EE 793 Cryptology Project
Solving the Learning With Errors (LWE) Problem via Linear Programming

1 Introduction

A **lattice** is defined as the set of all integer combinations

$$\mathcal{L}(b_1, \ldots, b_n) = \left\{ \sum_{i=1}^{n} x_i b_i : x_i \in \mathbb{Z} \text{ for } 1 \leq i \leq n \right\}$$

of n linearly independent vectors $b_1, \ldots, b_n$ in $\mathbb{R}^n$ (see Figure 1). The set of vectors $b_1, \ldots, b_n$ is called a *basis* for the lattice.

A basis can be represented by the matrix

$$B = [b_1 \ \cdots \ b_n] \in \mathbb{R}^{n \times n}$$

having the basis vectors as columns. Using matrix notation, the lattice generated by a matrix $B \in \mathbb{R}^{n \times n}$ can be defined as

$$\mathcal{L}(B) = \{Bx : x \in \mathbb{Z}^n\},$$

where Bx is the usual matrix-vector multiplication.

2 Problem Statement

The Learning With Errors (LWE) problem is fundamental in lattice-based, post-quantum cryptography. Given a known matrix $A \in \mathbb{Z}_q^{m \times n}$, a known vector $b \in \mathbb{Z}_q^m$, and a modulus q, there exists an integer secret $x \in \mathbb{Z}^n$ satisfying

$$A\,x + e \ \equiv \ b \pmod{q},$$

where $e \in \mathbb{Z}^m$ is a small "error" vector. Recovering x is conjectured hard when the dimensions and modulus are large and e is chosen from an appropriate distribution. This report describes a two-stage attack for small instances combining:

1. Lattice reduction (LLL) to obtain a reduced and approximate basis.
2. Mixed-Integer Linear Programming (MILP) to refine to the exact solution by minimizing the ℓ_1-error.

Given:

- $A \in \mathbb{Z}_q^{m \times n}$, a public matrix.
- $b \in \mathbb{Z}_q^m$, a public vector.
- Modulus $q \in \mathbb{Z}^+$.
- Unknown secret $x \in \mathbb{Z}^n$ and small error $e \in \mathbb{Z}^m$.

we have for each $i = 1, \ldots, m$

$$A_i \cdot x + e_i = b_i + q\, k_i$$

for some integer k_i. The goal is to recover x given (A, b, q), leveraging the assumption that $\|e\|$ is small.

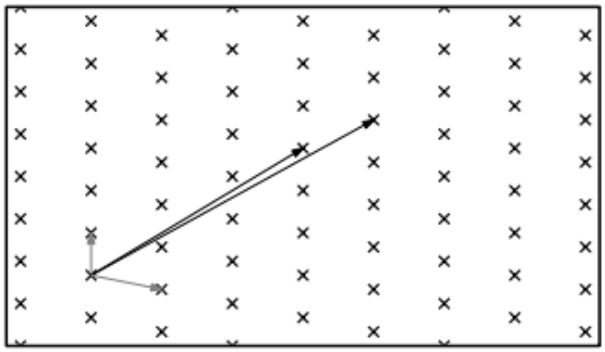

Fig. 1. A lattice in $\mathbb{R}^2$ and two of its bases

3 Phase I: LLL Lattice Reduction

Performs lattice basis reduction (Consists of shorter and more orthogonal vectors) as discussed in Class. The algorithm is mentioned below.

3.1 Embedding

Embed the system into a lattice:

$$B = \begin{pmatrix} q\, I_n & 0 & 0 \\ A & q\, I_m & 0 \\ b^\top & 0 & 1 \end{pmatrix} \in \mathbb{Z}^{(n+m+1)\times(n+m+1)}.$$

A short vector in the lattice generated by B heuristically corresponds to $(x, e, -1)$.

3.2 LLL Reduction

Apply the Lenstra–Lenstra–Lovasz (LLL) algorithm with parameter $\delta \approx 0.75$ to B. The first output basis vector v_{short} is likely to be "near" $(x, e, -1)$. Extract its first n coordinates as an approximate basis vector.

4 Phase II: MILP Refinement

Given an approximate $\tilde{x}$ from the LLL algorithm, we set up a Mixed-Integer Linear Program to minimize the total error.

4.1 Decision Variables

- $x_j \in \mathbb{Z}$, $x_{\min} \le x_j \le x_{\max}$: secret components.
- $k_i \in \mathbb{Z}$.
- $e_i^+, e_i^- \ge 0$: decompose error $e_i = e_i^+ - e_i^-$.
- $t_i \ge 0$: enforces $t_i \ge |e_i|$.

4.2 Constraints

For each $i = 1, \ldots, m$:

$$\sum_{j=1}^{n} A_{ij}\, x_j \;+\; q\, k_i \;-\; b_i \;=\; e_i^+ \;-\; e_i^-,$$

$$t_i \;\ge\; e_i^+, \quad t_i \;\ge\; e_i^-,$$

4.3 Objective

$$\min \sum_{i=1}^{m} t_i.$$

Since $t_i \ge |e_i|$, this minimizes ℓ_1-norm of the error, selecting the secret that yields the smallest total noise.

4.4 Warm-Start

Feed the MILP solver with $\tilde{x}$ from the LLL algorithm to tighten upper bounds on the objective and reduce computation by feeding it the reduced and more orthogonal basis vectors.

5 Combined Algorithm

1. **Input:** (A, b, q), secret bounds $(x_{\min}, x_{\max})$, noise bound E.
2. **LLL Phase:**
   (a) Build embedding matrix B.
   (b) Run LLL; extract $\tilde{x}$.
3. **MILP Phase:**
   (a) Define integer x_j, k_i and continuous $e_i^{\pm}, t_i$.
   (b) Add the linear constraints and objective $\min \sum t_i$.
   (c) Warm-start with $\tilde{x}$.
   (d) Solve via branch-and-bound.

4. **Output:** Optimal x and error $e_i = e_i^+ - e_i^-$.

6 Theoretical Guarantees

- **Exactness:** The MILP exactly encodes $Ax + e = b \mod q$ and ℓ_1-minimization.
- **Uniqueness:** Under LWE, the true secret yields strictly minimal $\|e\|_1$.

7 Analysis and Explanation of the LWE MILP Solver Code

The code can be found here: Colab Notebook for LWE solver

7.1 Overview

This document analyzes a mixed-integer linear programming (MILP) approach to solving the Learning With Errors (LWE) problem, optionally incorporating LLL reduction to improve numerical properties of the lattice defined by the matrix A.

7.2 Core Implementation Excerpts

```
from fpylll import IntegerMatrix, LLL

def lll_reduce_matrix(A):
    m, n = len(A), len(A[0])
    mat = IntegerMatrix(m, n)
    for i in range(m):
        for j in range(n):
            mat[i, j] = A[i][j]
    LLL.reduction(mat)
    A_reduced = [[mat[i, j] for j in range(n)] for i in range(m)]
    return A_reduced
```

Explanation: This function applies LLL reduction using the `fpylll` library to produce a numerically well-conditioned basis of the lattice generated by A.

```
prob = LpProblem("LWE_MILP_LLL", LpMinimize)

x = [LpVariable(f"x_{j}", cat="Integer", lowBound=x_min, upBound=x_max)
for j in range(n)]
k = [LpVariable(f"k_{i}", cat="Integer") for i in range(m)]
e_plus  = [LpVariable(f"e_plus_{i}",  lowBound=0) for i in range(m)]
e_minus = [LpVariable(f"e_minus_{i}", lowBound=0) for i in range(m)]
t       = [LpVariable(f"t_{i}",       lowBound=0) for i in range(m)]
```

```
prob += lpSum(t)
```

Explanation: Here, we define the MILP problem and create decision variables: the secret vector x, slack integers k, decomposed error terms e^+, e^-, and absolute error t. The objective is to minimize the total error $\sum_i t_i$.

```
for i in range(m):
    prob += (
        lpSum(A[i][j] * x[j] for j in range(n)) +
        q * k[i] - b[i] == e_plus[i] - e_minus[i]
    )
    prob += t[i] >= e_plus[i]
    prob += t[i] >= e_minus[i]
    prob += e_plus[i] <= error_bound
    prob += e_minus[i] <= error_bound
```

Explanation: These constraints encode the LWE equation $Ax + qk = b + e$, with $e_i = e_i^+ - e_i^-$ and $|e_i| \leq$ error_bound. The t_i variable represents the absolute error, which is minimized.

```
if prob.status == LpStatusOptimal:
    x_sol = [value(var) for var in x]
    errors = [value(e_plus[i]) - value(e_minus[i]) for i in range(m)]
    total_error = sum(value(t[i]) for i in range(m))
```

Explanation: Upon successful solution, the function extracts the secret vector x, individual errors, and total error from the MILP output.

Analytical Notes

- **LLL Reduction:** The use of LLL helps make the matrix A more orthogonal, which can improve MILP solver stability and convergence, especially when A is ill-conditioned.
- **MILP Formulation:** The absolute error terms t_i allow the problem to be posed as a linear objective, even though LWE is inherently noisy.
- **Modular Equation Handling:** By introducing integer slack variables k_i, the modular equation $Ax \equiv b \mod q$ is lifted to $\mathbb{Z}$, which is more amenable to MILP.
- **Error Decomposition:** Expressing e_i as $e_i^+ - e_i^-$ ensures that the absolute value can be handled linearly, satisfying MILP's requirement for linear constraints.

8 Conclusion

Algorithmic analysis shows that for addition of error terms up to the bounds as mentioned above does not change the secret key extracted key. In other words, the secret key is extracted correctly testing over various error values, given that they are within bounds. Thus, the LLL + MILP algorithm stated above can extract the secret key from the LWE system with the guarantees mentioned in Section 7.

References

[1] O. Regev. On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 84–93, 2005.

[2] O. Regev. Lattice-based Cryptography. In *Advances in Cryptology – EUROCRYPT 2006*, LNCS 4004, pages 131–141, 2006.

[3] D. Micciancio and O. Regev. Lattice-based Cryptography. In *Post-Quantum Cryptography*, Springer, 2009.

[4] V. Lyubashevsky, C. Peikert, and O. Regev. On Ideal Lattices and Learning with Errors over Rings. In *EUROCRYPT 2010*, LNCS 6110, pages 1–23, 2010.

[5] O. Regev. The Learning with Errors Problem. Available at: `https://en.wikipedia.org/wiki/Learning_with_errors`.

[6] D. Micciancio. CSE206A: Lattice Algorithms and Applications, Spring 2007. University of California, San Diego. `https://cseweb.ucsd.edu/classes/sp07/cse206a/`

[7] D. Micciancio. CSE207C: Lattices in Cryptography and Cryptanalysis, Winter 2002. University of California, San Diego. `https://cseweb.ucsd.edu/classes/wi02/cse207c/`