Lattice Reduction Algorithm and Applications

A Thesis

submitted to
Indian Institute of Science Education and Research Pune
in partial fulfillment of the requirements for the
BS-MS Dual Degree Programme

by

Gugulothu Ananda

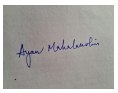
Indian Institute of Science Education and Research Pune
Dr. Homi Bhabha Road,
Pashan, Pune 411008, INDIA.

May, 2025

Supervisor: Dr.Ayan Mahalanobis

Certificate

This is to certify that this dissertation entitled **Lattice Reduction Algorithm and Applications** towards the partial fulfilment of the BS-MS dual degree programme at the Indian Institute of Science Education and Research, Pune represents study/work carried out by Gugulothu Ananda at the Indian Institute of Science Education and Research under the supervision of Dr.Ayan Mahalanobis, Assistant Professor, Department of Mathematics, Indian Institute of Science Education and Research, Pune, during the academic year 2024-2025.



Dr.Ayan Mahalanobis

Committee:

Dr.Ayan Mahalanobis

Dr.Krishna Kaipa

This thesis is dedicated to my supervisor, Dr.Ayan Mahalanobis

Declaration

I hereby declare that the matter embodied in the report entitled **Lattice Reduction Algorithm and Applications** are the results of the work carried out by me at the Department of Mathematics, Indian Institute of Science Education and Research, Pune, under the supervision of Dr.Ayan Mahalanobis and the same has not been submitted elsewhere for any other degree.

G.Ananda

Gugulothu Ananda

Acknowledgments

I extend my deepest gratitude to my supervisor, Dr.Ayan Mahalanobis, for his invaluable guidance, patience, and continuous support throughout this research. His profound knowledge and keen insights have played a crucial role in shaping my understanding and approach to this work. His encouragement, even during challenging phases, has given me the confidence to push forward and refine my research. I am truly fortunate to have had the opportunity to work under his mentorship, which has enhanced my technical skills and inspired me to think critically and independently. His unwavering belief in my abilities has been instrumental in the successful completion of this thesis.

I sincerely thank Dr.Krishna Kaipa for his valuable feedback and constructive suggestions. His insightful comments have helped me refine my research, identify key improvements, and enhance the overall quality of my work. His expertise and guidance have been highly beneficial in strengthening the depth and clarity of my findings.

I would also like to express my appreciation to IISER Pune for providing an excellent academic environment and the necessary resources to carry out this research. The institution's support, infrastructure, and intellectual atmosphere have greatly contributed to my learning experience and the smooth execution of this project.

Additionally, I am profoundly grateful to my friends and family for their constant support, motivation and encouragement. Their unwavering belief in me, especially during moments of doubt and difficulty, has been a source of strength. Their emotional and moral support has played a crucial role in helping me stay focused and determined throughout this journey.

Abstract

Cryptography is a fundamental field in ensuring secure communication, with public key cryptosystems playing a crucial role in modern security protocols. This thesis begins by discussing the GGH (Goldwasser-Goldwasser-Halevi) cryptosystem, a lattice-based encryption scheme, and then focuses on NTRU cryptosystem. NTRU is a lattice-based public-key cryptosystem known for its efficiency and resistance to quantum attacks and is used widely. The thesis explores its security, especially in the context of lattice reduction techniques like the Lenstra-Lenstra-Lovász (LLL) algorithm, highlighting how parameter selection and lattice dimensions impact decryption success. Through experimental analysis, the thesis concludes that NTRU demonstrates stronger security at higher dimensions but remains vulnerable to lattice-based attacks for smaller parameters. The results offer insights into optimizing NTRU's parameters to enhance its cryptographic resilience.

Contents

Chapter 1

Introduction

In today's digital world, protecting information is more critical than ever. With the increasing reliance on digital communication, e-commerce, and data storage, ensuring the confidentiality, integrity, and authenticity of information has become paramount. Cryptography has long been the primary tool used to secure digital interactions, providing techniques to safeguard sensitive data against unauthorized access and manipulation.

Moreover, the rise of quantum computing introduces new challenges to the field of cryptography. Algorithms such as Shor's algorithm threaten the security of public key cryptosystems, making them vulnerable to quantum attacks. As a result, researchers have been motivated to explore alternative cryptographic solutions that can withstand the power of quantum computers. Among these, lattice-based cryptography has gained significant attention due to its strong mathematical foundations and resistance to both classical and quantum attacks.

This thesis aims to provide a detailed exploration of classical cryptography, the emerging threats posed by quantum computing, and the promising solutions offered by lattice-based cryptography.

Chapter 2, begins by explaining fundamental concepts in cryptography. It begins with private-key cryptography, followed by public-key cryptosystems. The chapter also discusses the security of public key cryptosystems and introduces essential concepts like one-way and trapdoor functions, which are crucial for building secure cryptographic schemes.

In Chapter 3, the focus shifts towards cryptosystems designed to be secure against quantum attacks. It explores the GGH public key cryptosystem and its security analysis, providing insights into how lattice-based approaches can offer robust alternatives. The chapter then delves into the NTRU cryptosystem, explaining its structure, the use of convolution polynomial rings, and providing an overview of NTRUEncrypt.

In Chapter 4, a discussion of the mathematical techniques that are used in lattice-based cryptography is given. It covers Gram-Schmidt orthogonalization and Lenstra-Lenstra-Lovász (LLL) algorithm, which are essential for lattice reduction. The chapter also demonstrates how these algorithms are applied in the context of NTRUEncrypt and evaluates NTRUEncrypt as a lattice cryptosystem.

In Chapter 5, a specific focus is on analyzing the security aspects of the NTRU cryptosystem. It assesses the strengths and potential vulnerabilities of NTRU in the context of both classical and quantum attack models.

Chapter 2

Cryptography

Cryptography is a technique for securing information and communications using codes, ensuring that only authorized individuals can access and understand the data. The term originates from the Greek words 'crypt' (hidden) and 'graphy' (writing). Cryptographic techniques rely on mathematical concepts and algorithmic rules to transform messages into forms that are difficult to decipher. These algorithms are crucial in key generation, digital signatures, data privacy protection, secure web browsing, and safeguarding confidential transactions such as credit card and debit card payments.

Objectives of Cryptography

- **Confidentiality**: Confidentiality ensures that sensitive information, like a client's financial data, is only accessible by authorized individuals. For example, when a client sends their trading instructions to a stockbroker, encryption ensures that only the stockbroker can view the data, preventing unauthorized access by third parties, such as hackers.
- **Integrity**: Integrity ensures that data remains unchanged during transmission. For example, when a stockbroker receives an order from a client, cryptographic methods ensure that the order is not altered or tampered with during transit, protecting both the client and the broker from fraud or accidental changes.
- **Authentication**: Authentication verifies the identities of the users involved in the

communication, ensuring trust. In the context of online stock trading, it ensures that the stock broker receiving the client's instructions is the legitimate broker and that the client is authorized to make the trade.

- **Non-repudiation**: Non-repudiation ensures that a sender cannot deny having sent a message. For example, after a client places an order with a stockbroker, digital signatures can be used to provide proof that the client initiated the request, preventing the client from later denying the transaction.

- **Secure Communication**: Secure communication ensures that sensitive information is transmitted safely over insecure networks. When a client sends a stock trade order to a broker over the Internet, secure protocols such as SSL/TLS encrypt the communication to protect the transaction from eavesdropping and tampering by attackers.

- **Access Control**: Access control limits access to information based on security policies. In a stock brokerage system, access control ensures that only authorized employees can view or modify certain client data, such as account details, preventing unauthorized personnel from accessing sensitive financial information.

Cryptography plays a vital role in modern security systems, enabling the safe exchange of sensitive information across digital networks. It is broadly categorized into two main types: **private key cryptography** (symmetric encryption) and **public key cryptography** (asymmetric encryption). Private key cryptography relies on a single shared key for both encryption and decryption, offering high-speed encryption but presenting challenges in secure key distribution. In contrast, public key cryptography uses a pair of mathematically related keys, eliminating the need for a pre-shared secret and making it ideal for secure communications over untrusted networks. The following sections will explore these two cryptographic approaches in detail, discussing their mechanisms, advantages, and applications.

2.1 Private Key Cryptography

Private key cryptography, also known as symmetric cryptography, is one of the oldest known methods of secure communication. The concept of using a single secret key for both encryption and decryption dates back to ancient times. Historical examples include the Caesar

cipher used by Julius Caesar and the Vigenère cipher, which was widely used in the Renaissance period [5]. These early cryptographic techniques relied on manually exchanged secret keys, making key distribution a problem.

In the modern era, symmetric cryptography was formalized in the 20th century with the advent of computers. One of the first widely adopted symmetric encryption algorithms was the Data Encryption Standard (DES), developed by IBM in the early 1970s and later standardized by the U.S. National Institute of Standards and Technology (NIST) in 1977[6]. While DES provided a robust encryption mechanism for its time, its key length (56 bits) became vulnerable to brute-force attacks as computing power advanced. To address this, the Advanced Encryption Standard (AES) was introduced in 2001 after a public competition, providing stronger security with key sizes of 128, 192, and 256 bits[7].

Despite its efficiency and speed, symmetric cryptography has a fundamental drawback—the key exchange problem. Since the same key is used for both encryption and decryption, it must be securely shared between the sender and receiver before communication can begin. This requirement poses a significant security risk, especially in large-scale networks, as securely distributing and managing keys becomes increasingly complex[4].

The introduction of public key cryptography in the 1970s transformed the landscape of secure communication. The groundbreaking work of Whitfield Diffie and Martin Hellman in 1976 introduced the concept of asymmetric cryptography, which eliminated the need for a pre-shared secret [1]. Instead of using a single key, public key cryptography employs two mathematically linked keys: a public key, which is shared openly, and a private key, which is kept secret. This innovation solved the key distribution problem by allowing secure communication between parties who had never met before.

Public key cryptography paved the way for secure online transactions, digital signatures, and authentication mechanisms that form the foundation of modern e-commerce. Before its invention, secure communication over the internet was impractical due to the risk of key interception. With the adoption of protocols such as SSL/TLS, online banking, shopping, and secure email became feasible [7]. Digital certificates issued by trusted authorities ensure that websites are legitimate and that sensitive information, such as credit card details and passwords, is encrypted before transmission. Without public key cryptography, secure online commerce as we know it today would not be possible.

Thus, while private-key cryptography has been used for centuries, it was the invention of public-key cryptography that truly revolutionized secure communication, enabling the growth of the digital economy and making global e-commerce secure and reliable.

Here's how private key encryption and decryption works:

- **Key Generation**: A secret key is generated, which is a long string of random characters. This key is securely shared between the sender and the receiver.
- **Encryption**: The sender uses the secret key to transform the plaintext message into ciphertext using an encryption algorithm. The encrypted message is then transmitted to the receiver.
- **Decryption**: The receiver applies the same secret key to the ciphertext using the decryption algorithm. This process reverses the encryption, restoring the original plaintext message.

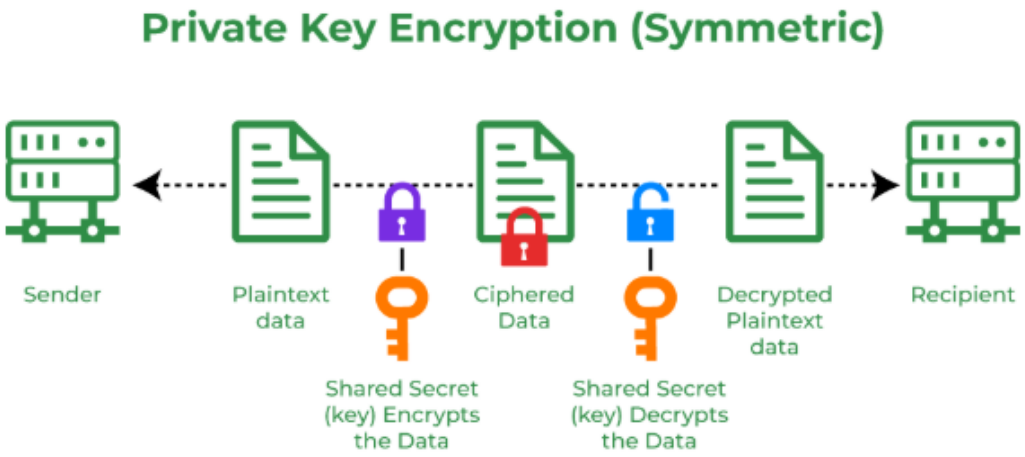


Figure 2.1: Symmetric Key Encryption Process: The sender encrypts the plaintext using a shared secret key, and the receiver decrypts it using the same key.

2.2 Public Key Cryptosystem

Public key cryptography, also known as asymmetric cryptography, was first introduced in 1976 by Whitfield Diffie and Martin Hellman [1], who proposed the concept of key pairs to enable secure communication over an insecure channel. Their work laid the foundation for modern cryptographic systems. Shortly after, RSA (Rivest-Shamir-Adleman) was developed

in 1977 [2], providing one of the first practical public-key encryption schemes. Over time, elliptic curve cryptography (ECC) [3] and other methods have been introduced, improving efficiency and security. Unlike traditional private-key (symmetric) cryptography, which uses the same key for encryption and decryption, public-key cryptography relies on two separate keys.

Public key cryptography, also known as asymmetric cryptography, is a revolutionary concept that addresses the limitations of traditional symmetric cryptography [1]. In symmetric cryptography, both the sender and receiver must share the same secret key, which creates challenges around securely distributing the key to both parties [2]. The primary issue is that if the key is intercepted during transmission, an unauthorized party could gain access to the encrypted information. Public key cryptography eliminates this problem by utilizing two separate keys: a public key and a private key.

The public key is made publicly accessible, and it is used for encryption or verifying digital signatures. Since the public key is shared openly, anyone can use it to encrypt a message or check the authenticity of a signature. However, the key is designed so that only the corresponding private key can decrypt the message or generate the signature. This means that even though the public key is freely available, it cannot be used by unauthorized individuals to gain access to encrypted data or forge signatures.

On the other hand, the private key is kept secret and is only known to the owner of the key pair. The private key is used to decrypt messages that have been encrypted with the corresponding public key, or to generate digital signatures that can be verified using the public key. Since the private key remains confidential, it ensures that only the intended recipient can read the encrypted message, and only the authorized entity can create valid signatures.

This dual-key approach eliminates the need for a pre-shared secret between communicating parties, which is a significant advantage over symmetric cryptography. As a result, public key cryptography solves the key distribution problem, making secure communication more practical and efficient. It allows individuals to exchange sensitive information over insecure channels, such as the Internet, without the need for a secure method of sharing a secret key beforehand.

In modern cryptographic systems, public key cryptography is widely used for a variety of

applications. It ensures the confidentiality of communications through encryption, authenticates the identities of users and systems, and guarantees the integrity of messages through digital signatures. Additionally, it plays a crucial role in securing online transactions, and email communications, and even in verifying software authenticity, making it an essential tool for ensuring privacy and security in the digital age.

Here's how Public key encryption and decryption work:

- **Key Generation**: A pair of cryptographic keys is generated—one public and one private. The public key is shared openly, while the private key is kept secret by the recipient.
- **Encryption**: The sender encrypts the plaintext message using the recipient's public key. This ensures that only the corresponding private key can decrypt the ciphertext.
- **Decryption**: The recipient applies their private key to decrypt the ciphertext, restoring the original plaintext message.

Figure 2.2: Public Key Encryption Process: The sender encrypts the plaintext using the recipient's public key, and only the recipient can decrypt it using their private key.

Difference Between Private Key and Public Key Cryptography

Private Key Cryptography	Public Key Cryptography
Faster than the public key.	Slower than a private key.
Uses the same key and algorithm for encryption and decryption.	Uses two keys: one for encryption and one for decryption.
The key is kept secret.	One of the keys is kept secret.
Symmetric, as it uses only one key.	Asymmetric, using both private and public keys.
Both sender and receiver share the same key.	Sender and receiver do not need to share the same key.
Efficient technology.	Inefficient technology.
Used for large amounts of text.	Used for short messages.
Private key is shared between two parties.	Public key can be used by anyone.
Used in algorithms like AES(Advanced Encryption Standard).	Used in algorithms like RSA(Rivest Shamir Adleman), DSA(Digital Signature Algorithm).
Private key is kept secret.	Public key is publicly accessible.

Figure 2.3: Comparison Between Private Key and Public Key Cryptography

2.3 Security of Public Key Cryptosystem

Public Key Cryptosystems (PKC) rely on asymmetric encryption, where a pair of keys (public and private) is used for secure communication. The security of these cryptosystems depend on the hardness of certain mathematical problems that are computationally infeasible to solve without the private key.

1. **Security Assumptions**

The security of a public key cryptosystem is based on the assumption that certain mathematical problems are difficult to solve efficiently. Some of the most common hardness assumptions include:

- **Integer Factorization Problem (IFP)**: Used in RSA, it assumes that factoring a large composite number into its prime factors is computationally infeasible [2].
- **Discrete Logarithm Problem (DLP)**: Used in Diffie-Hellman and ElGamal cryptosystems, it assumes that computing the discrete logarithm in a finite field is difficult [1].
- **Elliptic Curve Discrete Logarithm Problem (ECDLP)**: Used in ECC (Elliptic Curve Cryptography), it assumes that finding the discrete logarithm of a point on an elliptic curve is hard [3].
- **Lattice-based Assumptions**: Used in post-quantum cryptography, problems like the Shortest Vector Problem (SVP) and Learning With Errors (LWE) are assumed to be hard even for quantum computers [8].

2. **Attacks on Public Key Cryptosystems**
   Despite their security foundations, public key cryptosystems are vulnerable to various attacks:

   - **Brute Force Attacks**: Trying all possible private keys (infeasible for sufficiently large key sizes).
   - **Mathematical Attacks**: Exploiting weaknesses in the underlying hardness assumptions (e.g., number field sieve for factoring large integers in RSA) [20].
   - **Side-Channel Attacks**: Gaining information by analyzing timing, power consumption, or electromagnetic leaks from cryptographic implementations[21].
   - **Quantum Attacks**: Shor's algorithm can efficiently factor large integers, breaking RSA and ECC. Post-quantum cryptosystems aim to resist these attacks [9].

3. **Key Size and Security Level**
   To maintain security, key sizes must be chosen appropriately:

   - **RSA**: 2048-bit key is considered the minimum secure size; 4096-bit is recommended for long-term security [22].
   - **ECC**: 256-bit keys provide security equivalent to 3072-bit RSA [23].

- **Lattice-based Cryptography**: Key sizes vary but are generally larger due to the nature of lattice problems [24].

4. **Post-Quantum Security**
   Since quantum computers threaten traditional PKC, research has focused on post-quantum cryptography (PQC) based on lattice problems, hash functions, and other hard mathematical problems resistant to quantum attacks. NIST is currently standardizing PQC algorithms, such as Kyber, Dilithium, and Falcon [10].

5. **Best Practices for Secure Implementation**
   To ensure the security of a public key cryptosystem, best practices include:

   - Use sufficiently large key sizes to prevent brute force and mathematical attacks.
   - Implement side-channel-resistant algorithms to prevent leakage of cryptographic keys.
   - Regularly update cryptographic protocols to defend against emerging threats.
   - Transition to post-quantum cryptography for long-term security [11].

2.4 One-Way and Trapdoor Functions

Trapdoor functions and one-way functions both play a crucial role in securing sensitive data and maintaining its integrity, but they serve different purposes. While these terms are sometimes used interchangeably in cryptography, they are not identical. Both functions are fundamental to modern cryptographic systems, yet their unique properties make them suitable for distinct applications.

A one-way function is a mathematical function that is simple to compute in the forward direction but extremely difficult to invert. This means that while obtaining the output from a given input is straightforward, determining the original input from the output is computationally infeasible. This characteristic makes one-way functions valuable in cryptography, particularly for creating secure hashes and encrypting data.

Example: For appropriate choices of primes $a = 17$ and $d = 19$, the function $f(a, d) = ad$ is a one-way function: given a and d, computing $p = ad$ is easy, but given $p = 323$, finding

a and d (i.e., performing integer factorization) is difficult. This property, where multiplying large primes is computationally simple but factoring the product is hard, is foundational to the security of cryptographic systems like RSA. In this case, while it is easy to compute $p = 17 \times 19 = 323$, determining the prime factors of 323 is not feasible without advanced factorization techniques.

A trapdoor function is a special type of one-way function that includes a hidden "trapdoor" or secret key, allowing it to be easily reversed. Essentially, it behaves like a one-way function but can be efficiently inverted if the correct key is known.

Trapdoor functions play a crucial role in public key cryptography, where they are used to generate public and private key pairs. The public key is openly shared, while the private key remains confidential. Anyone can encrypt a message using the public key, but only the private key holder can decrypt it, ensuring secure communication without requiring a pre-shared secret.

The security of trapdoor functions is based on the complexity of certain mathematical problems, such as factoring large numbers. The private key provides the necessary information to reverse the function and decrypt encrypted data.

Example: An example of a trapdoor function is RSA. RSA is based on a one-way function, where encrypting data is easy, but decrypting it without the private key is computationally infeasible. The security of RSA relies on the difficulty of factoring large composite numbers, and the private key acts as the "trapdoor," allowing for efficient decryption.

Chapter 3

Quantum-Secure Cryptosystems

Quantum-secure cryptosystems are cryptographic systems designed to resist attacks from quantum computers. Traditional cryptographic schemes, such as RSA and ECC, rely on mathematical problems like integer factorization and discrete logarithms, which quantum algorithms (e.g., Shor's algorithm) can solve efficiently. Quantum-secure cryptography, also known as **post-quantum cryptography (PQC)**, aims to develop cryptographic techniques that remain secure even in the presence of powerful quantum computers.

Types of Quantum-Secure Cryptosystems

1. **Lattice-Based Cryptography**
    - Relies on the hardness of lattice problems such as the **Learning With Errors (LWE)** and **Shortest Vector Problem (SVP)**.
    - Example: NTRUEncrypt.
2. **Code-Based Cryptography**
    - Based on the difficulty of decoding using random linear codes.
    - Example: McEliece Cryptosystem.

3.1 GGH Public Key Cryptosystem

The Goldreich–Goldwasser–Halevi (GGH) lattice-based cryptosystem is a broken asymmetric cryptosystem based on lattices. However, the GGH signature scheme remains unbroken as

of today.

The GGH cryptosystem relies on the hardness of the Closest Vector Problem (CVP). It was introduced in 1997 by Oded Goldreich, Shafi Goldwasser, and Shai Halevi and utilizes a trapdoor one-way function based on the difficulty of lattice reduction [**?**]. The key idea behind this trapdoor function is that, given any lattice basis, generating a vector close to a lattice point is easy, for example, by taking a lattice point and adding a small error vector. However, reversing this process (recovering the original lattice point from the erroneous vector) requires a special basis, which serves as the trapdoor.

The GGH encryption scheme was broken in 1999 by Phong Q. Nguyen. Additionally, in 2006 [25] and [26] Nguyen and Oded Regev cryptanalyzed the related GGH signature scheme [12].

Key Generation: Devi starts by selecting a set of linearly independent vectors $b_1, b_2, \ldots, b_n \in \mathbb{Z}^n$ that are approximately orthogonal to each other. One method is to choose a parameter f and randomly assign the coordinates of $b_1, \ldots, b_n$ within the range $-f$ to f. To confirm that these vectors form a well-conditioned basis, Devi calculates the *Hadamard ratio* and ensures it is not too small. The vectors $b_1, \ldots, b_n$ constitute Devi's **private key**.

For simplicity, let B be the $n \times n$ matrix with rows $b_1, \ldots, b_n$, and let L represent the lattice generated by these vectors.

Then, Devi selects an $n \times n$ unimodular integer matrix D such that $\det(D) = \pm 1$. One approach to constructing D is by multiplying several randomly chosen elementary matrices. She then computes:

$$S = DB$$

The row vectors $s_1, \ldots, s_n$ of S constitute an alternative basis for L, which serves as Devi's **public key**.

Encryption: When David wants to send a message to Devi, he selects a small vector p with integer coordinates as his plaintext. For instance, p could be a binary vector. He also picks a small random perturbation vector r, which serves as noise or error. The coordinates of r are randomly chosen within the range $-\delta$ to δ, where δ is a fixed public parameter. He then computes the ciphertext c as follows:

$$c = pS + r = \sum_{i=1}^{n} p_i s_i + r$$

Here, c is *not* a lattice point, but it remains close to the lattice point pS due to the small perturbation r.

Decryption: Devi decrypts the ciphertext using *Babai's algorithm*. With the *good basis*

$b_1, \ldots, b_n$, she finds the closest lattice vector to c. Since she uses a well-conditioned basis and r is small, the lattice vector she recovers is mS.
Finally, she computes $S^{-1}(pS) = p$, thereby recovering the original plaintext p. We will use the theorem or algorithm below to illustrate an example of the GGH public key cryptosystem.
Theorem[Babai's Algorithm]: Let $Ł \subset \mathbb{R}^n$ be a lattice with basis $\{b_1, b_2, \ldots, b_n\}$, and let $y \in \mathbb{R}^n$ be an arbitrary vector. If the basis vectors are sufficiently orthogonal, the following algorithm finds the closest lattice vector to y:

1. Express y as a linear combination of the basis vectors:

$$y = c_1 b_1 + c_2 b_2 + \cdots + c_n b_n, \quad \text{where } c_i \in \mathbb{R}.$$

2. Set $d_i = \lfloor c_i \rceil$ for $i = 1, 2, \ldots, n$, where $\lfloor \cdot \rceil$ denotes rounding to the nearest integer.

3. Compute the lattice vector:

$$z = d_1 b_1 + d_2 b_2 + \cdots + d_n b_n.$$

4. Return z as the approximate closest vector.

If the basis vectors are reasonably orthogonal, the algorithm provides a good approximation to the closest vector problem (CVP). However, when the basis is highly non-orthogonal, the returned vector may be far from the actual closest lattice vector to y.
By using the Hadamard ratio, we can assess whether a reduced basis is more orthogonal than the original basis.
Definition 1. *The **Hadamard ratio** of a basis $B = \{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, \ldots, \mathbf{b}_n\}$ is defined as the following quantity:*

$$H(B) = \left(\frac{\det(L(B))}{\|\mathbf{b}_1\| \cdot \|\mathbf{b}_2\| \cdots \|\mathbf{b}_n\|} \right)^{1/n}$$

The closer the Hadamard ratio is to 1, the greater the orthogonality of the vectors in the basis. More on the Hadamard ratio and orthogonality can be found in [13].
Now we illustrate an example of a GGH Public Key Cryptosystem
Example: Devi begins by selecting an approximately orthogonal basis

$$\mathbf{b_1} = (-20, 8, 1), \quad \mathbf{b_2} = (14, 11, 23), \quad \mathbf{b_3} = (-18, 1, -12)$$

which are her private key basis. The lattice L spanned by $\mathbf{b_1}, \mathbf{b_2}, \mathbf{b_3}$ has a determinant of

$$\det(L) = 5280.$$

The Hadamard ratio of the basis is given by

$$H(\mathbf{b_1}, \mathbf{b_2}, \mathbf{b_3}) = \left(\frac{\det(L)}{||\mathbf{b_1}|| \cdot ||\mathbf{b_2}|| \cdot ||\mathbf{b_3}||} \right)^{1/3} \approx 0.7298$$

Devi chooses an unimodular matrix D to create the public key S. The matrix D satisfies

$$\det(D) = -1.$$

Then, the public key S is obtained by multiplying the basis matrix D by B:

$$S = DB.$$

The Hadamard ratio of the public key is

$$H(\mathbf{s_1}, \mathbf{s_2}, \mathbf{s_3}) = \left(\frac{\det(S)}{||\mathbf{s_1}|| \cdot ||\mathbf{s_2}|| \cdot ||\mathbf{s_3}||} \right)^{1/3} \approx 0.00002530$$

and is very small.
David decides to send a message to Devi and selects his plaintext as

$$p = (6, -5, -2).$$

He uses a small error vector

$$r = (-1, 0, 2).$$

The corresponding ciphertext c is computed as

$$c = (6, -5, -2) \cdot \begin{bmatrix} -248100 & 220074 & 332172 \\ -112192 & 99518 & 150209 \\ -216150 & 191737 & 289401 \end{bmatrix} + (-1, 0, 2).$$

Performing the computation, we get

$$c = (-495341, 439381, 663188).$$

Devi uses Babai's algorithm for decryption. First, she expresses the ciphertext c as a linear combination of the private key basis vectors:

$$c = q_1 b_1 + q_2 b_2 + q_3 b_3.$$

Using Babai's algorithm, she finds the coefficients q_1, q_2, and q_3 as:

$$(q_1, q_2, q_3) = (20698.1, 50981.1, 44173.0).$$

Thus, the ciphertext c can be approximated as

$$c \approx 20698.1 \cdot b_1 + 50981.1 \cdot b_2 + 44173.0 \cdot b_3.$$

She then rounds the coefficients to the nearest integers, obtaining the approximate closest vector:

$$h = 20698 \cdot b_1 + 50981 \cdot b_2 + 44173 \cdot b_3.$$

$$h = (-495340, 439380, 663185) = m * S$$

which is close to the ciphertext c. She then recovers the plaintext by computing $h \cdot S^{-1}$:

$$h \cdot S^{-1} = (6, -5, -2) = m.$$

If Shiv wants to decrypt David's message, he only knows the public key basis s_1, s_2, s_3. If he applies Babai's algorithm using a public key basis, he will get

$$c = e_1 s_1 + e_2 s_2 + e_3 s_3.$$

He finds the coefficients

$$(e_1, e_2, e_3) = (208.5, -361.5, -49.4).$$

Thus, the ciphertext c can be approximated as

$$c \approx 208.5 \cdot s_1 - 361.5 \cdot s_2 - 49.4 \cdot s_3$$

Rounding, he obtains a lattice vector

$$h' = 209 \cdot s_1 - 362 \cdot s_2 - 49 \cdot s_3 = (-760238, 674355, 1017850)$$

This lattice vector is not close to ciphertext c, and the obtained plaintext (209, -361, 49) is also incorrect. We can compare how close the closest vectors are to the ciphertext by computing the norms of c, h and h'. The obtained norms are

$$\|c - h\| \approx 2.236 \quad \text{and} \quad \|c - h'\| \approx 501168.150.$$

Here, we observe that the vector found using the good basis is very close to the lattice vector, as indicated by the small norm. In contrast, when using the bad basis, the norm is significantly larger, showing that the obtained vector is much farther from the lattice vector.

3.1.1 Security Of GGH Public Key Cryptosystem

The security of the GGH cryptosystem is fundamentally tied to the Closest Vector Problem (CVP), which is known to be hard in general. The idea is that, given a poor (highly non-orthogonal) basis, finding the closest lattice point to a given vector is computationally difficult. This hardness is what GGH relies on for encryption. However, in practice, this security assumption did not hold up.

One of the most significant vulnerabilities of the GGH encryption scheme lies in the construction of its public key. The public key is generated by applying an unimodular matrix transformation to the private key basis. This transformation, while ensuring the key's public nature, preserves certain structural properties that make it susceptible to attacks. Specifically, the structure of the public key allows adversaries to exploit lattice basis reduction techniques, such as the LLL (Lenstra-Lenstra-Lovász) algorithm. This algorithm finds a nearly orthogonal basis that can approximate the private key basis much better than expected, making decryption possible for an attacker.

Additionally, Babai's Nearest Plane Algorithm, which provides an efficient approximation to solving the CVP, allows attackers to recover plaintexts with high accuracy. The choice of the error vector in the encryption process further complicate things: if the error is too small, an attacker can directly recover the plaintext; if it is too large, even the legitimate recipient may fail to decrypt correctly.

As a result, the GGH encryption scheme was broken in 1999 by Phong Q. Nguyen. Despite this, the related GGH signature scheme has remained unbroken as of today. However, its

security also depends on careful parameter choices, as small basis vectors could lead to forgeries.
Ultimately, while GGH was an important step in lattice-based cryptography, it is no longer considered secure. Modern lattice-based cryptosystems, such as NTRU and Learning With Errors (LWE)-based schemes, have been developed to address its vulnerabilities and are now leading candidates for post-quantum cryptography.

3.2 Lattice-Based Cryptosystem - NTRU

Lattice-based cryptography is the generic term for the construction of cryptographic primitives that involve lattices, either in the construction itself or in the security proof. Lattice-based constructions support important standards of post-quantum cryptography. In 1996, Miklós Ajtai introduced the first lattice-based cryptographic construction whose security was based on the hardness of well-studied lattice problems. Around the same time, Cynthia Dwork demonstrated that a specific average-case lattice problem, known as the Short Integer Solutions (SIS) problem, is at least as hard as a worst-case lattice problem [28]. She further introduced a cryptographic hash function whose security is directly linked to the computational hardness of SIS.
In 1998, Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman proposed NTRU, a lattice-based public-key encryption scheme. However, the security of NTRU has not been proven to be as hard as solving a worst-case lattice problem [29].
A major breakthrough came in 2005 when Oded Regev introduced the first lattice-based public-key encryption scheme with security proven under worst-case hardness assumptions, along with the Learning With Errors (LWE) problem. Subsequent research has focused on strengthening Regev's security proof and enhancing the efficiency of the original scheme. Furthermore, significant efforts have been made to develop additional cryptographic primitives based on LWE and related problems. Notably, in 2009, Craig Gentry introduced the first fully homomorphic encryption scheme, leveraging lattice-based cryptography.
To understand the structure and operations within the NTRU public key cryptosystem, it is essential to explore convolution polynomial rings, as they form the foundational algebraic structure for key generation, encryption, and decryption processes.

3.3 Convolution Polynomial Rings

We describe a special class of polynomial quotient rings used in the NTRU public key cryptosystem.

Definition 2. *Let N be a positive integer. The ring of convolution polynomials (of rank N) is defined as the quotient ring*

$$F = \frac{\mathbb{Z}[y]}{\langle y^N - 1 \rangle},$$

where $\mathbb{Z}[y]$ is the polynomial ring over the integers, and $\langle y^N - 1 \rangle$ is the ideal generated by the polynomial $y^N - 1$ in the polynomial ring $\mathbb{Z}[y]$. In a similar manner, the quotient ring represents the ring of convolution polynomials modulo q is

$$F_q = \frac{(\mathbb{Z}/q\mathbb{Z})[y]}{\langle y^N - 1 \rangle},$$

where $\mathbb{Z}/q\mathbb{Z}$ denotes the integers modulo q and the ideal $\langle y^N - 1 \rangle$ is generated in $(\mathbb{Z}/q\mathbb{Z})[y]$. Every element of F or F_q can be uniquely represented as a polynomial of the form

$$a_0 + a_1 y + a_2 y^2 + \cdots + a_{N-1} y^{N-1},$$

with coefficients in $\mathbb{Z}$ or $\mathbb{Z}/q\mathbb{Z}$ respectively. These polynomials are subject to the equivalence relation $y^N = 1$, meaning that powers of $y \geq N$ can be reduced modulo N.

Simplified Computations: Performing computations in the rings F and F_q is simpler than in general polynomial quotient rings, primarily because the polynomial $y^N - 1$ has a straightforward form. By factoring out $y^N - 1$, we essentially impose the condition that $y^N = 1$. As a result, any occurrence of y^N in a polynomial can be replaced by 1. For instance, if y^k appears, we write $k = iN + j$ where $0 \leq j < N$, and then set

$$y^k = y^{iN+j} = (y^N)^i \cdot y^j = 1^i \cdot y^j = y^j.$$

In other words, the exponents of powers of y are reduced modulo N.

Polynomial Representation as Vectors: It is often convenient to identify a polynomial

$$a(y) = a_0 + a_1 y + a_2 y^2 + \cdots + a_{N-1} y^{N-1} \in R$$

by its vector of coefficients

$$(a_0, a_1, a_2, \ldots, a_{N-1}) \in \mathbb{Z}^N$$

similar representation for polynomials in F_q. This vector representation facilitates addition and other operations.

Addition of Polynomials: The addition of polynomials in F and F_q corresponds to the usual addition of vectors. Specifically, if $a(y) = (a_0, a_1, \ldots, a_{N-1})$ and $b(y) = (b_0, b_1, \ldots, b_{N-1})$ are polynomials, then their sum is given by:

$$a(y) + b(y) \longleftrightarrow (a_0 + b_0, a_1 + b_1, \ldots, a_{N-1} + b_{N-1}).$$

Multiplication of Polynomials: In both F and F_q, multiplication of two polynomials $a(y)$ and $b(y)$ is defined as follows:

- The product of two polynomials $a(y)$ and $b(y)$ in F is given by:

$$a(y) \star b(y) = c(y) \quad \text{where} \quad c_k = \sum_{i+j \equiv k \pmod N} a_i b_j$$

  Here, c_k is the coefficient of y^k in the product polynomial, and the sum runs over all i and j such that $i + j \equiv k \pmod N$.

- The product of two polynomials $a(y)$ and $b(y)$ in F_q follows the same formula:

$$a(y) \star b(y) = c(y) \quad \text{where} \quad c_k = \sum_{i+j \equiv k \pmod N} a_i b_j \mod q$$

  In this case, each c_k is reduced modulo q.

Definition 3. *Let $a(y) \in F_q$. The center-lift of $a(y)$ to F is the unique polynomial $\tilde{a}(y) \in F$ satisfying*

$$\tilde{a}(y) \mod q = a(y),$$

where the coefficients of $\tilde{a}(y)$ are chosen such that

$$-\frac{q}{2} < a_i \leq \frac{q}{2}.$$

When q is a prime, the extended Euclidean algorithm for polynomials determines which polynomials are units in F_q and provides a method to compute their inverses.

Proposition:[The extended Euclidean algorithm for $\mathbb{Q}[y]$]: Let $\mathbb{Q}$ be a field and let c and d be polynomials in $\mathbb{Q}[y]$ with $d \neq 0$. Then the greatest common divisor g of c and d

exists, and there exist polynomials e and f in $\mathbb{Q}[y]$ such that

$$c \cdot e + d \cdot f = g.$$

Proposition: Let q be a prime. Then $c(y) \in F_q$ has a multiplicative inverse if and only if

$$\gcd(c(y), y^N - 1) = 1 \quad \text{in } (\mathbb{Z}/q\mathbb{Z})[y]. \tag{3.1}$$

If this equation holds, then the inverse $c(y)^{-1} \in F_q$ can be computed using the extended Euclidean algorithm to find polynomials $e(y), f(y) \in (\mathbb{Z}/q\mathbb{Z})[y]$ satisfying

$$a(y) \cdot e(y) + (y^N - 1) \cdot f(y) = 1.$$

Then the inverse is given by

$$c(y)^{-1} = e(y) \quad \text{in } F_q.$$

Proof. The previous proposition states that we can find polynomials $e(y)$ and $f(y)$ in the polynomial ring $(\mathbb{Z}/q\mathbb{Z})[y]$ such that

$$c(y) \cdot e(y) + (y^N - 1) \cdot f(y) = \gcd(c(y), y^N - 1).$$

If the greatest common divisor is 1, then reducing modulo $y^N - 1$ gives

$$c(y) \cdot e(y) = 1 \quad \text{in } F_q.$$

This shows that $c(y)$ has a multiplicative inverse in F_q. Conversely, suppose $c(y)$ is a unit in F_q, meaning there exists a polynomial $e(y)$ such that

$$c(y) \cdot e(y) = 1 \quad \text{in } F_q.$$

By the definition of F_q, this implies

$$c(y) \cdot e(y) \equiv 1 \pmod{y^N - 1}.$$

By the definition of congruences, there must exist a polynomial $f(y)$ such that

$$c(y) \cdot e(y) - 1 = (y^N - 1)f(y),$$

so,

$$c(y) \cdot e(y) + (y^N - 1) \cdot f(y) = 1.$$

Thus, $\gcd(c(y), y^N - 1) = 1$, which completes the proof. □

3.4 NTRU Cryptosystem Overview

The cryptosystem is set up by choosing a positive prime integer N and two moduli p and q, with the condition that:

$$\gcd(N, q) = \gcd(p, q) = 1,$$

We impose the condition $q > (6d + 1)p$. This condition will be explained later.The public parameters (N, p, q, b) define the polynomial rings:

$$F = \mathbb{Z}[y]/(y^N - 1), \quad F_p = (\mathbb{Z}/p\mathbb{Z})[y]/(y^N - 1), \quad F_q = (\mathbb{Z}/q\mathbb{Z})[y]/(y^N - 1).$$

Key Generation: The private key consists of two ternary polynomials $f(y)$ and $g(y)$. A ternary polynomial is defined as:

Definition 4 (Ternary Polynomials:)**.** *The set $S(N, b_1, b_2)$ is defined as:*

$$S(N, b_1, b_2) = \left\{ b(y) \in F : \begin{array}{c} b_1 \text{ number of coefficients are } 1, \\ b_2 \text{ number of coefficients are } -1, \\ \text{remaining coefficients are } 0 \end{array} \right\}.$$

The private key consists of $f(y) \in S(N, b + 1, b)$ and $g(y) \in S(N, b, b)$. We compute the inverses:

$$S_q \equiv f^{-1} \mod q, \quad S_p \equiv f^{-1} \mod p.$$

If $f(y)$ is not invertible, we discard it and select a new one. The public key is:

$$k(y) \equiv S_q(y) \star g(y) \mod q.$$

The private key is the pair $(f(y), S_p(y))$.

Encryption: To encrypt a message $p(y)$, the sender represents it as a polynomial in the polynomial ring F, with coefficients between $-p/2$ and $p/2$. The sender picks a random polynomial $r(y) \in S(N, b, b)$ and computes the ciphertext c:

$$c(y) \equiv k(y) \star r(y) + p(y) \mod q.$$

Decryption: The recipient decrypts by computing:

$$d(y) \equiv f(y) \star c(y) \mod q.$$

To recover $p(y)$, center-lifting $d(y)$ to F and doing a mod p computation gives:

$$a(y) \equiv S_p(y) \star d(y) \mod p.$$

Again, center-lifting $a(y)$ to F retrieves the original message $p(y)$.
Example: Let us illustrate NTRUEncrypt by choosing the public parameters (N, p, q, b) as (11, 5, 41, 2). The recipient chooses two ternary polynomials:

$$f(y) = -y^8 - y^7 + y^6 + y^5 + y^2 \in S(3, 2)$$

$$g(y) = y^7 - y^6 - y^5 + y \in S(2, 2)$$

Then compute the inverses:

$$Sp(y) \equiv 4y^{10} + 3y^9 + 2y^8 + 4y^7 + 2y^6 + 2y^5 + 3y^4 + 3y^3 + 4y^2 + y + 3 \mod p$$

$$Sq(y) \equiv 2y^{10} + 6y^9 + 32y^8 + 22y^7 + 18y^6 + 3y^5 + 39y^4 + 28y^3 + 5y^2 + 2y + 8 \mod q$$

The public key is:

$$k(y) \equiv 33y^{10} + 11y^9 + 32y^8 + 19y^7 + 36y^6 + 35y^5 + 11y^4 + 30y^3 + 7y^2 + 12y + 20 \mod q$$

To encrypt a message $p(y)$, the sender picks a polynomial in F as plaintext and a random polynomial.

$$p(y) = y^{10} + y^9 - y^8 + 2y^7 + y^6 + 2y^5 - y^4 - y^3 + 2y^2 + 2$$

$$r(y) = -y^{10} + y^9 - y^8 + y^3$$

The cipher text $c(y)$ is as follows:

$$c(y) \equiv 21y^{10} + 15y^9 + 20y^8 + 33y^7 + 38y^6 + 38y^5 + 10y^4 + 39y^3 + 20y^2 + 33y + 28 \mod q$$

The recipient decrypts the message by computing:

$$d(y) \equiv f(y) \star c(y) \equiv 9y^{10} + 29y^9 + 36y^8 + y^7 + 35y^6 + 12y^5 + 3y^4 + 38y^3 + 5y^2 + 3y + 1 \mod q$$

To recover $p(y)$, center-lifting $d(y)$ to F and performing a mod p computation gives:

$$a(y) \equiv S_p(y) \star d(y) \equiv 4y^{10} + 3y^9 + y^7 + 4y^6 + 2y^5 + 3y^4 + 2y^3 + 3y + 1 \mod p$$

Now center-lifting $a(y)$ to an element of F retrieves the original $p(y)$.
Original message:

$$p(y) = y^{10} + y^9 - y^8 + 2y^7 + y^6 + 2y^5 - y^4 - y^3 + 2y^2 + 2$$

Decrypted message:

$$b(y) = y^{10} + y^9 - y^8 + 2y^7 + y^6 + 2y^5 - y^4 - y^3 + 2y^2 + 2$$

After following the steps for NTRU encryption and decryption, we observe that the decryption process successfully retrieves the original message in many cases. The correctness of the decryption heavily depends on the careful selection of parameters (N, p, q) and the properties of the polynomials used in the encryption process. These parameters determine the structure of the lattice and influence how easily the encryption can be broken or decrypted. The failure probability of decryption in NTRU is closely related to lattice reduction techniques, especially the effectiveness of lattice basis reduction algorithms like LLL (Lenstra-Lenstra-Lovász). The core idea of these techniques is to approximate the shortest vector in a given lattice, which in the case of NTRU corresponds to recovering the secret key. Lattice basis reduction transforms the lattice into a more manageable form, where the secret key can be more easily identified.
However, LLL does not always provide short enough vectors to successfully decrypt the message. While it can reduce the lattice, it may not produce the shortest possible vectors needed for accurate decryption, especially when the lattice is high-dimensional or poorly structured (e.g., when N is large or the parameters p and q are chosen improperly). This failure leads to decryption errors. The decryption failure probability increases as the difficulty of finding the shortest vector in the lattice increases.
Further analysis of this failure probability can give insights into the security of NTRU cryptosystem.

Chapter 4

Lattice Reduction Algorithms

Lattice reduction algorithms play a crucial role in computational number theory and cryptography, particularly in problems involving integer lattices. These algorithms aim to transform a given lattice basis into a nearly orthogonal and shorter basis, improving numerical stability and facilitating tasks such as solving the closest vector problem (CVP) and shortest vector problem (SVP).

Two important techniques related to lattice structures are the Gram-Schmidt Orthogonalization (GSO) and the Lenstra-Lenstra-Lovász (LLL) algorithm. GSO is a basis reduction method that orthogonalizes vectors in an arbitrary basis, helping to understand the lattice structure. However, it does not guarantee the shortest basis and works over fields, not integers. In contrast, the LLL algorithm is a lattice reduction technique that operates on integer vectors, reducing the basis iteratively to produce shorter, nearly orthogonal vectors. This makes LLL a widely used algorithm in cryptanalysis, integer programming, and post-quantum cryptography.

In this section, we discuss the mathematical formulation of these algorithms, their computational properties, and their significance in lattice-based cryptanalysis.

4.1 Gram-Schmidt Orthogonalization

The Gram-Schmidt orthogonalization process is a linear algebra method that is used to construct an orthogonal (or orthonormal) basis from a given set of vectors. It is performed on an arbitrary basis $(b_1, b_2, \ldots, b_n)$ of $\mathbb{R}^n$ to generate an orthogonal basis $(b_1^*, b_2^*, \ldots, b_n^*)$.

- The first vector of the orthogonal basis is $b_1^* = b_1$.

- For each subsequent vector b_i, it is adjusted by subtracting the projections onto the already orthogonalized vectors

$$b_i^* = b_i - \sum_{1 \le j < i} m_{ij} b_j^*$$

where the coefficients m_{ij} are given by

$$m_{ij} = \frac{b_i \cdot b_j^*}{\|b_j^*\|^2}.$$

This ensures the resulting vectors are orthogonal to each other.

Matrix Representation: Let B represent the matrix of the original vectors as rows and B^* the matrix of the orthogonalized vectors. The Gram-Schmidt process can also be expressed using the matrix M (which contains the coefficients m_{ij}), leading to the following relationships:

$$B = M \cdot B^*$$

$$B = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ m_{21} & 1 & 0 & \cdots & 0 \\ m_{31} & m_{32} & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & m_{n3} & \cdots & 1 \end{bmatrix} \cdot \begin{bmatrix} b_1^* \\ b_2^* \\ b_3^* \\ \vdots \\ b_n^* \end{bmatrix}$$

This method is essential in various applications, including the solving of linear systems, QR decomposition, and algorithms like the LLL (Lenstra-Lenstra-Lovász) lattice reduction algorithm, which builds on orthogonalization techniques.
For more details on Gram-Schmidt Orthogonalization, refer to the book by von zur Gathen and Gerhard [14, Chapter 16].
Example: Let $n = 3$, and define the basis vectors as:

$$b_1 = (2, 2, 0), \quad b_2 = (2, 0, 2), \quad b_3 = (0, 2, 2).$$

We know that:

$$b_1^* = b_1 = (2, 2, 0).$$

Compute the coefficient:

$$m_{21} = \frac{b_2 \cdot b_1^*}{\|b_1^*\|^2} = \frac{1}{2}.$$

Then, the second orthogonal vector is:

$$b_2^* = b_2 - m_{21} b_1^* = (1, -1, 2).$$

Similarly, compute:

$$m_{31} = \frac{b_3 \cdot b_1^*}{\|b_1^*\|^2} = \frac{1}{2},$$
$$m_{32} = \frac{b_3 \cdot b_2^*}{\|b_2^*\|^2} = \frac{1}{3}.$$

Thus, the third orthogonal vector is:

$$b_3^* = b_3 - m_{31} b_1^* - m_{32} b_2^* = \left(-\frac{4}{3}, \frac{2}{3}, \frac{2}{3}\right).$$

$$B = \begin{pmatrix} 2 & 2 & 0 \\ 2 & 0 & 2 \\ 0 & 2 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{2} & \frac{1}{3} & 1 \end{pmatrix} \cdot \begin{pmatrix} 2 & 2 & 0 \\ 1 & -1 & 2 \\ -\frac{4}{3} & \frac{4}{3} & \frac{4}{3} \end{pmatrix} = M \cdot B^*$$

The obtained orthogonal basis b_1^*, b_2^*, b_3^* are as follows:

$$b_1^* = (2, 2, 0), \quad b_2^* = (1, -1, 2), \quad b_3^* = \left(-\frac{4}{3}, \frac{4}{3}, \frac{4}{3}\right)$$

4.2 LLL Algorithm

The Lenstra–Lenstra–Lovász (LLL) algorithm, introduced by Arlert Lenstra, Hendrik Lenstra, and László Lovász in 1982[15], is a polynomial-time algorithm used for the reduction of the lattice basis. Lattice reduction refers to the process of transforming a given lattice basis into another one that is more orthogonal, with smaller vectors. The LLL algorithm aims to find a reduced basis, where the vectors are not only shorter but also closer to being mutually orthogonal, though not necessarily perfectly orthogonal.

The primary strength of the LLL algorithm lies in its ability to provide a nearly orthogonal basis, which helps to find relatively short vectors in a lattice. The algorithm is particularly efficient and runs in polynomial time, which is crucial for practical cryptographic applications. It operates on a lattice basis matrix and iteratively reduces it by performing specific transformations that involve Gram-Schmidt orthogonalization and lattice vector swaps. The output is a basis in which the vectors are shorter and better aligned, with certain bounds

on the length of the shortest vector in the reduced basis.
In lattice-based cryptography, the LLL algorithm is a fundamental tool because many cryptographic schemes rely on the hardness of lattice problems, such as finding the shortest vector or the closest vector in a lattice. The LLL algorithm helps break certain lattice-based cryptosystems, such as those based on NTRU or other polynomial rings, by reducing the complexity of these problems, although its effectiveness diminishes for larger parameter sizes due to the exponential growth of the search space and the increasing difficulty of the lattice reduction.
Additionally, LLL has a significant role in integer factorization problems. It has been used as part of techniques to factor large integers, particularly in methods such as the factorization of numbers represented as sums of squares. The algorithm's applications extend beyond cryptography and number theory to problems in coding theory, computational geometry, and algebraic number theory [15, 30, 31].
However, the LLL algorithm is not always capable for breaking cryptographic systems that rely on harder lattice problems, especially when the dimension of the lattice is large or the parameters are not suitable for LLL to be effective. In such cases, more advanced lattice reduction techniques, such as the BKZ-LLL algorithm, are used to provide more powerful reductions, often offering better results in breaking harder cryptographic systems.

Definition 5 (**LLL Reduced Basis:**). *Let $B = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n\}$ be a basis for a lattice L, and let $B^* = \{\mathbf{b}_1^*, \mathbf{b}_2^*, \ldots, \mathbf{b}_n^*\}$ be the associated Gram-Schmidt orthogonal basis. The basis B is said to be LLL reduced if it satisfies the following two conditions:*

1. ***Size Condition:***

$$|m_{i,j}| = \frac{\mathbf{b}_i \cdot \mathbf{b}_j^*}{\|\mathbf{b}_j^*\|^2} \leq \frac{1}{2} \quad \textit{for all } 1 \leq j < i \leq n.$$

2. ***Lovász Condition:***

$$\|\mathbf{b}_i^*\|^2 \geq \left(\frac{3}{4} - m_{i,i-1}^2\right) \|\mathbf{b}_{i-1}^*\|^2 \quad \textit{for all } 1 < i \leq n.$$

We will now describe an algorithm that generates a reduced basis for a lattice $L \subseteq \mathbb{Z}^n$ starting from an arbitrary basis. This algorithm can also be applied to determine a reduced basis for a lattice in $\mathbb{Q}^n$ by scaling the given basis vectors with a common denominator. For any real number m, we define $\lceil m \rceil$ as $\lceil m + \frac{1}{2} \rfloor$, which gives the integer that is closest to m.

Algorithm 1 LLL Algorithm

Input: Linearly independent row vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{Z}^n$.
Output: A reduced basis $(\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*)$ of the lattice $L = \sum_{i=1}^{n} \mathbb{Z}\mathbf{b}_i \subseteq \mathbb{Z}^n$.
Set $i = 2$.
$\mathbf{b}_1^* = \mathbf{b}_1$.
for $i = 2$ to n **do**
  for $j = i - 1$ to 1 **do**
    Compute $m_{i,j} = \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle}$.
    Update $\mathbf{b}_i^* = \mathbf{b}_i - \lfloor m_{i,j} \rceil \mathbf{b}_j^*$.
    for $k = 1$ to j **do**
      Update $m_{i,k} = m_{i,k} - \lfloor m_{i,j} \rceil m_{j,k}$.
    end for
  end for
  if Lovász Condition $\left(\|\mathbf{b}_i^*\|^2 \geq (\frac{3}{4} - m_{i,i-1}^2)\|\mathbf{b}_{i-1}^*\|^2\right)$ **then**
    $i = i + 1$.
  else
    Swap $\mathbf{b}_i$ and $\mathbf{b}_{i-1}$.
    $i = \max(i - 1, 2)$ and start from step-5.
  end if
end for
Return: Reduced basis $(\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*)$.

4.3 Application of LLL on NTRU Cryptosystem

The LLL algorithm is applied to the NTRUEncrypt cryptosystem to help with lattice reduction. NTRUEncrypt's security is based on the difficulty of solving certain lattice problems, such as the Shortest Vector Problem (SVP). The idea behind the cryptosystem is that finding short vectors in the lattice or solving the closest vector problem is computationally difficult. The LLL algorithm assists by reducing the lattice basis and providing an approximation of the shortest vector.

By reducing the lattice basis, LLL tries to find vectors that are as short as possible. These short vectors are important in cryptanalysis, as they can help reveal information about the private key. However, the effectiveness of LLL diminishes when applied to larger lattices. As the size of the lattice increases (due to larger parameter values in NTRU), the complexity of the lattice also increases, and LLL may fail to find sufficiently short vectors.

In such cases, the lattice reduction may not be precise enough to reveal the private key, leading to decryption failure. This highlights a limitation of the LLL algorithm: for larger values of N and more complex lattices, it may not provide an accurate approximation of the shortest vector, impacting the cryptosystem's performance.

4.3.1 NTRU as a Lattice Cryptosystem

We have the public key

$$k(y) = k_0 + k_1 y + \cdots + k_{N-1} y^{N-1}.$$

The NTRU lattice associated with $k(y)$ is the $2N$-dimensional lattice spanned by the rows of the matrix M_{NTRU} is called the NTRU lattice. The M_{NTRU} matrix is defined as follows:

$$M_{\text{NTRU}} = \left(\begin{array}{cccccccc} 1 & 0 & \cdots & 0 & k_0 & k_1 & \cdots & k_{N-1} \\ 0 & 1 & \cdots & 0 & k_{N-1} & k_0 & \cdots & k_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & k_1 & k_2 & \cdots & k_0 \\ \hline 0 & 0 & \cdots & 0 & q & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & q \end{array}\right)$$

Here, each block matrix is of size $n \times n$.
One can find a private key if one finds the shortest vector in the NTRU lattice. So, the security of NTRU depends on the difficulty of solving SVP in the NTRU lattice. By applying the LLL algorithm, one can find a shortest vector in the NTRU lattice.
If Shiv wants to find the private key. The only information he has is a public key. He can apply the LLL algorithm on the NTRU matrix to get the approximately shortest vector because the private key is very short. These propositions below will ensure that the vector $(f, g) \in L_{\text{NTRU}}$ the NTRU lattice and explain how the security of NTRU depends on solving the Shortest Vector Problem (SVP) in the NTRU lattice.

Proposition: Assuming that $f(x) \star k(x) \equiv g(x) \pmod{q}$, let $b(x) \in F$ be the polynomial satisfying

$$f(x) \star k(x) = g(x) + qb(x). \tag{1}$$

Then

$$(f, -b) \cdot M_{\text{NTRU}} = (f, g), \tag{2}$$

so the vector (f, g) lies in the NTRU lattice L_{NTRU}.

Proof. The first N coordinates of the product in (2) correspond to the vector f, as the left-hand side of M_{NTRU} is the identity matrix on top of a zero matrix. Next, consider the result of multiplying the column of M_{NTRU}, where the top entry is k_i, by the vector $(f, -b)$. The product gives the expression

$$k_i f_0 + k_{i-1} f_1 + \cdots + k_{i+1} f_{N-1} - q b_i,$$

which represents the i-th entry of the vector $f(x) \star h(x) - qu(x)$. From equation (1), this is the i-th entry of the vector g, so the second N coordinates of the product form the vector g. Finally, this indicates that we can obtain the vector (f, g) by taking a specific linear combination of the rows of M_{NTRU}. Hence, $(f, g) \in L_{\text{NTRU}}$.

Proposition. Let (N, p, q, b) be NTRU parameters, where for simplicity we will assume that

$$p = 3 \quad \text{and} \quad b \approx \frac{N}{3} \quad \text{and} \quad q \approx 6pb \approx 2pN.$$

Let L_{NTRU} be an NTRU lattice associated with the private key (f, g).

(a) $\det(L_{\text{NTRU}}) = q^N$.

(b) $(f, g) \approx \sqrt{4b} \approx \frac{4N}{3} \approx 1.155\sqrt{N}$.

(c) The Gaussian heuristic predicts that the shortest nonzero vector in the NTRU lattice has length

$$\sigma_{L_{\text{NTRU}}} \approx \frac{Nq}{\pi e} \approx 0.838N.$$

Hence, if N is large, there is a high probability that the shortest nonzero vectors in L_{NTRU} are (f, g) and its rotations. Further,

$$\frac{(f, g)}{\sigma(L)} \approx \frac{1.38}{\sqrt{N}},$$

so the vector (f, g) is a factor of $O\left(\frac{1}{\sqrt{N}}\right)$ shorter than predicted by the Gaussian heuristic.

The above proposition implies that Shiv can recover the private NTRU key if he can find

the shortest vector in the NTRU lattice L_{NTRU}. Therefore, the security of NTRUEncrypt is reliable on the difficulty of solving the Shortest Vector Problem (SVP) in the lattice L_{NTRU}. He has a public key:

$$k(y) \equiv 33y^{10} + 11y^9 + 32y^8 + 19y^7 + 36y^6 + 35y^5 + 11y^4 + 30y^3 + 7y^2 + 12y + 20 \mod q$$

He will compute the associated NTRU matrix M_{NTRU} with the public key:

$$\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 20 & 12 & 7 & 30 & 11 & 35 & 36 & 19 & 32 & 11 & 33 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 33 & 20 & 12 & 7 & 30 & 11 & 35 & 36 & 19 & 32 & 11 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 11 & 33 & 20 & 12 & 7 & 30 & 11 & 35 & 36 & 19 & 32 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 32 & 11 & 33 & 20 & 12 & 7 & 30 & 11 & 35 & 36 & 19 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 19 & 32 & 11 & 33 & 20 & 12 & 7 & 30 & 11 & 35 & 36 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 36 & 19 & 32 & 11 & 33 & 20 & 12 & 7 & 30 & 11 & 35 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 35 & 36 & 19 & 32 & 11 & 33 & 20 & 12 & 7 & 30 & 11 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 11 & 35 & 36 & 19 & 32 & 11 & 33 & 20 & 12 & 7 & 30 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 30 & 11 & 35 & 36 & 19 & 32 & 11 & 33 & 20 & 12 & 7 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 7 & 30 & 11 & 35 & 36 & 19 & 32 & 11 & 33 & 20 & 12 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 12 & 7 & 30 & 11 & 35 & 36 & 19 & 32 & 11 & 33 & 20 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 41 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 41 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 41 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 41 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 41 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 41 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 41 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 41 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 41 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 41 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 41
\end{pmatrix}$$

Then he applies the LLL algorithm on the NTRU matrix. This is how the matrix M_{LLL} will look like:

```
[-1 -1  1  1  0  0  0  0 -1  0  0  1  1 -1  0  0  0  0 -1  0  0  0]
[ 1  1  0  0  0  0 -1  0  0 -1 -1 -1  0  0  0  0 -1  0  0  0  1  1]
[ 0  0  1  1 -1 -1  0  0  0  0  1  0  0 -1 -1  1  0  0  0  0  1  0]
[ 0  0  0  0 -1  0  0 -1 -1  1  1  0  0  0 -1  0  0  0  1  1 -1  0]
[ 0  1  0  0  1  1 -1 -1  0  0  0  1  0  0  0 -1 -1  1  0  0  0  0]
[ 0  0  0  1  0  0  1  1 -1 -1  0  0  0  1  0  0  0 -1 -1  1  0  0]
[ 1  0  0  0  0 -1  0  0 -1 -1  1  0  0  0  0 -1  0  0  0  1  1 -1]
[ 0  0 -1  0  0 -1 -1  1  1  0  0  0 -1  0  0  0  1  1 -1  0  0  0]
[-1  0  0 -1 -1  1  1  0  0  0  0  0  0  0  1  1 -1  0  0  0  0 -1]
[ 1  1  1  1  1  1  1  1  1  1  1  0  0  0  0  0  0  0  0  0  0  0]
[ 0  1  1 -1 -1  0  0  0  0  1  0  0 -1 -1  1  0  0  0  0  1  0  0]
[ 4  8 -4 -3 -1 -5  1  6 -4  4 -4  7  5 -2 -4  2  1 -2  1 -2 -2 -4]
[-3  6 -6 -1 -1  2  0  6 -2  0  2  0  1 -7 -5 -6 -1 -9 -1 -7 -2 -4]
[ 5  3  1  4 -1 -6  4 -2  5 -4 -8  1  4 -3  1  2 -1  3  1  4 -6 -6]
[-7  5  3  1  4 -2 -6  4 -3  4 -3 -6  1  4 -3  0  2 -1  3  2  5 -7]
[-5  2  8 -4 -3 -1 -4  2  5 -4  3 -4  8  5 -1 -4  3  0 -3  1 -3 -2]
[ 7 -7 -1 -2  1 -2  6 -2  0  2 -2  0 -7 -6 -6 -1 -8 -1 -6 -2 -3 -1]
[ 4 -3  5 -3 -7  4  3  1  4  0 -5  4  1  4 -7 -7  3  4 -3  1  1 -1]
[-2  6 -3  0  2 -3  6 -6  0 -2  1 -8 -2 -6 -2 -3  0  1 -8 -6 -6 -1]
[ 3  1  4 -2 -6  4 -3  4 -3 -7  5  4 -3  0  2 -1  3  2  5 -7 -6  1]
[ 5 -3  0  2 -2  7 -6 -1 -2  2 -1  0 -6 -2 -3 -1  1 -7 -6 -6 -2 -9]
[-4  4 -4 -8  4  3  1  5 -1 -6  4  2  4 -7 -5  2  4 -2 -1  2 -1  2]
```

Comparing the Hadamard ratio of both matrices will give information about the orthogonality of the basis before and after the reduction. The Hadamard ratio of both matrices is as follows:

$$H(M_{\text{NTRU}}) \approx 0.1104028037 \quad \text{and} \quad H(M_{\text{LLL}}) \approx 0.8442368639.$$

He will find the shortest non-zero row vector from the matrix. The shortest vector in the LLL matrix is:

$$(-1, -1, 1, 1, 0, 0, 0, 0, -1, 0, 0, 1, 1, -1, 0, 0, 0, 0, -1, 0, 0, 0).$$

This is the top row of the matrix. Splitting this vector into two equal parts will give the polynomials:

$$f'(y) = -y^8 + y^3 + y^2 - y - 1$$

$$g'(y) = -y^7 - y^2 + y + 1$$

The polynomials $f'(y)$ and $g'(y)$ are different from the original private polynomials $f(y)$ and

$g(y)$. Now, Shiv can decrypt the message using $f'(y)$ and $g'(y)$ polynomials as his private key. Let's see how this works.
First, Shiv starts with the computed polynomials as his private key

$$f'(y) = -y^8 + y^3 + y^2 - y - 1$$

$$g'(y) = -y^7 - y^2 + y + 1$$

Then he computes inverses of $f'(y)$:

$$S'_p(y) \equiv 3y^{10} + 2y^9 + 2y^8 + y^7 + 4y^6 + 2y^5 + y^4 + 2y^3 + 3y^2 + y + 3 \mod p$$

$$S'_q(y) \equiv 38y^{10} + 2y^9 + 13y^8 + 36y^7 + 39y^6 + 33y^5 + 39y^4 + 35y^3 + 9y^2 + 19y + 23 \mod q$$

He has access to the public key, ciphertext given below:

$$k(y) \equiv 33y^{10} + 11y^9 + 32y^8 + 19y^7 + 36y^6 + 35y^5 + 11y^4 + 30y^3 + 7y^2 + 12y + 20 \mod q$$

$$c'(y) \equiv 21y^{10} + 15y^9 + 20y^8 + 33y^7 + 38y^6 + 38y^5 + 10y^4 + 39y^3 + 20y^2 + 33y + 28 \mod q$$

To decrypt the message $p(y)$, he comutes the polynomial

$$d'(y) \equiv f'(y) \star c'(y) \equiv 38y^{10}+3y^9+36y^8+38y^7+40y^6+32y^5+12y^4+5y^3+40y^2+6y+29 \mod q$$

He then center lifts the polynomial $d'(y)$ to an element of F and does a modulo p computation and then computes

$$a'(y) \equiv S'_p(y) \star d'(y) \equiv 2y^{10} + 3y^9 + 2y^7 + 4y^6 + y^5 + 2y^4 + 4y^2 + y + 3 \mod p$$

Center-lifting an $a'(y)$ say $b'(y)$ to F retrieves the message $p(y)$ as follows:
Original message $p(y)$:

$$p(y) = y^{10} + y^9 - y^8 + 2y^7 + y^6 + 2y^5 - y^4 - y^3 + 2y^2 + 2$$

Decrypted message $b'(y)$:

$$b'(y) = y^{10} + y^9 - y^8 + 2y^7 + y^6 + 2y^5 - y^4 - y^3 + 2y^2 + 2$$

This process using computed polynomials (f', g') demonstrates the successful encryption and

decryption of a message using a modified NTRU-based cryptographic scheme. One of the key aspects of this method is the comparison of the Hadamard ratio of the original NTRU basis and the reduced basis obtained through the LLL (Lenstra-Lenstra-Lovász) algorithm. The Hadamard ratio serves as a measure of the orthogonality of a basis, with higher values indicating a more orthogonal set of vectors. After lattice reduction, the basis becomes significantly more orthogonal, which is crucial for improving the efficiency of the decryption process. A more orthogonal basis reduces computational complexity, enhances numerical stability, and improves the effectiveness of the private key. By leveraging lattice reduction, the scheme ensures that decryption remains feasible and efficient while maintaining security. As I mentioned earlier in this section, as the size of parameters and lattice increases, LLL will fail to find sufficiently short vectors. This we can observe in the below graph.

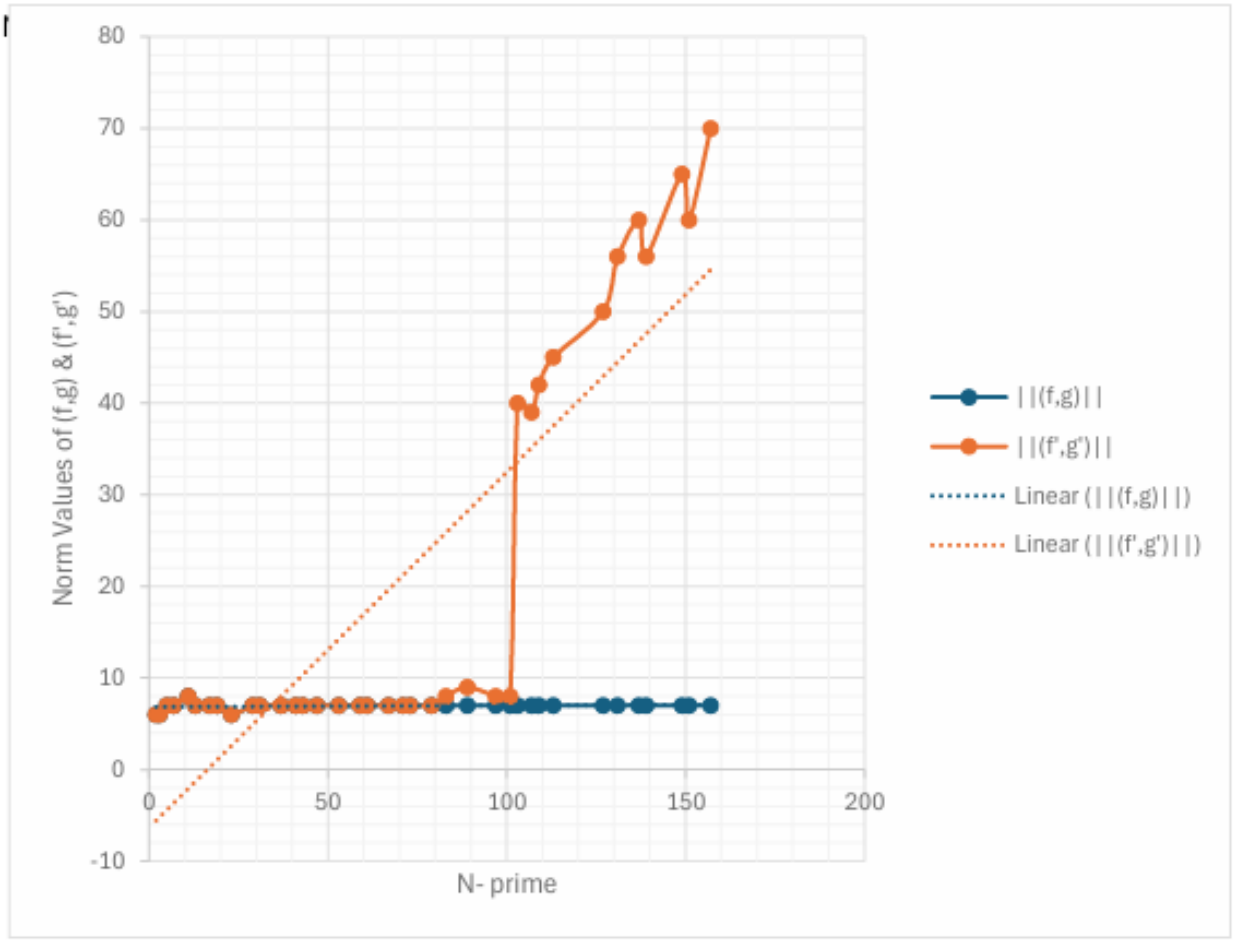


Figure 4.1: Effect of Increasing N on LLL Reduction: As N increases, the shortest vector (f', g') found by LLL deviates from the original private vector (f, g), leading to decryption failure using (f', g').

The graph above illustrates the effect of increasing N on the performance of the LLL algorithm in finding short vectors. As N grows, the LLL-reduced basis vectors (f', g') begin to deviate significantly from the original private key vectors (f, g). This deviation occurs because the LLL algorithm provides only an approximation to the shortest vector and becomes less effective in high-dimensional lattices. As a result, when (f', g') diverges too far from (f, g), decryption using (f', g') in the NTRU cryptosystem fails, as the transformation no longer correctly reconstructs the original message.

Chapter 5

Security of NTRU

The security of NTRU is based on the difficulty of solving certain lattice problems, particularly the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP). These problems are computationally hard and provide the foundation of NTRU's encryption strength. The system uses a ring of polynomials, and the hardness of lattice-based problems, such as finding short vectors in high-dimensional lattices, ensures the security of the scheme. Even with the threat of quantum computing, solving these lattice problems remains difficult, making NTRU resilient against quantum attacks.

However, NTRU's security is highly dependent on choosing the right cryptographic parameters. If parameters like N, q, p, and b used in the encryption process are not carefully selected, it could lead to weaknesses that make the system vulnerable to attacks. Lattice reduction algorithms like LLL and BKZ, designed to find short vectors in a lattice, can exploit poor parameter choices [16]. Such attacks may allow an adversary to break the encryption without the private key, highlighting the importance of a careful parameter selection.

Another key aspect of NTRU's security is the use of center-lifting polynomials. The way f, g polynomials are chosen and structured can influence the overall strength of the system. Incorrect choices can result in easier attacks, as they may alter the lattice in a way that makes it more susceptible to reduction. Therefore, both the selection of parameters and the use of appropriate center-lifting polynomials are crucial for maintaining the security of NTRU.

Proposition: If the NTRU parameters (N, p, q, b) are chosen to satisfy

$$q > (6b + 1)p$$

then the decrypted message $b(y)$ is equal to the original message $p(y)$.

Proof. We have the polynomial $d(y)$:

$$d(y) \equiv f(y) \star c(y) \mod q. \tag{5.1}$$

where

$$c(y) \equiv p \cdot p_u(y) \star r(y) + p(y) \mod q.$$

Substituting this in the above equation gives:

$$d(y) \equiv f(y) \star (p \cdot p_u(y) \star r(y) + p(y)) \mod q. \tag{5.2}$$

Substituting

$$p_u(y) \equiv S_q(y) \star g(y) \mod q$$

yields:

$$d(y) \equiv pf(y) \star S_q(y) \star g(y) \star r(y) + f(y) \star p(y) \mod q. \tag{5.3}$$

This simplifies to:

$$d(y) \equiv p \cdot g(y) \star r(y) + f(y) \star p(y) \mod q. \tag{5.4}$$

Now, considering the polynomial in F instead of modulo q, we have:

$$pg(y) \star r(y) + f(y) \star p(y). \tag{5.5}$$

Bounding the largest possible coefficient: Since $g(y), r(y) \in S(b, b)$, their convolution $g(y) \star r(y)$ has a maximum coefficient of $2b$. Similarly, since $f(y) \in S(b+1, b)$ and the coefficients of $p(y)$ are between $-\frac{1}{2}p$ and $\frac{1}{2}p$, the largest possible coefficient of $f(y) \star p(y)$ is $(2b+1) \cdot \frac{1}{2}p$. Thus, the largest coefficient of the expression is at most:

$$p \cdot 2b + (2b+1) \cdot \frac{1}{2}p = 3b + \frac{1}{2}p. \tag{5.6}$$

The assumption

$$q > (6b+1)p$$

ensures that every coefficient has a magnitude strictly smaller than $\frac{1}{2}q$. Therefore, when computing $d(y)$ modulo q and then lifting it back, the exact value is recovered:

$$d(y) = p \cdot g(y) \star r(y) + f(y) \star p(y). \tag{5.7}$$

Next, multiplying $d(y)$ by $S_p(y)$, the inverse of $f(y)$ modulo p, and reducing modulo p, we get:

$$b(y) \equiv S_p(y) \star d(y) \mod p. \tag{5.8}$$

Substituting for $d(y)$,

$$b(y) \equiv S_p(y) \star (p \cdot g(y) \star r(y) + f(y) \star p(y)) \mod p. \tag{5.9}$$

Since $p \cdot g(y) \star r(y) \equiv 0 \mod p$, reducing modulo p gives:

$$b(y) \equiv S_p(y) \star f(y) \star p(y) \mod p. \tag{5.10}$$

Using the property $S_p(y) \star f(y) \equiv 1 \mod p$, we obtain:

$$b(y) \equiv p(y) \mod p. \tag{5.11}$$

Thus, $b(y)$ and $p(y)$ are the same modulo p, completing the proof. □

The proof shows that if $q > (6b + 1)p$, then decrypted message $b(y)$ will match the original message $p(y)$ modulo p. This condition ensures the coefficients remain within bounds, preventing decryption errors. We can better understand the importance of this condition $q > (6b + 1)p$ through the following example.

Example Let's take NTRU parameters $N, p, q, b = (11, 13, 41, 2)$. Here

$$41 = q < (6 \cdot 2 + 1) \cdot 13 = 149.$$

Let's choose two random polynomials $f(y)$ and $g(y)$ as

$$f(y) = -y^9 - y^6 + y^5 + y^4 + y^3 \in S(3, 2)$$

$$g(y) = y^9 - y^6 - y^4 + 1 \in S(2, 2)$$

Then, the inverse of $f(y)$ in S_q and S_p are as follows:

$$S_p(y) = 2y^{10} + 10y^9 + 8y^8 + 3y^7 + 4y^4 + 8y^3 + 10y^2 + 12y + 9 \in F_p$$

$$S_q(y) = 6y^{10} + y^9 + 38y^8 + 5y^7 + 38y^6 + 20y^5 + 40y^4 + 6y^3 + 19y^2 + 23y + 10 \in F_q$$

The public key is:

$$k(y) = S_q(y) \star g(y) = 33y^{10}+26y^9+26y^8+18y^7+6y^6+37y^5+26y^4+23y^3+12y^2+35y+4 \in F_q$$

For encryption, take a plaintext $p(y) \in F$ and a random polynomial $r(y) \in S(2,2)$, and compute the ciphertext $c(y)$ as follows:

$$p(y) = -5y^{10} + 6y^9 + y^8 + 5y^7 + 6y^6 - 5y^5 + 6y^4 - y^3 + 3y - 6$$

$$r(y) = y^8 - y^5 - y^2 + 1$$

$$c(y) = p \cdot k(y) \star r(y) + p(y) = 7y^{10}+22y^9+14y^8+31y^7+39y^6+12y^5+34y^4+34y^3+y^2+10y+11 \in F_q$$

Then, for decrypting the plaintext or message, compute the polynomial:

$$d(y) = f(y) * c(y) = 38y^{10}+40y^9+31y^8+37y^7+20y^6+25y^5+8y^4+14y^3+19y^2+35y+30 \in F_q$$

Next, center-lifting this polynomial will give this polynomial as an element of F, and perform a mod p computation:

$$a(y) = S_p(y) \star d(y) = 10y^{10} + 12y^9 + 3y^8 + 9y^7 + 7y^6 + 10y^5 + 8y^4 + y^3 + 6y^2 + 7y + 2 \in F_p$$

Center-lifting of $a(y)$ (say $b(y)$) to an element of F should retrieve the message $p(y)$ as follows:
Decrypted message:

$$b(y) = -5y^{10} + 2y^9 - 3y^8 - y^7 + y^6 - 6y^5 + 2y^4 + 3y^3 + y^2 - 5y - 5$$

Original message:

$$p(y) = -5y^{10} + 6y^9 + y^8 + 5y^7 + 6y^6 - 5y^5 + 6y^4 - y^3 + 3y - 6$$

Both the original message and the decrypted message are not the same, and the decryption failed. This example ensures that satisfying the condition $q > (6d + 1)p$ will not lead to decryption failure.
During experimenting with different parameters, I noticed that when there is a decryption success, the center-lifted polynomial $d(y)$ to an element of F is the same as a polynomial $v(y) = p \cdot g(y) + f(y) \cdot p(y)$, which is an element of F because all polynomials have smaller

coefficients.
However, when there is a decryption failure, these two polynomials are not the same, although they should be. From this observation, I conclude that center-lifting a polynomial plays a vital role in the success or failure of decryption.
The security of the NTRU cryptosystem depends on the computational hardness of lattice problems, particularly the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP). Our analysis demonstrates that for small values of N, the LLL algorithm does not significantly impact security. However, as N increases, LLL reduction fails to provide a sufficiently short basis, leading to decryption failures. This indicates that NTRU remains resistant to basic lattice reduction techniques like LLL. Further exploration using the BKZ algorithm is necessary to evaluate its resilience against more powerful attacks. Additionally, parameter selection plays a crucial role in security and decryption reliability.

Chapter 6

Conclusion

The growing importance of secure communication in the digital age has underscored the need for robust cryptographic systems. Traditional cryptographic schemes, though effective against classical computational threats, face significant vulnerabilities in the advent of quantum computing. This thesis has addressed these challenges by focusing on lattice-based cryptography, which offers a promising foundation for quantum-resistant security protocols.
Central to our study was the detailed exploration of the NTRU cryptosystem, a prominent lattice-based public key scheme. We began by examining the underlying algebraic structures of NTRU, particularly convolution polynomial rings, which form the basis for its encryption and decryption processes. The theoretical framework of NTRU was systematically analyzed to demonstrate how its design capitalizes on the inherent complexity of lattice problems.
A key emphasis was placed on understanding the role of lattice reduction algorithms, especially the Lenstra–Lenstra–Lovász (LLL) algorithm. We analyzed how these reduction techniques can potentially weaken the security of NTRU by simplifying its associated lattice structures. Through practical application, we illustrated how the LLL algorithm attempts to recover private keys by finding short vectors within the NTRU lattice.
Our results revealed that while NTRU exhibits certain vulnerabilities when small parameter sizes are used, its security significantly strengthens as the lattice dimension increases. The analysis demonstrated that careful parameter selection is crucial to ensuring that the NTRU cryptosystem remains resistant to attacks based on lattice reduction. Additionally, the experiments highlighted the impact of center-lifting and its relation to decryption correctness, offering further insights into the internal mechanics of NTRU.
Overall, this thesis underscores the critical relationship between lattice theory, reduction algorithms, and cryptographic resilience. By systematically analyzing the interplay between

these components, we have provided a comprehensive understanding of the strengths and limitations of lattice-based cryptosystems. This work reinforces the viability of NTRU as a secure and efficient solution for safeguarding digital communications against both classical and emerging quantum threats.

Bibliography

[1] Whitfield Diffie and Martin Hellman, "New Directions in Cryptography", *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, pp. 644-654, 1976.

[2] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", *Communications of the ACM*, vol. 21, no. 2, pp. 120-126, 1977.

[3] Neal Koblitz, "Elliptic Curve Cryptosystems", *Mathematics of Computation*, vol. 48, no. 177, pp. 203-209, 1990.

[4] Simon Singh, *The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography*, Anchor, 1999.

[5] David Kahn, *The Codebreakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet*, Scribner, 1996.

[6] Bruce Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd ed., Wiley, 1996.

[7] Eric Rescorla, *SSL and TLS: Designing and Building Secure Systems*, Addison-Wesley, 2000.

[8] Oded Regev, "On Lattices, Learning with Errors, Random Linear Codes, and Cryptography", *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 84-93, 2005.

[9] Peter W. Shor, "Algorithms for Quantum Computation: Discrete Logarithms and Factoring", *Proceedings of the 35th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 124-134, 1994.

[10] Lindy Chen and Yong Zhang, "Post-Quantum Cryptography: An Overview", *Cryptography and Communications*, vol. 8, no. 2, pp. 131–164, 2016.

[11] Daniel J. Bernstein, Tanja Lange, and Peter Schwabe, "The Security of Cryptographic Algorithms: An Overview", *Springer Handbook of Information Security*, pp. 1061-1082, 2019.

[12] Phong Q. Nguyen and Oded Regev, "Cryptanalysis of the GGH Signature Scheme", 2006.

[13] John H. Conway and Neil J. A. Sloane, *Sphere Packings, Lattices and Groups*, 3rd ed., Springer, 1999.

[14] J. von zur Gathen and J. Gerhard, *Modern Computer Algebra*, Cambridge University Press, 2013, Chapter 16.

[15] Arjen Lenstra, Hendrik Lenstra, and László Lovász. *Factoring polynomials with rational coefficients*. Mathematics of Computation, 44(170):129-133, 1982. .

[16] J. Hoffstein, J. Pipher, and J. H. Silverman, *An Introduction to Mathematical Cryptography*, Springer, 2008.

[17] Alfred Menezes, Paul C. van Oorschot, and Scott A. Vanstone, *Handbook of Applied Cryptography*, 2018.

[18] N. Howgrave-Graham, P. Q. Nguyen, D. Pointcheval, J. Proos, J. H. Silverman, A. Singer, W. Whyte, "The Impact of Decryption Failures on the Security of NTRU Encryption", In *Proceedings of the Annual International Cryptology Conference (Crypto 2003)*.

[19] D. Coppersmith, A. Shamir, "Lattice Attacks on NTRU", In Fumy, W. (Ed.), *Advances in Cryptology —EUROCRYPT 1997*, Lecture Notes in Computer Science, vol. 1233, Springer, Berlin, Heidelberg.

[20] Adleman, L. M., Pomerance, C., & Rumely, R. (1983). On distinguishing prime numbers from composite numbers. *Journal of the ACM (JACM), 30*(2), 304-322.

[21] Kocher, P., Jaffe, J., & Jun, B. (1999). Differential Power Analysis. In *Advances in Cryptology—CRYPTO '99* (pp. 388-397). Springer.

[22] National Institute of Standards and Technology (NIST). (2019). Recommendation for Key Management: Part 1: General. *NIST Special Publication 800-57, Revision 5*.

[23] National Institute of Standards and Technology (NIST). (2013). Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revised). *NIST Special Publication 800-56A, Revision 3*.

[24] Regev, O. (2009). Lattice-based cryptography. In *Handbook of Applied Cryptography* (pp. 787-804). CRC Press.

[25] Nguyen, P. Q. (1999). Breaking the GGH encryption scheme. *Lecture Notes in Computer Science, 1807*, 191-204.

[26] Nguyen, P. Q., & Regev, O. (2006). Cryptanalysis of the GGH signature scheme. *Journal of Cryptology, 19*(4), 465-495.

[27] Miklós Ajtai, "Generating Hard Instances of Lattice Problems," in *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC)*, 1996, pp. 99–108.

[28] Cynthia Dwork, "The Short Integer Solutions Problem and Cryptography," in *Advances in Cryptology - EUROCRYPT 1996*, Springer, 1996, pp. 109–120.

[29] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman, "NTRU: A Ring-Based Public Key Cryptosystem," in *Proceedings of the Third International Algorithmic Number Theory Symposium*, vol. 1423, pp. 267-288, Springer-Verlag, 1998.

[30] Schnorr, C. P. (1987). A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical Computer Science, 53*(2-3), 225-253.

[31] Kannan, R. (1987). Polynomial time algorithms for integer programming and related problems. *Mathematics of Operations Research, 12*(3), 415-446.