wijayanti et al. 2023], they take the values for d near $\dfrac{2n^2}{3}$ and we follow them to keep our results comparable to theirs. As an example, we use $d = 5$ for $n = 3$ just like them for some specific experiments here, although the precise d value should be 6 according to the formula.

Matrix NTRU over Integral Domains works as follows.

1. Key Generation: Let p and q be prime numbers. The private and public keys are $n \times n$ matrices randomly chosen from the set $\mathcal{L}(d)$. First, we choose a pair of ternary matrices F, G such that $F \mod p$ and $F \mod q$ are invertible. Then we compute the matrices $F_p = F^{-1} \mod p$ and $F_q = F^{-1} \mod q$. The parameters F, G, F_p, F_q should be kept secret. Now, we can compute the public key by performing the calculation
$$H = pF_qG \mod q.$$
2. Encryption: To encrypt a message, we encode it as a matrix M whose coefficients are in the form $a + b\sqrt{-3}$ with $a, b \in \mathbb{F}_p$. Next, we generate a random ternary matrix $R \in M_n(\mathbb{Z})$. Now, we compute the ciphertext as
$$E = HR + M \mod q.$$
3. Decryption: To decrypt, we compute
$$A = FE \mod q$$
applying the center-lift operation, which means that all A coefficients have the form $a + b\sqrt{-3}$ with $a, b \in \mathbb{F}_q$. Now we compute $B = F_pA \mod p$ and the calculation of
$$C = F_pB \mod p$$
gives the desired plaintext.

The reader who wants to be sure that the decryption algorithm works can take a look at [Wijayanti et al. 2023].

3.2. Decryption failure in detail and Key generation process

Depending on the parameters selected, there can be decryption errors when operating on Matrix NTRU over Integral Domains just like other NTRU variants cryptosystems and even NTRU itself. That can happen because matrices C and M, as defined on previous subsection, are not always equal and we say there is a decryption failure in that case. This is an important issue concerning any probabilistic cryptosystem since attackers can take advantage of it, recovering secret data by exploring those decryption failures (see [Jaulmes and Joux 2000] or [Gama and Nguyen 2007]). This justifies the importance of having sufficient conditions under which decryption failures are avoided.

We notice that the authors [Wijayanti et al. 2023] did address that question, although there are corrections to perform regarding decription failures on Matrix NTRU over Integral Domains.

Theorem 1 (Theorem 2 in [Wijayanti et al. 2023], Incorrect)**.** *If the parameters n, p, q, d satisfy*
$$q > 3\sqrt{7}np,$$
then the system decrypts correctly.

As we demonstrate in the next section, Theorem 1 is incorrect, as shown by a counterexample. Fortunately, we are able to correct the decryption failure condition and validate it extensively, particularly near the boundary cases, to support the new claim alongside the mathematical proofs.

Regarding the key generation process, although the method the authors [Wijayanti et al. 2023] applied for matrix inversion is correct, we discovered that a slight modification can significantly accelerate the generation of the coefficients in Theorem 1 from [Wijayanti et al. 2023].

In a short summary, we could say that the authors in [Wijayanti et al. 2023] made a severe restriction when dealing with the generation of the coefficients on the Extended Euclidean Algorithm in the ring $\mathbb{Z}[\sqrt{-3}]$. We notice that the same is true regarding the determinant of the private key F. We show how to speed up the key generation by letting the coefficients α and β in Theorem 1 of [Wijayanti et al. 2023] take values in $\mathbb{Z}[\sqrt{-3}]$ as well as the determinant of the matrix F.

4. A Counterexample and Correction to the Decryption Failure Theorem

In such public key system it is advisable to provide necessary conditions under which decryption failure does not happen. An attempt to provide such condition was done in [Wijayanti et al. 2023] and is described in Theorem 1. For $n = 3, p = 3$ this condition boils down to $q > 71.44$. The authors present examples where they tested the values for $n = 3, p = 3$ and q varying from 13 to 673, but they tested the encryption and decryption process only 100 times for each case which was not enough to spot inconsistencies in their result. They spotted no errors for $q = 73$ which abides to the condition $q > 71.44$ but as we show in Example 4.1, $q = 73$ gives indeed decryption failure when tested more intensively.

Example 4.1 (Counterexample to Theorem 1)**.** *This is an example of a decryption failure for $N = 3, p = 3, q = 73, d = 5$, where the chosen parameters satisfy the theorem's hypotheses and should, in principle, permit correct decryption of the ciphertext. However, as we detail below, this does not occur. The private keys chosen are F and G:*

$$F = \begin{bmatrix} 0-\sqrt{-3} & 0 & -1 \\ -1 & 1-\sqrt{-3} & -1 \\ \sqrt{-3} & 1+\sqrt{-3} & 1 \end{bmatrix} \quad G = \begin{bmatrix} -\sqrt{-3} & -1+\sqrt{-3} & 0 \\ 0 & -1 & 1 \\ -\sqrt{-3} & 1-\sqrt{-3} & 1+\sqrt{-3} \end{bmatrix}$$

The corresponding inverses F_p and F_q are:

$$F_p = \begin{bmatrix} 2 & 2+2\sqrt{-3} & 2+\sqrt{-3} \\ 1+2\sqrt{-3} & 0 & 1+2\sqrt{-3} \\ 2+\sqrt{-3} & \sqrt{-3} & 2\sqrt{-3} \end{bmatrix}$$

and

$$F_q = \begin{bmatrix} 37 & 18+18\sqrt{-3} & 55+18\sqrt{-3} \\ 55+18\sqrt{-3} & 0 & 55+18\sqrt{-3} \\ 72+36\sqrt{-3} & 54+55\sqrt{-3} & 54+18\sqrt{-3} \end{bmatrix}$$

The public key is $H = pF_qG$:

$$H = \begin{bmatrix} 16+16\sqrt{-3} & 19+16\sqrt{-3} & 57+54\sqrt{-3} \\ 32+35\sqrt{-3} & 0 & 3 \\ 48+60\sqrt{-3} & 60+54\sqrt{-3} & 16+16\sqrt{-3} \end{bmatrix}$$

Consider we want to encrypt the message M using a random matrix R:

$$M = \begin{bmatrix} -\sqrt{-3} & \sqrt{-3} & 1-\sqrt{-3} \\ 1 & -1+\sqrt{-3} & -1 \\ -\sqrt{-3} & \sqrt{-3} & 0 \end{bmatrix} \quad R = \begin{bmatrix} \sqrt{-3} & -1-\sqrt{-3} & \sqrt{-3} \\ \sqrt{-3} & -1-\sqrt{-3} & -1 \\ 1 & \sqrt{-3} & 0 \end{bmatrix}$$

The corresponding ciphertext is computed as $E = pHR + M \pmod{q}$:

$$E = \begin{bmatrix} 25+12\sqrt{-3} & 57+64\sqrt{-3} & 10+69\sqrt{-3} \\ 45+32\sqrt{-3} & 72+10\sqrt{-3} & 40+32\sqrt{-3} \\ 39+50\sqrt{-3} & 40+14\sqrt{-3} & 52+67\sqrt{-3} \end{bmatrix}$$

To decrypt, we first compute $A = FE \pmod{q}$:

$$A = \begin{bmatrix} 70+71\sqrt{-3} & 6+2\sqrt{-3} & 9+69\sqrt{-3} \\ 4+71\sqrt{-3} & 5+6\sqrt{-3} & 1+2\sqrt{-3} \\ 25+6\sqrt{-3} & 36+7\sqrt{-3} & 8+3\sqrt{-3} \end{bmatrix}$$

and

$$A_{\textit{center\_lift}} = \begin{bmatrix} -3-2\sqrt{-3} & 6+2\sqrt{-3} & 9-4\sqrt{-3} \\ 4-2\sqrt{-3} & 5+6\sqrt{-3} & 1+2\sqrt{-3} \\ 25+6\sqrt{-3} & 36+7\sqrt{-3} & 8+3\sqrt{-3} \end{bmatrix}$$

Then we compute $B = A \pmod{p}$:

$$B = \begin{bmatrix} \sqrt{-3} & 2\sqrt{-3} & 2\sqrt{-3} \\ 1+\sqrt{-3} & 2 & 1+2\sqrt{-3} \\ 1 & \sqrt{-3} & 2 \end{bmatrix} \quad B_{\textit{center\_lift}} = \begin{bmatrix} \sqrt{-3} & -\sqrt{-3} & -\sqrt{-3} \\ 1+\sqrt{-3} & -1 & 1-\sqrt{-3} \\ 1 & \sqrt{-3} & -1 \end{bmatrix}$$

Finally, we compute $C = F_pB \pmod{p}$:

$$C = \begin{bmatrix} 2\sqrt{-3} & 1 & 1+2\sqrt{-3} \\ 1 & 0 & 2 \\ 2\sqrt{-3} & 0 & 0 \end{bmatrix} \quad C_{\textit{center\_lift}} = \begin{bmatrix} -\sqrt{-3} & 1 & 1-\sqrt{-3} \\ 1 & 0 & -1 \\ -\sqrt{-3} & 0 & 0 \end{bmatrix}$$

Note that $C \neq M$.

We found that [Wijayanti et al. 2023] did not took into account the fact that decryption failures can occur if any real or complex component of the matrices involved during decryption is not smaller than $q/2$.

A corrected statement of a necessary condition for decryption failure will be given in the next Section which also includes further experimental investigation of boundary cases to strengthen and test this new condition. The corrected condition requires $q > 96$, i.e., the minimum value of q to create a system without decryption failure is $q = 97$.

4.1. Correcting the decryption failure condition

Theorem 2. *Suppose that the system parameters n, p, q satisfy*

$$q > 8n\left(p + \left\lceil \frac{p-1}{2} \right\rceil\right).$$

Then the system decrypts correctly.

Proof. We first determine the shape of the calculation of A during decryption. Considering that

$$A \equiv pGR + FM(\mod q)$$

we need to restrict its largest possible entries of $pGR + FM$.

More specifically, none of the components (real or imaginary) of any entry of the above matrix should exceed $q/2$ in order for the decryption to work (see [Silverman et al. 2008, Proposition 6.48]). This is a direct consequence of the fact that in the matrix NTRU over integral domain, the module q operation for matrices was defined component-wise in (1).

Since the product of the matrices involves products of elements in $Z(\sqrt{-3})$ we start by considering the sets

$$\mathbb{T}_1 = \{-1, 0, 1\}$$

and

$$\mathbb{T}_k = \{-k, \dots, 0, \dots, k\}, \quad k \in \mathbb{Z}.$$

Given arbitrary $\alpha = a + b\sqrt{-3}, \beta = c + d\sqrt{-3} \in \mathbb{Z}[\sqrt{-3}]$ we have

$$\alpha\beta = (a + b\sqrt{-3})(c + d\sqrt{-3}) = ac - 3bd + (ad + bc)\sqrt{-3},$$

If $a, b, c, d \in \mathbb{T}_1$ then

$$\max\{ac - 3bd\} = 4 \quad \text{and} \quad \max\{ad + bc\} = 2$$

so the maximum absolute value of the entries of $\alpha\beta$ in this scenario is 4. In a similar way, If $a, b \in \mathbb{T}_1$ and $c, d \in \mathbb{T}_k$ then

$$\max\{ac - 3bd\} = k - 3(-k) = 4k \quad \text{and} \quad \max\{ad + bc\} = k + k = 2k$$

so the maximum absolute value of the entries of $\alpha\beta$ in this second scenario is $4k$.

Now we will bound the largest value of the components of the matrices involved in

$$A \equiv pGR + FM \mod q.$$

Since both G and R are in $\mathcal{L}$, their entries are elements whose components are in $\mathbb{T}_1$ and therefore the largest entry of pGR is $4np$. Now for the product FM, the entries of F have components in $\mathbb{T}_1$ but the entries of M have components in $\mathbb{T}_k$ with $k = \lceil \frac{p-1}{2} \rceil$. Therefore, by the same argument we see that the largest entry is $n(4\lceil \frac{p-1}{2} \rceil)$.

Putting it all together, the largest entry in $pGR + FM$ is $4n(p + k)$ and since it should be smaller than $q/2$ ([Silverman et al. 2008, proof of Proposition 6.48]) the decryption failure condition boils down to

$$q/2 > 4n\left(p + \left\lceil \frac{p-1}{2} \right\rceil\right),$$

which completes the proof. $\square$

For $p = 3, n = 3$ the condition is $q > 96$ which differs from the one found by [Wijayanti et al. 2023] ($q > 71.44$ and it is now clear why the previous incorrect condition would be prone to decryption failure.

4.2. Experimental Support for Theorem 2 Correctness in Boundary Scenarios

In order to assess empirically the correctness of our new decryption failure Theorem 2 it is crucial to test it at the boundary, which the decryption failure condition is tested more often. Fixing $p = 3, n = 3$ in Theorem 2 gives the condition $q > 96$. Since q must be prime, the values $q = 89$ and $q = 97$ are particularly interesting: for $q = 89$, decryption is expected to fail at some point, while for $q = 97$, no failure should occur. The example below illustrates the case $q = 89$, listing all matrices generated that satisfy the system's hypotheses and demonstrating that decryption fails. For $q = 97$ we do intensive encryption and decryption testing.

Example 4.2 (Testing boundary values of Theorem 2)**.** *Let* $p = 3, n = 3, q = 89, d = 5.$ *The value* $q = 89$ *corresponding to a prime slightly smaller than the required restriction* $q > 96$*, which tell us there might be a decryption failure. The private keys chosen are*

$$F = \begin{bmatrix} -1-\sqrt{-3} & 1-\sqrt{-3} & -1-\sqrt{-3} \\ 1 & \sqrt{-3} & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

and

$$G = \begin{bmatrix} -1-\sqrt{-3} & 1-\sqrt{-3} & -1-\sqrt{-3} \\ 0 & 1 & 1 \\ \sqrt{-3} & 0 & 1 \end{bmatrix}.$$

The corresponding inverses F_p *and* F_q *are:*

$$F_p = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1+\sqrt{-3} & 0 \\ 2\sqrt{-3} & 1+2\sqrt{-3} & 2 \end{bmatrix} \quad F_q = \begin{bmatrix} 0 & 0 & 1 \\ 44 & 44+44\sqrt{-3} & 0 \\ 45\sqrt{-3} & 44+45\sqrt{-3} & 88 \end{bmatrix}$$

The public key is $H = pF_qG$*:*

$$H = \begin{bmatrix} 3\sqrt{-3} & 0 & 3 \\ 46+46\sqrt{-3} & 86 & 0 \\ 49+40\sqrt{-3} & 3+3\sqrt{-3} & 0 \end{bmatrix}$$

Consider we want to encrypt the message M using a random matrix R:

$$M = \begin{bmatrix} 1-\sqrt{-3} & 0 & 0 \\ -1-\sqrt{-3} & 0 & \sqrt{-3} \\ 1-\sqrt{-3} & -1+\sqrt{-3} & 1 \end{bmatrix} \quad R = \begin{bmatrix} 1-\sqrt{-3} & \sqrt{-3} & 1 \\ -1-\sqrt{-3} & 0 & -1 \\ 1-\sqrt{-3} & 1 & 0 \end{bmatrix}$$

The corresponding ciphertext is computed as $E = pHR + M \pmod q$:

$$E = \begin{bmatrix} 13+88\sqrt{-3} & 83 & 3\sqrt{-3} \\ 8+2\sqrt{-3} & 40+46\sqrt{-3} & 49+47\sqrt{-3} \\ 87+73\sqrt{-3} & 57+50\sqrt{-3} & 47+37\sqrt{-3} \end{bmatrix}$$

To decrypt, we first compute $A = FE \pmod q$:

$$A = \begin{bmatrix} 41 & 10+83\sqrt{-3} & 85 \\ 5+80\sqrt{-3} & 2+\sqrt{-3} & 84 \\ 13+88\sqrt{-3} & 83 & 3\sqrt{-3} \end{bmatrix}$$

and

$$A_{\textit{center\_lift}} = \begin{bmatrix} 41 & 10-6\sqrt{-3} & -4 \\ 5-9\sqrt{-3} & 2+\sqrt{-3} & -5 \\ 13-\sqrt{-3} & -6 & 3\sqrt{-3} \end{bmatrix}$$

Then we compute $B = A \pmod p$:

$$B = \begin{bmatrix} 2 & 1 & 2 \\ 2 & 2+\sqrt{-3} & 1 \\ 1+2\sqrt{-3} & 0 & 0 \end{bmatrix} \quad B_{\textit{center\_lift}} = \begin{bmatrix} -1 & 1 & -1 \\ -1 & -1+\sqrt{-3} & 1 \\ 1-\sqrt{-3} & 0 & 0 \end{bmatrix}$$

Finally, we compute $C = F_pB \pmod p$:

$$C = \begin{bmatrix} 1+2\sqrt{-3} & 0 & 0 \\ 1+2\sqrt{-3} & 0 & \sqrt{-3} \\ 1 & 2+\sqrt{-3} & 1 \end{bmatrix} \quad C_{\textit{center\_lift}} = \begin{bmatrix} 1-\sqrt{-3} & 0 & 0 \\ 1-\sqrt{-3} & 0 & \sqrt{-3} \\ 1 & -1+\sqrt{-3} & 1 \end{bmatrix}$$

Note that $C_{\textit{center\_lift}} \neq M$ (specifically at entries (2,1) and (2,2)).

The value for $q = 97$ is worth being intensively tested. We had crafted two scenarios. The first one has private keys, messages M and random R's crafted in order to maximaze the magnitude of the entry at position (1,1) of the matrix $pGR + FM$ to force a possible decryption failure. We tested 10^7 times and no decryption failure were found. We have also devised another experimental setting where both keys and messages were chosen randomly in $\mathcal{L}(d)$, and no decryption failure appeared after running the experiment 10^6 times.

We have also devised an experiment varying the values of n and d. The value of q was choosen to be the smallest prime satysfying the conditions of Theorem 2. The results are shown in Table 1.

Table 1. Decryption failure rate for the NTRU-like matrix scheme using the faster and more general key generation process. Here, q is the smallest prime satisfying Theorem 2, and d is approximately $\frac{2}{3}n^2$ as in [Wijayanti et al. 2023].

n	q	d	Number of Decryption Failure
3	97	6	0
4	131	11	0
5	163	17	0
6	193	24	0
7	227	33	0
10	331	67	0
15	487	150	0

5. Key Generation: Limitations, Corrections, and Efficient Approaches

5.1. Limitations of Existing Key Generation Methods and challenges

The key generation process proposed in [Wijayanti et al. 2023] for the matrix NTRU over integral domain require very strict conditions on the determinant of F in order for it to be invertible modulo p and q. They require the existence of integers α and β such that $\alpha\text{det}(F) + \beta p = 1$, and similarly for q. This led to a very slow key generation process, which was reported in [Wijayanti et al. 2023, Table III], where key generation for $n = 7$ takes around 14 minutes on a personal computer.

The authors argue that such slow process is probably due to the non existence of the determinant inverse modulo p or q, but we show that if one allows both α and β to be in $\mathbb{Z}[\sqrt{-3}]$, then the inverse of F is not only fast to calculate but it exists in most cases. In fact, valid keys can be generated for $n = 7$ in 0.01 seconds, which significantly improves over the process proposed in [Wijayanti et al. 2023].

5.2. Efficient Key Generation Using the Extended Euclidean Algorithm in $\mathbb{Z}[\sqrt{-3}]$ and a simulation study of its efficiency

A critical step in the Matrix-NTRU cryptosystem over the ring $\mathbb{Z}[\sqrt{-3}]$ is the generation of a matrix $F \in M_n(\mathbb{Z}[\sqrt{-3}])$ that admits modular inverses modulo two distinct primes p and q. That is, we require $F^{-1} \bmod p$ and $F^{-1} \bmod q$ to exist in order to compute the private and public key pair. This condition depends directly on the invertibility of $\det(F)$ in the ring $\mathbb{Z}[\sqrt{-3}]$.

In [Wijayanti et al. 2023]'s approach, this invertibility was achieved by requiring the solution of the Diophantine equation

$$\alpha \det(F) + \beta p = 1$$

with $\alpha, \beta \in \mathbb{Z}$. This restriction dramatically impacts efficiency: since $\det(F)$ lies in $\mathbb{Z}[\sqrt{-3}]$, finding integer coefficients satisfying the above equation becomes increasingly unlikely as n grows, leading to impractical key generation times. For instance, for $n = 7$, it was reported that generating a valid matrix F could take over 10 minutes, severely limiting scalability (see [Wijayanti et al. 2023, Table III]).

To overcome this bottleneck, we propose lifting the restriction that α, β must be integers. Instead, we solve the extended Euclidean equation in the domain $\mathbb{Z}[\sqrt{-3}]$ itself.

That is, given a prime p, we apply the extended Euclidean algorithm in $\mathbb{Z}[\sqrt{-3}]$ to find $\alpha, \beta \in \mathbb{Z}[\sqrt{-3}]$ satisfying:

$$\alpha \det(F) + \beta p = 1.$$

When such α exists, we define the modular inverse of F modulo p as:

$$F^{-1} \bmod p = \alpha \text{adj}(F),$$

where $\text{adj}(F)$ denotes the adjugate matrix of F. In the case the equation resolves to -1, we take $-\alpha\text{adj}(F)$ as the modular inverse.

This generalization dramatically improves efficiency: most matrices F with nonzero determinants in $\mathbb{Z}[\sqrt{-3}]$ admit modular inverses modulo p and q under this extended notion. As a result, key generation becomes not only feasible but fast, even for larger values of n, as the experimental results in Table 2 demonstrate.

We performed a Simulation with Generalized Key Generation using parameters that abide to the Corrected Decryption Failure Bounds from Theorem 2. The results are shown in Table 2 where we see that new key generation not only outperforms the initial approach suggested in [Wijayanti et al. 2023] but makes it fast for moderate values of n. It is worth noting that the values of p, q did not significantly affect the running time of the system as reported in [Wijayanti et al. 2023] and therefore we think this comparisons makes sense.

Table 2. Comparison of key generation times (in seconds) between the reference method ([Wijayanti et al. 2023]) and the method presented in this work. Average time execution is presented for our work based on 1000 executions. The values for the computational time for key generation in [Wijayanti et al. 2023] are the correspondent ones presented in their Table III where it is not clear which p and q was used and therefore, we reported their minimum time for each dimension for a somewhat fair comparison.

n	q	d	KeyGen [Wijayanti et al. 2023] (s)	KeyGen THIS WORK (s)
3	97	6	0.17336	0.00327
4	131	11	0.93104	0.00495
5	163	17	7.33099	0.00719
6	193	24	83.7658	0.01045
7	227	33	766.048	0.01313
10	331	67	not available	0.03368
15	487	150	not available	0.11421

The most important lesson from Table 2 is that [Wijayanti et al. 2023]'s approach to generate keys for the matrix NTRU over the integral domain was overly restrictive making it hard to find keys even for moderate values of n, but this is immediately solved by using extended Euclidean algorithm in the integral domain $\mathbb{Z}[\sqrt{-3}]$.

6. Conclusion

We have studied the theory behind the new post-quantum system Matrix NTRU over integral domain and corrected an important result on its decryption failure condition. In addition, we replaced the previous approach of key generation by one that is more practical and much faster for this purpose.

Although we have made significant strides in making the matrix NTRU over integral domain more practical it is important to note that we are not recommending its deployment at this stage. Future work should focus on a thorough analysis of its security. We believe that the system may still be vulnerable to attacks in a severe way, particularly through the breaking of the private key matrix in individual rows combined with an integer lattice attack using the isomorphism presented in [Monica Nevins 2010]. This is an interesting future research question. Source code is made available at `https://github.com/thiagopod17/integral_domain_matrix_ntru`.

References

Chen, C., Danba, O., Hoffstein, J., Hulsing, A., Rijneveld, J., Schanck, J. M., Schwabe, P., Whyte, W., and Zhang, Z. (2020). Ntru: algorithm specifications and supporting documentation (2019). *URL: https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions.*, 1.

Coglianese, M. and Goi, B.-M. (2005). Matru: A new ntru-based cryptosystem. In *Progress in Cryptology-INDOCRYPT 2005: 6th International Conference on Cryptology in India, Bangalore, India, December 10-12, 2005. Proceedings 6*, pages 232–243. Springer.

Daniel J. Bernstein, Tanja Lange, Chitchanok Chuengsatiansup, and Peter Schwabe (Accessed: 2024). NTRU Prime. `https://ntruprime.cr.yp.to`. Website.

Gama, N. and Nguyen, P. Q. (2007). New chosen-ciphertext attacks on ntru. In *International Workshop on Public Key Cryptography*, pages 89–106. Springer.

Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219.

Hoffstein, J., Pipher, J., and Silverman, J. H. (1998). Ntru: A ring-based public key cryptosystem. In *International algorithmic number theory symposium*, pages 267–288. Springer.

Hungerford, T. W. (2012). *Algebra*, volume 73. Springer Science & Business Media.

Jaulmes, É. and Joux, A. (2000). A chosen-ciphertext attack against ntru. In *Annual international cryptology conference*, pages 20–35. Springer.

Koblitz, N. (1987). Elliptic curve cryptosystems. *Mathematics of computation*, 48(177):203–209.

Miller, V. S. (1985). Use of elliptic curves in cryptography. In *Conference on the theory and application of cryptographic techniques*, pages 417–426. Springer.

Mittal, S. and Ramkumar, K. (2022). A retrospective study on ntru cryptosystem. In *AIP Conference Proceedings*, volume 2451. AIP Publishing.

Monica Nevins, Camelia KarimianPour, A. M. (2010). Ntru over rings beyond z. *Designs, Codes and Cryptography*, 56:65–78.

Nayak, R., Sastry, C., and Pradhan, J. (2008). A matrix formulation for ntru cryptosystem. In *2008 16th IEEE International Conference on Networks*, pages 1–5. IEEE.

NIST. Post-Quantum Cryptography Standardization. https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization.

Rivest, R. L., Shamir, A., and Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126.

Salleh, N. and Kamarulhaili, H. (2020). Ntru public-key cryptosystem and its variants: An overview. *Int. l J. of Cryptology Research*, 10(1):1–21.

Shor, P. W. (1994). Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. Ieee.

Silverman, J. H., Pipher, J., and Hoffstein, J. (2008). *An introduction to mathematical cryptography*, volume 1. Springer.

Singh, S. and Padhye, S. (2016). Generalisations of ntru cryptosystem. *Security and Communication Networks*, 9(18):6315–6334.

Sousa, T. and Neto, T. S. (2024). Lattice base reduction attack on matrix ntru. In *Anais do XXIV Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*, pages 431–444, Porto Alegre, RS, Brasil. SBC.

Wijayanti, I. E., Isnaini, U., Sari, A. K., Ali, S., and Aji, N. C. (2023). Matrix ntru cryptosystem over integral domain. In *2023 IEEE International Conference on Cryptography, Informatics, and Cybersecurity (ICoCICs)*, pages 35–40. IEEE.