# LEARNING WITH ERRORS

ETHAN YANG

Abstract. This article discusses the Learning with Errors and the Ring Learning with Errors problem and their cryptographic applications in a gentle fashion.

## 1. Introduction

Most of current cryptosystems in use today are based on either the discrete logarithm problem, the factoring problem, or the elliptic curve discrete logarithm problem. However, in 1999, Shor [Sho99] showed that these problems can be solved quickly using a quantum computer. Although current quantum computers are relatively far from becoming a serious threat, a lot of current cryptographic research focuses on developing quantum resistant cryptography, or algorithms that are believed to be secure against attacks from quantum computers.

One such area of research is in problems related to lattices. For a more in-depth treatement of lattice-based cryptography, see [Pei16]. We will focus on the Learning with Errors (LWE) problem, as well as the related Ring Learning with Errors (RLWE) problem. It was introduced by Regev [Reg10] in 2005, RLWE was introduced by Lyubashevsky, Peikert, and Regev [LPR13] in 2013. Regev recieved the 2018 Gödel prize for his work in 2005.

NIST has a project focusing on the standardization of post-quantum cryptography. Many of the algorithms within their competition are based on lattices, LWE, and RLWE. One such example is from Alkim et al. [ADPS16], who have submitted their algorithm based on RLWE, NewHope, to the project and have made it to the second round along with 25 other candidates. Google previously tested CECPQ1, a cipher based on RLWE, to make web browsers secure to quantum computers.

The rest of this article is structured as follows. Section 2 will give some definitions and a gentle introduction to the problem. Section 3 will serve to provide a formal definition for LWE and provide some results about the problem. Sections 4 and 6 describe variants of Learning with Errors, and sections 5 and 7 describe cryptographic applications for LWE.

## 2. Definitions and Overview

Let $\mathbb{Z}$ be the ring of integers and let $\mathbb{Z}/q\mathbb{Z}$ denote the integers modulo $q$. Let $\mathbf{a}, \mathbf{s} \in (\mathbb{Z}/q\mathbb{Z})^n$ be vectors, and let $\langle \mathbf{a}, \mathbf{s} \rangle = \sum_{i=1}^{n} a_i s_i$ be the standard inner product or dot product of two vectors. Let poly$(n)$ mean "some polynomial in $n$."

We first give an extremely easy variant of the Learning with Errors problem, the Learning without Errors problem. Imagine two characters, Jeb and Eliza, where Eliza has locked Jeb in a room and he is only given access to a button. This button has some known information about it, namely that it has a dimension $n \in \mathbb{N}$ and a modulus $q \in \mathbb{N}$. However, it also contains a secret vector $\mathbf{s} \in (\mathbb{Z}/q\mathbb{Z})^n$. Upon pressing this button, Jeb receives a random

*Date*: December 9, 2019.

vector $\mathbf{a} \in (\mathbb{Z}/q\mathbb{Z})^n$ along with the inner product modulo $q$, $\langle \mathbf{a}, \mathbf{s} \rangle \in \mathbb{Z}/q\mathbb{Z}$. Jeb can hit this button as many times as he wants, and his goal is to quickly figure out what the secret $\mathbf{s}$ is, preferably in a time that is polynomial in $n$. If he figures out this secret, then Eliza will let him out of the room.

After writing down what the button gives him, Jeb realizes that this is a system of linear equations, modulo $q$. Since Jeb is smart and remembers from Algebra 2 that he can solve a system of linear equations using Gaussian Elimination, he can always find the secret vector $\mathbf{s}$ in polynomial time once he is given $n$ independent vectors.

*Example.* Imagine that $n = 2$ and $q = 5$. Jeb pushes the button three times and gets the equations

$$\begin{aligned} 2s_1 + s_2 &\equiv 2 \pmod 5 \\ s_1 + 3s_2 &\equiv 1 \pmod 5 \\ 2s_1 + 3s_2 &\equiv 3 \pmod 5. \end{aligned}$$

The first and second equations are dependent. Using Gaussian Elimination, Jeb realizes that $\mathbf{s} = (2, 3)$.

Now, Jeb has solved the Learning without Errors problem and escaped the room! Unfortunately, Eliza is extremely evil and he is unfortunately trapped in a different room with another, more menacing button. This button represents the Learning with Errors problem. Along with the dimension $n$ and the modulus $q$, the button is also known to have an error parameter $B \in \mathbb{Z}/q\mathbb{Z}$. When Jeb pushes the button, he is presented a random vector $\mathbf{a} \in (\mathbb{Z}/q\mathbb{Z})^n$ and the inner product plus some error term $\langle \mathbf{a}, \mathbf{s} \rangle + e \in \mathbb{Z}/q\mathbb{Z}$, where, for now, we can think of $e$ being randomly chosen in the interval $[-B, B]$.

Now, instead of being given linear equations, Jeb is essentially only given some noisy approximate linear equations, where Jeb does not know how much it errs. His Gaussian Elimination algorithm does not work, and Jeb cannot think of any algorithms to solve the problem, so he is stuck in the room forever and Eliza is happy.

*Example.* Imagine that $n = 2$, $q = 5$, and $B = 1$. One possible scenario is that Jeb pushes the button many times and gets the equations

$$\begin{aligned} 2s_1 + s_2 &\approx 1 \pmod 5 \\ s_1 + 3s_2 &\approx 1 \pmod 5 \\ 2s_1 + 3s_2 &\approx 3 \pmod 5 \\ 4s_1 + 2s_2 &\approx 4 \pmod 5 \\ 3s_1 + 4s_2 &\approx 4 \pmod 5 \\ &\vdots \\ 2s_1 + s_2 &\approx 2 \pmod 5. \end{aligned}$$

Jeb, being bored and hopeless, sits in the room and just continuously presses the button. Eventually, he is presented with a vector $\mathbf{a}$ that is of the form $(1, 0, 0, \ldots)$. He realizes that the inner product $\langle \mathbf{a}, \mathbf{s} \rangle + e = s_1 + e$ told him an approximate value of $s_1$. Excited, Jeb uses his expert skill at pressing buttons and presses the button extremely quickly. He is able to

eventually produce many vectors of the form $(1, 0, 0, \ldots)$. Given enough of these, Jeb can guess with relatively high probability what $s_1$ is. Jeb repeats the process for the whole vector $\mathbf{s}$, and is let out of the room by Eliza.

Of course, in reality, Jeb's algorithm would take an enormously long amount of time for large values of $n$. After escaping the room, Jeb is extremely excited to learn more about this problem, so he reads the rest of this paper.

## 3. Learning with Errors

We now formally define the LWE problem.

We set a couple of parameters. $n$ is called the degree, or the security parameter. $q$ is the modulus. Instead of an interval of error, we set the probability distribution $\chi$ on $\mathbb{Z}/q\mathbb{Z}$ to be the error distribution. Let $A_{\mathbf{s},\chi} \in \mathbb{Z}/q\mathbb{Z}^n \times \mathbb{Z}/q\mathbb{Z}$ be the probability distribution obtained by choosing a vector $\mathbf{a} \in \mathbb{Z}/q\mathbb{Z}^n$ uniformly at random and $e \in \mathbb{Z}/q\mathbb{Z}$ chosen according to $\chi$, and outputting $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e)$. An algorithm is said to solve LWE with modulus $q$ and error distribution $\chi$ if, for any vector $\mathbf{s} \in \mathbb{Z}/q\mathbb{Z}^n$, given an arbitrary amount of independent samples from $A_{\mathbf{s},\chi}$, it outputs $\mathbf{s}$ with large probability.

This is called the search variant of the LWE problem, as the goal is to find such the secret $\mathbf{s}$. There is also a decision variant, which is to distinguish the LWE samples from uniformly random samples.

### 3.1. Parameters.

Generally, we let $q$ be polynomial in $n$. We take the error distribution $\chi$ to be the discrete Gaussian distribution with standard deviation $\alpha q$, where $\alpha > 0$ is $1/\operatorname{poly}(n)$. One can think of this as the Gaussian distribution with standard deviation $\alpha q$, taking values modulo $q$ and rounding every point to the nearest integer. It turns out that the hardness of this problem is mainly dependent on $n$. Taking large $q$ does make it the problem slightly more difficult, but it has the disadvantage of making cryptographic algorithms inefficient since it requires computation with large numbers.

Of course, there are certain parameters that would be bad choices for our problem. For example, choosing $\chi$ to be the uniformly random distribution across all values of $\mathbb{Z}/q\mathbb{Z}$ will lead to no one ever being able to find the secret. Furthermore, we need $q > \sqrt{n}$ and the error cannot be always smaller than $\sqrt{n}$ due to a security vulnerability from Arora and Ge [AG11], which we will discuss shortly. We let the reader research the topic further for more informed choices of parameters.

### 3.2. Attacks.

The easiest algorithm, the one that our character Jeb used, where he tries to find vectors of the form $\mathbf{a} = (\ldots, 0, 1, 0, \ldots)$ has a complexity of $2^{O(n \log n)}$. This is because finding such a polynomial is expected to take $q^{-n}$, and we need slightly more equations to determine $\mathbf{s}$.

Improving this slightly, Blum, Kalai, and Wasserman [BKW03] had an algorithm to solve LWE in $2^{O(n)}$. The reader can view their article for more information on their algorithm.

Arora and Ge [AG11] created an attack for when the error is small. In fact, this is the vulnerability mentioned above for why the restrictions on the error and modulus exist. Again, the reader is referred to their paper for the full details of their algorithm, which is slightly sub-exponential. We give a brief example to give insight as to the general argument of the algorithm. Imagine that the error was uniformly chosen from the set $\{-1, 0, 1\}$. Then, if $A_{\mathbf{s},\chi}$ outputs tuples $(\mathbf{a}, b)$, we know that

$$(b - (\langle \mathbf{a}, \mathbf{s} \rangle - 1))(b - (\langle \mathbf{a}, \mathbf{s} \rangle))(b - (\langle \mathbf{a}, \mathbf{s} \rangle + 1)) = 0.$$

Expanding this gives us a degree three polynomial in terms of $s_i$, such as

$$\sum_{i,j,k} \alpha_{i,j,k} s_i s_j s_k + \sum_{i,j} \beta_{i,j} s_i s_j + \sum_i \gamma_{i,j,k} s_i + \delta = 0,$$

for some known and easy-to-compute coefficients $\alpha, \beta, \gamma, \delta$. We can relabel the products of $s_i$ with other variables, and this equation becomes linear. Given enough of these, we can solve the equation using Gaussian Elimination.

3.3. **Hardness.** The current fastest known algorithm to solve the LWE problem is exponential. Furthermore, LWE is known to be as least as hard as certain problems regarding lattices. In particular, it relies on the *worst-case* hardness of lattice problems, contrasting algorithms that rely on factoring, for example. If factoring was found to be easy in the average case, then cryptographic schemes relying on factoring would be broken. One can see [Reg09] for more information.

## 4. Decisional Learning with Errors

To discuss the Decisional Learning with Errors problem, we refer back to our characters Eliza and Jeb. Again, Eliza has trapped Jeb in one of two possible identical rooms, each with a button inside. The only difference between the rooms is that the button in one room outputs $(\mathbf{a}, u)$, where $u \in \mathbb{Z}/q\mathbb{Z}$ is chosen randomly. The other button is the exact same button as the one mentioned earlier. Now, Jeb's goal is to, with high probability, tell Eliza which room he is in. Again, this is hard, and it turns out to be equivalent to the search variant.

Formally, an algorithm solves decisional LWE if, with high probability, can distinguish between whether it has access to $A_{\mathbf{s},\chi}$ or randomly chosen outputs. In the following proofs, let $n \geq 1$, $2 \leq q \leq \text{poly}(n)$, and $\mathbf{s}$ be the secret vector.

**Lemma 4.1** (Search to Decision)**.** *Given a procedure that solves search LWE in polynomial time, there exists a polynomial time algorithm that solves decision LWE.*

*Proof.* Our procedure to solve decision LWE is as follows. Take enough samples from our distribution to feed into our procedure for search LWE. The procedure outputs some guess for the secret $\mathbf{s}$. Now, it is easy to verify whether it is the correct secret, by asking for more samples and checking whether the error follows the distribution $\chi$. If it does, then we have access to the $A_{\mathbf{s},\chi}$ distribution. If it looks uniformly random, then we have access to the other. ■

The other direction is slightly more difficult than the first.

**Lemma 4.2** (Decision to Search)**.** *Given a procedure that solves decision LWE in polynomial time, there exists a polynomial time algorithm that solves search LWE.*

*Proof.* We show ow a procedure can find the value of $s_1$ in $\text{poly}(n)$ time. Let $k \in \mathbb{Z}/q\mathbb{Z}$ be our guess for $s_1$. We repeat the following with all values of $k$. Given enough samples from our distribution $A_{\mathbf{s},\chi}$, for each $(\mathbf{a}, b)$, we feed our procedure that solves decision LWE the tuple $(\mathbf{a} + (r, 0, \ldots, 0), b + r \cdot k)$, where $r \in \mathbb{Z}/q\mathbb{Z}$ is chosen uniformly at random.

Clearly, if we guessed correctly and $k = s_1$, then our tuples are exactly in the distribution $A_{\mathbf{s},\chi}$. Otherwise, if we guessed incorrectly, then as long as $q$ is prime, the tuples we output look like the random distribution. ■

## 5. Encryption

We give a simple cryptosystem as a proof of concept here. We parameterize it as follows. We have the integers $n$, $m$, $q$, and a real $\alpha > 0$, where $m$ is the number of samples, and the rest of the variables are as defined as the statement of LWE. We do have to be careful with our choices of parameters to ensure correctness. One such choice is $q$ to be prime between $n^2$ and $2n^2$, $m = 1.1 \cdot n \log q$, and $\alpha = 1/(\sqrt{n} \log^2 n)$.

Our private key is the secret vector $\mathbf{s}$. Our public key consists of $m$ samples $(\mathbf{a_i}, b_i)$from the LWE distribution.

For encryption, we do the following to encrypt a message $M$ with bits $M_1, M_2, \ldots$. Choose a random subset $S$ of $[m]$. The encryption is $(\sum_{i \in S} a_i, \sum_{i \in S} b_i + \lfloor \frac{q}{2} \rfloor M_i)$.

To decrypt a pair $(\mathbf{a}, b)$, we choose based on whether $b - \langle \mathbf{a}, \mathbf{s} \rangle$ is closer to 0 than to $\lfloor \frac{q}{2} \rfloor$.

Note that the public key is of size $n^2$, which is impractical when $n$ is large.

To see that this cryptosystem is correct, one just has to make sure that the sum of error terms is not larger than $q/4$ (otherwise we would decrypt incorrectly). It turns out, with basic statistics, that the parameters we chose lead to this happening with extremely small probability.

## 6. Ring Learning with Errors

One impracticality with using learning with errors for cryptography is the fact that one needs $n$ vectors, each of length $n$, leading to $O(n^2)$ elements in a key. The idea behind RLWE is to use one vector of length $n$ to generate all other vectors, leaving a key size of a substantially smaller $O(n)$ elements.

Here is the formal statement of the RLWE problem, taken from [Reg10]. Let $n$ be a power of two, and let $q$ be a prime satisfying $q \equiv 1 \pmod{2n}$. Define $R_q$ as the quotient ring of polynomials $Z_q[x]/\langle x^n + 1 \rangle$. We are given samples of the form $(\mathbf{a}, \mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \mathbf{e}) \in R_q \times R_q$ where $\mathbf{s} \in R_q$ is a fixed secret, $\mathbf{a} \in R_q$ is chosen uniformly, and $\mathbf{e}$ is an error term chosen independently from some error distribution over $R_q$. The error distribution is typically so every coefficient of the polynomial is independently and identically distributed (a normal distribution in higher dimension).

Lyubashevsky, Peikert, and Regev [LPR13] proved that the RLWE problem has the same hardness as the LWE problem. The reader can read more about RLWE in the given references.

The cryptosystems being proposed that are based on LWE are actually based on RLWE because of the necessary efficiency that it brings.

The applications for LWE in cryptography is extremely broad. One area that is particularly interesting is that it is possible to do fully homomorphic encryption with RLWE, see [BV11]. There are also algorithms for digital signatures that are based on RLWE.

## 7. Ring Learning with Errors Key Exchange

Ding, Xie, and Lin [DXL] have developed a simple key exchange relying on both LWE and RLWE. We give a sketch of their RLWE key exchange algorithm here.

Alice chooses a secret $s_A$ and a public $m \in R_q$. She computes $p_A = ms_A + 2e_A \pmod{q}$, where $e_A$ is chosen according to the error distribution. She sends $p_A$ and $m$.

Bob chooses a secret $s_B$. Bob then computes $K_B = p_A \cdot s_B + 2e'_B \pmod{q}$ and $p_B = m \cdot s_B + 2e_B \pmod{q}$ where $e_B, e'_B$ are chosen according to the error distribution. Finally,

Bob sends $p_B$ and a signal (which is defined precisely in their paper, we will take it as a sort of "hint" $\sigma$ of the value $K_B$) to Alice. He obtains the shared key using a "robust extractor" (also defined in their paper), which extracts a value $E(K_B, \sigma)$ from the computed value $K_B$ and the hint $\sigma$. The goal of this extractor is such that if $x, y$ are close, then the values $E(x, \sigma)$ and $E(y, \sigma)$ are the same.

Alice computes $K_A = s_A p_B + 2e'_A \pmod q$ with some error $e'_A$ and obtains the shared key $E(K_A, \sigma)$.

Since $K_A$ and $K_B$ differ in terms of the error terms, they are close with extremely high probability and thus the shared keys are the same with high probability.

## References


[ADPS16] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange—a new hope. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 327–343, 2016.

[AG11] Sanjeev Arora and Rong Ge. New algorithms for learning in presence of errors. In *International Colloquium on Automata, Languages, and Programming*, pages 403–415. Springer, 2011.

[BKW03] Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM (JACM)*, 50(4):506–519, 2003.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In *Annual cryptology conference*, pages 505–524. Springer, 2011.

[DXL] Jintai Ding, Xiang Xie, and Xiaodong Lin. A simple provably secure key exchange scheme based on the learning with errors problem.

[LPR13] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *Journal of the ACM (JACM)*, 60(6):43, 2013.

[Pei16] Chris Peikert. A decade of lattice cryptography. *Foundations and Trends® in Theoretical Computer Science*, 10(4):283–424, 2016.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009.

[Reg10] Oded Regev. The learning with errors problem (invited survey). In *2010 IEEE 25th Annual Conference on Computational Complexity*, pages 191–204. IEEE, 2010.

[Sho99] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.