耐量子計算機暗号 ML-KEM の
安全性に関する調査及び評価

安田 雅哉
（立教大学理学部）

2026 年 1 月

第 1 章

調査結果・評価結果の概要（エグゼクティブサマリー）

FIPS 標準化された ML-KEM [64] は，加群格子上の LWE である Module-LWE 問題に基づく KEM で，通常（構造化なし）の LWE 問題に基づく方式に比べて効率的である．

■ML-KEM の構成概要　ML-KEM の基礎環は $R = \mathbb{Z}[X]/(X^n+1)$ $(n = 256)$ で，素数 $q = 3329$ を法とする剰余環 $R_q = R/qR$ を用いる．また，安全性レベルに応じて，3 種類の階数 $k \in \{2, 3, 4\}$ を選択する．ML-KEM では，秘密鍵 $\boldsymbol{s} = (s_1, \ldots, s_k) \in R_q^k$ とノイズ $\boldsymbol{e} = (e_1, \ldots, e_k) \in R_q^k$ の成分多項式 $s_i, e_i \in R_q$ のすべての $\mathbb{Z}_q$ 係数は，中心二項分布 CBD_η $(\eta \in \{2, 3\})$ からサンプルされる．本章では，すべてのベクトルは列ベクトルとする．このとき，すべての成分が R_q 上一様ランダムに選ばれた行列 $\boldsymbol{A} \in R_q^{k \times k}$ に対して，組 $(\boldsymbol{A}, \boldsymbol{t})$ を ML-KEM の公開鍵とする．ただし，$\boldsymbol{t} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e} \in R_q^k$ とする．また，各係数が $\{0, 1\}$ に属する平文 $m \in R_q$ に対し，すべての $\mathbb{Z}_q$ 係数を CBD_η からサンプルした $\boldsymbol{y}, \boldsymbol{e}_1 \in R_q^k$ と $e_2 \in R_q$ を選び，公開鍵 $(\boldsymbol{A}, \boldsymbol{t})$ を用いて，

$$c = (\boldsymbol{u}, v) = \left(\boldsymbol{A}^\top \boldsymbol{y} + \boldsymbol{e}_1, \boldsymbol{t}^\top \boldsymbol{y} + e_2 + \left\lceil \frac{q}{2} \right\rfloor \cdot m\right) \in R_q^k \times R_q \tag{1.1}$$

を m の暗号文とする．さらに，暗号文に対して，$v - \boldsymbol{s}^\top \boldsymbol{u} \in R_q$ の各 $\mathbb{Z}_q$ 係数をノイズ補正することで復号できる．このように構成した公開鍵暗号方式を，藤崎-岡本変換により KEM（ML-KEM）に変換する．ML-KEM では，R_q における乗算を高速化するために，数論変換が多用される．

■ML-KEM の証明可能安全性　ML-KEM の秘密鍵 $\boldsymbol{s} \in R_q^k$ に対して，R_q^k 上で一様にサンプルされた $\boldsymbol{a} \in R_q^k$ と $t = \boldsymbol{a}^\top \boldsymbol{s} + e \in R_q$ の組 $(\boldsymbol{a}, t) \in R_q^k \times R_q$ を Module-LWE サンプルという．ただし，$e \in R_q$ の各 $\mathbb{Z}_q$ 係数は CBD_η からサンプルされる．Module-LWE サンプルの分布が $R_q^k \times R_q$ 上の一様ランダム分布と識別困難な Module-LWE 仮定の下で，ML-KEM の基盤である公開鍵暗号方式は IND-CPA 安全である．したがって，藤崎-岡本変換で得られる ML-KEM は IND-CCA 安全である．その安全性証明は，ランダムオラクルモデル（ROM）においてはタイトである一方，量子ランダムオラクルモデル（QROM）においてはノンタイトである（文献 [13, §4] を参照）．しかし，いくつかの自然な仮定の下では，QROM においてもタイトな安全性帰着がある．

表 1.1　ML-KEM における 3 種類のパラメータとゲートコストによる攻撃計算量の見積もり

ML-KEM パラメータ（k：階数パラメータ）	512（$k=2$）	768（$k=3$）	1024（$k=4$）
攻撃可能な BKZ の最小ブロックサイズ β	413	637	894
攻撃に必要なゲートコスト（ビット）	151.5	215.1	287.3
NIST 標準化の安全性レベル [63]	レベル 1	レベル 3	レベル 5
要求される古典ゲート数（ビット）	143	207	272

■ML-KEM の安全性を支える Module-LWE 問題に対する攻撃計算量　現時点で，ML-KEM の安全性を支える Module-LWE 問題に対する最良の攻撃法は，$\mathbb{Z}_q$ 上の LWE 問題に帰着した上で，BKZ 基底簡約などの $\mathbb{Z}$ 格子上のアルゴリズムを適用するものである．暗号文 (1.1) の形から，攻撃に利用できる Module-LWE サンプル数は最大 $k+1$ 個であり，ML-KEM に対しては primal 攻撃と dual 攻撃が有効となる．表 1.1 に，ML-KEM パラメータと，それらに対する攻撃計算量の見積もりをまとめる．具体的には，ML-KEM パラメータに対し primal 攻撃と BKZ 基底簡約の組み合わせが有効で，攻撃者に有利な観点で，BKZ の progressive 化とシミュレーション [29]，dimension-for-free [32] など最新技術の効果を考慮する．また，BKZ の内部 SVP オラクルで呼び出す篩アルゴリズムの最内部にある繰り返し関数に対して，文献 [6] の解析に基づくゲートコストを表 1.1 に示す（詳細は文献 [13, Table 4] を参照）．表 1.1 から，各 ML-KEM パラメータに対して，攻撃に必要なゲートコストは耐量子計算機暗号の NIST 標準化 [63] の安全性レベル 1, 3, 5 で要求される古典ゲート数を上回り，十分な安全性を持つと考えられる．

篩アルゴリズムの解析の精密化・改良による影響　篩アルゴリズムの多角的な解析の精密化と，将来予想されるアルゴリズム的改良を考慮すると，表 1.1 内のゲートコスト評価は $-16 \sim 14$ 程度変動する可能性がある [13, §5.3, Summary]．最悪の場合，NIST 標準化で要求される古典ゲート数を下回る可能性があるが，これはあくまで攻撃者に最も有利な条件下での評価に過ぎない．実際には，文献 [56, §4.1.1] で指摘されているように，篩アルゴリズムのメモリアクセスのコストを現実的に反映した条件下では，NIST 標準化で要求される古典ゲート数は維持されると考えられる．（文献 [79] によるメモリアクセスのコスト削減は実用的なもので，[13] の予想の範囲内と考えられる．）

最新の dual-sieve 攻撃による影響　文献 [23] で新しい dual-sieve 攻撃が提案され，NIST 標準化で要求される古典ゲート数を下回ると主張している．しかし，その解析は理想的な理論モデルに基づき，オーバーヘッドが隠れている．また，LWE チャレンジで検証されている primal 攻撃に比べ，dual 攻撃の実用性の解析は進んでおらず，文献 [35] の指摘のように，dual 攻撃の成功確率は実際よりかなり高く見積もられている．したがって，文献 [23] の攻撃計算量は実際よりかなり小さく見積もられている可能性が高く，表 1.1 内のゲートコスト評価には影響しないと考えられる．

代数構造を利用した格子アルゴリズムの影響　加群格子上の BKZ 基底簡約は，$\mathbb{Z}$ 格子上のアルゴリズムと同程度の品質の基底を出力するか不明で，実用的な動作のための実装基盤も現時点では整備されていない．また，イデアル格子上の SVP に対する量子アルゴリズムは，Module-LWE への適用には障壁があり，文献 [56, Appendix C] の指摘のように，ML-KEM に対する実用的な攻撃に繋がる可能性は低い．以上から，現時点で，代数構造を利用したアルゴリズムは，代数構造を利用しないアルゴリズムよりも影響が大きいとは言えない．

第 2 章

ML-KEM の構成に関する解説

本章では，FIPS 標準として制定された耐量子計算機暗号の鍵カプセル化メカニズム（KEM）である ML-KEM [64] の構成について解説する．ML-KEM の安全性は Module-LWE 問題の計算困難性に基づく．具体的には，ML-KEM は，$\mathbb{Z}_q$ 上の LWE 問題に基づく Regev [67] の公開鍵暗号方式をひな形とし，それを Module-LWE 問題に一般化した公開鍵暗号方式を藤崎-岡本変換により KEM 変換した暗号方式である．

2.1 LWE 問題と Regev による公開鍵暗号方式

本節では，ML-KEM のひな形である LWE 問題に基づく Regev [67] による公開鍵暗号方式について解説する．そのために，$\mathbb{Z}_q$ 上の LWE 問題から述べる．

2.1.1 $\mathbb{Z}_q$ 上の LWE 問題

LWE（Learning with Errors）問題は機械学習理論から派生した計算問題で，奇素数 q による整数剰余類環 $\mathbb{Z}_q$ 上の秘密ベクトル $\boldsymbol{s} \in \mathbb{Z}_q^n$ に関するランダムな連立線形「近似」方程式が与えられたとき，その秘密ベクトル $\boldsymbol{s}$ を復元する問題である．具体的な数値例として，$n = 4$, $q = 17$ に対して，秘密ベクトル $\boldsymbol{s} = (s_1, s_2, s_3, s_4) \in \mathbb{Z}_{17}^4$ に関する連立線形近似方程式

$$
\begin{cases}
14s_1 + 15s_2 + 5s_3 + 2s_4 \approx 8 \pmod{17} \\
13s_1 + 14s_2 + 14s_3 + 6s_4 \approx 16 \pmod{17} \\
6s_1 + 10s_2 + 13s_3 + s_4 \approx 12 \pmod{17} \\
10s_1 + 4s_2 + 12s_3 + 16s_4 \approx 12 \pmod{17} \\
9s_1 + 5s_2 + 9s_3 + 6s_4 \approx 9 \pmod{17} \\
3s_1 + 6s_2 + 4s_3 + 5s_4 \approx 16 \pmod{17} \\
\qquad\qquad \vdots \qquad\qquad\qquad\quad \vdots \\
6s_1 + 7s_2 + 16s_3 + 2s_4 \approx 3 \pmod{17}
\end{cases}
$$

が与えられたとする（この数値例は文献 [68] から引用）．ただし，各線形方程式の値は近似値であり，その誤差はこの例では ± 1 以内と仮定する．LWE 問題は，この連立線形近似方程式の解

$\boldsymbol{s} \in \mathbb{Z}_q^n$ を求める計算問題である．ちなみに，この数値例では $\boldsymbol{s} = (0, 13, 9, 11) \in \mathbb{Z}_{17}^4$ が解となる．LWE 問題で注意すべきことは，連立線形近似方程式に誤差がない場合は，ガウスの消去法により効率的に解を求めることができる点である．逆に言うと，連立線形近似方程式で与えられる誤差の大きさが，LWE 問題の求解を困難にする．

定式化された $\mathbb{Z}_q$ 上の LWE 問題 [67] は，以下である．

定義 2.1 (LWE 問題)**.** n を正の整数とし，q を奇素数とする．また，χ を $\mathbb{Z}_q$ 上のノイズ分布とする（例えば，χ として平均 0，標準偏差 $\sigma > 0$ の $\mathbb{Z}$ 上の離散ガウス分布 $D_{\mathbb{Z},\sigma}$ をとる）．$\mathbb{Z}_q^n$ 上一様ランダムに選ばれた秘密ベクトル $\boldsymbol{s} \in \mathbb{Z}_q^n$ を固定する．また，各成分が $\mathbb{Z}_q$ 上一様ランダムに選ばれた $\boldsymbol{a} \in \mathbb{Z}_q^n$ とノイズ分布 χ からサンプルされた $e \in \mathbb{Z}_q$ に対して，

$$(\boldsymbol{a}, t) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$$

の組を出力する確率分布を $L_{\boldsymbol{s},\chi}$ とする．ただし，

$$t = \langle \boldsymbol{a}, \boldsymbol{s} \rangle + e \bmod q \in \mathbb{Z}_q$$

とする（2 つのベクトル $\boldsymbol{v}$ と $\boldsymbol{w}$ の内積を $\langle \boldsymbol{v}, \boldsymbol{w} \rangle$ で表す）．このとき，次の 2 つの問題を考える．

- **判定問題**：与えられた複数の組 $(\boldsymbol{a}_i, t_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ $(i = 1, 2, \ldots, m)$ が，LWE における確率分布 $L_{\boldsymbol{s},\chi}$ からサンプルされた元か，$\mathbb{Z}_q^n \times \mathbb{Z}_q$ 上一様ランダムに生成された元かを決定せよ．
- **探索問題**：LWE における確率分布 $L_{\boldsymbol{s},\chi}$ からサンプルされた複数の組 $(\boldsymbol{a}_i, t_i)$ $(i = 1, 2, \ldots, m)$ から秘密ベクトル $\boldsymbol{s} \in \mathbb{Z}_q^n$ を復元せよ．

注意 2.1. 上記の LWE 問題について，探索問題の解である秘密ベクトル $\boldsymbol{s} \in \mathbb{Z}_q^n$ を得ることができれば，明らかに判定問題を解くことができる．逆に，探索問題は判定問題に帰着可能で [67, Lemma 4.2]，判定問題を解くオラクルを用いて探索問題を解くことができる．また，秘密ベクトル $\boldsymbol{s}$ を $\mathbb{Z}_q^n$ 一様ランダムに選んだ場合と，離散ガウス分布から選んだ場合の LWE 問題の計算困難性は等しい（詳細は [55] を参照）．

一般に，上記の 2 つの問題において，LWE における確率分布 $L_{\boldsymbol{s},\chi}$ は任意個の組 $(\boldsymbol{a}, t)$ をサンプルするオラクルとしてみなす．具体的には，ある固定したサンプル数 $m > 0$ に対して，LWE における確率分布 $L_{\boldsymbol{s},\chi}$ からサンプルされた異なる m 個の組

$$\begin{cases} (\boldsymbol{a}_1, t_1), & t_1 = \langle \boldsymbol{a}_1, \boldsymbol{s} \rangle + e_1 \bmod q \\ (\boldsymbol{a}_2, t_2), & t_2 = \langle \boldsymbol{a}_2, \boldsymbol{s} \rangle + e_2 \bmod q \\ \quad \vdots & \\ (\boldsymbol{a}_m, t_m), & t_m = \langle \boldsymbol{a}_m, \boldsymbol{s} \rangle + e_m \bmod q \end{cases}$$

から, LWE 問題を解くことを考える．特に, 求解に要する計算時間が最も短くなるような m を攻撃者が選べることを想定する．第 i 行ベクトルを $\boldsymbol{a}_i$ とする $m \times n$ 行列を $\boldsymbol{A}$ とし, $\boldsymbol{t} = (t_1, t_2, \ldots, t_m)$ とおく．このとき, 上記の m 個の LWE サンプルの組は

$$\big(\boldsymbol{A}, \boldsymbol{t}\big) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m \tag{2.1}$$

と簡潔に表せて，関係式

$$\boldsymbol{t} \equiv \boldsymbol{s}\boldsymbol{A}^{\top} + \boldsymbol{e} \pmod{q} \tag{2.2}$$

を満たす．ただし，$\boldsymbol{e} = (e_1, e_2, \ldots, e_m) \in \mathbb{Z}_q^m$ をノイズベクトルとする（各 e_i は χ からサンプルされた元であることに注意）．

2.1.2 Regev による公開鍵暗号方式

Regev による公開鍵方式 [67] の構成には, 以下の 4 つのパラメータが必要である．

- n: LWE 次元
- m: LWE サンプルの個数（$m \geq 1.1 \cdot n \log q$ となる最小の整数を選ぶ）
- q: 剰余パラメータ（$n^2 \leq q \leq 2n^2$ を満たす素数を選ぶ）
- $\alpha > 0$: 離散ガウス分布の標準偏差を定めるノイズパラメータ（$\alpha = 1/(\sqrt{n} \cdot \log^2 n)$）

以下に，具体的な公開鍵暗号方式の構成を示す．

■鍵生成　一様ランダムに秘密鍵ベクトル $\boldsymbol{s} \leftarrow \mathbb{Z}_q^n$ を選ぶ．次に，平均 0, 標準偏差 $\sigma = \alpha q$ の $\mathbb{Z}$ 上の離散ガウス分布 $\chi = D_{\mathbb{Z},\sigma}$ を用いて，秘密鍵ベクトル $\boldsymbol{s}$ による LWE 分布 $L_{\boldsymbol{s},\chi}$ から生成した m 個のサンプル

$$\{(\boldsymbol{a}_i, t_i)\}_{i=1}^{m}, \quad t_i = \langle \boldsymbol{a}_i, \boldsymbol{s} \rangle + e_i \bmod q \in \mathbb{Z}_q \quad (i = 1, 2, \ldots, m) \tag{2.3}$$

を公開鍵とする．ただし，各 e_i は χ からサンプリングされた元とする．

■暗号化　集合 $\{1, 2, \ldots, m\}$ の中から，一様ランダムに選んだ部分集合を S とする．このとき，上記の公開鍵を用いて，平文 $\mu \in \{0, 1\}$ の暗号文を次で定める．

$$c = (\boldsymbol{u}, v) = \left(\sum_{i \in S} \boldsymbol{a}_i,\ \mu \cdot \left\lfloor \frac{q}{2} \right\rceil + \sum_{i \in S} t_i \right) \in \mathbb{Z}_q^n \times \mathbb{Z}_q \tag{2.4}$$

■復号　式 (2.4) の形の暗号文 $c = \big(\boldsymbol{u}, v\big) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ に対し，秘密鍵ベクトル $\boldsymbol{s}$ を用いて

$$\big[v - \langle \boldsymbol{u}, \boldsymbol{s} \rangle \big]_q$$

を計算し, その値が十分 0 に近い場合は 0 を出力し, それ以外の場合は 1 を出力する．ただし, $[z]_q$ は元 $z \in \mathbb{Z}_q$ を $\big[-\frac{q}{2}, \frac{q}{2}\big)$ に収めた値とする．具体的には，法 q による整数 z の値が $0 \leq z < \frac{q}{2}$ であれば $[z]_q = z$ とし，$\frac{q}{2} \leq z < q$ であれば $[z]_q = z - q$ と定める．

復号の正当性

復号について，式 (2.4) の暗号文 $c = (\boldsymbol{u}, v)$ に対して，$\sigma \ll q$ であれば

$$v - \langle \boldsymbol{u}, \boldsymbol{s} \rangle = \sum_{i \in S} e_i + \mu \cdot \left\lfloor \frac{q}{2} \right\rceil \approx \mu \cdot \left\lfloor \frac{q}{2} \right\rceil \in \mathbb{Z}_q$$

が成り立つ．これより，$\mu = 0$ の場合は $[v - \langle \boldsymbol{u}, \boldsymbol{s} \rangle]_q \approx 0$, $\mu = 1$ の場合は $[v - \langle \boldsymbol{u}, \boldsymbol{s} \rangle]_q \approx \pm \frac{q}{2}$ が成り立つ．具体的には，

$$\left| \sum_{i \in S} e_i \right| \lesssim \sigma m < \frac{q}{4} \Longleftrightarrow 4\sigma m < q$$

であれば，高い確率で復号に成功する．

注意 2.2. 上記の暗号方式の構成において，式 (2.3) の公開鍵を，式 (2.1) のように $(\boldsymbol{A}, \boldsymbol{t})$ と行列表示する．ただし，公開鍵 $(\boldsymbol{A}, \boldsymbol{t})$ は LWE 関係式 (2.2) を満たす．このとき，式 (2.4) の形の暗号文は

$$c = (\boldsymbol{u}, v) = \left(\boldsymbol{y}\boldsymbol{A},\ \mu \cdot \left\lfloor \frac{q}{2} \right\rceil + \langle \boldsymbol{y}, \boldsymbol{t} \rangle \right)$$

と表すことができる．ただし，$\{1, 2, \ldots, m\}$ の部分集合 S に対して，$\boldsymbol{y} = (y_1, \ldots, y_m)$ の各成分 $y_i \in \{0, 1\}$ は

$$y_i = \begin{cases} 1 & (i \in S) \\ 0 & (i \notin S) \end{cases}$$

と定める．

2.2 ML-KEM の構成

ML-KEM は CRYSTALS-Kyber [13] に基づく加群格子上の KEM 方式で，その安全性は加群格子上の LWE 問題（Module-LWE 問題）の計算量困難性に基づく．具体的には，ML-KEM では，2 のべき数 $n = 2^8 = 256$ に対して，

$$R \coloneqq \mathbb{Z}[X]/(X^n + 1) \tag{2.5}$$

を基本環とし，素数 $q = 3329$ に対して

$$R_q \coloneqq R/qR = \mathbb{Z}_q[X]/(X^n + 1) \tag{2.6}$$

を R の剰余環とする．環 R_q の任意の元は $\mathbb{Z}_q$ を係数とする $n - 1$ 以下の次数の多項式

$$f(X) = f_0 + f_1 X + \cdots + f_{n-1} X^{n-1} \quad (f_i \in \mathbb{Z}_q)$$

と表せ，その係数ベクトル

$$\boldsymbol{f} = (f_0, f_1, \ldots, f_{n-1}) \in \mathbb{Z}_q^n$$

を対応させることで，$\mathbb{Z}_q$ 加群として R_q は $\mathbb{Z}_q^n$ と同型である．ML-KEM は，3 種類の（R_q 上の自由加群としての）階数パラメータ $k \in \{2, 3, 4\}$ に対し，$\mathbb{Z}_q$ 加群 $R_q^k \simeq (\mathbb{Z}_q^n)^k$ 上の LWE 問題を安全性の根拠とした KEM である．以下で，定式化した Module-LWE 問題を述べておく [21, 51]（$\mathbb{Z}_q$ 上の LWE 問題への帰着については，後述の 4.1 節を参照）．

定義 2.2 (Module-LWE 問題)**.** 秘密の元 $\boldsymbol{s}(X) = \big(s_1(X), \ldots, s_k(X)\big) \in R_q^k$ を固定する．また，一様ランダムに選ばれた $\boldsymbol{a}(X) = \big(a_1(X), \ldots, a_k(X)\big) \in R_q^k$ と R_q 上のノイズ分布 χ からサンプルされた $e(X) \in R_q$ に対して，

$$\begin{gathered} \big(\boldsymbol{a}(X), t(X)\big) \in R_q^k \times R_q, \\ t(X) = \big\langle \boldsymbol{a}(X), \boldsymbol{s}(X) \big\rangle + e(X) = \sum_{i=1}^{k} a_i(X) s_i(X) + e(X) \end{gathered} \tag{2.7}$$

の組を出力する確率分布を $L_{\boldsymbol{s}(X),\chi}$ とする（ML-KEM では，χ として中心二項分布サンプリングをとる）．ただし，$\big\langle \boldsymbol{a}(X), \boldsymbol{s}(X) \big\rangle \in R_q$ は，R_q を成分とする長さ k の 2 つのベクトル $\boldsymbol{a}(X), \boldsymbol{s}(X) \in R_q^k$ の内積とする．このとき，次の 2 つの問題を考える．ただし，サンプル数 m は，攻撃者を有利とする観点から適当に選べると仮定することが多い．

- **判定問題**：与えられた複数の組 $\big(\boldsymbol{a}_j(X), t_j(X)\big) \in R_q^k \times R_q$ $(j = 1, 2, \ldots, m)$ が，Module-LWE における確率分布 $L_{\boldsymbol{s}(X),\chi}$ からサンプルされた元か，$R_q^k \times R_q$ 上一様ランダムに生成された元かを決定せよ．
- **探索問題**：Module-LWE における確率分布 $L_{\boldsymbol{s}(X),\chi}$ からサンプルされた複数の組 $\big(\boldsymbol{a}_j(X), t_j(X)\big)$ $(j = 1, 2, \ldots, m)$ から秘密の元 $\boldsymbol{s}(X) \in R_q^k$ を復元せよ．

注意 2.3. 円分体 $K = R \otimes_{\mathbb{Z}} \mathbb{Q}$ のイデアル R の双対を $R^\vee$ とし，$R_q^\vee = R^\vee / qR^\vee$ とする．文献 [51] では，Module-LWE の秘密 $\boldsymbol{s}(X)$ は $(R_q^\vee)^k$ 上の一様サンプリングから選ばれる．ただし，式 (2.5) の形の R に対しては，$R^\vee = \frac{1}{n} R$ より，単純なスケーリングにより，秘密 $\boldsymbol{s}$ は $(R_q)^k$ 上の一様サンプリングから選ばれるとしてよい．通常の LWE 問題と同様で，$\boldsymbol{s}$ が $(R_q)^k$ 上一様ランダムに選ばれた場合と，$\boldsymbol{s}$ の成分多項式 $s_i(X)$ の $\mathbb{Z}_q$ 係数が $[-\eta, \eta]$ $(1 \leq \eta \ll q)$ 上一様ランダムに選ばれた場合の Module-LWE 問題の困難性は等しい [20]．ML-KEM では，$\boldsymbol{s}$ の各成分多項式 $s_i(X) \in R_q$ の $\mathbb{Z}_q$ 係数は中心二項分布からサンプリングされるが，この場合の Module-LWE 問題が $(R_q)^k$ 上一様ランダムに選ばれた場合と同程度の困難性をもつかどうかは証明されていない．さらに，注意 2.1 と同じように，Module-LWE 問題においても，探索問題が解ければ判定問題を解くことができる．また，探索問題は判定問題に多項式時間帰着可能なので，判定問題と探索問題は多項式時間帰着の意味で等価である（詳細は文献 [51, 57] を参照）．

ML-KEM では，R_q における乗算を高速化するために，Number-Theoretic Transform（NTT）とよばれる数論変換を利用する．以降では，ML-KEM の最も基本となる構成要素である NTT を説明したのちに，ML-KEM の構成について説明する．

2.2.1 数論変換：Number-Theoretic Transform (NTT)

NTT は，環 R_q の元 $f(X)$ を R_q と同型な環 T_q の元 $\widehat{f}$ に写し，T_q における乗算を利用して効率的に R_q の 2 つの元の乗算を行う手法である．これは複素数体 $\mathbb{C}$ 上の高速フーリエ変換による多項式乗算と同じアイデアで，NTT はその $\mathbb{Z}_q$ 上版とみなせる．上述したように，ML-KEM では

2 のべき数 $n = 2^8 = 256$ と素数 $q = 3329$ で定まる剰余環 R_q を用いる（ML-KEM の暗号パラメータについては，後述の 2.2.3 節を参照）．これらの暗号パラメータの組

$$(n, q) = \big(256, 3329\big)$$

において，$\mathbb{Z}_q^* \coloneqq \mathbb{Z}_q \setminus \{0\}$ は位数 $q - 1 = 3328 = 2^8 \cdot 13$ の巡回群で，$\mathbb{Z}_q^*$ は位数 $2^8 = 256 = n$ の巡回部分群 $\langle\zeta\rangle$ を唯一つ含む．具体的には，$\mathbb{Z}_q$ において

$$\zeta \coloneqq 17 \bmod q \in \mathbb{Z}_q$$

が 1 の原始 n 乗根で，ζ の奇数べきによる集合

$$\big\{\zeta, \zeta^3, \zeta^5, \ldots, \zeta^{n-1}\big\}$$

が $\mathbb{Z}_q$ に含まれる 1 の原始 n 乗根全体の集合である．ここで，$N = \frac{n}{2} = 128$ とおくと，各 $i = 0, 1, \ldots, N-1$ に対して，

$$\zeta^{(2i+1)N} \equiv -1 \pmod{q}$$

が成り立つ．ゆえに，多項式環 $\mathbb{Z}_q[X]$ において，$X^n + 1$ は次のように N 個の 2 次式の積に分解できる．

$$X^n + 1 = \prod_{i=0}^{N-1} \left(X^2 - \zeta^{2i+1}\right) = \prod_{i=0}^{N-1} \left(X^2 - \zeta^{2\mathsf{BitRev}_7(i)+1}\right) \in \mathbb{Z}_q[X]$$

ただし，$\mathsf{BitRev}_7(i)$ は符号なし 7 ビット整数 i のビット逆順整数を表し，実装上の都合のため ML-KEM ではこの順序を利用する．以下では，数論変換の原理を説明するために，$i = 0, 1, \cdots, N-1$ の単純な順序を用いる．上記の $X^n + 1$ の分解により，次の（$\mathbb{Z}_q$ 加群としての）同型を得る．

$$R_q = \mathbb{Z}_q[X]/(X^n + 1) \simeq \bigoplus_{i=0}^{N-1} \mathbb{Z}_q[X]/\left(X^2 - \zeta^{2i+1}\right) =: T_q$$

具体的には，この同型は

$$\begin{aligned} \mathsf{NTT} : R_q &\longrightarrow T_q, \\ f(X) &\longmapsto \widehat{f} \coloneqq \big(f \bmod (X^2 - \zeta^{2i+1})\big)_{i=0}^{N-1} \end{aligned} \tag{2.8}$$

で定まる．ここで，T_q を **NTT 空間**（NTT domain），$\widehat{f} = \mathsf{NTT}\big(f\big) \in T_q$ を $f(X) \in R_q$ の **NTT 表現**（NTT representation）とよぶ．

NTT 表現について

R_q の元 $f(X) = f_0 + f_1 X + \cdots + f_{n-1} X^{n-1}$ の偶数（even）と奇数（odd）の次数に関する多項式をそれぞれ

$$\begin{cases} f_e(Y) \coloneqq f_0 + f_2 Y + f_4 Y^2 + \cdots + f_{2N-2} Y^{N-1}, \\ f_o(Y) \coloneqq f_1 + f_3 Y + f_5 Y^2 + \cdots + f_{2N-1} Y^{N-1} \end{cases}$$

とおく（$N = \frac{n}{2}$ に注意）．この構成から，明らかに

$$f(X) = f_e(X^2) + f_o(X^2)X \tag{2.9}$$

が成り立つ．ここで，各 $i = 0, 1, \ldots, N-1$ に対して，

$$\begin{cases} \widehat{f}_{2i} \coloneqq f_e(\zeta^{2i+1}) = \displaystyle\sum_{j=0}^{N-1} f_{2j}\zeta^{(2i+1)j}, \\ \widehat{f}_{2i+1} \coloneqq f_o(\zeta^{2i+1}) = \displaystyle\sum_{j=0}^{N-1} f_{2j+1}\zeta^{(2i+1)j} \end{cases} \tag{2.10}$$

とおくと，式 (2.9) より

$$f(X) \equiv \widehat{f}_{2i} + \widehat{f}_{2i+1}X \pmod{\left(X^2 - \zeta^{2i+1}\right)} \tag{2.11}$$

が成り立つ（X^2 に ζ^{2i+1} を代入したと考えればよい）．これより，$f(X) \in R_q$ の NTT 表現は

$$\widehat{f} = \left(\widehat{f}_{2i} + \widehat{f}_{2i+1}X\right)_{i=0}^{N-1} \in T_q$$

とかける（式 (2.8) を参照）．

NTT 表現の行列表示

$\mathbb{Z}_q$ の元を成分とする $N \times N$ 行列を

$$\boldsymbol{B} = A(\zeta) \coloneqq \begin{pmatrix} 1 & \zeta & \zeta^2 & \cdots & \zeta^{N-1} \\ 1 & \zeta^3 & \zeta^6 & \cdots & \zeta^{3(N-1)} \\ 1 & \zeta^5 & \zeta^{10} & \cdots & \zeta^{5(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \zeta^{2N-1} & \zeta^{2(2N-1)} & \cdots & \zeta^{(N-1)(2N-1)} \end{pmatrix} \in \mathbb{Z}_q^{N \times N}$$

とおく．R_q の元 $f(X) = f_0 + f_1X + \cdots + f_{n-1}X^{n-1}$ の偶数と奇数の次数に関するそれぞれの係数ベクトル $(f_0, f_2, \ldots, f_{2N-2}), (f_1, f_3, \ldots, f_{2N-1}) \in \mathbb{Z}_q^N$ に対して，式 (2.10) より

$$\begin{pmatrix} \widehat{f}_0 \\ \widehat{f}_2 \\ \widehat{f}_4 \\ \vdots \\ \widehat{f}_{2N-2} \end{pmatrix} = \begin{pmatrix} f_e(1) \\ f_e(\zeta^3) \\ f_e(\zeta^5) \\ \vdots \\ f_e(\zeta^{2N-1}) \end{pmatrix} = \boldsymbol{B} \begin{pmatrix} f_0 \\ f_2 \\ f_4 \\ \vdots \\ f_{2N-2} \end{pmatrix},$$

$$\begin{pmatrix} \widehat{f}_1 \\ \widehat{f}_3 \\ \widehat{f}_5 \\ \vdots \\ \widehat{f}_{2N-1} \end{pmatrix} = \begin{pmatrix} f_o(1) \\ f_o(\zeta^3) \\ f_o(\zeta^5) \\ \vdots \\ f_o(\zeta^{2N-1}) \end{pmatrix} = \boldsymbol{B} \begin{pmatrix} f_1 \\ f_3 \\ f_5 \\ \vdots \\ f_{2N-1} \end{pmatrix}$$

が成り立つ．つまり，$f \in R_q$ の偶数と奇数の次数の係数ベクトルはそれぞれ行列 $\boldsymbol{B}$ による線形変換（つまり，離散フーリエ変換）で $\widehat{f} \in T_q$ の偶数と奇数の添え字番号のベクトルに写る．行列 $\boldsymbol{B}$ の逆行列は

$$\boldsymbol{C} = \frac{1}{N} A(\zeta^{-1}) \in \mathbb{Z}_q^{N \times N}$$

で与えられるので，式 (2.8) の NTT 写像の逆写像 NTT^{-1} は，行列 $\boldsymbol{C}$ を用いて同様に計算できる（つまり，逆離散フーリエ変換から計算可能．具体的な NTT の計算アルゴリズムは，FIPS 仕様書 [64, Algorithms 9, 10] を参照）．

NTT 空間における乗算

R_q の 2 つの元 $f(X), g(X)$ に対して，その積を

$$h(X) = f(X) \cdot g(X) \in R_q$$

とおく．$h(X)$ の NTT 表現 $\widehat{h} \in T_q$ について，式 (2.11) から，各 $i = 0, 1, \ldots, N-1$ に対して

$$\begin{aligned} \widehat{h}_{2i} + \widehat{h}_{2i+1} X &\equiv h(X) = f(X) \cdot g(X) \\ &\equiv \left(\widehat{f}_{2i} + \widehat{f}_{2i+1} X\right) \left(\widehat{g}_{2i} + \widehat{g}_{2i+1} X\right) \pmod{\left(X^2 - \zeta^{2i+1}\right)} \end{aligned}$$

が成り立つ．ここで，2 つの NTT 表現

$$\widehat{f} = \left(\widehat{f}_{2i} + \widehat{f}_{2i+1} X\right)_{i=0}^{N-1}, \quad \widehat{g} = (\widehat{g}_{2i} + \widehat{g}_{2i+1} X)_{i=0}^{N-1} \in T_q$$

の積を

$$\begin{aligned} \widehat{f} \circ \widehat{g} &\coloneqq \left(\left(\widehat{f}_{2i} + \widehat{f}_{2i+1} X\right) \cdot \left(\widehat{g}_{2i} + \widehat{g}_{2i+1} X\right) \bmod \left(X^2 - \zeta^{2i+1}\right)\right)_{i=0}^{N-1} \\ &= \left(\widehat{f}_{2i} \widehat{g}_{2i} + \widehat{f}_{2i+1} \widehat{g}_{2i+1} \zeta^{2i+1} + \left(\widehat{f}_{2i} \widehat{g}_{2i+1} + \widehat{f}_{2i+1} \widehat{g}_{2i}\right) X\right)_{i=0}^{N-1} \in T_q \end{aligned} \tag{2.12}$$

と定める（法 $(X^2 - \zeta^{2i+1})$ において，$X^2 = \zeta^{2i+1}$ であることに注意）．このとき，

$$\begin{aligned} &\mathsf{NTT}(f \cdot g) = \mathsf{NTT}(f) \circ \mathsf{NTT}(g) \\ \iff\ &f(X) \cdot g(X) = \mathsf{NTT}^{-1}\left(\widehat{f} \circ \widehat{g}\right) \in R_q \end{aligned}$$

が成り立つ（つまり，式 (2.8) の NTT 写像は環の同型写像である）．特に，NTT 空間 T_q における乗算は，成分ごとの演算であるため，（R_q における乗算に比べて）効率的に計算可能である．具体的には，R_q における乗算には $O(n^2)$ 回の $\mathbb{Z}_q$ 上の乗算が必要であるのに対し，式 (2.12) から，NTT 空間における乗算には $4n = O(n)$ 回だけの $\mathbb{Z}_q$ 上の乗算を要する．また，式 (2.8) の NTT 写像の計算は高速数論変換を用いて $O(n \log n)$ で可能であるため，NTT 変換の計算時間を含めても R_q での乗算よりも高速となる．

2.2.2 ML-KEM の基本構成と処理概要

KEM は公開チャネル上で二者が安全に秘密情報を共有するためのアルゴリズム群である．安全に共有された秘密情報は共通鍵暗号の鍵生成の乱数シードなどに用いられ，暗号や認証などの安全なやり取りの中で重要な役割を果たす．（R_q 上の自由加群としての）階数 k の R_q 加群 $R_q^k \simeq (\mathbb{Z}_q^n)^k$ 上の LWE 問題に基づく ML-KEM は，次の 2 つのステップで構成される．

- 1 つ目は，Module-LWE 問題の計算困難性に基づく公開鍵暗号（K-PKE）を構成する．
- 2 つ目は，K-PKE を藤崎-岡本変換により KEM（ML-KEM）に変換する．

藤崎-岡本変換の性質により，公開鍵暗号方式から構成される KEM はより一般的な攻撃モデルにおいて安全であり，IND-CCA2 安全性を満たす（詳しくは次章を参照）．本項では，FIPS 文書 [64] に合わせて，すべてのベクトルは列ベクトルとする．

K-PKE の処理概要

ここでは，K-PKE の処理概要とその原理が分かるように，簡略化した形で各アルゴリズムの処理を説明する．特に，処理の高速化のために，NTT 変換を適宜利用する．

■K-PKE 鍵生成アルゴリズム Algorithm 1 に，鍵生成アルゴリズム（[64, Algorithm 13], K-PKE.KeyGen）の主な処理をまとめる．具体的には，乱数 d を入力として，暗号鍵 $\mathtt{ek}_{\mathtt{PKE}}$ と復号鍵 $\mathtt{dk}_{\mathtt{PKE}}$ を出力する．ステップ 2 において，NTT 表現の公開鍵行列の各成分 $\widehat{\boldsymbol{A}}[i,j]$ は，入力する乱数から擬似ランダムな T_q の元を出力する SampleNTT 関数 [64, Algorithm 7] を用いて生成する（具体的には，$\widehat{\boldsymbol{A}}[i,j] \leftarrow \mathsf{SampleNTT}(\rho \| i \| j)$ により生成）．ステップ 3, 4 において，各多項式 $\boldsymbol{s}[i]$ または $\boldsymbol{e}[i]$ のすべての十分小さい $\mathbb{Z}_q$ 係数は，SamplePolyCBD 関数 [64, Algorithm 8] を用いて生成する．具体的には，$\eta \in \{2,3\}$ に対する $\mathbb{Z}_q$ 上の中心二項分布 CBD_η（Centered Binomial Distribution）を

(i) $(x_1, \ldots, x_\eta, y_1, \ldots, y_\eta) \in \{0,1\}^{2\eta}$ を一様ランダムにサンプルする
(ii) $\sum_{i=1}^{\eta}(x_i - y_i) \bmod q \in \mathbb{Z}_q$ を出力

と定め，$\boldsymbol{s}[i]$ と $\boldsymbol{e}[i]$ の各 $\mathbb{Z}_q$ 係数は CBD_η からサンプルする．ただし，CBD_η の乱数シードとして，ステップ 1 で生成した σ を用いる．ステップ 8 において，[64] では $\left(\widehat{\boldsymbol{t}}, \rho\right)$ と $\widehat{\boldsymbol{s}}$ をそれぞれ符号化関数 ByteEncode [64, Algorithm 5] で符号化したものを暗号鍵 $\mathtt{ek}_{\mathtt{PKE}}$ と復号鍵 $\mathtt{dk}_{\mathtt{PKE}}$ とする．この鍵生成アルゴリズムにおいて，ρ から NTT 表現の公開鍵行列 $\widehat{\boldsymbol{A}}$ が復元可能なので, 暗号鍵 $\mathtt{ek}_{\mathtt{PKE}}$ は NTT 表現の Module-LWE インスタンスの組

$$\left(\widehat{\boldsymbol{A}}, \widehat{\boldsymbol{t}}\right)$$

に対応する．特に，それらの NTT 逆変換を

$$\boldsymbol{t} = \mathsf{NTT}^{-1}\left(\widehat{\boldsymbol{t}}\right) \in R_q^k, \quad \boldsymbol{A} = \mathsf{NTT}^{-1}\left(\widehat{\boldsymbol{A}}\right) \in (R_q)^{k \times k}$$

Algorithm 1 K-PKE.KeyGen：K-PKE 鍵生成アルゴリズム（[64, Algorithm 13] の簡略版）

入力： 乱数 d

出力： 暗号鍵 $\mathsf{ek}_{\mathsf{PKE}}$ と復号鍵 $\mathsf{dk}_{\mathsf{PKE}}$

1: $(\rho, \sigma) \leftarrow \mathsf{G}(d \| k)$ ▷ ハッシュ関数 G を用いて擬似ランダムな乱数の組 (ρ, σ) を生成

2: $\widehat{\boldsymbol{A}} = \left(\widehat{\boldsymbol{A}}[i,j]\right)_{i,j=0}^{k-1} \in (T_q)^{k \times k}$ ▷ 乱数 ρ から NTT 表現の公開鍵行列を生成

3: $\boldsymbol{s} = \left(\boldsymbol{s}[i]\right)_{i=0}^{k-1} \in R_q^k$
▷ 各 $\boldsymbol{s}[i] \in R_q$ のすべての $\mathbb{Z}_q$ 係数は中心二項分布 CBD_η からサンプル（十分小さい）

4: $\boldsymbol{e} = \left(\boldsymbol{e}[i]\right)_{i=0}^{k-1} \in R_q^k$
▷ 各 $\boldsymbol{e}[i] \in R_q$ のすべての $\mathbb{Z}_q$ 係数は CBD_η からサンプル（十分小さい）

5: $\widehat{\boldsymbol{s}} = \left(\mathsf{NTT}(\boldsymbol{s}[i])\right)_{i=0}^{k-1} \in T_q^k$ ▷ 各 $\boldsymbol{s}[i]$ を NTT 変換

6: $\widehat{\boldsymbol{e}} = \left(\mathsf{NTT}(\boldsymbol{e}[i])\right)_{i=0}^{k-1} \in T_q^k$ ▷ 各 $\boldsymbol{e}[i]$ を NTT 変換

7: $\widehat{\boldsymbol{t}} = \widehat{\boldsymbol{A}} \circ \widehat{\boldsymbol{s}} + \widehat{\boldsymbol{e}} = \left(\sum_{j=0}^{k-1} \widehat{\boldsymbol{A}}[i,j] \circ \widehat{\boldsymbol{s}}[j] + \widehat{\boldsymbol{e}}[i]\right)_{i=0}^{k-1} \in T_q^k$
▷ NTT 空間上で，R_q^k 上の LWE 関係式 $\boldsymbol{t} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e}$ を生成（式 (2.13) を参照）

8: $\mathsf{ek}_{\mathsf{PKE}} = \left(\widehat{\boldsymbol{t}}, \rho\right)$, $\mathsf{dk}_{\mathsf{PKE}} = \widehat{\boldsymbol{s}}$
▷ $\widehat{\boldsymbol{A}}$ は ρ から復元できるので，$\mathsf{ek}_{\mathsf{PKE}}$ は Module-LWE インスタンスの組 $\left(\widehat{\boldsymbol{A}}, \widehat{\boldsymbol{t}}\right)$ に対応

9: **return** $\left(\mathsf{ek}_{\mathsf{PKE}}, \mathsf{dk}_{\mathsf{PKE}}\right)$

とおく．ただし，ベクトルまたは行列に対する NTT^{-1} は，各成分に対する NTT 逆変換とする．このとき，R_q^k 上の LWE 関係式

$$\begin{aligned} & \boldsymbol{t} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e} \in R_q^k \\ \iff & \boldsymbol{t}[i] = \sum_{j=0}^{k-1} \boldsymbol{A}[i,j] \cdot \boldsymbol{s}[j] + \boldsymbol{e}[i] \in R_q \quad (i = 0, 1, \ldots, k-1) \end{aligned} \tag{2.13}$$

が成り立つ（式 (2.7) を参照）．ただし，$\boldsymbol{t}[i], \boldsymbol{e}[i], \boldsymbol{A}[i,j], \boldsymbol{s}[j]$ はそれぞれ R_q の元で，$\mathbb{Z}_q$ 係数の $n-1$ 次以下の多項式で表されることに注意する．一方，復号鍵 $\mathsf{dk}_{\mathsf{PKE}}$ は NTT 表現の LWE の秘密 $\widehat{\boldsymbol{s}}$ であるので，暗号鍵から復号鍵を見つけるのは $T_q^k \simeq R_q^k$ 上の探索 LWE 問題（つまり，探索 Module-LWE 問題）である．特に，適切な暗号パラメータ（後述の 2.2.3 節を参照）を利用した場合，その Module-LWE 問題を解くのは計算量的に非常に困難である．また，鍵生成アルゴリズムにおいて，NTT 空間上で公開鍵行列 $\widehat{\boldsymbol{A}}$ を直接生成すると共に，ステップ 7 で NTT 空間上で Module-LWE 関係式を生成することで，計算の高速化を図る．

■K-PKE 暗号化アルゴリズム Algorithm 2 に，暗号化アルゴリズム（[64, Algorithm 14], K-PKE.Encrypt）の主な処理をまとめる．具体的には，暗号化鍵 $\mathsf{ek}_{\mathsf{PKE}}$，平文 m と乱数 r を入力し，暗号文 c を出力する．ステップ 2, 3, 4 において，r をシードとした擬似乱数を引数とした SamplePolyCBD 関数で，すべての $\mathbb{Z}_q$ 係数が十分小さい多項式を生成する．ステップ 7 では，バイト列で表現された平文 m を ByteDecode 関数 [64, Algorithm 6] でビット列

Algorithm 2 K-PKE.Encrypt：K-PKE 暗号化アルゴリズム（[64, Algorithm 14] の簡略版）

入力： 暗号化鍵 $\mathtt{ek}_{\mathtt{PKE}} = \left(\widehat{\boldsymbol{t}}, \rho\right)$，平文 m と乱数 r

出力： 暗号文 c

1: ρ から NTT 表現の公開鍵行列 $\widehat{\boldsymbol{A}} \in (T_q)^{k \times k}$ を復元

2: $\boldsymbol{y} = \left(\boldsymbol{y}[i]\right)_{i=0}^{k-1} \in R_q^k$

▷ 各 $\boldsymbol{y}[i] \in R_q$ のすべての $\mathbb{Z}_q$ 係数は中心二項分布 CBD_η からサンプル（十分小さい）

3: $\boldsymbol{e}_1 = (\boldsymbol{e}_1[i])_{i=0}^{k-1} \in R_q^k$

▷ 各 $\boldsymbol{e}[i] \in R_q$ のすべての $\mathbb{Z}_q$ 係数は CBD_η からサンプルする（十分小さい）

4: $e_2 \in R_q$ ▷ すべての $\mathbb{Z}_q$ 係数は CBD_η からサンプルする（十分小さい）

5: $\widehat{\boldsymbol{y}} = (\mathsf{NTT}\left(\boldsymbol{y}[i]\right))_{i=0}^{k-1} \in T_q^k$ ▷ 各 $\boldsymbol{y}[i]$ を NTT 変換

6: $\boldsymbol{u} = \mathsf{NTT}^{-1}\left(\widehat{\boldsymbol{A}}^\top \circ \widehat{\boldsymbol{y}}\right) + \boldsymbol{e}_1 = \boldsymbol{A}^\top \boldsymbol{y} + \boldsymbol{e}_1 = \left(\sum_{j=0}^{k-1} \boldsymbol{A}[j,i]\boldsymbol{y}[j] + \boldsymbol{e}_1[i]\right)_{i=0}^{k-1} \in R_q^k$

▷ ただし，$\boldsymbol{A} = \mathsf{NTT}^{-1}(\widehat{\boldsymbol{A}}) = (\boldsymbol{A}[i,j])_{i,j=0}^{k-1} \in (R_q)^{k \times k}$ とする

7: $\mu = \mathsf{Decompress}\left(\mathsf{ByteDecode}(m)\right) \in R_q$

▷ 平文 m をビット列化した後に R_q の元に変換（式 (2.14) と (2.15) を参照）

8: $v = \mathsf{NTT}^{-1}\left(\widehat{\boldsymbol{t}}^\top \circ \widehat{\boldsymbol{y}}\right) + e_2 + \mu = \boldsymbol{t}^\top \boldsymbol{y} + e_2 + \mu \in R_q$

9: **return** $c = \left(\boldsymbol{u}, v\right) \in R_q^k \times R_q$ ▷ 式 (2.16) を参照

▷ FIPS 203 では，$\boldsymbol{u}, v$ を Compress 関数で圧縮したあと，ByteEncode 関数で符号化

$(m_0, m_1, \ldots, m_{n-1}) \in \{0,1\}^n$ に変換した後に，各ビット $m_i \in \{0,1\}$ を Decompress 関数で

$$\mu_i = \left\lceil \frac{q}{2} \cdot m_i \right\rfloor \in \mathbb{Z}_q \tag{2.14}$$

に変換する．ただし，$\lceil a \rfloor$ は有理数 $a \in \mathbb{Q}$ の最近似整数への丸め込みとする．また，各 μ_i を係数とする多項式を

$$\mu = \mu_0 + \mu_1 X + \cdots + \mu_{n-1} X^{n-1} \in R_q \tag{2.15}$$

とする．正確には，FIPS 203 [64] では，ステップ 9 において，$\boldsymbol{u}$ と v はそれぞれ Compress 関数で圧縮した後，ByteEncode 関数で符号化する．出力する暗号文は

$$c = (\boldsymbol{u}, v) = \left(\boldsymbol{A}^\top \boldsymbol{y} + \boldsymbol{e}_1, \boldsymbol{t}^\top \boldsymbol{y} + e_2 + \mu\right) \in R_q^k \times R_q \tag{2.16}$$

の形で，LWE に基づく Regev の公開鍵暗号方式の暗号文 (2.4) と同じである （注意 2.2 を参照）．また，ステップ 6 と 8 において，NTT 空間上で行列-ベクトル積 $\boldsymbol{A}^\top \boldsymbol{y}$ と内積 $\boldsymbol{t}^\top \boldsymbol{y} = \langle \boldsymbol{t}, \boldsymbol{y} \rangle$ を計算することで，計算の高速化を図る．

■K-PKE 復号アルゴリズム アルゴリズム 3 に，復号アルゴリズム（[64, Algorithm 15], K-PKE.Decrypt）の主な処理をまとめる．具体的には，復号鍵 $\mathtt{dk}_{\mathtt{PKE}}$ と暗号文 c を入力とし，復号文 m' を出力する．ステップ 2 において，多項式表現の R_q の元 $w = w_0 + w_1 X + \cdots + w_{n-1} X^{n-1}$

Algorithm 3 K-PKE.Decrypt：K-PKE 復号アルゴリズム（[64, Algorithm 15] の簡略版）

入力： 復号鍵 $\mathtt{dk}_{\mathtt{PKE}} = \widehat{\boldsymbol{s}}$ と暗号文 $c = (\boldsymbol{u}, v)$

出力： 復号文 m'

1: $w = v - \mathsf{NTT}^{-1}\left(\widehat{\boldsymbol{s}}^{\top} \circ \mathsf{NTT}(\boldsymbol{u})\right) = v - \boldsymbol{s}^{\top}\boldsymbol{u} \in R_q$ ▷ メインの復号処理

2: **return** $m' = \mathsf{ByteEncode}\left(\mathsf{Compress}(w)\right)$

に対して，各係数 $w_i \in \mathbb{Z}_q$ を Compress 関数で

$$z_i = \left\lceil \frac{2}{q} \cdot w_i \right\rfloor \bmod 2 \in \{0, 1\} \tag{2.17}$$

に変換する．また，ビット列 $(z_0, z_1, \ldots, z_{n-1})$ を ByteEncode 関数 [64, Algorithm 5] でバイト列に変換する．特に，ByteEncode 関数と ByteDecode 関数はお互いの逆関数である．

復号の正当性

式 (2.16) の暗号文 $c = (\boldsymbol{u}, v)$ に対して，R_q^k 上の LWE 関係式 (2.13) から，

$$\begin{aligned}
w &= v - \boldsymbol{s}^{\top}\boldsymbol{u} && \text{(Algorithm 3 のステップ 1 を参照)} \\
&= \left(\boldsymbol{t}^{\top}\boldsymbol{y} + e_2 + \mu\right) - \boldsymbol{s}^{\top}\left(\boldsymbol{A}^{\top}\boldsymbol{y} + \boldsymbol{e}_1\right) && \text{(式 (2.16) を利用)} \\
&= \boldsymbol{t}^{\top}\boldsymbol{y} + e_2 + \mu - (\boldsymbol{A}\boldsymbol{s})^{\top}\boldsymbol{y} - \boldsymbol{s}^{\top}\boldsymbol{e}_1 \\
&= \boldsymbol{t}^{\top}\boldsymbol{y} + e_2 + \mu - (\boldsymbol{t} - \boldsymbol{e})^{\top}\boldsymbol{y} - \boldsymbol{s}^{\top}\boldsymbol{e}_1 && \text{(式 (2.13) を利用)} \\
&= \mu + \underbrace{e_2 + \boldsymbol{e}^{\top}\boldsymbol{y} - \boldsymbol{s}^{\top}\boldsymbol{e}_1}_{\text{すべての } \mathbb{Z}_q \text{ 係数が十分小さい}} \in R_q
\end{aligned}$$

が成り立つ．ここで，$\boldsymbol{s}, \boldsymbol{e}, \boldsymbol{e}_1, \boldsymbol{y} \in R_q^k$ の各成分 $\boldsymbol{s}[i], \boldsymbol{e}[i], \boldsymbol{e}_1[i], \boldsymbol{y}[i] \in R_q$ と $e_2 \in R_q$ のすべての $\mathbb{Z}_q$ 係数は十分小さいことに注意する（すべて $\mathbb{Z}_q$ 上の中心二項分布 CBD_η からサンプリング）．よって，Compress 関数による各 $\mathbb{Z}_q$ 係数におけるノイズ補正（式 (2.17) を参照）により

$$\begin{aligned}
\mathsf{Compress}(w) &= \mathsf{Compress}(\mu) \\
&= (m_0, m_1, \ldots, m_{n-1}) \in \{0, 1\}^n
\end{aligned}$$

が成り立つ．ただし，2 段目の式変形については，各 $\mathbb{Z}_q$ 係数において式 (2.14) と (2.15) から

$$\left\lceil \frac{2}{q} \cdot \mu_i \right\rfloor \bmod 2 = \left\lceil \frac{2}{q} \cdot \left\lceil \frac{q}{2} \cdot m_i \right\rfloor \right\rfloor \bmod 2 = m_i \in \{0, 1\}$$

であることによる．最後に，ByteEncode 関数により，平文のビット列 $(m_0, m_1, \ldots, m_{n-1})$ をバイト列に変換することで，元の平文 m に復号できる（つまり，復号文 m' は平文 m に一致する）．また，ステップ 1 において，NTT 空間上で内積 $\boldsymbol{s}^{\top}\boldsymbol{u} = \langle \boldsymbol{s}, \boldsymbol{u} \rangle \in R_q$ を計算することで，計算の高速化を図る．

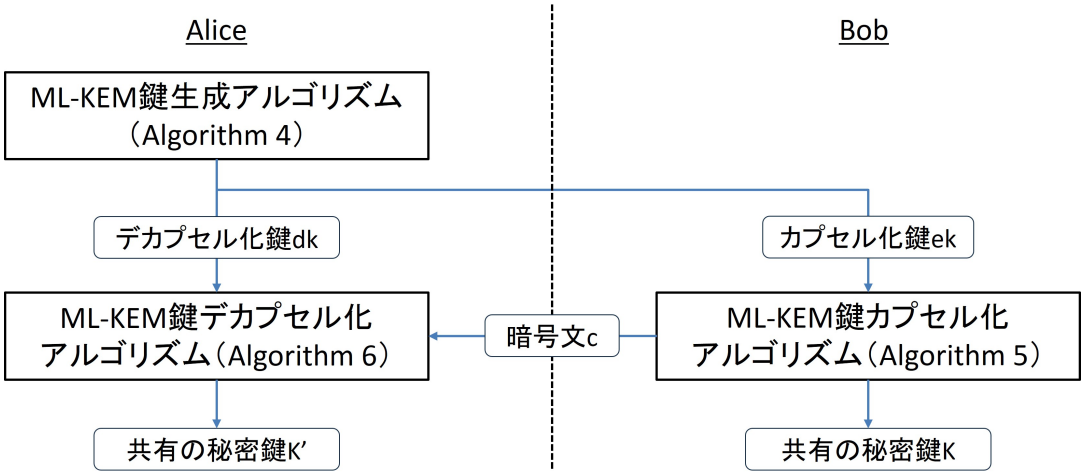

図 2.1　ML-KEM の全体の処理の流れ（文献 [64, Figure 1] を参照）

ML-KEM の処理概要

上記で構成した K-PKE 方式（Algorithm 1, 2, 3）を用いて，ML-KEM は下記の 3 つのアルゴリズムで構成される（ML-KEM の全体の処理の流れは図 2.1 を参照）．

■ML-KEM 鍵生成アルゴリズム　Algorithm 4 に，鍵生成アルゴリズム [64, Algorithm 16] の主な処理をまとめる．具体的には，K-PKE 鍵生成アルゴリズム（Algorithm 1）を用いて，入力する 2 つの乱数 d, z から，カプセル化鍵 $\mathtt{ek}$ とデカプセル化鍵 $\mathtt{dk}$ を出力する．

■ML-KEM 鍵カプセル化アルゴリズム　Algorithm 5 に，鍵カプセル化アルゴリズム [64, Algorithm 17] の主な処理をまとめる．具体的には，K-PKE 暗号化アルゴリズム（Algorithm 2）を用いて，カプセル化鍵 $\mathtt{ek}$ と乱数 m から共有の秘密鍵 K と暗号文 c を出力する．特に，m と $\mathtt{ek}$ のハッシュ値 (K, r) に対して，暗号文 c は $(\mathtt{ek}, m, r)$ から一意的に（つまり，決定的に）生成する．

■ML-KEM 鍵デカプセル化アルゴリズム　Algorithm 6 に，鍵デカプセル化アルゴリズム [64, Algorithm 18] の主な処理をまとめる．具体的には，K-PKE 復号アルゴリズム（Algorithm 3）を用いて，デカプセル化鍵 $\mathtt{dk}$ と暗号文 c から，共有の秘密鍵（のコピー）K' を出力する．特に，入力する暗号文 c が改竄されていないことを保証するために，K-PKE 暗号化アルゴリズム（Algorithm 2）で復号文から暗号文 c' を生成し，c と c' が一致するか検証する．ただし，上記の鍵カプセル化アルゴリズムと同じように，復号文 m' と $\mathtt{ek}$ のハッシュ値 (K', r') に対して，暗号文 c' は $(\mathtt{ek}, m', r')$ から一意的に生成する．

ML-KEM の正当性

鍵デカプセル化アルゴリズム（Algorithm 6）に入力する暗号文 c が，鍵カプセル化アルゴリズム（Algorithm 5）に入力する $(\mathtt{ek}, m)$ から正当に計算されたものとする．このとき，Algorithm 6

Algorithm 4 ML-KEM 鍵生成アルゴリズム [64, Algorithm 16]

入力： 2 つの乱数 d, z

出力： カプセル化鍵 $\mathtt{ek}$ とデカプセル化鍵 $\mathtt{dk}$

1: K-PKE 鍵生成アルゴリズム（Algorithm 1）で，乱数 d から $(\mathtt{ek}_{\mathtt{PKE}}, \mathtt{dk}_{\mathtt{PKE}})$ を生成

2: $\mathtt{ek} = \mathtt{ek}_{\mathtt{PKE}}$

3: $\mathtt{dk} = \big(\mathtt{dk}_{\mathtt{PKE}}, \mathtt{ek}, \mathsf{H}(\mathtt{ek}), z\big)$ ▷ H はハッシュ関数

4: **return** $\big(\mathtt{ek}, \mathtt{dk}\big)$

Algorithm 5 ML-KEM 鍵カプセル化アルゴリズム [64, Algorithm 17]

入力： カプセル化鍵 $\mathtt{ek}$ と乱数 m

出力： 共有の秘密鍵 K と暗号文 c

1: $(K, r) = \mathsf{G}\left(m \| \mathsf{H}(\mathtt{ek})\right)$ ▷ G はハッシュ関数

2: K-PKE 暗号化アルゴリズム（Algorithm 2）で，$(\mathtt{ek}, m, r)$ から暗号文 c を生成
▷ c は $(\mathtt{ek}, m, r)$ から一意的に生成されることに注意

3: **return** (K, c)

Algorithm 6 ML-KEM 鍵デカプセル化アルゴリズム [64, Algorithm 18]

入力： デカプセル化鍵 $\mathtt{dk} = (\mathtt{dk}_{\mathtt{PKE}}, \mathtt{ek}, \mathsf{H}(\mathtt{ek}), z)$ と暗号文 c

出力： 共有の秘密鍵 K

1: K-PKE 復号アルゴリズム（Algorithm 2）で，復号鍵 $\mathtt{dk}_{\mathtt{PKE}}$ と暗号文 c から，復号文 m' を生成

2: $(K', r') = \mathsf{G}\left(m' \| \mathsf{H}(\mathtt{ek})\right)$

3: $\bar{K} = \mathsf{J}(z \| c)$ ▷ J はハッシュ関数

4: K-PKE 暗号化アルゴリズム（Algorithm 2）で，$(\mathtt{ek}, m', r')$ から暗号文 c' を生成
▷ c' は $(\mathtt{ek}, m', r')$ から一意的に生成されることに注意

5: $c \neq c'$ の場合は，$K' = \bar{K}$ とおく

6: **return** K'

のステップ 1 で，c の復号文は $m' = m$ となる．これより，Algorithm 6 のステップ 2 は，鍵カプセル化アルゴリズムのステップ 1 と同じ組 (K, r) を生成する．また，Algorithm 6 のステップ 4 で，$(\mathtt{ek}, m', r') = (\mathtt{ek}, m, r)$ より，同じ暗号文 $c = c'$ を生成する．よって，Algorithm 6 は，鍵カプセル化アルゴリズムと同じ共有の秘密鍵

$$K' = K$$

を出力する．一方，入力する暗号文 c' が改竄されていれば，$c \neq c'$ なので，$K' = \bar{K} \neq K$ となる．

ML-KEM と CRYSTALS-Kyber との違い

ここでは，ML-KEM 方式と CRYSTALS-Kyber 方式 [13] の違いについてまとめておく（詳細は文献 [64, Appendix C] を参照）．

- Kyber 方式では，共有の秘密鍵 K は長さが可変な値として扱われていた．一方，ML-KEM 方式では，K の長さは 256 ビットに固定している．また，K は直接共通鍵として利用することも，秘密鍵生成の乱数シードとして用いることもできる．
- ML-KEM の鍵カプセル化と鍵デカプセル化のアルゴリズムでは，Kyber の第 3 ラウンド仕様書 [13] とは異なる藤崎-岡本変換を利用する．具体的には，ML-KEM 鍵カプセル化アルゴリズム（Algorithm 5）では共有する秘密鍵 K の導出において，暗号文 c のハッシュ値を含まない（具体的には，Algorithm 5 のステップ 1 で，入力する乱数 m とカプセル化鍵 ek のハッシュ値から K を生成）．また，ML-KEM 鍵デカプセル化アルゴリズムではその変更に合わせている（Algorithm 6 のステップ 2 を参照）．
- Kyber の第 3 ラウンドの仕様書 [13] では，鍵カプセル化アルゴリズム内の初期乱数 m は使う前にハッシュされる．具体的には，Kyber における鍵カプセル化アルゴリズムの 1 行目と 2 行目の間に，

  $$m \leftarrow H(m)$$

  の処理ステップがあった．一方，ML-KEM 鍵カプセル化アルゴリズム（Algorithm 5）では，m の生成には NIST 承認の乱数生成器が用いられるため，その処理は不必要で行わない．
- ML-KEM では，Kyber の第 3 ラウンドの仕様書 [13] にはなかった入力データの検証ステップを含む．例えば，ML-KEM 鍵カプセル化アルゴリズムでは，カプセル化キーを含むバイト配列が，モジュラー還元なしで q を法とする整数配列に正しくデコードされることを必要とする（ただし，上記の Algorithm 5 では，詳しく説明してない．入力データの整合性チェックに関しては，FIPS 仕様書 [64, §7] の Key pair check, Encapsulation key check, Decapsulation input check の段落をそれぞれ参照）．

2.2.3 ML-KEM における暗号パラメータ

表 2.1 に，ML-KEM における主な暗号パラメータ，対応する鍵・暗号文のサイズ，安全性レベルなどをまとめる．ただし，RBG（Random Bit Generator）強度は，乱数生成器が出力するビット列に対する攻撃困難性を表す．具体的には，(Module-)LWE の次元 $n = 256$ と剰余素数 $q = 3329$ は ML-KEM-512, -768, -1024 の 3 種類の暗号パラメータで共通であるが，主に 3 種類の階数パラメータ $k \in \{2, 3, 4\}$ により安全性レベルが異なる．特に，ML-KEM のパラメータ名は，

$$n \times k \in \{512, 768, 1024\}$$

表 2.1　ML-KEM における暗号パラメータ・安全性レベル・暗号文サイズなど（ただし，RBG（Random Bit Generator）強度は，乱数生成器が出力するビット列に対する攻撃困難性を表す）

ML-KEM パラメータ		ML-KEM-512	ML-KEM-768	ML-KEM-1024
暗号パラメータ	(n, q)	(256, 3329)	(256, 3329)	(256, 3329)
	k	2	3	4
	(η_1, η_2)	(3, 2)	(2, 2)	(2, 2)
	(d_u, d_v)	(10, 4)	(10, 4)	(11, 5)
デカプセル化（復号）失敗確率		$2^{-138.8}$	$2^{-164.8}$	$2^{-174.8}$
要求される RBG 強度（ビット）		128	192	256
NIST 安全性レベル [63]		レベル 1	レベル 3	レベル 5
サイズ（単位：バイト）	カプセル化鍵	800	1184	1568
	デカプセル化鍵	1632	2400	3168
	暗号文	768	1088	1568
	共有の秘密鍵	32	32	32

の値により名づけられている．また，各暗号パラメータ $(n, q, k, \eta_1, \eta_2, d_u, d_v)$ は下記のように選択されている（詳細は，文献 [13, Section 1.4] を参照）．

- ML-KEM 内の K-PKE 暗号アルゴリズム（Algorithm 2）において，256 ビットの平文を扱うので，n は 256 以上が必要であるため，$n = 256$ が選ばれている．
- 2.2.1 項で述べた NTT 処理を行うため，$n = 256 \mid q - 1$ を満たす小さな素数 q を選択する必要がある．この条件を満たす素数として，257 と 769 があるが，CCA 安全性に関する失敗確率が無視できない．具体的に，LWE ノイズを要因とする K-PKE の復号エラー率が無視できない大きさになり，CCA 安全性の帰着効率に影響する．これより，次に小さな素数である $q = 3329$ が選ばれている．
- 安全性レベル（ML-KEM の安全性を支える Module-LWE 問題の計算困難性）に応じて，階数 $k \in \{2, 3, 4\}$ を調整している（ML-KEM に対する具体的な攻撃計算量などは，後述の表 4.1 と 4.2 を参照）．
- その他のパラメータ η_1, η_2, d_u, d_v は安全性，暗号文サイズ，デカプセル化（復号）失敗確率のバランスを取るように選ばれている．特に，復号失敗を利用した攻撃（または，文献 [43] などの攻撃の改良）の脅威を避けるために，失敗確率が 2^{-128} より小さくなるようにパラメータの値が選ばれている．
  - η_1 は，秘密ベクトル $\boldsymbol{s}$ と公開鍵のノイズベクトル $\boldsymbol{e}$ と，暗号文における $\boldsymbol{y}$ の中心二項分布の大きさを定める．また，η_2 は，暗号文における $\boldsymbol{e}_1$ と e_2 のノイズを定める．具体的に，ML-KEM-512 では，$\eta_1 = 3 > \eta_2 = 2$ と設定されている（公開鍵に比べて，暗号文 $c = (\boldsymbol{u}, v)$ における $\boldsymbol{e}_1, e_2$ によるノイズは小さいが，$\boldsymbol{u}, v$ ともに Compress 関数

で圧縮するので，暗黙的にノイズが増大する）．一方，その他の ML-KEM-768, -1024 では $\eta_1 = \eta_2 = 2$ と設定されている．

– 上記では詳しく説明してないが，(d_u, d_v) は Compress/Decompress 関数，ByteEncode/ByteDecode 関数で使われるパラメータである．具体的には，(d_u, d_v) は暗号文サイズ圧縮のためのベクトル量子化に用いられる．

第 3 章

ML-KEM の安全性証明に関する調査結果

本章では，ML-KEM の安全性証明に関する調査結果を述べる．具体的には，古典ランダムオラクルモデル（ROM）と量子ランダムオラクルモデル（QROM）における Module-LWE 問題（定義 2.2）からの安全性帰着について述べる．

3.1 安全性仮定

ML-KEM の安全性を支える計算問題は，定義 2.2 の Module-LWE 問題である．ただし，ML-KEM における秘密 $\boldsymbol{s} \in R_q^k$ の成分多項式 s_i $(i = 1, \ldots, k)$ とノイズ多項式 $e \in R_q^k$ のすべての $\mathbb{Z}_q$ 係数は，中心二項分布 CBD_η $(\eta \in \{2, 3\})$ からサンプリングされる．ここで，B_η をすべての $\mathbb{Z}_q$ 係数が CBD_η からサンプルされる R_q 上のサンプル分布とする．ML-KEM の安全性証明においては，次の 2 種類のサンプルを識別する判定版の Module-LWE 問題を考える．

- 一様ランダムなサンプル $\left(\boldsymbol{a}_i, t_i\right) \leftarrow R_q^k \times R_q$
- Module-LWE サンプル $\left(\boldsymbol{a}_i, t_i\right) \in R_q^k \times R_q$（$\boldsymbol{a}_i \leftarrow R_q^k$ は一様ランダムにサンプルされ，式 (2.7) のように $t_i = \langle \boldsymbol{a}_i, \boldsymbol{s} \rangle + e_i \in R_q$ とする．ただし，$\boldsymbol{s} \leftarrow B_\eta^k$ はすべての Module-LWE サンプルに対し共通である一方，$e_i \leftarrow B_\eta$ は毎回選ばれる．）

より正確には，判定版の Module-LWE 問題に対する攻撃者（またはアルゴリズム）$\mathscr{A}$ に対して，

$$
\begin{aligned}
&\mathbf{Adv}^{\mathrm{mlwe}}_{m,k,\eta}(\mathscr{A}) \\
&= \left|\Pr\left[b' = 1 :\ \boldsymbol{A} \leftarrow R_q^{m \times k};\ (\boldsymbol{s}, \boldsymbol{e}) \leftarrow B_\eta^k \times B_\eta^m;\ \boldsymbol{t} = \boldsymbol{s}\boldsymbol{A}^\top + \boldsymbol{e};\ b' \leftarrow \mathscr{A}(\boldsymbol{A}, \boldsymbol{t})\right]\right. \\
&\qquad \left. - \Pr\left[b' = 1 :\ \boldsymbol{A} \leftarrow R_q^{m \times k};\ \boldsymbol{t} \leftarrow R_q^m;\ b' \leftarrow \mathscr{A}(\boldsymbol{A}, \boldsymbol{t})\right]\right| \geq 0
\end{aligned}
\tag{3.1}
$$

と定める．ただし，m は Module-LWE サンプル数で，$b' \leftarrow \mathscr{A}(\boldsymbol{A}, \boldsymbol{t})$ は攻撃者 $\mathscr{A}$ による出力結果とする．以下の ML-KEM の安全性証明においては，暗号文の形 (2.16) から，$m = k + 1$ の場合の $\mathbf{Adv}^{\mathrm{mlwe}}_{k+1,k,\eta}(\mathscr{A})$ が十分 0 に近い Module-LWE 仮定の下で考える．

3.2 ROM における Module-LWE からのタイトな帰着

次の定理が示すように，ランダムオラクルモデル（ROM）において，Module-LWE 仮定の下で，ML-KEM の基盤である公開鍵暗号方式 K-PKE（Algorithms 1, 2, 3 で構成）はタイトな IND-CPA 安全である（ただし，[13, Theorem 1, §4.3.1] において，“Kyber.CPAPKE” を “ML-KEM.K-PKE” に変更）．その証明は，Module-LWE 仮定の下で、公開鍵と暗号文が擬似ランダムであることから従う．

定理 3.1 ([13], Theorem 1)**.** XOF と G はランダムオラクルとする．このとき，任意の攻撃者 $\mathscr{A}$ に対して，$\mathscr{A}$ と同程度の処理能力を持つ攻撃者 $\mathscr{B}, \mathscr{C}$ が存在して，

$$\mathbf{Adv}^{\mathrm{cpa}}_{\text{ML-KEM.K-PKE}}(\mathscr{A}) \leq 2 \cdot \mathbf{Adv}^{\mathrm{mlwe}}_{k+1,k,\eta}(\mathscr{B}) + \mathbf{Adv}^{\mathrm{prf}}_{\mathrm{PRF}}(\mathscr{C})$$

を満たす．ただし，$\mathbf{Adv}^{\mathrm{cpa}}_{\text{ML-KEM.K-PKE}}$ と $\mathbf{Adv}^{\mathrm{prf}}_{\mathrm{PRF}}$ はそれぞれ式 (3.1) と同じように定める．

ML-KEM（Algorithms 4, 5, 6 で構成）は K-PKE の（微調整した）藤崎-岡本変換から得られる．2 つのハッシュ関数 G,H をランダムオラクルとモデル化した仮定の下で，次の定理が示すように ML-KEM は IND-CCA2 安全である（ただし，[13, Theorem 2, §4.3.1] において，“Kyber.CCAKEM” を “ML-KEM” に変更した）．特に，次の定理におけるタイトな上界は，上記の定理 3.1 と文献 [42] の結果から得られる．

定理 3.2 ([13], Theorem 2)**.** XOF と 2 つのハッシュ関数 G, H はランダムオラクルとする．このとき，XOF，G, H のランダムオラクルに高々 q_{RO} 回問い合わせることができる古典攻撃者 $\mathscr{A}$ に対して，$\mathscr{A}$ と同程度の処理能力を持つ攻撃者 $\mathscr{B}, \mathscr{C}$ が存在して，

$$\mathbf{Adv}^{\mathrm{cca}}_{\text{ML-KEM}}(\mathscr{A}) \leq 2 \cdot \mathbf{Adv}^{\mathrm{mlwe}}_{k+1,k,\eta}(\mathscr{B}) + \mathbf{Adv}^{\mathrm{prf}}_{\mathrm{PRF}}(\mathscr{C}) + 4q_{RO} \cdot \delta$$

を満たす．ただし，$\mathbf{Adv}^{\mathrm{cca}}_{\text{ML-KEM}}$ は式 (3.1) と同じように定め，δ は ML-KEM の基盤である K-PKE における復号失敗確率とする．

3.3 QROM における Module-LWE からのノンタイトな帰着

次の定理が示すように，量子ランダムオラクルモデル（QROM）において，Module-LWE 仮定の下で，ML-KEM の基盤である公開鍵暗号方式 K-PKE が IND-CPA 安全であれば，ML-KEM は IND-CCA2 安全である（詳細は文献 [42, 69] を参照．定理 3.2 と同じように，[13, Theorem 3] において，“Kyber.CCAKEM” を “ML-KEM” に変更した）．

定理 3.3 ([13], Theorem 3)**.** XOF と 2 つのハッシュ関数 G,H は量子ランダムオラクルとする．このとき，XOF，G, H の量子ランダムオラクルに高々 q_{RO} 回問い合わせることができる量子攻撃者

$\mathscr{A}$ に対して，$\mathscr{A}$ と同程度の処理能力を持つ量子攻撃者 $\mathscr{B}, \mathscr{C}$ が存在して，

$$\mathbf{Adv}^{\mathrm{cca}}_{\text{ML-KEM}}(\mathscr{A}) \leq 4q_{RO} \cdot \sqrt{\mathbf{Adv}^{\mathrm{mlwe}}_{k+1,k,\eta}(\mathscr{B})} + \mathbf{Adv}^{\mathrm{prf}}_{\mathrm{PRF}}(\mathscr{C}) + 8q_{RO}^2 \cdot \delta \tag{3.2}$$

を満たす．

上記の定理における上界はノンタイトである．ゆえに，上記の定理は，量子ランダムオラクルモデルにおける ML-KEM の IND-CCA 安全性について漸近的な示唆を与えるだけである．

一方，標準的ではない仮定の下で，量子ランダムオラクルモデルにおいて，よりタイトな上界を得ることができる．具体的には，ML-KEM の基盤である公開鍵暗号方式 K-PKE の確定版が，量子ランダムオラクルモデルにおいて擬似ランダムと仮定する．つまり，暗号化時に利用するランダムコインが，$r = G(m)$ のように平文 m によって確定的に決定すると仮定する．また，確定版の暗号化における擬似ランダム性とは，ランダムに選ばれた平文の暗号文 (c_1, c_2) が，一様ランダムな $(\boldsymbol{u}, v)$ に対する暗号文 $\big(\mathsf{Compress}_q(\boldsymbol{u}), \mathsf{Compress}(v)\big)$ と計算量的に識別困難であることをいう．このとき，式 (3.2) よりタイトな上界

$$\mathbf{Adv}^{\mathrm{cca}}_{\text{ML-KEM}}(\mathscr{A}) \leq 2\mathbf{Adv}^{\mathrm{mlwe}}_{k+1,k,\eta}(\mathscr{B}) + \mathbf{Adv}^{\mathrm{pr}}_{\text{DK-PKE}}(\mathscr{C}) + \mathbf{Adv}^{\mathrm{prf}}_{\mathrm{PRF}}(\mathscr{D}) + 8q_{RO}^2 \cdot \delta$$

が成り立つ [13, §4.3.2]．ただし，DK-PKE は K-PKE の確定版とし，“pr” はその擬似ランダム性（Pseudo-Randomness）とする．

3.4 Module-LWE 問題以外への攻撃

上記の ML-KEM に対する安全性証明から，Module-LWE 問題を解くことなく ML-KEM を攻撃する方法として次がある．

- 安全性証明の帰着仮定である共通プリミティブを攻撃する．
- ML-KEM におけるデカプセル化（復号）失敗確率におけるノンタイト性を利用する．

3.4.1 共通鍵暗号プリミティブへの攻撃可能性について

前章では詳しく説明しなかったが，次のように共通鍵暗号プリミティブは FIPS 202 標準 [62] からの関数でインスタント化する．

- `XOF`：SHAKE-128
- `H`：SHA3-256
- `G`：SHA3-512
- $\texttt{PRF}(s, b)$：SHAKE-256$(s \| b)$（擬似ランダム関数 PRF: PseudoRandom Function）
- `KDF`：SHAKE-256

これらの共通の構成要素は，Keccak（SHA-3）から導出された関数群でインスタント化される．具

体的には，乱数シード ρ から一意的に $\boldsymbol{A}$ を展開する際は，SHAKE-128 を利用して一様擬似ランダムな成分をもつ行列を出力し，安全性を支える格子問題におけるバックドアを発生させない．また，ノイズ（誤差）生成時には，秘密と公開の入力を結合し，その結合値を SHAKE-256 に入力することで，安全な擬似ランダムな関数を構成する．これらの SHAKE の性質を破ることは，SHAKE の暗号解析における重大なブレークスルーで，KEM の中の SHAKE を他の擬似ランダム関数に置き換える必要があるが，現状そのような攻撃は現実的ではない．安全性証明では，SHAKE-128, SHA3-256, SHA3-512 をランダムオラクルとしてモデル化している．`XOF`, `G`, `H` の関数の SHAKE と SHA3 によるインスタンスを利用する攻撃は，一般に Keccak またはランダムオラクル証明における重大なブレークスルーであり，そのような攻撃は現状では現実的ではない．

3.4.2 復号失敗を利用した攻撃の可能性について

K-PKE，ML-KEM ともに復号エラーが起きた場合には，plaintext checking, key mismatch 等のオラクルを構成し，それをベースに鍵復元を行うサイドチャネル攻撃が数多く提案されている（例えば，文献 [66, 73, 77] を参照）．しかし，表 2.1 に示すように，ML-KEM における復号失敗確率 δ の見積もりは 2^{-128} 未満であり，実用的には無視できるほど小さい（IETF ドラフト [44, Section 8.1] も参照）．一方で，復号失敗を起こすような正規の暗号文（weak ciphertexts）を効率的に探索する手法も提案されているが [36]，ML-KEM の安全性に影響を与えるほど実用的なものではない．

第 4 章

Module 構造の考慮の有無に応じた計算量評価に関する調査結果

本章では，ML-KEM の安全性を支える Module-LWE 問題を解く攻撃法とその計算量見積もりを示す．具体的には，Module-LWE 問題を $\mathbb{Z}_q$ 上の LWE 問題に帰着し，さらに $\mathbb{Z}_q$ 上の LWE 問題を格子問題に帰着する．また，帰着した格子問題を解く現時点で最良とされる格子アルゴリズムの計算量評価に基づき，ML-KEM の暗号パラメータに対する攻撃計算量の見積もりを示す．本章では，特に断らない限り，すべてのベクトルは行ベクトルで統一する．また，$\mathbb{R}$ 成分のベクトル $\boldsymbol{v} = (v_1, \ldots, v_d) \in \mathbb{R}^d$ のノルムは，すべて Euclid ノルム

$$\|\boldsymbol{v}\| = \sqrt{\langle \boldsymbol{v}, \boldsymbol{v} \rangle} = \sqrt{v_1^2 + \cdots + v_d^2}$$

とする．

4.1 Module-LWE 問題の $\mathbb{Z}_q$ 上の LWE 問題への帰着

本節では，定義 2.2 の Module-LWE 問題を，通常の $\mathbb{Z}_q$ 上の LWE 問題の形に帰着できることを述べる（数学的には，$\mathbb{Z}_q$ 加群としての同型 $R_q^k \simeq (\mathbb{Z}_q^n)^k$ を明示的に与えることに相当する）．Module-LWE 問題における剰余環 $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ において，変数 X に関するベクトルを

$$\boldsymbol{X} = \left(1, X, X^2, \ldots, X^{n-1}\right) \in R_q^n$$

とおく．このとき，R_q の任意の元 $f(X) = f_0 + f_1 X + \cdots + f_{n-1} X^{n-1}$ とその係数ベクトル $\boldsymbol{f} = (f_0, f_1, \ldots, f_{n-1}) \in \mathbb{Z}_q^n$ に対して，

$$f(X) = \boldsymbol{f}\boldsymbol{X}^\top \in R_q$$

が成り立つ（右辺は 2 つのベクトル $\boldsymbol{f}, \boldsymbol{X}$ の内積）．また，$f(X)$ の係数ベクトル $\boldsymbol{f}$ に対する回転操作を

$$\mathrm{rot}(\boldsymbol{f}) \coloneqq \left(-f_{n-1}, f_0, f_1 \ldots, f_{n-2}\right) \in \mathbb{Z}_q^n \tag{4.1}$$

と定める．このとき，(1 回の）回転ベクトル $\mathrm{rot}(\boldsymbol{f})$ は，$f(X)$ に X を乗じた多項式 $Xf(X)$ の係数ベクトルである．より一般に，X^i を乗じた多項式 $X^i f(X)$ の係数ベクトルは，i 回の回転操作を施したベクトル $\mathrm{rot}^i(\boldsymbol{f}) = \mathrm{rot}\left(\mathrm{rot}(\cdots \mathrm{rot}\left(\boldsymbol{f}\right))\right)$ で与えられる．特に，環 R_q において $X^n = -1$ なので，n 回の回転ベクトル $\mathrm{rot}^n(\boldsymbol{f})$ は $-\boldsymbol{f}$ に一致する（つまり，$\mathrm{rot}^n(\boldsymbol{f}) = -\boldsymbol{f}$ である）．

定義 2.2 と同じように，Module-LWE 問題の秘密を $\boldsymbol{s}(X) = \big(s_1(X), s_2(X), \ldots, s_k(X)\big) \in R_q^k$ と表す．また，式 (2.7) の Module-LWE サンプルの組 $\big(\boldsymbol{a}(X), t(X)\big) \in R_q^k \times R_q$ に対して，$\boldsymbol{a}(X) = (a_1(X), a_2(X), \ldots, a_k(X))$ と表す．このとき，環 R_q において，

$$t(X) = \sum_{i=1}^{k} a_i(X) s_i(X) + e(X)$$

が成り立つことを意味する．さらに，R_q の元 $s_i(X), a_i(X)$ $(i = 1, \ldots, k)$ と $e(X), t(X)$ にそれぞれ対応する係数ベクトルを $\boldsymbol{s}_i, \boldsymbol{a}_i$ $(i = 1, 2, \ldots, k)$ と $\boldsymbol{e}, \boldsymbol{t} \in \mathbb{Z}_q^n$ とする．このとき，

$$\begin{aligned}
\boldsymbol{t}\boldsymbol{X}^\top = t(X) &= \sum_{i=1}^{k} a_i(X) s_i(X) + e(X) \\
&= \sum_{i=1}^{k} \boldsymbol{s}_i \boldsymbol{X}^\top a_i(X) + \boldsymbol{e}\boldsymbol{X}^\top \\
&= \sum_{i=1}^{k} \boldsymbol{s}_i \begin{pmatrix} a_i(X) \\ X a_i(X) \\ \vdots \\ X^{n-1} a_i(X) \end{pmatrix} + \boldsymbol{e}\boldsymbol{X}^\top = \sum_{i=1}^{k} \boldsymbol{s}_i \begin{pmatrix} \boldsymbol{a}_i \boldsymbol{X}^\top \\ \mathrm{rot}(\boldsymbol{a}_i)\boldsymbol{X}^\top \\ \vdots \\ \mathrm{rot}^{n-1}(\boldsymbol{a}_i)\boldsymbol{X}^\top \end{pmatrix} + \boldsymbol{e}\boldsymbol{X}^\top \\
&= \left(\sum_{i=1}^{k} \boldsymbol{s}_i \boldsymbol{A}_i + \boldsymbol{e} \right) \boldsymbol{X}^\top
\end{aligned}$$

が成り立つ（各 $j = 0, 1, \ldots, n-1$ に対し，$X^j a_i(X) = \mathrm{rot}^j(\boldsymbol{a}_i)\boldsymbol{X}^\top$ であることに注意）．ただし，各 $i = 1, 2, \ldots, k$ に対して，

$$\boldsymbol{A}_i = \begin{pmatrix} \boldsymbol{a}_i \\ \mathrm{rot}(\boldsymbol{a}_i) \\ \vdots \\ \mathrm{rot}^{n-1}(\boldsymbol{a}_i) \end{pmatrix} \in \mathbb{Z}_q^{n \times n}$$

とする．これより，変数ベクトル $\boldsymbol{X}$ の成分による集合 $\{1, X, X^2, \ldots, X^{n-1}\}$ は自由 $\mathbb{Z}_q$ 加群 R_q の基底（つまり，$R_q = \mathbb{Z}_q \oplus \mathbb{Z}_q X \oplus \cdots \oplus \mathbb{Z}_q X^{n-1}$）なので，

$$\boldsymbol{t} \equiv \sum_{i=1}^{k} \boldsymbol{s}_i \boldsymbol{A}_i + \boldsymbol{e} \pmod{q} \iff \boldsymbol{t} \equiv (\boldsymbol{s}_1 \mid \cdots \mid \boldsymbol{s}_k) \begin{pmatrix} \boldsymbol{A}_1 \\ \hline \vdots \\ \hline \boldsymbol{A}_k \end{pmatrix} + \boldsymbol{e} \pmod{q} \tag{4.2}$$

が成り立つ．各 $j = 0, 1, \ldots, n-1$ に対して，両辺の X^j の係数に対応する列を比較することで，次元 nk の $\mathbb{Z}_q$ 上の LWE サンプルの関係式が得られる．具体的には，$\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k$ の連結ベクトル

$$\boldsymbol{s} = (\boldsymbol{s}_1 \mid \cdots \mid \boldsymbol{s}_k) \in (\mathbb{Z}_q^n)^k \tag{4.3}$$

を秘密とした n 個の $\mathbb{Z}_q$ 上の nk 次元の LWE サンプル

$$\big(\boldsymbol{a}_j, t_j\big), \quad t_j \equiv \langle \boldsymbol{a}_j, \boldsymbol{s} \rangle + e_j \pmod{q} \quad (j = 0, 1, \ldots, n-1) \tag{4.4}$$

が得られる．ただし，$\boldsymbol{a}_j$ は式 (4.2) の右辺の $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_k$ を縦に連結した $nk \times n$ 行列の $j+1$ 列目のベクトル，t_j, e_j はそれぞれ $t(X), e(X) \in R_q$ の X^j 係数とする．また，攻撃者有利の観点から，任意に Module-LWE サンプルを生成し，その中から $\mathbb{Z}_q$ 上の LWE サンプルの m 個を行列表示

$$\big(\boldsymbol{A}, \boldsymbol{t}\big), \quad \boldsymbol{t} \equiv \boldsymbol{s}\boldsymbol{A}^\top + \boldsymbol{e} \pmod{q} \tag{4.5}$$

しておく．ただし，$\boldsymbol{A} \in \mathbb{Z}_q^{m \times nk}$，$\boldsymbol{t}, \boldsymbol{e} \in \mathbb{Z}_q^m$ とする．また，$\mathbb{Z}_q$ 上の LWE サンプル数 m は，一般に攻撃が最も有利となるものを想定する．この行列表示により，行列 $\boldsymbol{A}$ の行ベクトル $\boldsymbol{a}_j \in \mathbb{Z}_q^{nk}$ と秘密ベクトル $\boldsymbol{s} \in \mathbb{Z}_q^{nk}$ の内積 $\langle \boldsymbol{a}_j, \boldsymbol{s} \rangle$ について，式 (4.4) の関係が成り立つ．特に，ML-KEM では，秘密ベクトル $\boldsymbol{s}$ とノイズベクトル $\boldsymbol{e}$ のすべての $\mathbb{Z}_q$ 成分は，$\eta \in \{2, 3\}$ に対する $\mathbb{Z}_q$ 上の中心二項分布 CBD_η からサンプリングされる（ML-KEM における中心二項分布 CBD_η の具体的な計算手順については，2.2.2 項を参照）．

4.2 格子の基礎と格子アルゴリズム

本節では，格子の基礎と格子問題を解くための格子アルゴリズムについて簡単にまとめておく．

4.2.1 格子と基底

整数 $d \geq 2$ に対して，d 次元実ベクトル空間 $\mathbb{R}^d$ の一次独立な d 個のベクトル $\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_d$ の整数係数の線形結合全体の集合

$$L = \left\{ \sum_{i=1}^{d} x_i \boldsymbol{b}_i \;\middle|\; x_i \in \mathbb{Z},\ 1 \leq i \leq d \right\} = \mathbb{Z}\boldsymbol{b}_1 + \cdots + \mathbb{Z}\boldsymbol{b}_d$$

を（完全階数の）d 次元の**格子**（lattice）とよぶ．特に，格子 L は $\mathbb{R}^d$ の（離散）加法部分群である．また，格子 L を生成する一次独立な d 個のベクトルの組 $\{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_d\}$ を**基底**（basis）とよび，各 $\boldsymbol{b}_i$ $(i = 1, 2, \ldots, d)$ を**基底ベクトル**（basis vector）とよぶ．さらに，行ベクトルで表した d 個の基底ベクトル $\boldsymbol{b}_i \in \mathbb{R}^d$ $(i = 1, 2, \ldots, d)$ を行として持つ $d \times d$ 行列

$$\boldsymbol{B} = \begin{pmatrix} \boldsymbol{b}_1 \\ \vdots \\ \boldsymbol{b}_d \end{pmatrix} \in \mathbb{R}^{d \times d}$$

を格子 L の**基底行列**（basis matrix）とよぶ．2 次元以上の格子を生成する互いに異なる基底は無限に存在する．正確には，同じ格子を生成する 2 つの基底行列 $\boldsymbol{B}_1, \boldsymbol{B}_2$ に対し $\boldsymbol{B}_2 = \boldsymbol{V}\boldsymbol{B}_1$ を満たす $d \times d$ のユニモジュラ行列 $\boldsymbol{V}$ が必ず存在する．ただし，整数行列でその行列式が $\det(\boldsymbol{V}) = \pm 1$ である正方行列 $\boldsymbol{V}$ を**ユニモジュラ行列**（unimodular matrix）とよぶ．これより，2 次以上のユニ

モジュラ行列は無限に存在するため，2 次元以上の格子は互いに異なる基底を無限にもつ．また，格子 L の任意の基底行列 $\boldsymbol{B}$ を用いて，L の**体積**（volume）を

$$\mathrm{vol}(L) \coloneqq |\det(\boldsymbol{B})| > 0$$

と定める．ここで，互いに異なる格子基底行列はユニモジュラ行列で結ばれるので，格子の体積は基底行列の選び方には依存しない．格子 L の**第 1 逐次最小**（first successive minimum）は，L 上の最短な非零ベクトルのノルムを指し，$\lambda_1(L)$ の記号で表す．

双対格子とその基底

d 次元実ベクトル空間 $\mathbb{R}^d$ の完全階数の格子 $L \subset \mathbb{R}^d$ に対して，集合

$$\widehat{L} \coloneqq \left\{ \boldsymbol{x} \in \mathbb{R}^d \mid \langle \boldsymbol{x}, \boldsymbol{y} \rangle \in \mathbb{Z},\ \forall \boldsymbol{y} \in L \right\}$$

を L の**双対格子**（dual lattice）とよぶ．また，L の基底行列 $\boldsymbol{B} \in \mathbb{R}^{d \times d}$ に対して，

$$\widehat{\boldsymbol{B}} \coloneqq \left(\boldsymbol{B}^\top\right)^{-1} = \left(\boldsymbol{B}^{-1}\right)^\top$$

は双対格子 $\widehat{L}$ の基底行列となる（つまり，$\widehat{\boldsymbol{B}}$ の d 個の行ベクトルは，$\mathbb{R}$ 上一次独立で，$\widehat{L}$ を生成する）．この $\widehat{\boldsymbol{B}}$ を**双対基底行列**（dual basis matrix）とよぶ．また，単位行列 $\boldsymbol{I}_d$ に対して，明らかに $\boldsymbol{B}\widehat{\boldsymbol{B}}^\top = \boldsymbol{I}_d$ を満たすので，L の体積とその双対格子 $\widehat{L}$ の体積について

$$\mathrm{vol}(L) \times \mathrm{vol}(\widehat{L}) = 1 \tag{4.6}$$

が成り立つ．

Gauss のヒューリスティックと格子の第 1 次逐次最小

d 次元実ベクトル空間 $\mathbb{R}^d$ 内の完全階数の格子 L に対して，体積 $\mathrm{vol}(C)$ を持つ任意の集合 $C \subseteq \mathbb{R}^d$ との共通部分に含まれる格子ベクトルの個数はおおよそ

$$\#(L \cap C) \approx \frac{\mathrm{vol}(C)}{\mathrm{vol}(L)}$$

であることが期待できる．これを **Gauss のヒューリスティック**（Gaussian Heuristic）とよぶ．特に，集合 C として，格子 L の第 1 次逐次最小 $\lambda_1(L)$ を半径に持ち中心が零ベクトル $\mathbf{0} \in \mathbb{R}^d$ の d 次元開球 $\mathcal{B}(\mathbf{0}, \lambda_1(L))$ をとると，おおよそ

$$\frac{\mathrm{vol}(C)}{\mathrm{vol}(L)} \approx \#(L \cap C) \approx 1$$

と期待できる．さらに，d 次元単位球の体積を ω_d とすると $\mathrm{vol}(C) = \omega_d \times \lambda_1(L)^d$ が成り立つので，

$$\lambda_1(L) \approx \left(\frac{\mathrm{vol}(L)}{\omega_d}\right)^{1/d} \sim \sqrt{\frac{d}{2\pi e}}\mathrm{vol}(L)^{1/d} \tag{4.7}$$

が成り立つと期待できる．ただし，記号 $\sim$ は $d \to \infty$ のとき両辺の比が 1 に収束することを意味する．また，d 次元単位球の体積 ω_d に関しては，ガンマ関数 $\Gamma(x)$ を用いると

$$\omega_d = \frac{\pi^{d/2}}{\Gamma\left(1+\frac{d}{2}\right)} \sim \left(\frac{2\pi e}{d}\right)^{d/2} \tag{4.8}$$

が成り立つことが知られている．

4.2.2 格子問題と格子基底簡約

格子問題は格子に関する計算問題で，以下で最も代表的な 2 つの格子問題である**最短ベクトル問題**（Shortest Vector Problem, SVP）と**最近ベクトル問題**（Closest Vector Problem, CVP）を述べておく．これらの格子問題の求解困難性は，格子暗号の根本的な安全性の根拠となっている．

定義 4.1 (最短ベクトル問題, SVP)**.** 格子 L の基底 $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d\}$ が与えられたとき，格子の最短な非零ベクトル $\boldsymbol{v} \in L$ を見つけよ．つまり，$\|\boldsymbol{v}\| = \lambda_1(L)$ を満たす格子ベクトル $\boldsymbol{v} \in L$ を見つけよ．

定義 4.2 (最近ベクトル問題, CVP)**.** 格子 L の基底 $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d\}$ と $\boldsymbol{t} \in \mathbb{R}^d \setminus L$ が与えられたとき，$\boldsymbol{t}$ に最も近い格子ベクトル $\boldsymbol{v} \in L$ を見つけよ．つまり，$\boldsymbol{t}$ との距離 $\|\boldsymbol{t} - \boldsymbol{v}\|$ を最小にする格子ベクトル $\boldsymbol{v} \in L$ を見つけよ．

上述の SVP・CVP や（Module-）LWE などの格子問題を解くのに必須の格子アルゴリズムとして，**格子基底簡約**（lattice basis reduction）がある．格子基底簡約は，与えられた格子 $L \subset \mathbb{R}^d$ の基底から，各ベクトル $\boldsymbol{b}_i$ が短く・互いのベクトルが直交に近い L の新しい基底 $\{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_d\}$ を見つける操作（アルゴリズム）である．明確な基準があるわけではないが，このような基底を「簡約基底」（reduced basis）または「良い基底」（good basis）とよぶ．具体的な基底簡約のアルゴリズムを紹介するために，**Gram-Schmidt の直交化**（Gram-Schmidt orthogonalization）を説明しておく．格子 L の基底 $\{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_d\}$ の Gram-Schmidt ベクトル $\boldsymbol{b}_1^\star, \boldsymbol{b}_2^\star, \ldots, \boldsymbol{b}_d^\star$ は，次のように再帰的に定める．

$$\begin{cases} \boldsymbol{b}_1^\star \coloneqq \boldsymbol{b}_1, \\ \boldsymbol{b}_i^\star \coloneqq \boldsymbol{b}_i - \displaystyle\sum_{j=1}^{i-1} \mu_{i,j} \boldsymbol{b}_j^\star, \quad \mu_{i,j} = \frac{\langle \boldsymbol{b}_i, \boldsymbol{b}_j^\star \rangle}{\|\boldsymbol{b}_j^\star\|^2} \quad (i = 2, 3, \ldots, d). \end{cases}$$

Gram-Schmidt ベクトルについて，直交性 $\langle \boldsymbol{b}_i^\star, \boldsymbol{b}_j^\star \rangle = 0$ $(i \neq j)$ と，格子の体積について

$$\mathrm{vol}(L) = \prod_{i=1}^{d} \|\boldsymbol{b}_i^\star\| \tag{4.9}$$

が成り立つ．また，各 $2 \leq \ell \leq d$ に対して，$\mathbb{R}^d$ から $\mathbb{R}$-ベクトル空間 $\langle \boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_{\ell-1} \rangle_{\mathbb{R}}$ の直交補空間への直交射影を

$$\pi_\ell : \mathbb{R}^d \longrightarrow \langle \boldsymbol{b}_1, \ldots, \boldsymbol{b}_{\ell-1} \rangle_{\mathbb{R}}^{\perp} = \langle \boldsymbol{b}_\ell^\star, \ldots, \boldsymbol{b}_d^\star \rangle_{\mathbb{R}}, \quad \pi_\ell(\boldsymbol{x}) = \sum_{i=\ell}^{d} \frac{\langle \boldsymbol{x}, \boldsymbol{b}_i^\star \rangle}{\|\boldsymbol{b}_i^\star\|^2} \boldsymbol{b}_i^\star$$

とする．また，便宜上 π_1 は恒等写像としておく．さらに，$(d-\ell+1)$ 個の一次独立な射影ベクトル $\pi_\ell(\boldsymbol{b}_\ell),\ldots,\pi_\ell(\boldsymbol{b}_d)$ で生成させる格子を $\pi_\ell(L)$ と記し，L の**射影格子**（projected lattice）とよぶ．$\pi_\ell(\boldsymbol{b}_\ell),\ldots,\pi_\ell(\boldsymbol{b}_d)$ の Gram-Schmidt ベクトルは $\boldsymbol{b}_\ell^\star,\boldsymbol{b}_{\ell+1}^\star,\ldots,\boldsymbol{b}_d^\star$ であるので，式 (4.9) と同様に，射影格子 $\pi_\ell(L)$ の体積は $\prod_{i=\ell}^{d}\|\boldsymbol{b}_i^\star\|$ で与えられる．

以下で，代表的な 2 つの格子基底簡約アルゴリズムを紹介しておく．

Lentra-Lenstra-Lovász（LLL）基底簡約

LLL 基底簡約 [53] は，簡約パラメータ $\frac{1}{4}<\delta<1$ に対して，次の 2 つの条件を満たす格子の基底 $\{\boldsymbol{b}_1,\boldsymbol{b}_2,\ldots,\boldsymbol{b}_d\}$（LLL 簡約基底という）を見つけるアルゴリズムである．

(i) サイズ簡約されている．つまり，すべての Gram-Schmidt 係数が $|\mu_{i,j}|\leq\frac{1}{2}$ $(1\leq j<i\leq d)$ を満たす．
(ii) Lovász 条件を満たす．つまり，$\delta\|\boldsymbol{b}_{k-1}^\star\|^2\leq\|\pi_{k-1}(\boldsymbol{b}_k)\|^2$ $(k=2,3,\ldots,d)$ を満たす．

入力基底に対して，Lovász 条件が成り立たないとき，LLL 基底簡約内で隣り合う基底ベクトル $\boldsymbol{b}_{k-1},\boldsymbol{b}_k$ の交換を行い，(i) と (ii) の両方の条件を満たす基底（つまり，LLL 簡約基底）を見つける．また，LLL 基底簡約の時間計算量は，入力する基底が生成する格子の次元 d に関して多項式時間である．

Block Korkine-Zolotarev（BKZ）基底簡約

BKZ 基底簡約 [71] は，ブロックサイズ $\beta\geq 2$ による LLL 基底簡約の一般化である（$\beta=2$ の場合は LLL 基底簡約と本質的に同じ）．LLL 基底簡約に比べ，BKZ 基底簡約でより良い簡約基底を見つけることができるが，その計算量は β に関して指数時間である．具体的には，BKZ 基底簡約に入力するブロックサイズ β を増やすごとに，実行時間が非常に遅くなる一方，より短い基底ベクトルを出力する．より具体的には，ブロックサイズ $2\leq\beta\leq d$ に対して，BKZ 基底簡約は次の 2 つの条件を満たす格子 L の基底 $\{\boldsymbol{b}_1,\boldsymbol{b}_2,\ldots,\boldsymbol{b}_d\}$（$\beta$-BKZ 簡約基底）を見つける．

(i) LLL 基底簡約と同様，基底はサイズ簡約されている．
(ii) すべての $1\leq j\leq d$ に対して，$\|\boldsymbol{b}_j^\star\|=\lambda_1(L_{[j:k]})$ を満たす．ただし，$k=\min(j+\beta-1,d)$ とし，$(k-j+1)$ 個の射影ベクトル $\pi_j(\boldsymbol{b}_j),\pi_j(\boldsymbol{b}_{j+1}),\ldots,\pi_j(\boldsymbol{b}_k)$ で生成される L のブロック射影格子を $L_{[j:k]}$ とする（ブロック射影格子 $L_{[j:k]}$ は，射影格子 $\pi_j(L)$ の部分格子である）．

入力基底に対して，BKZ 基底簡約のアルゴリズム内ではブロック射影格子 $L_{[j:k]}$ 上の SVP オラクルを繰り返しよびだし，(i) と (ii) の両方の条件を満たす基底（つまり，β-BKZ 簡約基底）を見つける．以下で述べるように，BKZ 基底簡約の出力基底と計算量はブロックサイズ β に依存する．

4.2.3 BKZ 基底簡約の出力基底と計算量

これまで BKZ 2.0 [26] や pump & jump BKZ（pnj-BKZ）[5] などの効率的な BKZ 基底簡約の改良アルゴリズムが提案され，格子に基づく暗号技術の安全性評価において頻繁に利用されている．ここでは，BKZ 基底簡約の出力基底と計算量評価の見積もりについて述べる (詳細は [2] を参照) .

BKZ 基底簡約の出力基底の見積もり

格子基底簡約アルゴリズムが出力する簡約基底の「良さ」を測る指標として Hermite 因子がある．d 次元格子 $L \subset \mathbb{R}^d$ の基底が与えられたとき，基底簡約アルゴリズムが出力する最短な基底ベクトル（多くの場合は第 1 基底ベクトル）を $\boldsymbol{b} \in L$ とする．このとき，その基底簡約アルゴリズムの **Hermite 因子**（Hermite factor）を

$$\gamma \coloneqq \frac{\|\boldsymbol{b}\|}{\mathrm{vol}(L)^{1/d}}$$

と定める．これは，Hermite 因子が小さいほど，その基底簡約アルゴリズムはより短い基底ベクトルを出力することを意味する．100 以上の高次元のランダム格子に対して，LLL や BKZ などの基底簡約アルゴリズムの Hermite 因子の d 乗根 $\gamma^{1/d}$（つまり，root Hermite 因子）は定数に収束することが実験的に知られている [38]．特に，高い次元 d のランダム格子において，高いブロックサイズ $\beta \geq 50$ に対する BKZ 基底簡約の root Hermite 因子はおおよそ

$$\gamma^{\frac{1}{d}} \approx \left(\omega_\beta^{-\frac{1}{\beta}}\right)^{\frac{1}{\beta-1}} \approx \left(\frac{\beta}{2\pi e}(\pi\beta)^{\frac{1}{\beta}}\right)^{\frac{1}{2(\beta-1)}} =: \delta_\beta \tag{4.10}$$

に従うことが実験的に知られている [26, 78]．ただし，ω_β は β 次元の単位球の体積とする（式 (4.8) を参照）．例えば，$\beta = 85$ で $\gamma^{1/d} \approx 1.01$ となる．この root Hermite 因子の見積もりを用いて，格子に基づく暗号技術の安全性評価対象の格子問題の求解で必要となる BKZ のブロックサイズ β を求めることができる．より具体的には，$\beta \geq 50$ かつ $\beta \ll d$ を満たす大きなブロックサイズ β に対して，d 次元のランダム格子 L の β-BKZ 簡約基底 $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d\}$ の Gram-Schmidt ベクトル $\boldsymbol{b}_1^\star, \ldots, \boldsymbol{b}_d^\star$ のノルムはおおよそ

$$\|\boldsymbol{b}_i^\star\| \approx \delta_\beta^{d-2i+1} \cdot \mathrm{vol}(L)^{\frac{1}{d}} \quad (i = 1, 2, \ldots, d) \tag{4.11}$$

に従うことが実験的に知られている（例えば，文献 [4, 25, 26, 78] を参照）．ただし，δ_β は式 (4.10) の値とする．また，これは Gram–Schmidt ベクトルの対数ノルム $\log \|\boldsymbol{b}_i^\star\|$ $(i = 1, 2, \ldots, d)$ が直線上に並ぶという**幾何級数仮定**（Geometric Series Assumption, GSA）[70] と Gauss のヒューリスティックの下で得られる結果である（実際には，後半の添え字 $i \approx d$ に対して，$\boldsymbol{b}_i^\star$ のノルムは式 (4.11) には従わない．詳細は文献 [4, 78] を参照）.

BKZ 基底簡約の計算量の見積もり

ブロックサイズ β を利用する際の BKZ 基底簡約の計算量は，β 次元の（ブロック射影）格子上の「SVP オラクルの計算量」と「呼び出し回数」の積で見積もることができる．β 次元格子上の SVP オラクルに適したアルゴリズムとして篩 (sieving) と数え上げ (enumeration) があり，数え上げアルゴリズムは β に関して超指数時間の処理コストであるのに対し，篩アルゴリズムは指数時間の処理コストであり漸近的に数え上げアルゴリズムよりも効率的である．（ただし，数え上げアルゴリズムの空間計算量が β に関して多項式的であるのに対し，篩アルゴリズムの空間計算量は β に関して指数関数的である．）具体的には，β 次元格子上の篩アルゴリズムの時間計算量は

$$2^{c\beta+o(\beta)}$$

で，Locally Sensitive Hashing（LSH）技術を利用した改良により [15, 49]，古典計算機上では $c = 0.292$ である（dimension-for-free の技術を用いなければ，実用上，隠れた準指数部分の因子は 1 以上である．詳細は [32, 58] を参照）．また，多くの篩アルゴリズムは Grover の探索アルゴリズムにより高速化されるため [48, 50]，量子計算機上では $c = 0.265$ と見積もられる（ただし，実用的に量子高速化が可能かは依然として根拠が弱い．詳細は [6] を参照）．一方，数え上げアルゴリズムの時間計算量は古典計算機上で

$$2^{c_1\beta\log\beta+c_2\beta+c_3} \quad \text{または} \quad 2^{c_1\beta^2+c_2\beta+c_3}$$

で，Grover の探索アルゴリズムにより量子計算機上ではその指数部分が半分になると見積もられる（定数 c_1, c_2, c_3 に関しては様々な評価値があり，具体的な値については [2, Table 4] を参照）．一方で，BKZ 基底簡約のアルゴリズム内の SVP オラクルの呼び出し回数については，入力するブロックサイズの β，または入力する格子次元 d に対し $8d$ と見積もることがある（例えば，文献 [1] を参照）．また，最小回数である 1 回の β 次元の SVP アルゴリズムの計算量困難性を**コア SVP 困難性**（Core-SVP hardness）とよび [10]，攻撃者に有利な条件設定で格子に基づく暗号方式の安全性を評価・比較することが多い（コア SVP 困難性による解析の正当性は，文献 [7] 内の解析と実験から検証済み）．

4.3 q-ary 格子と LWE 問題の求解法

本節では，$\mathbb{Z}_q$ 上の LWE 問題を解くための特殊な q-ary 格子について述べると共に，q-ary 格子を用いた LWE 問題の求解法についてまとめておく（q-ary 格子の詳細については，[16] を参照）．

4.3.1 q-ary 格子

奇素数 q に対して，$q\mathbb{Z}^m \subseteq L \subseteq \mathbb{Z}^m$ を満たす完全階数の m 次元格子 L を **q-ary 格子**（q-ary lattice）とよぶ．

2 つの自然数 ℓ, m に対して，$\ell \times m$ の整数行列 $\boldsymbol{M} \in \mathbb{Z}^{\ell \times m}$ に対する 2 つの m 次元 q-ary 格子を

$$\begin{aligned} \Lambda_q(\boldsymbol{M}) &\coloneqq \left\{ \boldsymbol{y} \in \mathbb{Z}^m : \exists \boldsymbol{s} \in \mathbb{Z}^\ell \text{ s.t. } \boldsymbol{y} \equiv \boldsymbol{s}\boldsymbol{M} \pmod q \right\}, \\ \Lambda_q^\perp(\boldsymbol{M}) &\coloneqq \left\{ \boldsymbol{y} \in \mathbb{Z}^m : \boldsymbol{y}\boldsymbol{M}^\top \equiv \boldsymbol{0} \pmod q \right\} \end{aligned} \tag{4.12}$$

と定める（詳細は [16] を参照）．これらの 2 つの集合は，共に $\mathbb{R}^m$ の離散加法部分群なので，格子である．正規化の差を除き，これら 2 つの q-ary 格子は互いに双対の関係にある．より正確には，

$$\Lambda_q^\perp(\boldsymbol{M}) = q\widehat{\Lambda_q(\boldsymbol{M})}, \quad \Lambda_q(\boldsymbol{M}) = q\widehat{\Lambda_q^\perp(\boldsymbol{M})} \tag{4.13}$$

が成り立つ．また，群準同型写像

$$f : \mathbb{Z}^m \longrightarrow (\mathbb{Z}/q\mathbb{Z})^\ell, \quad \boldsymbol{y} \longmapsto \boldsymbol{y}\boldsymbol{M}^\top \bmod q \tag{4.14}$$

の核は q-ary 格子 $\Lambda_q^\perp(\boldsymbol{M})$ である．ここで，群の準同型定理から

$$\mathrm{vol}(\Lambda_q^\perp(\boldsymbol{M})) = [\mathbb{Z}^m : \Lambda_q^\perp(\boldsymbol{M})] = \#\mathrm{Im}(f)$$

が成り立つ（$\#S$ は集合 S の要素数とする）．ただし，群の指数 $[\mathbb{Z}^m : \Lambda_q^\perp(\boldsymbol{M})]$ は格子の体積の比

$$\frac{\mathrm{vol}(\Lambda_q^\perp(\boldsymbol{M}))}{\mathrm{vol}(\mathbb{Z}^m)} = \mathrm{vol}(\Lambda_q^\perp(\boldsymbol{M}))$$

に一致することに注意する．これより，$\mathrm{Im}(f)$ は $(\mathbb{Z}/q\mathbb{Z})^\ell$ の部分群なので，体積 $\mathrm{vol}\left(\Lambda_q^\perp(\boldsymbol{M})\right)$ は q^ℓ を割る．また，元の格子と双対格子の体積の関係式 (4.6) から，$q^{m-\ell}$ は体積 $\mathrm{vol}\left(\Lambda_q(\boldsymbol{M})\right)$ を割ることが分かる（式 (4.13) の双対関係に注意）．さらに，ほとんどの多くの行列 $\boldsymbol{M}$ に対して，式 (4.14) の群準同型写像 f は全射で，その場合は

$$\mathrm{vol}\left(\Lambda_q^\perp(\boldsymbol{M})\right) = q^\ell, \quad \mathrm{vol}\left(\Lambda_q(\boldsymbol{M})\right) = q^{m-\ell}$$

が成り立つ．一方，q-ary 格子 $\Lambda_q(\boldsymbol{M})$ 上の任意のベクトルは $\boldsymbol{y} = \boldsymbol{s}\boldsymbol{M} + q\boldsymbol{z}$ $(\exists \boldsymbol{s} \in \mathbb{Z}^\ell, \exists \boldsymbol{z} \in \mathbb{Z}^m)$ とかけるので，その格子は $(\ell + m) \times m$ の整数行列

$$\begin{pmatrix} \boldsymbol{M} \\ q\boldsymbol{I}_m \end{pmatrix} \in \mathbb{Z}^{(\ell+m)\times m}$$

の一次従属な $(\ell + m)$ 個の行ベクトルで生成される．この生成行列の Hermite Normal Form（HNF）を計算して一次独立なベクトルを求めることで，m 次元 q-ary 格子 $\Lambda_q(\boldsymbol{M})$ の基底行列 $\boldsymbol{B} \in \mathbb{Z}^{m\times m}$ が得られる．また，双対基底の性質から，もう片方の q-ary 格子 $\Lambda_q^\perp(\boldsymbol{M})$ の基底行列は

$$(q\boldsymbol{B}^{-1})^\top \in \mathbb{Z}^{m\times m}$$

で得られる（式 (4.13) の双対関係より，q 倍を乗じる必要がある）．

4.3.2 LWE 問題の格子問題への帰着

ここでは，$\mathbb{Z}_q$ 上の LWE 問題の求解のための格子問題への帰着について述べる．特に，後述の ML-KEM パラメータに対する攻撃計算量の評価のために，Module-LWE 問題を $\mathbb{Z}_q$ 上の LWE 問題に帰着し行列表示した式 (4.5) を考える．

Primal 攻撃：探索 LWE 問題に対する求解

探索 LWE 問題を，目標ベクトルがある格子ベクトルに近いという条件下での CVP である **BDD**（Bounded Distance Decoding）問題に帰着して解く方法を紹介する．

定義 4.3 (BDD 問題)**.** 格子 L と目標ベクトル $\boldsymbol{t}$ に対して，ある $0 < \mu \leq \frac{1}{2}$ が存在し

$$\mathrm{dist}(\boldsymbol{t}, L) \coloneqq \min_{\boldsymbol{v} \in L} \|\boldsymbol{t} - \boldsymbol{v}\| < \mu \lambda_1(L)$$

を満たすと仮定する．格子 L の基底が与えられたとき，目標ベクトル $\boldsymbol{t}$ に最も近い格子ベクトル $\boldsymbol{v} \in L$ を見つけよ．

次元 nk の $\mathbb{Z}_q$ 上の m 個の LWE サンプル $(\boldsymbol{A}, \boldsymbol{t}) \in \mathbb{Z}_q^{m \times nk} \times \mathbb{Z}_q^m$（式 (4.5) を参照）は，関係式

$$\boldsymbol{t} \equiv \boldsymbol{s}\boldsymbol{A}^\top + \boldsymbol{e} \pmod{q}$$

を満たすので，式 (4.3) の秘密ベクトル $\boldsymbol{s} \in \mathbb{Z}_q^{nk}$ を見つける探索 LWE 問題は，$\boldsymbol{t} \in \mathbb{Z}_q^m$ を目標ベクトルとする q-ary 格子 $\Lambda_q(\boldsymbol{A}^\top)$ 上の BDD 問題とみなせる．具体的には，

$$\boldsymbol{t} = \boldsymbol{s}\boldsymbol{A}^\top + \boldsymbol{e} + q\boldsymbol{w} \quad (\exists \boldsymbol{w} \in \mathbb{Z}^m) \tag{4.15}$$

と表した目標ベクトルに対して，q-ary 格子上のベクトルを

$$\boldsymbol{v} = \boldsymbol{s}\boldsymbol{A}^\top + q\boldsymbol{w} \in \Lambda_q\big(\boldsymbol{A}^\top\big)$$

とおくと，$\boldsymbol{t} - \boldsymbol{v} = \boldsymbol{e}$ が成り立つ．また，ノイズベクトル $\boldsymbol{e} \in \mathbb{Z}^m$ のすべての成分が中心が 0 で標準偏差が $\sigma > 0$ の離散分布からサンプルされた場合，そのノルムはおおよそ

$$\|\boldsymbol{e}\| \approx \sigma\sqrt{m}$$

と見積もれる．ゆえに，目標ベクトル $\boldsymbol{t}$ との距離がおおよそ $\sigma\sqrt{m}$ となる q-ary 格子 $\Lambda_q(\boldsymbol{A}^\top)$ 上の格子ベクトル $\boldsymbol{v}$ を見つけることで，ノイズベクトル $\boldsymbol{e}$ を復元することができる．また，ノイズベクトル $\boldsymbol{e}$ が復元できた場合，$\boldsymbol{t} - \boldsymbol{e} \equiv \boldsymbol{s}\boldsymbol{A}^\top \pmod{q}$ の関係式にガウスの消去法を適用すれば，秘密ベクトル $\boldsymbol{s} \in \mathbb{Z}_q^{nk}$ を見つけることができる．

一方，BDD 問題の求解には，CVP を SVP に帰着する Kannan や Bai-Galbraith らの**埋め込み法**（embedding techniques）[45, 14] が実用的である．特に，ML-KEM に対しては，秘密ベクト

ル $\boldsymbol{s} \in \mathbb{Z}_q^{nk}$ のノルムが非常に短いので，Bai-Galbraith の埋め込み法 [14] が有効である．具体的には，$d = m + nk + 1$ 次の正方行列

$$\boldsymbol{B} = \begin{pmatrix} \boldsymbol{O}_{m,nk} & q\boldsymbol{I}_m & \boldsymbol{0}_m^\top \\ \boldsymbol{I}_{nk} & \boldsymbol{A}^\top & \boldsymbol{0}_{nk}^\top \\ \boldsymbol{0}_{nk} & -\boldsymbol{t} & 1 \end{pmatrix} \in \mathbb{Z}^{d \times d} \tag{4.16}$$

を考える．ただし，$\boldsymbol{O}_{m,\ell}$ は $m \times \ell$ の零行列，$\boldsymbol{0}_\ell$ はすべての成分が 0 の長さ ℓ の行ベクトルとする．ここで，$\boldsymbol{B}$ の d 個の行ベクトルで生成される d 次元の格子を L とする．特に，L の基底行列の構成から，格子 L の体積は

$$\mathrm{vol}(L) = |\det(\boldsymbol{B})| = q^m$$

である．このとき，式 (4.15) より，

$$\boldsymbol{z} = \big(\boldsymbol{s} \mid -\boldsymbol{e} \mid 1\big) = \big(\boldsymbol{w} \mid \boldsymbol{s} \mid 1\big)\boldsymbol{B} \in L \tag{4.17}$$

である．つまり，秘密ベクトルとノイズベクトルを連結した非常に短いベクトル $\boldsymbol{z} \in \mathbb{Z}^d$ が，格子 L の非零な最短ベクトルとして埋め込まれる．（式 (4.7) から，L がランダムな格子であれば $\lambda_1(L) = O\left(q^{m/d}\right)$ と予想されるが，q が十分大きく，特に $\|\boldsymbol{z}\| \ll q^{m/d}$ の場合，$\boldsymbol{z}$ が格子 L の非零な最短ベクトルと予想される．）そこで，BKZ 基底簡約などの SVP を解く格子アルゴリズムを利用して，$\boldsymbol{z} \in L$ を復元することができれば，秘密ベクトル $\boldsymbol{s}$ とノイズベクトル $\boldsymbol{e}$ を同時に見つけることができる．このように，秘密ベクトル $\boldsymbol{s}$ を見つける探索 LWE 問題を，BDD 問題に帰着させたのちに，埋め込み法で解く方法は **primal 攻撃**とよばれる（例えば，文献 [2] を参照）．

注意 4.1. 近年，機械学習を利用した（$\mathbb{Z}_q$ 上の）LWE 問題に対する攻撃法とその改良法がいくつか提案されている [54, 72, 74]．ただし，それらの機械学習を利用した攻撃法は，成分が疎かつ小さい秘密ベクトルを持つ LWE 問題にのみに有効で，秘密ベクトルの成分（または係数）が中心二項分布 CBD_η でサンプリングされる ML-KEM には現状では脅威とはならない（成分が疎かつ小さい秘密ベクトルを持つ LWE インスタンスに対する機械学習を利用した攻撃を含む各種攻撃のベンチマーク実験については，文献 [75] を参照）．

Dual 攻撃：判定 LWE 問題に対する求解

次は，判定 LWE 問題を **SIS**（Short Integer Solution）問題に帰着して解く方法を紹介する．

定義 4.4 (SIS 問題)**.** 奇素数 q と $0 < \xi < q$ を満たす（小さい）実数 ξ を固定する．すべての成分が $\mathbb{Z}_q$ 上一様ランダムに選ばれた $\ell \times m$ 整数行列 $\boldsymbol{M}$ に対して，

$$\|\boldsymbol{x}\| \le \xi \quad \text{かつ} \quad \boldsymbol{x}\boldsymbol{M}^\top \equiv \boldsymbol{0} \pmod{q}$$

を満たす非零ベクトル $\boldsymbol{x} \in \mathbb{Z}^m$ を見つけよ．これは，q-ary 格子 $\Lambda_q^\perp(\boldsymbol{M})$ 上の短い非零ベクトルを見つける問題と言い換えることができる（具体的な q-ary 格子の構成については，式 (4.12) を参照）．

次元 nk の $\mathbb{Z}_q$ 上の m 個の LWE サンプル $(\boldsymbol{A}, \boldsymbol{t}) \in \mathbb{Z}_q^{m \times nk} \times \mathbb{Z}_q^m$（式 (4.5) を参照）に対して，$nk \times m$ の転置行列 $\boldsymbol{A}^\top$ に対する SIS 問題の十分短い解ベクトル

$$\boldsymbol{x} \in \Lambda_q^\perp\big(\boldsymbol{A}^\top\big) \iff \boldsymbol{x}\boldsymbol{A} \equiv \boldsymbol{0} \pmod{q}$$

が得られたとする．このとき，行列表示の m 個の LWE サンプル $(\boldsymbol{A}, \boldsymbol{t})$ は $\boldsymbol{t} \equiv \boldsymbol{s}\boldsymbol{A}^\top + \boldsymbol{e} \pmod{q}$ を満たすので，

$$\begin{aligned} \langle \boldsymbol{x}, \boldsymbol{t} \rangle &\equiv \langle \boldsymbol{x}, \boldsymbol{s}\boldsymbol{A}^\top + \boldsymbol{e} \rangle \\ &= \langle \boldsymbol{x}\boldsymbol{A}, \boldsymbol{s} \rangle + \langle \boldsymbol{x}, \boldsymbol{e} \rangle \equiv \langle \boldsymbol{x}, \boldsymbol{e} \rangle \pmod{q} \end{aligned} \tag{4.18}$$

が成り立つ．ここで，ノイズベクトル $\boldsymbol{e}$ のすべての成分が中心が 0 で標準偏差が $\sigma > 0$ の離散分布からサンプルされたとすると，

$$z = \langle \boldsymbol{x}, \boldsymbol{t} \rangle \bmod q = \langle \boldsymbol{x}, \boldsymbol{e} \rangle \in \mathbb{Z}_q$$

は標準偏差が σm の離散分布に従う．一方，組 $(\boldsymbol{A}, \boldsymbol{t}) \in \mathbb{Z}_q^{m \times nk} \times \mathbb{Z}_q^m$ が，LWE サンプルでなく $\mathbb{Z}_q$ 上一様ランダムにサンプルされたものであれば，$z = \langle \boldsymbol{x}, \boldsymbol{t} \rangle \bmod q$ は $\mathbb{Z}_q$ 上一様分布に従う．これより，判定 LWE 問題に対して，

$$\varepsilon = 4\exp\left(-2\pi^2\tau^2\right) \quad \left(\tau = \frac{\sigma m}{q}\right) \tag{4.19}$$

の advantage を持つ．このような判定 LWE 問題に対する解法は **dual 攻撃**とよばれる（例えば，文献 [2] を参照）．

primal 攻撃における Bai-Galbraith 埋め込み法のように，秘密ベクトル $\boldsymbol{s}$ が短い場合は（例えば，ML-KEM 方式に対しては），$(m + nk) \times nk$ の整数行列

$$\boldsymbol{A}' = \begin{pmatrix} \boldsymbol{A} \\ -\boldsymbol{I}_{nk} \end{pmatrix}$$

とおき，q-ary 格子

$$\Lambda' = \Lambda_q^\perp(\boldsymbol{A}'^\top) = \big\{(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{Z}^m \times \mathbb{Z}^{nk} \mid \boldsymbol{x}\boldsymbol{A} \equiv \boldsymbol{y} \pmod{q}\big\}$$

を考えるのが有用である．実際，q-ary 格子 Λ' の基底行列に十分大きなブロックサイズ β の BKZ 基底簡約を適用し，短い非零ベクトル $\boldsymbol{b} = (\boldsymbol{x}, \boldsymbol{y}) \in \Lambda'$ を見つけたとする（一般に，$\boldsymbol{b}$ として，Λ' の β-BKZ 簡約基底の第 1 基底ベクトル $\boldsymbol{b}_1$ をとる）．このとき，式 (4.18) と同じように，

$$\begin{aligned} \langle \boldsymbol{x}, \boldsymbol{t} \rangle &\equiv \langle \boldsymbol{x}, \boldsymbol{s}\boldsymbol{A}^\top + \boldsymbol{e} \rangle \\ &= \langle \boldsymbol{x}\boldsymbol{A}, \boldsymbol{e} \rangle + \langle \boldsymbol{x}, \boldsymbol{e} \rangle \equiv \langle \boldsymbol{y}, \boldsymbol{s} \rangle + \langle \boldsymbol{x}, \boldsymbol{e} \rangle \pmod{q} \end{aligned}$$

が成り立つ．これより，秘密ベクトル $\boldsymbol{s}$ とノイズベクトル $\boldsymbol{e}$ のすべての成分が中心が 0 で標準偏差 $\sigma > 0$ の離散分布からサンプルされたとすると，上式から

$$|\langle \boldsymbol{x}, \boldsymbol{t} \rangle \bmod q| = |\langle \boldsymbol{x}, \boldsymbol{e} \rangle + \langle \boldsymbol{y}, \boldsymbol{s} \rangle| \lessapprox \sigma\ell \tag{4.20}$$

が成り立つ．ただし，$\ell = \|\boldsymbol{b}\| = \|(\boldsymbol{x}, \boldsymbol{y})\|$ とする．一方，4.3.1 項で述べた q-ary 格子の性質から，q-ary 格子 Λ' の次元は m で，体積は q^{kn} なので，式 (4.11) から

$$\ell = \|\boldsymbol{b}\| \approx \delta_\beta^{m+kn-1} \cdot q^{\frac{kn}{m+kn}}$$

と見積もれる．式 (4.19) と (4.20) より，ε の advantage をもつ攻撃者は，不等式

$$-2\pi^2\tau^2 \geq \ln\left(\frac{\varepsilon}{4}\right) \quad \left(\tau = \frac{\sigma\ell}{q}\right)$$

を満たす（最小の）ブロックサイズ β を入力とする BKZ 基底簡約の計算時間を必要とする．実際の攻撃では少なくとも $\frac{1}{2}$ の advantage が必要なため，攻撃者はおよそ $\frac{1}{\varepsilon^2}$ 個の Λ' の短いベクトルを生成して，攻撃の成功確率を増幅させる必要がある．特に，篩アルゴリズムでは $2^{0.2075\beta}$ 個のベクトルを生成するので，攻撃者は少なくとも

$$R = \max\left\{1,\ \frac{1}{2^{0.2075\beta}\varepsilon^2}\right\}$$

回の繰り返しを必要とする．ただし，篩アルゴリズムが出力するすべての格子ベクトルが非零な最短ベクトルと同程度に短いという保守的な（攻撃者に有利な）仮定の下での議論である（詳細は文献 [13, Section 5.1.3] を参照）．

注意 4.2. 成分が $0, \pm 1$ のいずれかの秘密ベクトル $\boldsymbol{s}$ を持つ（$\mathbb{Z}_q$ 上の）LWE 問題に対して，2021 年に May [60] は格子アルゴリズムと中間一致攻撃を組み合わせた求解法を提案した．具体的には，LWE 関係式 $\boldsymbol{t} \equiv \boldsymbol{s}\boldsymbol{A}^\top + \boldsymbol{e} \pmod q$ において，秘密ベクトルを 2 分割 $\boldsymbol{s} = (\boldsymbol{s}_1, \boldsymbol{s}_2)$ して，

$$\boldsymbol{t} - \boldsymbol{s}_1\boldsymbol{A}_1^\top \equiv \boldsymbol{s}_2\boldsymbol{A}_2^\top + \boldsymbol{e} \pmod q \tag{4.21}$$

を考える．ただし，$\boldsymbol{A}_1, \boldsymbol{A}_2$ は，秘密鍵ベクトルの分割 $\boldsymbol{s} = (\boldsymbol{s}_1, \boldsymbol{s}_2)$ に合わせて公開行列 $\boldsymbol{A}$ を分割した行列とする．上式の両辺に対して，中間一致攻撃を適用して，$(\boldsymbol{s}_1, \boldsymbol{s}_2)$ を探す（この求解法の進展として，primal 攻撃との組み合わせは文献 [41]，dual 攻撃との組み合わせは文献 [17] などを参照）．中間一致攻撃との組み合わせによる求解法は，成分が疎かつ（$0, \pm 1$ のような）小さい秘密ベクトル $\boldsymbol{s}$ を持つ LWE 問題にしか有効ではなく，$\boldsymbol{s}$ のすべての成分が中心二項分布 CBD_η ($\eta \in \{2, 3\}$) でサンプリングされる ML-KEM 方式には有効ではない（具体的には，ML-KEM の秘密ベクトル $\boldsymbol{s}$ の数え上げの計算量について，後述の 5.1 節を参照）．

4.4 ML-KEM パラメータに対する攻撃計算量の見積もり

本節では，ML-KEM の暗号パラメータ（2.2.3 項）に対する攻撃計算量を見積もる．文献 [8, 19] で述べられているように，$\mathbb{Z}_q$ 上の LWE 問題に対して様々な攻撃法がある．しかし，ML-KEM では，式 (2.16) の暗号文の形から，得られる $\mathbb{Z}_q$ 上の LWE サンプル数が最大 $(k+1)n$ なので，非常に多くの LWE サンプル数を必要とする BKW 型攻撃 [47] と線形攻撃 [12] を除外することができる．これより，ML-KEM に対しては，本質的には 4.3.2 項で説明した primal 攻撃・dual 攻撃（と BKZ 基底簡約による求解との組み合わせ）の 2 つの攻撃法だけが対象となる．

4.4.1 コア SVP 困難性による攻撃計算量の見積もり

ここでは，ML-KEM の暗号パラメータに対して，攻撃で必要となる BKZ 基底簡約のブロックサイズ β を見積もるとともに，BKZ 基底簡約内の 1 回の β 次元の SVP アルゴリズムのコア SVP 困難性 [10] による攻撃計算量を見積もる．また，文献 [13, Section 5.2] で議論されているように，コア SVP 困難性による見積もりでは dual 攻撃の方が primal 攻撃よりほんの少し計算量が低くなるが，実際にはより多くの攻撃計算量を要する．具体的には，上述の dual 攻撃において，篩アルゴリズムで指数関数的に多くの格子ベクトルを生成できると仮定しているが，それらの多くは $\sqrt{4/3}$ 倍程度長い．さらに，指数関数的に多くの短い格子ベクトルを生成できるという仮定は，篩アルゴリズムに関する近年の改良と整合性が取れない（例えば，dimension-for-free 改良 [32] など）．一方，これらの余分な短い格子ベクトルを仮定しない解析 [2] では，primal 攻撃よりも dual 攻撃の方がかなり計算コストがかかると予想している．そのため，以下では primal 攻撃の計算量についてのみ議論する．

4.3.2 項で説明した primal 攻撃では，式 (4.17) の形の LWE 問題の秘密とノイズの連結ベクトル $\boldsymbol{z}$ を，体積が q^m で次元が $d = m + nk + 1$ の Bai-Galbraith 埋め込み格子 L の最短ベクトルとして埋め込む．また，primal 攻撃では，格子 L の基底に BKZ 基底簡約アルゴリズムを適用することで，最短ベクトル $\boldsymbol{z}$ を見つけることを考える．ここで，$\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d\}$ を格子 L の β-BKZ 簡約基底とし，$\{\boldsymbol{b}_1^\star, \ldots, \boldsymbol{b}_d^\star\}$ をその Gram-Schmidt ベクトルとする．GSA と Gauss のヒューリスティックから，各 Gram-Schmidt ベクトルのノルムについて，式 (4.11) が成り立つとする（これは攻撃者に有利なモデルで，かなり楽観的な仮定である）．このとき，目的の最短ベクトル $\boldsymbol{z}$ の $d - \beta + 1$ の位置における射影ベクトル $\pi_{d-\beta+1}(\boldsymbol{z}) \in \pi_{d-\beta+1}(L)$ のノルムが

$$\begin{aligned} \sigma\sqrt{\beta} \approx \|\pi_{d-\beta+1}(\boldsymbol{z})\| \leq \|\boldsymbol{b}_{d-\beta+1}^\star\| \approx \delta_\beta^{2\beta-d-1} \cdot \mathrm{vol}(L)^{1/d} \\ \iff \sigma\sqrt{\beta} \leq \delta_\beta^{2\beta-d-1} \cdot q^{m/d} \end{aligned} \tag{4.22}$$

を満たせば，BKZ 基底簡約の第 1 基底ベクトルとして目的の $\boldsymbol{z} \in L$ を見つけることができる（探索 LWE 問題に対する BKZ による求解実験については，文献 [4, 7, 65] を参照）．ただし，式 (4.17) の形の $\boldsymbol{z}$ のノルムは

$$\|\boldsymbol{z}\| \approx \sigma\sqrt{kn + m} \approx \sigma\sqrt{d} \tag{4.23}$$

と見積もれ，その射影ベクトル $\pi_{d-\beta+1}(\boldsymbol{z})$ のノルムは，文献 [7, Section 4.1] から

$$\|\pi_{d-\beta+1}(\boldsymbol{z})\| \approx \sqrt{\frac{\beta}{d}} \cdot \|\boldsymbol{z}\| \approx \sigma\sqrt{\beta}$$

と見積もれる．また，δ_β は式 (4.10) の値とする．不等式 (4.22) を満たす最小の β が，primal 攻撃が成功する最小の BKZ 基底簡約のブロックサイズと期待される．

表 4.1 に，ML-KEM の 3 つの暗号パラメータ（2.2.3 項）に対して，不等式 (4.22) を満たす primal 攻撃に必要な最小の BKZ ブロックサイズ β と，BKZ 基底簡約のサブルーチンである β 次

表 4.1　ML-KEM の安全性を支える Module-LWE 問題に対するコア SVP による攻撃計算量見積もり [13, Table 4]（帰着する $\mathbb{Z}_q$ 上の LWE サンプル $(\boldsymbol{A}, \boldsymbol{b}) \in \mathbb{Z}_q^{m \times nk} \times \mathbb{Z}_q^m, \boldsymbol{b} = \boldsymbol{s}\boldsymbol{A}^\top + \boldsymbol{e}$ に対して，d 次元の Bai-Galbraith 埋め込み格子を用いた primal 攻撃の計算量見積もり）

ML-KEM パラメータ（$n \times k$ の値）	512	768	1024
$\boldsymbol{s}, \boldsymbol{e}$ に関する中心二項分布 CBD_{η_1} の η_1	$\eta_1 = 3$	$\eta_1 = 2$	$\eta_1 = 2$
攻撃に利用する格子次元 $d = m + nk + 1$	999	1419	1885
攻撃に必要な BKZ の最小ブロックサイズ β	406	626	878
コア SVP の古典計算量（ビット）	118	183	256
コア SVP の量子計算量（ビット）	107	166	232

元 SVP アルゴリズムの 1 回の計算量であるコア SVP 困難性の古典と量子の計算量見積もりをまとめる（詳細は [13, Table 4] を参照）．ただし，攻撃に必要なサンプル数 m，BKZ 基底簡約の最小ブロックサイズ β，コア SVP 困難性の古典計算量と量子計算量は下記のように見積もる．

- ML-KEM では，秘密ベクトル $\boldsymbol{s}$ と公開鍵のノイズベクトル $\boldsymbol{e}$ のすべての $\mathbb{Z}_q$ 係数は中心二項分布 CBD_{η_1} からサンプリングされる．具体的には，ML-KEM-512 の場合は $\eta_1 = 3$，ML-KEM-768, 1024 の場合は $\eta_1 = 2$ で，式 (4.23) に対して中心二項分布 CBD_{η_1} の標準偏差は $\sigma = \sqrt{\eta_1/2}$ である（つまり，分散は $\sigma^2 = \eta_1/2$ である）．特に，ML-KEM-512 の場合，暗号文におけるノイズ $\boldsymbol{e}_1, e_2$ は $\eta_2 = 2$ による CBD_{η_2} からサンプリングされるが，Compress 関数により暗号文におけるノイズは暗黙的に増幅するので，ここでは η_1 の値にのみ着目すればよい（参考程度であるが，$\eta_2 = 2$ でコア SVP の古典計算量を算出すると 112 ビットとなり，表 4.1 内の 118 ビットよりも 6 ビット下がる）．
- primal 攻撃に利用するサンプル数 m と不等式 (4.22) を満たす BKZ 基底簡約のブロックサイズ β の最良の組 (m, β) は，primal 攻撃が最も有効となるサンプル数 m を 1 から $(k+1)n$ の中から求めた上で，不等式 (4.22) を満たす最小のブロックサイズ β を求める．具体的には，GitHub：`https://github.com/pq-crystals/security-estimates` 内の Kyber.py コードから求まる（コード内では，dual 攻撃に必要な BKZ 基底簡約のブロックサイズも求めている）．
- また，BKZ 基底簡約のサブルーチンである β 次元におけるコア SVP 困難性（篩アルゴリズム）の計算量として，4.2.3 項から，古典計算機で $2^{0.292\beta}$，量子計算機で $2^{0.265\beta}$ と見積もる（これは攻撃者にかなり有利な計算量見積もりである）．

図 4.1 に，表 4.1 の ML-KEM の暗号パラメータに対するコア SVP の計算量見積もりの検証用 Sage コードを示す（Sage Math Cell：`https://sagecell.sagemath.org/` 上で動作可能）．具体的には，ML-KEM に関する暗号パラメータ (n, k, q, η_1) と表 4.1 にある攻撃に利用する最良のサンプル数 m の値を代入すれば，攻撃に必要な BKZ の最小ブロックサイズ β と，その時のコア SVP の古典計算量と量子計算量を算出する．ただし，$n \times k = 768, 1024$ の場合，不等式 (4.22) を

図 4.1　表 4.1 のコア SVP 困難性の計算量見積もりの検証用 Sage コード

```
k = 2; n = 256; q = 3329; eta = 3
sigma = RR(sqrt(eta/2))
m = 486; d = m+k*n+1

for b in range(100, 1000):
    A = sigma*sqrt(b)
    delta = ((math.pi*b)^(1.0/b)*b/(2*math.pi*exp(1)))^(1.0/(2*(b-1)))
    B = delta^(2*b-d-1)*q^(m/d)
    if RR(A) < RR(B):
        print("攻撃に必要な最小ブロックサイズ = ", b)
        print("古典計算量（ビット）= ", 0.292*b)
        print("量子計算量（ビット）= ", 0.265*b)
        break
```

満たす primal 攻撃に必要な BKZ の最小ブロックサイズ β は，表 4.1 の β から -1 または -2 程度ずれるが，コア SVP の計算量には大きなずれはない．

4.4.2　最新の技術と解析による攻撃計算量の見積もり

ここでは，BKZ 基底簡約におけるシミュレーションおよび progressive 化や dimension-for-free[32] などの最新の技術の効果を考慮した，ML-KEM の安全性を支える Module-LWE 問題に対する primal 攻撃に必要となる BKZ のブロックサイズ β の再見積もりを行う．また，文献 [6] の解析に基づくゲートコストの見積もりを示す．まず，BKZ 基底簡約の progressive 化によるオーバーヘッド定数を

$$C = \lim_{\beta\to\infty} \frac{\sum_{i=0}^{\beta} 2^{0.292i}}{2^{0.292\beta}} = \frac{2^{0.292}}{2^{0.292}-1} \approx 5.46$$

と定める．以下では，ML-KEM-512 の暗号パラメータ（$nk = 512$）に対してのみ議論する．

BKZ シミュレーションの利用

Primal 攻撃の成功条件に関する不等式 (4.22) は，GSA 仮定下の BKZ 簡約基底の Gram-Schmidt ベクトルのノルム評価 (4.11) に依存する．しかし，実際の Gram-Schmidt ベクトル $\boldsymbol{b}_1^\star, \ldots, \boldsymbol{b}_d^\star$ では，後半の添え字 $i \approx d$ に対して，$\boldsymbol{b}_i^\star$ のノルムは式 (4.11) には必ずしも従わない [4, 78]．そこで，leaky-LWE-estimator [29] の一部のシミュレーターを使う．これは progressive-BKZ 基底簡約 [5, 11] を利用しており，このシミュレーターでは攻撃に必要な BKZ ブロックサイズとして

$$\beta = 413$$

が必要であると見積もれる．（正確には，$\beta = 412$ から $\beta = 413$ にすることで，コストが $2^{0.292} \approx 1.224$ 倍増幅する一方で，成功確率は 1.373 倍増幅する．）また，利用する格子次元は $d = 1025$ である．β 次元の SVP に対する篩アルゴリズムの計算コストと比べて，progressive-BKZ

基底簡約の計算コストはおおよそ

$$C \cdot (d - \beta) = 5.46 \times (1025 - 413) \approx 3340 \tag{4.24}$$

倍に増加する．

Dimension-for-free 技術の効果

文献 [32] の dimension-for-free（d_{4f}）技術の効果を考慮すると，BKZ ブロックサイズ $\beta = 413$ に対して

$$d_{4f} = \frac{\beta \ln(4/3)}{\ln(\beta/(2\pi e))} \approx 37.3$$

を得る（整数に切り上げることで，$d_{4f} = 38$ とする）．これより，BKZ 基底簡約内で呼び出される β 次元の SVP オラクルとして，

$$\beta' = \beta - d_{4f} = 375$$

次元の篩アルゴリズムを用いればよい．（文献 [5] の “on-the-fly lifting” と “pump-down sieves” の 2 つのトリックを用いると，上記の d_{4f} の値よりもわずかに多くの free 次元を得られる可能性がある．実用的には，メモリを $2^{0.5}$ 程度削減できるが，計算時間への影響は限定的である．）

篩アルゴリズムのゲートコスト

古典および量子回路による篩アルゴリズムの計算コストの最新の解析 [6] では，“AllPairSearch” 関数に着目している．具体的には，篩に関する球面キャップとくさびの正確な体積を求め，最内側の繰り返し関数に対する正確なゲートカウントを計算し，パラメータの自動最適化を行うことで，古典と量子の計算コストを得る．最良の古典アルゴリズムに関して，文献 [6] の解析では，$\beta' = 375$ 次元における AllPairSearch に対しては，おおよそ

$$2^{137.4}$$

ゲートのコストと結論づけている．素朴な篩アルゴリズムでは多項式回の AllPairSearch の呼び出しが必要であるが，progressive 型の篩アルゴリズム [32, 49] では，実用的には比較的少ない呼び出し回数で十分である．そこで，progressive 型の篩アルゴリズムにおいて，各次元ごとに 1 回の AllPairSearch 関数を呼ぶと仮定すると，$\beta' = 375$ 次元までで

$$C \cdot 2^{137.4} \tag{4.25}$$

ゲートのコストがかかる．

最終的なゲートコスト

式 (4.24) と (4.25) から，最終的なゲートコストとして

$$G = (1025 - 413) \cdot C^2 \cdot 2^{137.4} = 2^{151.5}$$

表 4.2　BKZ シミュレーション [29] と dimension-for-free（d_{4f}）技術 [32] を考慮した ML-KEM の安全性を支える Module-LWE 問題に対する primal 攻撃計算量の再見積もりと文献 [6] によるゲートコストとメモリの見積もり（文献 [13, Table 4] を参照）

ML-KEM パラメータ（nk の値）	512	768	1024
$\boldsymbol{s}, \boldsymbol{e}$ に関する中心二項分布 CBD_{η_1} の η_1	$\eta_1 = 3$	$\eta_1 = 2$	$\eta_1 = 2$
攻撃に利用する格子次元 d	1025	1467	1918
攻撃可能な BKZ の最小ブロックサイズ β	413	637	894
篩アルゴリズムの SVP 次元 $\beta' = \beta - d_{4f}$	375	586	829
攻撃に必要なゲートコスト（ビット）	151.5	215.1	287.3
攻撃に必要なメモリ（ビット）	93.8	138.5	189.7
NIST 標準で要求される安全性レベル [63]	143	207	272
（古典ゲート数換算，ビット）	AES-128 相当	AES-192 相当	AES-256 相当

を得る．篩アルゴリズムで利用する格子ベクトルの各成分を 1 バイトで表現できると仮定すると，文献 [6] の解析に従えば，必要なメモリを見積もることが可能である．ML-KEM における 3 つの暗号パラメータに対しては，`https://github.com/lducas/leaky-LWE-Estimator/tree/NIST-round3` のスクリプトから算出できる．

表 4.2 に，BKZ シミュレーションおよび progressive 化や dimension-for-free 技術を考慮した ML-KEM の安全性を支える Module-LWE 問題に対する primal 攻撃計算量の再見積もりと，文献 [6] の解析に従ったゲートコストとメモリの見積もりを示す（文献 [13, Table 4] を参照）．

注意 4.3 (理想的な近傍探索)**.** 文献 [6] の解析では，文献 [15] の篩におけるバケットが球上完全に一様に分布すると仮定している．しかしながら，文献 [15] の篩アルゴリズムは，バケットの分布に存在するある構造を利用しているに違いない．元の解析では，各ペアを見つける成功確率は，その構造により $2^{\widetilde{O}(\sqrt{\beta})}$ の準指数よりも大きな因子には影響しないことを示している．しかしながら，この漸近的な解析はタイトではなく，準備的な実験では，実用的には成功確率のロスはそれほど大きくないことを示唆している．さらに，パラメータに依存するが，文献 [6] の理想化に比べて，文献 [15] の篩アルゴリズムはオーバーヘッドを持つ．具体的には，メモリ使用量を最小化するには，時間計算量として $2^{O(\beta/\log\beta)}$ のオーバーヘッドが生じる．原則的に，メモリ使用量と時間計算量にトレードオフがある．

4.4.3　Dual-sieve 攻撃とその影響

文献 [40] では，$\mathbb{Z}_q$ 上の LWE 問題に対する dual 攻撃とヒューリスティックな推定ステップを組み合わせた改良手法が提案されており，dimension-for-free 技術 [32] の利用と単一の篩によって多数の短い格子ベクトルを生成する手法が示されている．この LWE 問題に対する攻撃手法は dual-sieve 攻撃と呼ばれる．その後，文献 [59] では，FFT（Fast-Fourier-Transformation）

に基づく識別法を利用した dual 攻撃の改良が提案され，ML-KEM（Kyber [13]）に対する攻撃計算量が評価されている．この FFT に基づく dual-sieve 攻撃は dual-sieve-FFT 攻撃と呼ばれ，量子アルゴリズムによる亜種 [9] や符号理論からのアイデアを取り入れた改良 [24]，さらに Module-LWE の代数構造を利用した改良 [76] などが提案されている．文献 [40, 59, 9, 24] で提案された dual-sieve-FFT 攻撃とその改良の実用性については，文献 [35] で理論と実験の両面で検証が行われ，dual-sieve-FFT 攻撃の成功確率は実際よりかなり高く見積もられていると結論付けている．

近年，文献 [23] では，符号理論のアイデアに基づく新しい dual-sieve 攻撃が提案されている．具体的には，文献 [59] で用いられていた modulus switching 技術を，効率的な復号アルゴリズムに置き換える dual-sieve-FFT 攻撃の改良である．この新しい dual-sieve-FFT 攻撃による ML-KEM（Kyber）-512, 768, 1024 の安全性レベル [13, Table 4]（表 4.2 を参照）は，耐量子計算機暗号の NIST 標準化が要求する安全性レベル 1, 3, 5 に対応うる古典ゲート計算量 143, 207, 272 ビット [63] よりも少なくともそれぞれ

$$3.5,\ 11.9,\ 12.3 \text{ ビット}$$

は下回ると主張されている（詳細は [23, Table 5.1] を参照）．ただし，文献 [23] における dual-sieve-FFT 攻撃の計算量評価は，最近傍探索に関する文献 [15] による理想的な理論モデルに基づく（注意 4.3 を参照）．また，文献 [33] で言及されているように，文献 [15] による理想的な理論モデルでは復号コストを過小評価し，積符号が持つレート-歪み特性の非最適性を考慮してないため，いくつかのオーバーヘッドが隠れている．具体的には，380 次元における篩アルゴリズムに対して，最近傍探索の実際の計算量は，文献 [15] による理想的な理論モデルより 2^6 倍程度増加する（詳細は文献 [33] を参照）．

4.5 Module-LWE 問題に対する代数構造を利用した攻撃とその影響

Module-LWE 問題に対する上記の攻撃アルゴリズムは，ML-KEM 方式の構成における基礎環 $R = \mathbb{Z}[X]/(X^n + 1)$ $(n = 256)$ に内在する代数構造を一切利用していない．本節では，代数構造を利用した Module-LWE 問題に対する攻撃とその影響について述べる．

4.5.1 加群格子上の格子アルゴリズム

円分体 $K = R \otimes_{\mathbb{Z}} \mathbb{Q} \cong \mathbb{Q}(\zeta_{2n})$ のイデアル格子上の篩アルゴリズムは，処理性能とメモリ使用量の両面において有効である．具体的には，メモリ使用量については自由 $\mathbb{Z}$ 加群 $R = \mathbb{Z}[X]/(X^n+1)$ の階数 n の因子分だけ削減できる．また，処理性能については，近傍探索テクニックを利用するかどうかに依存するものの，$O(n)$ から $O(n^2/\log n)$ の間の因子分の高速化が期待できる．これは，イデアル格子の対称性から，式 (4.1) の回転操作を利用すれば，1 つの格子ベクトル（つまり，R

の元 $f(x)$ の係数ベクトル $\boldsymbol{f} \in \mathbb{Z}^n$）から同じノルムを持つ n 個の格子ベクトル

$$\mathrm{rot}^i(\boldsymbol{f}) \quad (i = 1, 2, \ldots, n)$$

を効率的に生成することができる．また，加群格子上の数え上げアルゴリズムについても，その計算量は漸近的に改善されることが文献 [46] において示されている．

また，加群格子上の BKZ 基底簡約アルゴリズム [61] は存在し，内部で呼び出す SVP オラクルは加群格子の対称性の恩恵を受ける可能性がある．しかし，下記のように，このアプローチによる本質的な恩恵を妨げる問題が数多くある．

- まず，$\mathbb{Z}$ 階数 $r \geq 2$ の加群格子上の BKZ 基底簡約を，ML-KEM の基礎環 $R \cong \mathbb{Z}^n$ 上で適用するためには，$r \mid n = 256 = 2^8$ の条件を満たす必要がある（扱う加群の $\mathbb{Z}$ 階数 r が小さいほど，加群格子の対称性から得られる恩恵は限定的となる）．さらに，$\mathbb{Z}$ 階数 r の加群格子上の BKZ 基底簡約のブロックサイズ β は

  $$r = \gcd\bigl(\beta, 256\bigr)$$

  の条件を満たす必要がある．この制約により，攻撃に最適と考えられるブロックサイズ β を自由に選択することが困難となる．特に，ブロックサイズ選択の自由度が低いため，加群格子上の BKZ 基底簡約の progressive 化に支障をきたす．また，dimension-for-free 技術と組み合わせる場合には，

  $$r = \gcd\bigl(\beta, d_{4f}, 256\bigr)$$

  の条件を満たす必要があるため，ブロックサイズ β の選択に更なる制限が課される．加えて，Kannan の埋め込み法を $\mathbb{Z}$ 階数が r の加群構造に適用する際，特別な調整が必要となり，全体の格子次元は 1 次元ではなく，少なくとも r 次元分増加させる必要がある．
- 次に，同じブロックサイズ β を利用したとしても，通常（つまり $\mathbb{Z}$ 格子上）の BKZ と加群版の BKZ が同程度の品質を持つ基底を出力するかどうかは明らかではない．（文献 [34] では，非構造化格子よりも加群格子上の BKZ 基底簡約はより多くのブロックサイズを必要とすると結論付けている．）これは，$\beta = 2$ の BKZ 基底簡約である LLL 基底簡約の場合でさえ未解決である（詳細は [31, 52] を参照）．その主な理由は，一般に代数体の整数環は Euclid 整域ではないので，$\mathbb{Z}$ 上の LLL 基底簡約を加群上に一般化することは困難である．特に，2 次元のガウス基底簡約に対応するサイズ基底簡約で行う divide-and-swap アルゴリズムを加群上で行うことができない．また，加群格子上で LLL や BKZ の基底簡約アルゴリズムを実用的に動作させるための実装基盤も，現時点では十分に整備されていない．

以上のような数多くの問題点を踏まえると，文献 [56, Appendix C] で言及されている通り，Module-LWE 問題を実際に解く際には，代数構造を利用しない $\mathbb{Z}$ 上の BKZ 基底簡約を用いるのが現時点では最も有効である．

4.5.2 イデアル格子上の SVP に対する量子アルゴリズム

イデアル格子上の SVP に対する量子アルゴリズムが提案されている [37, 22, 18, 27, 28]. しかし，文献 [28] では，Ring-LWE に対する量子攻撃に向けた障壁について言及しているが，Module-LWE では更なる障壁が生じると言及している．また，文献 [3] では，Ring-LWE から Module-LWE への帰着を構築し，それはあるパラメータにおける Ring-LWE に対する多項式時間アルゴリズムは Module-LWE に対する攻撃に変換できることを示唆している．しかしながら，実用的な観点から，この攻撃は加群格子の次元が増加するごとにかなり処理性能（効率性）が下がる．つまり，加群格子の次元の増加が暗号方式の安全性を高めることを示唆している．特に，ML-KEM-768 にこの帰着を適用すると，非常に大きな剰余と誤差を持つ Ring-LWE を導き，攻撃者に 1 個以上のサンプルを要求する．文献 [56, Appendix C] で言及されているように，イデアル格子上の SVP に対する量子アルゴリズムは，ML-KEM を含む格子暗号方式に対する実用的な攻撃につながる可能性は低いと考えられる．

第 5 章

ML-KEM に特化した解析手法とその評価に関する調査結果

本章では，ML-KEM に特化した解析手法とその評価に関する調査結果をまとめる．具体的には，ML-KEM の安全性を支える秘密ベクトル $\boldsymbol{s}$ が中心二項分布 CDB_η からサンプルされる Module-LWE 問題に対する攻撃アルゴリズムとその影響について述べる．

5.1 ML-KEM における秘密ベクトルの数え上げ計算量

2.2.2 項で述べたように，ML-KEM の安全性を支える Module-LWE 問題では，秘密ベクトル $\boldsymbol{s} \in \mathbb{Z}_q^{nk}$ の各成分は中心二項分布 CBD_η $(\eta = 2, 3)$ からサンプルされる．文献 [39] では，式 (4.21) を用いる May の秘密ベクトル探索アルゴリズム [60] を拡張し，ML-KEM における秘密ベクトル $\boldsymbol{s} \in \mathbb{Z}_q^{nk}$ を数え上げる計算量を解析している．具体的には，

$$N = nk \in \{512, 768, 1024\}$$

に対して，すべての成分が中心二項分布 CBD_η からサンプルされた秘密ベクトル $\boldsymbol{s} \in \mathbb{Z}_q^N$ を数え上げる時間・空間計算量はともに，

$$\begin{array}{ll} O\big(2^{0.36N}\big) & (\eta = 2 \text{ の場合}), \\ O\big(2^{0.37N}\big) & (\eta = 3 \text{ の場合}) \end{array} \tag{5.1}$$

と評価されている [39, Table 12, Appendix A]．一方，4.3.2 項で説明した primal 攻撃では，攻撃に必要な BKZ 基底簡約のブロックサイズ β は，表 4.1 から，おおよそ

$$\beta \approx \frac{4}{5}N$$

と見積もられる．また，BKZ 基底簡約の内部で β 次元の SVP オラクルとして用いられる篩アルゴリズムの時間計算量がおおよそ

$$O\big(2^{0.292\beta}\big)$$

であることと比較すると，式 (5.1) の数え上げ計算量は大きい．以上より，文献 [39] で提案された秘密ベクトルの数え上げアルゴリズムが，BKZ 基底簡約の内部で呼び出す篩アルゴリズムより（漸近的に）有効になることはなく，表 4.2 の ML-KEM に対する攻撃計算量に実質的な影響は与えない．

第 6 章

ML-KEM の暗号強度に関する考察

本章では，ML-KEM の暗号強度に関する考察をまとめる．第 3 章で議論したように，ML-KEM の帰着仮定である共通プリミティブであるハッシュ関数・擬似ランダム関数やデカプセル化（復号）失敗確率を利用することが実用的に困難であるため，ML-KEM の暗号強度は Module-LWE 問題の攻撃計算量に依存する．以下では，ML-KEM の安全性を支える Module-LWE 問題の攻撃計算量について考察する．

6.1 ML-KEM の安全性を支える Module-LWE 問題の攻撃計算量

ML-KEM の安全性を支える Module-LWE 問題では，秘密 $\boldsymbol{s}(X) = (s_1(X), \ldots, s_k(X)) \in R_q^k$ の成分多項式 $s_i(X)$ $(i = 1, \ldots, k)$ とノイズ $e(X) \in R_q$ のすべての $\mathbb{Z}_q$ 係数は，中心二項分布 CBD_η $(\eta \in \{2, 3\})$ からサンプルされる．現時点では，このような Module-LWE 問題に対する最良の攻撃法は，4.1 節で述べた方法により $\mathbb{Z}_q$ 上の nk 次元の LWE 問題に帰着した後，BKZ 基底簡約などの $\mathbb{Z}$ 格子上のアルゴリズムを適用するものである．また，ML-KEM の暗号文の形 (2.16) から，攻撃に利用可能な Module-LWE サンプル数は最大 $(k+1)$ であり，これらを $\mathbb{Z}_q$ 上の LWE サンプルに帰着すると，その個数は最大 $(k+1)n$ である．このサンプル数の制限により BKW 型攻撃と線形攻撃は適用できず，$\mathbb{Z}_q$ 上の LWE 問題に対する primal 攻撃と dual 攻撃のみが有効となる．また，4.4 節で議論したように，表 2.1 の ML-KEM の暗号パラメータ $nk \in \{512, 768, 1024\}$ に対しては，primal 攻撃と BKZ 基底簡約（内部 SVP オラクルは篩アルゴリズム）の組み合わせが有効である．攻撃者に有利な観点から，BKZ の progressive 化とシミュレーション [29]，dimension-for-free 技術 [32] などの効果を考慮すると，ML-KEM-512, 768, 1024 を攻撃するために必要な BKZ の最小ブロックサイズ β はそれぞれ

$$\beta = 413,\ 637,\ 894$$

と見積もられる（表 4.2 を参照）．さらに，それらの β までの progressive 型の BKZ 基底簡約において，内部 SVP オラクルとして呼び出す篩アルゴリズムの最内部にある繰り返し関数の文献 [6]

の解析に基づくゲートコストはそれぞれ

$$G = 2^{151.5},\ 2^{215.1},\ 2^{287.3} \tag{6.1}$$

と見積もられる（表 4.2 を参照）．これは，耐量子計算機暗号の NIST 標準化 [63] の安全性レベル 1, 3, 5 でそれぞれ要求される古典ゲート数

$$2^{143},\ 2^{207},\ 2^{272} \tag{6.2}$$

を上回る（詳細は [63, §4.A.5] を参照）．

■篩アルゴリズムの解析の精密化・改良による影響　式 (6.1) のゲートコスト評価は，文献 [15] の篩アルゴリズムの計算量に依存する．注意 4.3 で述べた通り，文献 [6] の理想的な近傍探索に比べて，文献 [15] の篩アルゴリズムはオーバーヘッドをもつ．文献 [13, Section 5.3, Summary] で述べられているように，篩アルゴリズムの多角的な解析の精密化と，将来的に予想されるアルゴリズム的改良を考慮すると，式 (6.1) のゲートコスト評価は $2^{-16} \sim 2^{14}$ 倍程度ずれる可能性がある．最悪の場合，式 (6.1) のゲートコスト評価は NIST 標準化で要求される式 (6.2) の古典ゲート数を下回る可能性があるが，これはあくまで攻撃者に最も有利な条件下の評価に過ぎない．実際には，文献 [56, §4.1.1] で指摘されているように，篩アルゴリズムのメモリアクセスのコストを現実的に反映した条件下で，NIST 標準化で要求される式 (6.2) の古典ゲート数は維持されると考えられる．（近年，文献 [79] で，文献 [15] の篩アルゴリズムのメモリアクセスのコストを従来のおおよそ 40% に削減する改良が提案されている．これは実用的な改良で，文献 [13, Section 5.3, Summary] で予想されている改良の範囲に収まるものと思われる．）

■最新の dual-sieve 攻撃による影響　近年，文献 [23] で符号理論のアイデアに基づく新しい dual-sieve 攻撃が提案されている．この dual-sieve 攻撃により，耐量子計算機暗号の NIST 標準化が要求する式 (6.2) の古典ゲート数よりも少なくとも $2^{3.5}$, $2^{11.9}$, $2^{12.3}$ 下回ると主張している [23, Table 5.1]．ただし，文献 [23] による解析は理想的な理論モデルに基づき，いくつかのオーバーヘッドが隠れている．また，LWE チャレンジ [30] に対する大規模な解読などで理論と実験の両面で検証されている primal 攻撃に比べて，dual 攻撃の実用的な解析は進んでおらず，dual-sieve 攻撃による実用性を検証している文献 [35] で述べられているように，dual-sieve 攻撃の成功確率は実際よりかなり高く見積もられている．これより，文献 [23] が主張する攻撃計算量評価は実際よりかなり低く見積もられている可能性が高く，primal 攻撃による式 (6.1) のゲートコスト評価には影響しないと思われる．

■ML-KEM に特化した攻撃手法による影響　ML-KEM の秘密鍵 $\boldsymbol{s}$ の成分多項式のすべての $\mathbb{Z}_q$ 係数が中心二項分布 CBD_η からサンプルされることを利用した攻撃が文献 [39] で提案されている．5.1 節で述べたように，その攻撃の計算量は primal 攻撃よりも大きいため，primal 攻撃による式 (6.1) のゲートコスト評価には影響しない．

6.2 代数構造を利用した格子アルゴリズムの影響

式 (6.1) のゲートコスト評価においては，ML-KEM の構成における基礎環 $R = \mathbb{Z}[X]/(X^n + 1)$ $(n = 256)$ の代数構造を利用していない．4.5.1 項で述べたように，環 R の代数構造を利用することで，篩アルゴリズムなどの SVP アルゴリズムの高速化が期待できる．しかし，加群格子上の BKZ 基底簡約アルゴリズムの内部 SVP オラクルとして呼び出すことにはいくつかの技術的障壁がある．具体的には，加群格子上の BKZ 基底簡約のブロックサイズ β の選択に制限があるため，primal 攻撃に最適な β を利用することが困難となる．また，加群格子上の BKZ 基底簡約アルゴリズムが，$\mathbb{Z}$ 格子上の BKZ 基底簡約と同程度の品質を持つ基底を出力するかどうかは不明である．（文献 [34] では，非構造化格子よりも加群格子上の BKZ 基底簡約はより多くのブロックサイズを必要とすると結論付けている．）さらに，加群格子上で BKZ 基底簡約を実用的に動作させるための実装基盤も現時点では十分に整備されていない．一方，イデアル格子上の SVP に対する量子アルゴリズムについても，4.5.2 項で述べたように Module-LWE に適用するには障壁があり，文献 [56, Appendix C] で言及されているように，ML-KEM に対する実用的な攻撃に繋がる可能性は低い．以上のように，ML-KEM に対する攻撃において，代数構造を利用した格子アルゴリズムは現時点で有効とはならない．

謝辞

本報告書の執筆に関しては，JST 経済安全保障重要技術育成プログラム【JPMJKP24U2】の支援を受けたものです．

参考文献

[1] Martin R Albrecht. On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL. In *Advances in Cryptology–EUROCRYPT 2017*, volume 10211 of *Lecture Notes in Computer Science*, pages 103–129. Springer, 2017.

[2] Martin R. Albrecht, Benjamin R. Curtis, Amit Deo, Alex Davidson, Rachel Player, Eamonn W. Postlethwaite, Fernando Virdia, and Thomas Wunderer. Estimate all the {LWE, NTRU} schemes! In Dario Catalano and Roberto De Prisco, editors, *Security and Cryptography for Networks - 11th International Conference, SCN 2018, Amalfi, Italy, September 5-7, 2018, Proceedings*, volume 11035 of *Lecture Notes in Computer Science*, pages 351–367. Springer, 2018. `https://estimate-all-the-lwe-ntru-schemes.github.io/docs/`.

[3] Martin R. Albrecht and Amit Deo. Large modulus Ring-LWE $\geq$ Module-LWE. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I*, volume 10624 of *Lecture Notes in Computer Science*, pages 267–296. Springer, 2017.

[4] Martin R Albrecht and Léo Ducas. Lattice attacks on NTRU and LWE: A history of refinements. In *Computational Cryptography: Algorithmic Aspects of Cryptology*, volume 469 of *London Mathematical Society Lecture Note Series*, pages 15–40. Cambridge University Press, 2021.

[5] Martin R. Albrecht, Léo Ducas, Gottfried Herold, Elena Kirshanova, Eamonn W. Postlethwaite, and Marc Stevens. The general sieve kernel and new records in lattice reduction. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part II*, volume 11477 of *Lecture Notes in Computer Science*, pages 717–746. Springer, 2019.

[6] Martin R Albrecht, Vlad Gheorghiu, Eamonn W Postlethwaite, and John M Schanck. Estimating quantum speedups for lattice sieves. In *Advances in Cryptology–ASIACRYPT 2020*, volume 12492 of *Lecture Notes in Computer Science*, pages 583–613. Springer, 2020.

[7] Martin R. Albrecht, Florian Göpfert, Fernando Virdia, and Thomas Wunderer. Revisiting the expected cost of solving uSVP and applications to LWE. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I*, volume 10624 of *Lecture Notes in Computer Science*, pages 297–322. Springer, 2017.

[8] Martin R Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015.

[9] Martin R Albrecht and Yixin Shen. Quantum augmented dual attack. *arXiv preprint arXiv:2205.13983*, 2022. `https://arxiv.org/pdf/2205.13983`.

[10] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange - A new hope. In Thorsten Holz and Stefan Savage, editors, *25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016*, pages 327–343. USENIX Association, 2016.

[11] Yoshinori Aono, Yuntao Wang, Takuya Hayashi, and Tsuyoshi Takagi. Improved progressive BKZ algorithms and their precise cost estimation by sharp simulator. In *Advances in Cryptology–EUROCRYPT 2016*, volume 9665 of *Lecture Notes in Computer Science*, pages 789–819. Springer, 2016.

[12] Sanjeev Arora and Rong Ge. New algorithms for learning in presence of errors. In *International Colloquium on Automata, Languages, and Programming (ICALP 2011)*, volume 6755 of *Lecture Notes in Computer Science*, pages 403–415. Springer, 2011.

[13] Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancréde Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Kyber – algorithm specifications and supporting documentation (version 3.02). `https://pq-crystals.org/kyber/data/kyber-specification-round3-20210804.pdf`, 2021.

[14] Shi Bai and Steven D. Galbraith. Lattice decoding attacks on binary LWE. In Willy Susilo and Yi Mu, editors, *Information Security and Privacy - 19th Australasian Conference, ACISP 2014, Wollongong, NSW, Australia, July 7-9, 2014. Proceedings*, volume 8544 of *Lecture Notes in Computer Science*, pages 322–337. Springer, 2014.

[15] Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In *Proceedings of the twenty-seventh annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2016)*, pages 10–24. SIAM, 2016.

[16] Buchmann Johannes Bernstein, Daniel J and Erik Dahmen. *Post-quantum cryptography*. Springer, 2009.

[17] Lei Bi, Xianhui Lu, Junjie Luo, and Kunpeng Wang. Hybrid dual and meet-LWE attack.

In *Information Security and Privacy (ACISP 2022)*, volume 13494 of *Lecture Notes in Computer Science*, pages 168–188. Springer, 2022.

[18] Jean-François Biasse and Fang Song. Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields. In *ACM-SIAM Symposium on Discrete Algorithms (SODA 2016)*, pages 893–902. SIAM, 2016.

[19] Nina Bindel, Johannes Buchmann, Florian Göpfert, and Markus Schmidt. Estimation of the hardness of the learning with errors problem with a restricted number of samples. *Journal of Mathematical Cryptology*, 13(1):47–67, 2019.

[20] Katharina Boudgoust, Corentin Jeudy, Adeline Roux-Langlois, and Weiqiang Wen. On the hardness of module learning with errors with short distributions. *Journal of Cryptology*, 36(1):1–70, 2023.

[21] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):1–36, 2014.

[22] Peter Campbell, Michael Groves, and Dan Shepherd. Soliloquy: A cautionary tale. ETSI 2nd quantum-safe crypto workshop, 2014, 2014. `https://docbox.etsi.org/workshop/2014/201410_CRYPTO/S07_Systems_and_Attacks/S07_Groves_Annex.pdf`.

[23] Kevin Carrier, Charles Meyer-Hilfiger, Yixin Shen, and Jean-Pierre Tillich. Assessing the impact of a variant of MATZOV's dual attack on Kyber. In *Advances in Cryptology–CRYPTO 2025*, volume 16000 of *Lecture Notes in Computer Science*, pages 444–476. Springer, 2025.

[24] Kevin Carrier, Yixin Shen, and Jean-Pierre Tillich. Faster dual lattice attacks by using coding theory. 2024. `https://hal.science/hal-04519755/document`.

[25] Yuanmi Chen. *Réduction de réseau et sécurité concrete du chiffrement completement homomorphe*. PhD thesis, Paris 7, 2013.

[26] Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better lattice security estimates. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, volume 7073 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2011.

[27] Ronald Cramer, Léo Ducas, Chris Peikert, and Oded Regev. Recovering short generators of principal ideals in cyclotomic rings. In *Advances in Cryptology–EUROCRYPT 2016*, volume 9666 of *Lecture Notes in Computer Science*, pages 559–585. Springer, 2016.

[28] Ronald Cramer, Léo Ducas, and Benjamin Wesolowski. Short stickelberger class relations and application to ideal-svp. In *Advances in Cryptology–EUROCRYPT 2017*, volume 10210 of *Lecture Notes in Computer Science*, pages 324–348. Springer, 2017.

[29] Dana Dachman-Soled, Léo Ducas, Huijing Gong, and Mélissa Rossi. LWE with side information: Attacks and concrete security estimation. In *Advances in Cryptology–CRYPTO 2020*, volume 12171 of *Lecture Notes in Computer Science*, pages 329–358. Springer, 2020.

[30] TU Darmstadt and UC San Diego. LWE challenge. `https://www.latticechallenge.org/lwe_challenge/challenge.php`. 2025-12-31 閲覧.

[31] Gabrielle De Micheli and Daniele Micciancio. A fully classical LLL algorithm for modules. *Cryptology ePrint Archive, Paper 2022/1356*, 2022. `https://eprint.iacr.org/2022/1356.pdf`.

[32] Léo Ducas. Shortest vector from lattice sieving: a few dimensions for free. In *Advanves in Cryptology–EUROCRYPT 2018*, volume 10820 of *Lecture Notes in Computer Science*, pages 125–145. Springer, 2018.

[33] Léo Ducas. Estimating the hidden overheads in the BDGL lattice sieving algorithm. In *Post-Quantum Cryptography (PQCrypto 2022)*, volume 13512 of *Lecture Notes in Computer Science*, pages 480–497. Springer, 2022.

[34] Léo Ducas, Lynn Engelberts, and Paola de Perthuis. Predicting module-lattice reduction. In *Advances in Cryptology–ASIACRYPT 2025*, volume 16247 of *Lecture Notes in Computer Science*, pages 133–166. Springer, 2025.

[35] Léo Ducas and Ludo N Pulles. Does the dual-sieve attack on learning with errors even work? In *Advances in Cryptology–EUROCYPT 2023*, volume 14083 of *Lecture Notes in Computer Science*, pages 37–69. Springer, 2023.

[36] Jan-Pieter D' anvers and Senne Batsleer. Multitarget decryption failure attacks and their application to Saber and Kyber. In *Public-Key Cryptography (PKC 2022)*, volume 13177 of *Lecture Notes in Computer Science*, pages 3–33. Springer, 2022.

[37] Kirsten Eisenträger, Sean Hallgren, Alexei Kitaev, and Fang Song. A quantum algorithm for computing the unit group of an arbitrary degree number field. In *ACM Symposium on Theory of computing (STOC 2014)*, pages 293–302, 2014.

[38] Nicolas Gama and Phong Q Nguyen. Predicting lattice reduction. In *Advances in Cryptology–EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 31–51. Springer, 2008.

[39] Timo Glaser and Alexander May. How to enumerate LWE keys as narrow as in Kyber/Dilithium. In *Cryptology and Network Security (CANS 2023)*, volume 14342 of *Lecture Notes in Computer Science*, pages 75–100. Springer, 2023.

[40] Qian Guo and Thomas Johansson. Faster dual lattice attacks for solving LWE with applications to CRYSTALS. In *Advances in Cryptology–ASIACRYPT 2021*, volume 13093 of *Lecture Notes in Computer Science*, pages 33–62. Springer, 2021.

[41] Minki Hhan, Jiseung Kim, Changmin Lee, and Yongha Son. Let's meet ternary keys on Babai's plane: A hybrid of lattice-reduction and meet-LWE. *Cryptology ePrint Archive,*

Paper 2022/1473, 2022. https://eprint.iacr.org/2022/1473.

[42] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 341–371. Springer, 2017.

[43] Nick Howgrave-Graham. A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In *Advances in Cryptology–CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 150–169. Springer, 2007.

[44] Internet Engineering Task Force (IETF). Post-quantum cryptography for engineers, February 2024. https://datatracker.ietf.org/doc/html/draft-ietf-pquip-pqc-engineers-03.

[45] Ravi Kannan. Improved algorithms for integer programming and related lattice problems. In David S. Johnson, Ronald Fagin, Michael L. Fredman, David Harel, Richard M. Karp, Nancy A. Lynch, Christos H. Papadimitriou, Ronald L. Rivest, Walter L. Ruzzo, and Joel I. Seiferas, editors, *Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 25-27 April, 1983, Boston, Massachusetts, USA*, pages 193–206. ACM, 1983.

[46] Jiseung Kim, Changmin Lee, and Yongha Son. Worst-case analysis of lattice enumeration algorithm over modules. *Cryptology ePrint Archive, Paper 2025/480*, 2025. https://eprint.iacr.org/2025/480.pdf.

[47] Paul Kirchner and Pierre-Alain Fouque. An improved BKW algorithm for LWE with applications to cryptography and lattices. In *Advances in Cryptology–CRYPTO 2015*, volume 9215 of *Lecture Notes in Computer Science*, pages 43–62. Springer, 2015.

[48] Thijs Laarhoven. Search problems in cryptography: from fingerprinting to lattice sieving. *PhD thesis, Eindhoven University of Technology*, 2016. https://pure.tue.nl/ws/files/14673128/20160216_Laarhoven.pdf.

[49] Thijs Laarhoven and Artur Mariano. Progressive lattice sieving. In *Post-Quantum Cryptography (PQCrypto 2018)*, volume 10786 of *Lecture Notes in Computer Science*, pages 292–311. Springer, 2018.

[50] Thijs Laarhoven, Michele Mosca, and Joop Van De Pol. Finding shortest lattice vectors faster using quantum search. *Designs, Codes and Cryptography*, 77(2):375–400, 2015.

[51] Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Des. Codes Cryptogr.*, 75(3):565–599, 2015.

[52] Changmin Lee, Alice Pellet-Mary, Damien Stehlé, and Alexandre Wallet. An LLL algorithm for module lattices. In *Advances in Cryptology–ASIACRYPT 2019*, volume 11922 of *Lecture Notes in Computer Science*, pages 59–90. Springer, 2019.

[53] A. K. Lenstra, H. W. Lenstra, and Lovász L. Factoring polynomials with rational coeffi-

cients. *Mathematische Annalen*, 261(4):515–534, 12 1982.

[54] Cathy Yuanchen Li, Jana Sotáková, Emily Wenger, Mohamed Malhou, Evrard Garcelon, François Charton, and Kristin Lauter. SalsaPicante: A machine learning attack on LWE with binary secrets. In *ACM SIGSAC Conference on Computer and Communications Security (ACM CCS 2023)*, pages 2606–2620, 2023.

[55] Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In Aggelos Kiayias, editor, *Topics in Cryptology - CT-RSA 2011 - The Cryptographers' Track at the RSA Conference 2011, San Francisco, CA, USA, February 14-18, 2011. Proceedings*, volume 6558 of *Lecture Notes in Computer Science*, pages 319–339. Springer, 2011.

[56] National lnstitute of Standards and Technology (NIST). NIST IR 8413-upd1: Status report on the third round of the NIST post-quantum cryptography standardization process. 2022. `https://nvlpubs.nist.gov/nistpubs/ir/2022/NIST.IR.8413-upd1.pdf`.

[57] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23. Springer, 2010.

[58] Artur Mariano, Thijs Laarhoven, and Christian Bischof. A parallel variant of LDSieve for the SVP on lattices. In *2017 25th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*, pages 23–30. IEEE, 2017.

[59] MATZOV. Report on the security of LWE: Improved dual lattice attack, 2022. `https://zenodo.org/records/6412487`.

[60] Alexander May. How to meet ternary LWE keys. In *Advances in Cryptology–CRYPTO 2021*, volume 12826 of *Lecture Notes in Computer Science*, pages 701–731. Springer, 2021.

[61] Tamalika Mukherjee and Noah Stephens-Davidowitz. Lattice reduction for modules, or how to reduce ModuleSVP to ModuleSVP. In *Advances in Cryptology–CRYPTO 2020*, volume 12171 of *Lecture Notes in Computer Science*, pages 213–242. Springer, 2020.

[62] National Institute of Standards and Technology (NIST). FIPS 202: SHA-3 standard: Permutation-based hash and extendable-output functions. `https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf`, August 2015.

[63] National Institute of Standards and Technology (NIST). Submission requirements and evaluation criteria for the post-quantum cryptography standardization process, 2016. `https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf`.

[64] National Institute of Standards and Technology (NIST). FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism Standard. `https://nvlpubs.nist.gov/nistpubs/FIPS/`

NIST.FIPS.203.pdf, August 13, 2024.

[65] Eamonn W. Postlethwaite and Fernando Virdia. On the success probability of solving unique SVP via BKZ. In Juan A. Garay, editor, *Public-Key Cryptography - PKC 2021 - 24th IACR International Conference on Practice and Theory of Public Key Cryptography, Virtual Event, May 10-13, 2021, Proceedings, Part I*, volume 12710 of *Lecture Notes in Computer Science*, pages 68–98. Springer, 2021.

[66] Gokulnath Rajendran, Prasanna Ravi, Jan-Pieter D'anvers, Shivam Bhasin, and Anupam Chattopadhyay. Pushing the limits of generic side-channel attacks on LWE-based KEMs-parallel PC oracle attacks on Kyber KEM and beyond. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2023(2):418–446, 2023.

[67] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 84–93. ACM, 2005.

[68] Oded Regev. The learning with errors problem (invited survey). In *Proceedings of the 25th Annual IEEE Conference on Computational Complexity, CCC 2010, Cambridge, Massachusetts, USA, June 9-12, 2010*, pages 191–204. IEEE Computer Society, 2010.

[69] Tsunekazu Saito, Keita Xagawa, and Takashi Yamakawa. Tightly-secure key-encapsulation mechanism in the quantum random oracle model. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 520–551. Springer, 2018.

[70] Claus Peter Schnorr. Lattice reduction by random sampling and birthday methods. In *Symposium on Theoretical Aspects of Computer Science (STACS 2003)*, volume 2607 of *Lecture Notes in Computer Science*, pages 145–156. Springer, 2003.

[71] Claus-Peter Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Math. Program.*, 66:181–199, 1994.

[72] Samuel Stevens, Emily Wenger, Cathy Li, Niklas Nolte, Eshika Saxena, François Charton, and Kristin Lauter. Salsa Fresca: Angular embeddings and pre-training for ML attacks on learning with errors. *arXiv preprint arXiv:2402.01082*, 2024. https://arxiv.org/pdf/2402.01082.

[73] Yutaro Tanaka, Rei Ueno, Keita Xagawa, Akira Ito, Junko Takahashi, and Naofumi Homma. Multiple-valued plaintext-checking side-channel attacks on post-quantum KEMs. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2023(3):473–503, 2023.

[74] Emily Wenger, Mingjie Chen, Francois Charton, and Kristin E Lauter. SALSA: Attack-

ing lattice cryptography with transformers. *Advances in Neural Information Processing Systems (NeurIPS 2022)*, 35:34981–34994, 2022.

[75] Emily Wenger, Eshika Saxena, Mohamed Malhou, Ellie Thieu, and Kristin Lauter. Benchmarking attacks on learning with errors. In *IEEE Symposium on Security and Privacy (SP)*, pages 279–297. IEEE, 2025.

[76] Han Wu and Guangwu Xu. Enhancing the dual attack against MLWE: Constructing more short vectors using its algebraic structure. *Cryptology ePrint Archive, Paper 2022/1661*, 2022. `https://eprint.iacr.org/2022/1661`.

[77] Keita Xagawa, Akira Ito, Rei Ueno, Junko Takahashi, and Naofumi Homma. Fault-injection attacks against NIST's post-quantum cryptography round 3 KEM candidates. In *Advances in Cryptology–ASIACRYPT 2021*, volume 13091 of *Lecture Notes in Computer Science*, pages 33–61. Springer, 2021.

[78] Yang Yu and Léo Ducas. Second order statistical behavior of LLL and BKZ. In Carlisle Adams and Jan Camenisch, editors, *Selected Areas in Cryptography (SAC 2017) - 24th International Conference, Ottawa, ON, Canada, August 16-18, 2017, Revised Selected Papers*, volume 10719 of *Lecture Notes in Computer Science*, pages 3–22. Springer, 2017.

[79] Ziyu Zhao, Jintai Ding, and Bo-Yin Yang. Sieving with streaming memory access. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2025(2):362–384, 2025.