École Normale Supérieure
Département d'Informatique

Université Paris 7
Denis Diderot

Signatures Fondées sur les Réseaux Euclidiens: Attaques, Analyses et Optimisations

Thèse

présentée et soutenue publiquement le 12 Novembre 2013 par

Léo Ducas-Binda

pour l'obtention du

Doctorat de l'Université Paris Diderot
(spécialité informatique)

Devant le jury composé de :

Directeur de thèse :	Phong Q. Nguyen	(INRIA, École Normale Supérieure)
Rapporteurs :	Fabien Laguillaumie	(Université de Lyon 1)
	Oded Regev	(New York University)
Examinateurs :	Jean-Charles Faugère	(INRIA, Université Pierre et Marie Curie)
	Guillaume Hanrot	(École Normale Supérieure de Lyon)
	Vadim Lyubashevsky	(INRIA, École Normale Supérieure)
	David Pointcheval	(CNRS, École Normale Supérieure)
	Pascal Paillier	(CryptoExperts)

Travaux effectués au Laboratoire d'Informatique de l'École Normale Supérieure

Remerciements

Mes premières pensées vont à ma famille. À mon père, qui a su cultiver en moi le goût du savoir dès mon plus jeune âge. Je ne vois de plus beaux cadeaux ce celui d'une vie où le travail et la *passion* ne font qu'un. À ma mère pour l'autre moitié de mon éducation, sans laquelle je ne saurais être *libre* et *heureux*. Merci.

Je souhaite ensuite remercier Phong Nguyễn en premier lieu pour m'avoir proposé ce sujet de thèse qui, en plus d'être un sujet porteur, m'a offert de nombreuses opportunités d'en découvrir encore et toujours plus sur la géométrie, l'algèbre et l'algorithmique, ravivant ma curiosité de taupin. Je ne peux que louer la finesse de ses réactions, son recul soigneusement dosé, pour m'enseigner d'abord la rigueur puis l'autonomie dans le métier de chercheur. Je souhaite aussi remercier David Pointcheval, directeur officiel du début de cette thèse de m'avoir accueilli au laboratoire d'informatique de l'ENS, et d'avoir contribué a la direction de cette thèse avec beaucoup de bienveillance.

Mon humble reconnaissance se porte vers les rapporteurs de cette thèse, Oded Regev et Fabien Laguillaumie, pour avoir eu la patience de lire mon manuscrit et de m'avoir fait part de leurs commentaires avisés. J'ai conscience de la durée et de la difficulté de se plonger dans un document aussi long, pointu et technique qu'une thèse. Je me sens tout à fait honoré de l'intérêt qu'ils ont porté à mes travaux.

Je remercie aussi tous ceux qui ont, par leur encadrement, leur collaboration, ou même de simples discussions scientifiques, contribué aux articles publiés durant cette thèse : Dan Boneh, Oded Regev, Chris Peikert, Vadim Lyubashevsky, Damien Stehlé. Je m'estime très chanceux d'avoir pu côtoyer des scientifiques de ce calibre. Je remercie aussi Jeff Hoffstein, qui en plus de son intérêt pour mes travaux, m'a offert l'une de mes premières opportunités de présenter mes résultats en public. Je remercie enfin mes pairs et co-auteurs Tancrède Lepoint et Alain Durmus, à qui je souhaite une carrière scientifique à la hauteur de leurs talents.

Au-delà de l'encadrement officiel, j'ai aussi ressenti un grand soutien de la part de Pierre-Alain Fouque et de Damien Vergnaud, toujours curieux et à l'écoute du dernier sujet qui m'enthousiasme ; et partageant en retour leurs sujets avec autant d'enthousiasme. Je garde précieusement les quelques conseils de David Naccache distillés lors de ces passages au laboratoire. Je remercie chaleureusement Michel Abdalla *what else*, toujours attentif à mon état de frustration, me faisant comprendre aux heures avancées que demain ce serait tout aussi bien.

Quelques personnes ont aussi contribué a ce manuscrit en dehors du cadre scientifique ; Jill-Jênn, Jill-Jênn [1], Pascale et Salomé ont toute ma gratitude pour avoir aidé à la relecture et la correction de cette thèse.

Il me tiens a cœur de remercier aussi toutes les personnes qui ont contribué, de près ou de loin à la réalisation du jeu vidéo Cryptris ; ce fut une expérience professionnelle très intéressante et gratifiante ; c'est avec une grande joie que je vois notre projet aboutir. Je dois en particulier témoigner mon respect à Anthony Teston pour son travail remarquable à la coordination du projet et pour ses talents de médiateur scientifique. Je félicite aussi Mathieu Jouhet et son équipe pour leur création aux qualités graphiques, ergonomiques et technologiques bien au-delà de mes espérances initiales.

Je remercie également l'équipe administrative à l'Ens ainsi qu'à l'école doctorale, Isabelle, Régine, Joëlle et Michelle pour leur diligence, leur efficacité et parfois leur patience. Et bien sûr Valerie, que je remercie aussi pour sa vive et franche compagnie.

L'obtention d'un doctorat peut devenir un exercice terriblement solitaire. Heureusement, durant ces longues années, j'ai eu de formidables colocataires, sans qui le temps de la thèse eût ètè une expérience bien trop monotone. Eux comme moi ayant plutôt la bougeotte, ils sont fort nombreux ! Merci à (par ordre d'apparition) Damien, Xavier, Florian, Marion, Simoné, Mélodie, Vlad, Pablo, Nico, Maud, Jane, Matan, Olivier ; je n'aurais probablement pas eu le courage d'aller au bout sans les forces vives de leur présence au quotidien.

1. oui, deux fois

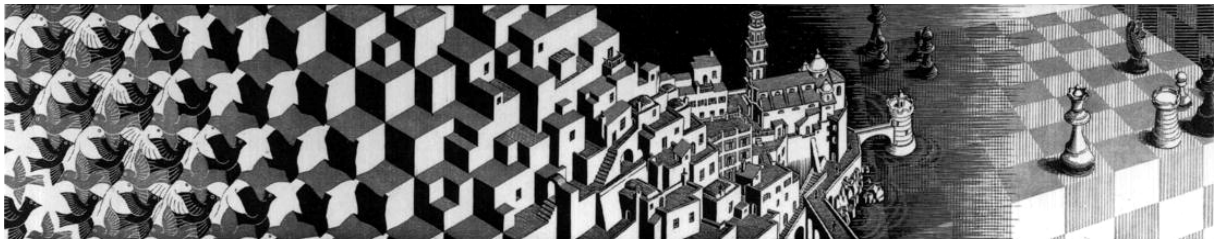
Mais de la vie, il y en a eu aussi au labo, dans ce grand *open space* ; elle a mis du temps a se développer, grands timides que nous sommes, ou peut être ai-je mis du temps à la decouvrir, grand timide que je suis. Aurore *Vive la Bretagne*©, Olivier *Paladin Loyal-Bisounours*, Tancrède *à Quatre épingles*, Mario *Sourire Toujours Sourire*, *Sa Majestée* Elizabeth, Jill-Jênn *Jamais-J'arrête* et Thomas *Thanks Bro* ; vous êtes de ces gens qui font oublier que l'on est pas censé mélanger ses collègues et ses amis. Je vous en remercie. J'ai aussi grandement apprecié les échanges, scientifiques ou non, avec Yuanmi 陈圆谜, Charles, Roch, Angelo, Miriam et Sorina.

Et puis, j'ai une pensée pour Stéphane, Églantine, Catherine, Till, Manon, Raph, Salomé, Patrick, Elsa, Florian (BiBi), Pierre, Martin, Jennie, Ségolène, Audrey, Denise, Anne-So ; en souvenir des instants partagés de la vie parisienne. Et une autre pensée pour les Lyonnaises et les Lyonnais : Margaux, Arthur, Julie, Corentin, Charlotte, Yannick, Marina, Paule, Leila. Je ne saurais non plus oublier Sarah et Viya pour leur soutien dématérialisé.

Je souhaite enfin remercier Senyang 黄森洋, Chengliang 田呈亮 et Xuexin (ou Sophia) 郑学欣 pour leur accueil durant mon passage à Beijing et leur souhaite une fin de thèse prolifique. Ce n'est pas sans une certaine fierté que j'écris le nom qu'ils m'ont donné : 李欧俊, à défaut de savoir le prononcer. 谢谢 !

Young man, in mathematics you don't understand things. You just get used to them.
Jeune homme, en mathématique on ne comprend pas les choses. On s'y habitue.

– John von Neumann



Extrait de *Metamorphosis III* – Maurits Cornelis Escher.

Table des matières

Chapitre 1

Prolégomènes

Cryptologie du grec $\kappa\rho\upsilon\pi\tau\sigma\sigma$ le secret, et $\lambda\sigma\gamma\sigma\sigma$, la science. En cryptologie se distinguent deux branches, la **cryptographie** ($\gamma\rho\alpha\phi\epsilon\iota\nu$, l'écriture) qui s'intéresse à la construction de codes difficiles à casser, et la **cryptanalyse** qui cherche à attaquer ces constructions. Au-delà de la notion de secret, la cryptographie s'étend à d'autres notions de sécurité de l'information, comme l'authenticité (certification sur l'origine du message) et l'intégrité (garantie que le message n'a pas été altéré).

1.1 Un bref historique des idées en cryptologie

Dans cette section, nous proposons au lecteur un historique partiel de la Cryptographie, permettant de mettre en avant les idées sans détails mathématiques. Certaines de ces idées, notamment les aspects statistiques de la cryptanalyse et les contre-mesures cryptographiques offrent un parallèle simple avec les travaux effectuées dans cette thèse. Cette section a aussi vocation à vulgariser l'histoire de la cryptographie ; les lecteurs experts seront priés de pardonner les inexactitudes simplificatrices. Pour une histoire plus détaillée de la cryptographie, les lecteurs interessés sont invités à consulter l'ouvrage *The Codebreakers* de Kahn [Kah96].

1.1.1 L'âge artisanal

Les premières techniques de cryptographie naissent avec l'art de la guerre, lorsqu'il devient évident qu'au-delà du nombre et de la puissance des armées, l'information sur les positions et les stratégies adverses permet une prise de décision favorable. Ainsi, de façon symétrique, il devient essentiel de se prémunir contre une prise d'information de l'adversaire, autant sur le terrain que dans les communications entre généraux.

Le Code de César. Le code de César constitue l'un des exemples les plus anciens et les plus simples de techniques cryptographiques ; il n'est cependant pas le plus ancien (la " scytale" le précède), et il semble que Jules César utilisait d'autres techniques plus complexes.

Le code de César consiste simplement en un décalage alphabétique, de trois rangs : pour chiffrer un message on remplace `A` par `D`, `B` par `E`, `C` par `F` et ainsi de suite, jusqu'à `W` par `Z` ; après quoi l'on reprend au début de l'alphabet : `X` par `A`, etc. Le récipiendaire effectuera le décalage inverse pour déchiffrer le message.

Il est bien sûr très aisé de casser un tel code ; mais au-delà de cette trop grande simplicité, un inconvénient majeur réside dans le fait que la confidentialité nécessite que la méthode de chiffrement elle-même reste secrète : il suffit d'un traître pour compromettre la confidentialité de toutes les communications entre César et ses généraux.

Cependant, le choix d'un décalage d'exactement trois caractères est arbitraire. Il est possible d'utiliser différents décalages pour chacun des interlocuteurs, limitant ainsi l'impact d'une fuite d'information. La valeur du décalage constitue alors *la clef secrète* du chiffrement, et il n'est plus nécessaire que la technique elle-même reste secrète.

Le Code Navajo. Même si le décalage n'est pas connu d'un adversaire, ce chiffre reste très faible : il est raisonnable d'essayer tous les décalages possibles (26 essais au plus). Notons tout de même que le

récipiendaire légitime pourra déchiffrer le message plus vite que l'adversaire. Cette avance peut suffire, la confidentialité du message pouvant n'être nécessaire que durant un court laps de temps.

Un exemple célèbre d'une telle application est *le code Navajo* : il s'agit d'un dialecte amérindien n'ayant aucun lien avec les langues orientales et occidentales ; et il fut utilisé à des fins cryptographiques par l'armée américaine durant la seconde guerre mondiale. Bien qu'il soit théoriquement possible d'inférer une partie de l'information en analysant une grande quantité de radio-communications, cette analyse requiert beaucoup d'efforts et de temps aux ennemis japonais ou allemands. Les radio-codeurs Navajo pouvaient au contraire traduire l'information en temps réel, sans recours à des machines de codage ni même de prise de note : il s'agissait d'une technique idéale de radio-communication chiffrée sur les théâtres d'opération pour des prises de décisions rapides. Cette histoire a été romancée par Collins, Rice et Batteer dans *Windtalkers* puis adaptée au cinéma par John Woo [CRB01, Woo02].

Cette idée reste essentielle jusque dans la cryptographie moderne : il n'est jamais (sauf si les clefs sont aussi longues que les messages, voir masque à usage unique) complètement impossible de casser un code, cependant on choisit ses paramètres de telle façon à ce que les attaques requièrent des efforts en termes de puissance et de temps de calcul inaccessibles en pratique.

Les codes de substitutions. Comme nous l'avons détaillé, la faiblesse principale du code de César est le nombre insuffisant de clefs possibles, rendant la recherche exhaustive tout à fait raisonnable. Si au lieu de juste un décalage, on s'autorise une permutation quelconque, comme :

clair	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
chiffrés	P M L O I K J U Y H N B G T R F V C D E Z A Q S X W

le nombre de clefs possibles est alors de

$$26! = 1 \times 2 \times 3 \times \cdots \times 26 = 403291461126605635584000000 \approx 10^{26} \approx 2^{88}.$$

Énumérer un tel nombre de combinaisons reste inaccessible même aux plus puissants super-ordinateurs construits en 2013 ! Malheureusement, il ne suffit pas de rendre l'ensemble des clefs grand pour rendre sûre une technique de chiffrement. Prenons par exemple le cryptogramme suivant :

```
KTIMNE LMDDKC XT LUNIICKGKTE FMC DXPDENEXENVT
   LKDE XT FKENE FKX LVGGK HVXKC MX FKTOX
```

La méthode la plus simple pour l'attaquer s'appelle l'analyse de fréquence : étant donné qu'une lettre est toujours remplacée de la même façon, on devrait retrouver dans les chiffrés les propriétés statistiques de la langue française. Par exemple, il y a fort à parier que la lettre apparaissant le plus souvent corresponde au `E` ; ici, il s'agit du `K`, présent 10 fois.

```
E..... ....E. .. ......E.E.. ... ............
   .E.. .. .E... .E. ....E ...E. .. .E...
```

En utilisant d'autres propriétés de la langue, comme le dédoublement de certaines lettres, on pourra retrouver l'intégralité du message déchiffré (le clair). Nous appellerons ces attaques, des attaques “ du premier ordre” : les biais statistiques à détecter sont inversement proportionnels à la taille de l'alphabet.

De telles méthodes de chiffrement ont cependant été utilisées, surtout autour du XV$^{\text{ème}}$ siècle ; et quelques traités de cryptographie font leur apparition. Il y sera recommandé d'utiliser des contre-mesures à ces attaques statistiques, notamment l'utilisation de plusieurs symboles différents pour coder une même lettre apparaissant trop souvent comme le `E`, et des symboles spéciaux pour cacher les lettres dédoublées comme `FF`. Ces contre-mesures restent cependant imparfaites : une analyse de fréquence “ du second ordre” est possible, consistant à mesurer non plus la fréquence de chaque symbole, mais la fréquence de chaque paire de symboles. Cette deuxième attaque nécessite cependant des messages beaucoup plus longs, car il faut détecter des biais statistiques de l'ordre de l'inverse du *carré* de la taille de l'alphabet.

Historiquement, certaines variations de ces contre-mesures ont résisté très longtemps à la cryptanalyse ; il a fallu par exemple attendre la fin du XIX$^{\text{ème}}$ avant que “le grand chiffre” utilisé par Louis XIV pour ses communications diplomatiques ne soit cassé par Bazeries, révélant enfin les dessous de l'histoire européenne [Baz01].

Le chiffre de Vigenère. Le chiffrement de Vigenère consiste lui aussi en une substitution monoalphabétique, mais propose que la substitution varie selon la position afin qu'une même lettre ne soit pas toujours chiffrée de la même façon ; l'objectif étant d'empêcher les attaques statistiques. Plus précisément, chaque lettre du message sera chiffrée par un décalage dont le rang est donné par un autre texte qui sert de clef, que l'on répète en boucle si nécessaire. Autrement dit, on obtient le chiffré en ajoutant le rang (modulo 26) de chaque lettre du clair avec une lettre de clef, en comptant à partir de `A` = 0.

	`ALATTENTIONDUROYHENRYIII`	clair
+	`CLEFCLEFCLEFCLEFCLEFCLEF`	clef
=	`CWEYVPRYKZRIWCSDJPRWATMN`	chiffré

Cette technique de chiffrement nécessite de choisir de longues clefs par rapport au message : en effet si l'on sait par exemple que la clef est courte et a pour longueur 4, alors en ne gardant qu'une lettre sur quatre du chiffré, on a de nouveau une substitution fixe : chaque lettre est toujours remplacée par la même lettre.

	`ATIUHY`	extrait du clair
+	`CCCCCC`	extrait de clef
=	`CVKWJA`	extrait de chiffré

Si cet extrait de texte chiffré est assez long, alors on peut de nouveau appliquer une analyse de fréquence du premier ordre pour retrouver la lettre correspondant au `E` dans l'extrait de chiffré ; et ainsi remonter à l'extrait de clef. Il n'y a plus qu'à recommencer pour tous les autres extraits de texte clair.

Si par contre le message n'est pas plus grand que la clef, ou simplement un petit facteur plus grand cette attaque ne s'applique plus car l'analyse de fréquence est faite sur de trop petits extraits. Mis au point au XVI$^{\text{ème}}$ siècle, il faudra attendre les travaux indépendants de Friedrich Kasiski (1863) et de Charles Babbage, pour casser ce chiffrement. Sans entrer dans les détails il s'agit encore une fois d'utiliser les propriétés statistiques présentes dans le message et dans la clef.

1.1.2 L'âge technique

Vers la fin du XIX$^{\text{ème}}$ et le début du XX$^{\text{ème}}$ siècle la cryptographie prend un nouveau tournant, avec l'introduction du formalisme mathématique ainsi que l'automatisation du chiffrement. Ce processus scientifique aboutira avec Alan Turing, qui concevra mathématiquement les premiers modèles d'ordinateur, et participera à leur réalisation durant la seconde guerre mondiale, pour la cryptanalyse des communications de l'Axe. Il est cependant intéressant de constater que Charles Babbage qui inventa les premières machines mécaniques à calculer 100 ans plus tôt, s'intéressait lui aussi à la cryptanalyse.

" Deciphering is, in my opinion, one of the most fascinating of arts, and I fear I have wasted upon it more time than it deserves." Charles Babbage, *Passages from the life of a philosopher* (1864)

Principes de Kerckhoffs. En 1883, Auguste Kerckhoffs énonce 6 *desiderata* de la cryptographie militaire [Ker83] :

i Le système doit être matériellement, sinon mathématiquement, indéchiffrable

ii Il faut qu'il n'exige pas le secret, et qu'il puisse sans inconvénient tomber entre les mains de l'ennemi

iii La clef doit pouvoir en être communiquée et retenue sans le secours de notes écrites, et être changée ou modifiée au gré des correspondants

iv Il faut qu'il soit applicable à la correspondance télégraphique

v Il faut qu'il soit portatif, et que son maniement ou son fonctionnement n'exige pas le concours de plusieurs personnes

vi Enfin, il est nécessaire, vu les circonstances qui en commandent l'application, que le système soit d'un usage facile, ne demandant ni tension d'esprit, ni la connaissance d'une longue série de règles à observer.

Le plus important en est le second, repris plus simplement par une maxime de Claude Shannon : " L'adversaire connaît le système ". Plus précisément, cela énonce que la confidentialité des communications ne doit pas requérir le secret du système de chiffrement, mais seulement le secret de la clef.

Notons cependant que le dernier principe était à l'époque très limitant (essentiellement des roues de codage manuel, ou des tables). Il faudra attendre les machines électromécaniques pour se permettre des opérations complexes en préservant la simplicité d'utilisation. Dans la cryptographie moderne ce dernier

principe s'interprète en termes de temps de calcul, mais aussi en termes de simplicité des opérations à effectuer et de parallélisabilité du calcul ; pour certaines applications, telle que la cryptographie sur carte à puce, on pourra préférer faire beaucoup d'opérations mathématiques simples, que peu d'opérations trop complexes.

Le masque à usage unique. L'invention du masque à usage unique est attribué à Gilbert Vernam (1917), bien qu'il semble que Frank Miller en ait posé les bases dès 1882 [Bel11, Mil82]. Il s'agit en fait d'un chiffrement de Vigenère, mais qui spécifie précisément que la clef doit être *parfaitement aléatoire* (et non pas un texte), aussi longue que le message à chiffrer, et surtout *jamais réutilisée*.

Cela semble à première vue contrer toutes les techniques de cryptanalyse du chiffrement de Vigenère ; en effet, Claude Shannon publia trente ans plus tard [Sha49] une preuve mathématique que ce chiffrement garantit une sécurité absolue ! L'idée de la preuve est que sans connaissance sur la clef, n'importe quel message clair peut être chiffré en n'importe quel message chiffré, et ce avec probabilité égale, ainsi, il est impossible de retrouver le message clair, ne serait-ce que partiellement, et même en disposant d'une puissance de calcul infinie. Plus fort encore, cette sécurité vaut aussi dans un " contexte Bayésien" : même si un attaquant a une connaissance partielle a priori du message, ayant vu sa version chiffrée, il n'apprendra rien de plus.

Machines de Chiffrement. Le masque à usage unique est surtout d'un intérêt théorique, car il requiert que les deux parties souhaitant communiquer, aient préalablement échangé des clefs secrètes aussi longues que le message. Il fut cependant utilisé pour garantir une confidentialité parfaite du *téléphone rouge* – il s'agissait en fait d'un télétexte et non d'un téléphone – entre Moscou et Washington ; régulièrement étaient échangées par valises diplomatiques d'énormes quantités (pour l'époque) de données parfaitement aléatoires stockées sur bande magnétique [Kah96, pp. 715-716].

Dans beaucoup de contextes, cet inconvénient est rédhibitoire ; et l'on souhaitera se contenter d'une courte clef pour échanger de longs messages. C'est ainsi que diverses machines électromécaniques de chiffrement furent inventées durant la première moitié du XX$^{\text{ème}}$, la plus célèbre d'entre elles étant la machine Enigma, utilisée par l'armée du III$^{\text{ème}}$ Reich durant la seconde guerre mondiale. Comme pour le chiffrement de Vigenère, il s'agit de chiffrement par substitution, où la substitution varie pour chaque lettre. Le fonctionnement de la machine est basé sur une série (trois ou quatre) de rotors, chacun effectuant une permutation par un câblage électrique. Mis bout à bout, on obtient la composition de ces permutations, et chaque rotation d'un rotor la modifie. À chaque frappe sur le clavier, un signal électrique traverse l'ensemble des rotors, allumant une lettre chiffrée sur l'écran ; après quoi les rotors tournent à des rythmes différents. La clef de ce chiffrement réside dans la position initiale des rotors. Ces systèmes s'avérèrent bien conçus, au sens où les meilleures attaques nécessitent une recherche presque exhaustive. Cependant, la puissance de calcul des Alliés fut sous-estimée, les Allemands ignorant les immenses progrès théoriques et technologiques réalisés par les Alliés, à Bletchley Park près de Londres.

Les premiers ordinateurs. Les premières cryptanalyses d'Enigma furent accomplies par des mathématiciens polonais, avec des moyens rudimentaires : des tables, des feuilles-masques, et beaucoup de patience. Lorsqu'ils furent accueillis à Bletchley Park, les cryptanalystes anglais furent stupéfaits de les voir venir à bout des versions les plus faibles d'Enigma avec si peu de moyens ; à bien des égards, ce fut un acte d'héroïsme mathématique. Leurs résultats furent d'un grand secours aux alliés, permettant de réduire l'espace de recherche des attaques exhaustives.

Cependant, ces méthodes manuelles prouvèrent vite leurs limites face à des tailles de clefs plus grandes. C'est ce qui motiva l'intérêt initial de développer de nouvelles machines de calcul, plus efficaces et plus souples. C'est ainsi qu'Alan Turing se consacrera à la cryptanalyse d'Enigma durant toute la seconde guerre mondiale ; lui qui avait posé les fondements théoriques de l'informatique, la science du calcul. Il avait inventé le premier modèle mathématique, d'une machine qui serait capable d'effectuer à la demande n'importe quel calcul : contrairement aux machines précédentes conçues pour effectuer un calcul en particulier, la Machine de Turing est programmable ; créant une distinction entre logiciel et matériel. Pour ces travaux, il est considéré aujourd'hui comme le fondateur de la discipline informatique en tant que science, et son nom a été donné au plus haut prix en informatique théorique, remis chaque année depuis 1966.

Du côté technologique, les premières machines de déchiffrement restent basiques : appelées *bombes*, elles sont simplement des répliques de la machine Enigma, mises en parallèles et cadencées aussi rapidement que la mécanique le permet : il n'y a pas encore de logique électronique.

Une première transition a lieu avec l'introduction des commutateurs, des interrupteurs contrôlés par signal électrique, grâce à un électro-aimant. C'est l'entreprise de téléphone Bell qui fut chargée de sa conception et de sa réalisation ; trouvant ainsi une nouvelle utilisation des relais électromécaniques, non plus pour le routage des communications, mais bel et bien pour le calcul. Les rotors sont ainsi remplacés par des multi-commutateurs. Bien que cette transition technologique offre une vitesse et programmabilité accrue, elle sera très vite replacée par le transistor à lampe, permettant d'effectuer des opérations logiques sans aucun mouvement mécanique, et donc beaucoup plus rapidement. La seule partie mobile de cette dernière machine est un ruban perforé codant le message à déchiffrer, se déplaçant à $40km/h$ et lu par un dispositif optique ; reflétant la construction théorique de la machine de Turing, avec son ruban, sa tête de lecture et son processeur logique.

Standardisation Internationale. Jusqu'à la fin des années 60, les techniques cryptographiques restent secrètes, developpées indépendamment dans chaque nation pour les applications diplomatiques et militaires. Avec l'arrivée des réseaux informatisés dans les entreprises, et bientôt ARPANET, le bureau états-unien des standards lance un appel à candidatures pour la standardisation en 1973. L'entreprise IBM proposera alors l'algorithme de chiffrement par bloc *Lucifer*, developpé deux ans plus tôt, notamment par Horst Feistel. Quelques modifications furent apportées, et le DES, Data Encryption Standard, fut adopté en 1976. Cela marque le début de la recherche académique en cryptographie.

La contribution majeure de Feistel sera souvent re-utilisée par la suite ; elle permet de transformer une fonction quelconque sur des blocs de taille n, en une permutation sur des blocs de taille de $2n$, et dont l'inverse est aussi facilement calculable. Cette construction s'est récemment avérée théoriquement sûre, permettant de démontrer l'équivalence entre les modèles de l'oracle aléatoire et le chiffrement par bloc idéal [HKT11].

1.1.3 L'âge paradoxal

Toutes les techniques de chiffrement vues jusqu'ici nécessitent que les parties voulant communiquer de façon confidentielle partagent au préalable une clef secrète ; et cette limitation semblait naturelle à bon nombre de cryptologues. En 1970, James H. Ellis affirme dans une note interne au quartier général britannique des télécommunications (GHCQ) [Ell70] qu'il est concevable d'effectuer des communications confidentielles sans secret partagé au préalable ; ouvrant la voie de la cryptographie dite à clef publique.

L'échange de clef. Le premier résultat en ce sens est dû à Ralph C. Merkle en 1974 (publié un peu plus tard [Mer78]) ; il permet à deux parties (Alice et Bob) de s'échanger efficacement une clef qui ne sera connue que d'eux, bien qu'ils communiquent à travers un canal non confidentiel. L'idée est la suivante : considérons tout d'abord un message m_1, que nous chiffrerons avec une petite clef k_1, obtenant un chiffré c_1 ; la clef étant choisie petite afin qu'il soit possible, bien que coûteux de faire une attaque exhaustive. Le chiffré c_1 est alors un *puzzle*, et sa résolution révèle le message m_1. Bob génère ainsi n puzzles, $c_1 \dots c_n$; et les envoie à Alice. Alice choisit alors l'un des puzzles, disons le i-ème, le résout, et obtient m_i. Chaque puzzle résolu m_i contient un identifiant aléatoire unique v_i, et une grande clef K_i. Alice retourne à Bob l'identifiant v_i ; et les deux parties utiliseront ensuite la grande clef K_i pour communiquer confidentiellement.

Alice n'a résolu qu'un seul puzzle, par contre un attaquant voulant retrouver la clef devra en résoudre n, nécessitant donc beaucoup plus de temps de calcul. Précisément, si générer un puzzle a un coût 1, le résoudre un coût ℓ, le temps de calcul de Alice et Bob est de $n + \ell$, alors que celui d'un attaquant serait de $n\ell$. Malheureusement un tel écart de 1 à n n'est pas suffisant : pour la cryptographie on requiert généralement un écart exponentiel ; néanmoins, cela confirme que l'idée de Ellis, bien qu' apparemment paradoxale est loin d'être absurde !

Il ne faudra attendre que deux ans avant que cette idée soit améliorée pour obtenir enfin un écart (quasi-) exponentiel entre le temps de calcul nécessaire aux parties légitimes et l'attaquant, avec le protocole d'échange de clef de Bailey W. Diffie, Martin E. Hellman [DH76] (et Ralph C. Merkle[1]).

L'idée novatrice est de s'appuyer sur une structure mathématique algébrique ; jusqu'alors toute forme de structure était évitée en cryptographie de peur qu'elle ne mène à des cryptanalyses. Cette structure

1. Usuellement, ce protocole est attribué à Diffie et Hellman seuls ; cependant, Martin Hellman affirme en 2002 : " *The system has since become known as Diffie–Hellman key exchange. While that system was first described in a paper by Diffie and me, it is a public key distribution system, a concept developed by Merkle, and hence should be called 'Diffie–Hellman–Merkle key exchange' if names are to be associated with it. I hope this small pulpit might help in that endeavor to recognize Merkle's equal contribution to the invention of public key cryptography.*" De plus, le brevet associé (U.S. Patent 4,200,770) crédite Diffie, Hellman et Merkle.

algébrique permet en effet des attaques plus rapides que la recherche exhaustive, mais l'essentiel est que ces attaques restent exponentielles.

Soit p un nombre premier, et $g < p$ en entier quelconque (ou presque), établi à l'avance, mais ne nécessitant pas d'être secret. Le protocole se déroule ainsi :

Alice		**Bob**
choisit a aléatoirement		choisit b aléatoirement
calcule $A = g^a \bmod p$		calcule $B = g^a \bmod p$
	$\xrightarrow{A}$	
	$\xleftarrow{B}$	
calcule $K = B^a \bmod p$		calcule $K = A^b \bmod p$

À la fin du protocole, les deux parties se sont mises d'accord sur une clef commune $K = g^{ab}$. L'attaquant à seulement eu accès à $A = g^a$, et $B = g^b$, et peut certes calculer certaines combinaisons, telles que $A \cdot B = g^{a+b}$. Par contre, pour retrouver g^{ab} la seule méthode connue est de retrouver a à partir de $A = g^a$ (ou symétriquement b à partir de $B = g^b$) c'est-à-dire résoudre le problème du *logarithme discret*; les seuls algorithmes connus pour résoudre ce problème sont super-polynomiaux.

Fonction à Trappe, Chiffrement à clef publique. En 1977, Ronald Rivest, Adi Shamir et Leonard Adleman proposent le premier algorithme de chiffrement à clef publique RSA [RSA78]. L'échange de clefs vu précédemment, ne permet à deux parties que de s'accorder de façon confidentielle sur une clef commune aléatoire, qui servira de clef privée à la suite des échanges; le chiffrement à clef publique RSA permet directement la transmission d'un message confidentiel, et ne nécessite pas d'interaction après la publication de la clef publique. Plus précisément, Alice souhaitant pouvoir recevoir des messages confidentiels, choisit deux grands nombres premiers p et q, qui constituent sa clef privée; de ces nombres elle déduit la clef publique $N = pq$, en calculant un simple produit. Ainsi retrouver la clef privée à partir de la clef publique est exactement le problème de la factorisation, pour lequel aucun algorithme efficace n'est connu, et ce malgré des décennies de recherche; il est raisonnable de croire qu'aucun algorithme efficace n'existe!

Cet entier N sert de paramètre à une fonction dite *à trappe* : $f_N(x) = x^e \bmod N$, pour une valeur de e partagée par tous, disons $e = 17$. Ainsi, connaissant N, il est possible à tous de calculer $y = f_N(x)$ pour n'importe quelle valeur de x, mais inverser la fonction, c'est-à-dire retrouver x à partir $y = f_N(x)$ ne semble être possible (au sens calculable efficacement) qu'en connaissance de la trappe : la factorisation de N en nombres premiers p et q. Précisément, connaissant p et q, on retrouve la taille (l'ordre) du groupe multiplicatif modulo N, par la fonction indicatrice d'Euler $\phi(N) = (p-1)(q-1)$. Cette structure de groupe assure (pour presque tout choix de e) l'existence d'un entier d, tel que pour tout x, $(x^e)^d = x^{ed} = x \bmod N$, et on peut retrouver ce d à partir de $\phi(N)$ et e, et donc de la factorisation p, q.

Preuves de Securité. Il reste cependant une question essentielle : la difficulté de factoriser N suffit-elle à garantir la confidentialité de ce schéma de chiffrement ? Cette question reste un problème ouvert : aucune méthode autre que la factorisation n'est connue à ce jour pour s'attaquer à ce chiffrement, mais les seules preuves établissant formellement un lien entre la confidentialité du schéma RSA et la difficulté de la factorisation ont été établies dans des modèles de calcul restreints.

Deux ans après la publication de RSA, Michael O. Rabin propose un autre cryptosystème [Rab79a], cette fois-ci vraiment *basé* sur la factorisation; au sens où il prouve mathématiquement que s'il existe un algorithme efficace compromettant la confidentialité de ce nouveau schéma, alors il existe un algorithme efficace de factorisation. De telles preuves mathématiques sont appelées réductions; on suppose l'existence d'un attaquant efficace, et l'on construit un algorithme qui fait appel à cet attaquant comme d'une sous-routine, par exemple ici pour résoudre le problème de la factorisation.

La sécurité offerte par le schéma de Rabin se limite cependant à une garantie de confidentialité limitée : en supposant que la factorisation est un problème insoluble en pratique, nous sommes assurés qu'il est impossible à un attaquant (passif) de retrouver l'intégralité du message clair en voyant un chiffré. Rien cependant n'assure qu'il ne soit pas capable de retrouver une portion de l'information contenue dans le message.

Un modèle bien plus général d'attaque sera proposé en 1982 par Shafi Goldwasser et Silvio Micali [GM82], appelé sécurité sémantique (ou indistinguabilité) assurant qu'aucune information même partielle sur le message ne soit calculable par l'attaquant. Cet article est souvent considéré comme l'article fondateur de la sécurité prouvée, sa méthodologie, incluant la définition d'un modèle d'attaque

pour une preuve de sécurité a depuis été repris par la majorité des constructions de cryptographie à clef publique : à ce titre, ils ont reçu en 2012 le prix Turing. Notons que les auteurs de RSA ont eux aussi été lauréats de ce prix, ainsi que Michael Rabin qui a partagé un prix Turing avec Dana Scott pour ses travaux sur les machines non déterministes.

Les attaques actives. Ces premières techniques de chiffrement asymétrique permettent donc à deux parties de communiquer de façon confidentielle en utilisant un canal non confidentiel. Cependant il est nécessaire de supposer que le canal garantisse l'intégrité des échanges : ces schémas résistent aux attaques dites passives (l'attaquant ne fait qu'écouter les données en transit, sans pouvoir les modifier), mais, sans autres contre-mesures, ils sont vulnérables aux attaques actives.

Par exemple, il existe une attaque active simple sur le protocole d'échange de clef Diffie-Hellman ; elle utilise le fait que l'on établisse effectivement une clef confidentielle avec *quelqu'un* à l'autre bout du canal ; mais rien ne garantit que cela soit avec la personne souhaitée. Ainsi, un attaquant, disons Charlie, qui aurait accès activement à ce canal, pourrait s'immiscer dans le processus de la façon suivante :

Alice		**Charlie**		**Bob**
choisit a aléatoirement		choisit c aléatoirement		choisit b aléatoirement
Calcul $A = g^a$		Calcul $C = g^c$		Calcul $B = g^a$
	$\xrightarrow{A}$		$\xrightarrow{C}$	
	$\xleftarrow{C}$		$\xleftarrow{B}$	
Calcul $K_1 = C^a$		Calcul $K_1 = A^c$ et $K_2 = B^c$		Calcul $K_2 = C^b$

Alors que Alice et Bob espéraient partager une clef confidentielle entre eux, ils partagent chacun une clef différente avec Charlie ! Il ne reste plus à Charlie qu'à déchiffrer tous les messages qu'il reçoit d'un coté et de les re-émettre chiffrés avec l'autre clef pour cacher sa présence tout en ayant accès à l'information déchiffrée.

La conclusion est que garantir l'identité des parties et l'intégrité des données échangées est généralement un prérequis pour assurer la confidentialité.

Identification et signature. Une solution au problème de l'identification est proposée par Uriel Feige, Amos Fiat et Adi Shamir en 1988 [FFS88] ; et la sécurité du schéma est basée sur le problème de la factorisation. Alice va prouver son identité à un vérifieur, Bob, par la connaissance d'un secret ; on souhaite cependant que personne d'autre qu'Alice ne connaisse ce secret, pas même Bob, auquel cas il pourrait lui-même se faire passer pour Alice auprès d'un tiers. C'est par un protocole interactif que cette identification a lieu, Bob ne connaissant qu'une clef publique pour la vérification ; cela se distingue de l'équivalent de l'identification de type symétrique par mot de passe, où le vérifieur Bob connaît le même secret qu'Alice.

Un tel protocole constitue *une preuve interactive de connaissance à divulgation nulle*, c'est-à-dire qu'Alice prouve la connaissance d'une clef secrète, en ne révélant aucune information nouvelle sur sa clef secrète ! Cette propriété, qui peut sembler paradoxale, s'établit par un argument de simulation : il est possible sans connaissance du secret de produire un faux transcript du déroulement normal du protocole, en jouant les deux rôles à la fois, celui d'Alice et de Bob. Cela prouve qu'obtenir un tel transcript n'apporte aucune information supplémentaire, puisque ce transcript peut être généré par n'importe qui. Cependant, produire de tels transcripts n'aide en rien à se faire passer pour Alice, car durant une identification réelle auprès de Bob, l'attaquant n'a plus le contrôle sur les choix secrets faits par Bob.

Un tel protocole permet donc l'identification d'une partie, une application directe est par exemple l'ouverture ou non d'un accès, physique ou informatique. Cependant, Fiat et Shamir proposent aussi une technique pour transformer un tel schéma d'identification en schéma de *signature*. Alors qu'une identification ne permet que de garantir son identité, la signature permet de lier un message choisi à son signataire c'est-à-dire garantir à la fois l'authenticité (l'identité du signataire) et son intégrité. Là encore, les arguments de simulation sont essentiels pour les preuves de sécurité. Par contre la signature n'étant pas un protocole interactif, l'argumentation de sécurité se place dans un modèle particulier, dit modèle de l'oracle aléatoire, et dont la simulation prendra le contrôle.

Infrastructure à clefs publiques. Les outils cryptographiques que nous venons de décrire sont les briques de base de la sécurité des communications, notamment sur le réseau internet. Lorsqu'un internaute souhaite se connecter, à un site de paiement en ligne, plusieurs garanties sont nécessaires : il faut premièrement que les données soient chiffrées afin que les identifiants bancaires ne soient pas interceptés

par un tiers malhonnête, mais il faut aussi que l'internaute soit sûr d'établir des communications avec le site de son choix, et non un attaquant actif. Ainsi, pour la première condition, il semble suffire que le site de paiement publie une clef publique de chiffrement, mais pour la seconde, il faut garantir que cette publication n'ait pas été altérée avant que l'internaute ne l'utilise.

Pour ce faire, on utilise une infrastructure à clefs publiques ; des autorités de certifications, aussi appelés tiers de confiance, certifient les clefs publiques de chiffrement des sites internet avant qu'elles ne soient utilisées par les internautes. Ainsi, l'internaute n'a besoin de connaître au préalable qu'une seule clef publique : celle permettant la vérification des signatures de l'autorité de certification, généralement pré-enregistrée dans le navigateur internet ; les sites internet quant à eux doivent faire certifier leur clef publique de chiffrement auprès de cette autorité par une signature-certificat. Il ne reste ainsi au site qu'à envoyer à l'internaute une clef publique de chiffrement associée à ce certificat ; la signature est vérifiée par le navigateur, assurant ainsi que la clef publique de chiffrement appartient bien au site souhaité. En utilisant cette clef publique certifiée, l'internaute a donc les deux garanties requises : l'identification du site internet et la confidentialité des données échangées.

Néanmoins, les infrastructures à clefs publiques sont relativement lourdes à mettre en place, surtout à une échelle globale ; elles nécessitent des serveurs dédiés, de nombreuses interactions, et la révocation de certificat reste un problème majeur pour lequel il n'y a pas de solution idéale. Simplifier cette logistique est l'une des motivations principales à l'invention de techniques de chiffrement à gestion de droits d'accès.

Chiffrement à gestion de droits d'accès. Bien que ces outils cryptographiques suffisent à répondre aux problèmes principaux de sécurité sur un réseau non sécurisé, certains scénarios requièrent de nouvelles fonctionnalités. Une restriction est que le chiffrement ou la signature seule n'intègrent pas de façon native de structures hiérarchiques permettant une gestion fine des droits d'accès ; si l'on veut chiffrer un message pour plusieurs personnes, il faut le chiffrer séparément pour chacune d'entre elles. Dans bien des cas, il serait beaucoup plus efficace de chiffrer le message une seule fois, en l'associant à une règle qui définit une règle d'accès ; surtout l'on souhaiterait que cette règle soit garantie par les constructions cryptographiques elles-mêmes, sans passer par un tiers donnant ou non l'accès (et qui donc aurait tous les droits). Le premier pas dans cette direction a lieu en 2001, lorsque Dan Boneh and Matthew K. Franklin proposèrent un schéma de chiffrement basé sur l'identité [BF03], résolvant un problème ouvert par Adi Shamir en 1984. Leur construction s'appuie sur le couplage de Weil, qu'Antoine Joux [Jou04] avait suggeré un an plus tôt comme puissant outil pour la construction de nouveaux protocoles. Pour ces travaux, Boneh, Franklin et Joux ont reçu en 2013 le Prix Gödel.

Cette première version n'offre pas encore de structure fine d'accès, mais simplifie les échanges entre l'autorité et les parties par rapport à l'infrastructure à clefs publiques. Le point essentiel pour la suite, est que les clefs ne sont plus choisies complètement indépendamment les unes des autres, mais toutes générées par une autorité unique, ce qui permet d'intégrer des structures de gestion de droits au sein même des clefs. La première évolution sera le chiffrement basé sur l'identité hiérarchique, permettant de chiffrer un message de telle façon qu'il soit déchiffrable par une entité précise mais aussi par tous ses supérieurs hiérarchiques. Ont suivi de nombreuses variations offrant de nouvelles règles d'accès de plus en plus fines. Pendant dix ans, presque toutes ces constructions étaient basées sur les couplages de courbes elliptiques ; plus récemment, nombre d'entre elles ont été adaptées aux réseaux euclidiens. Ce nouvel objet mathématique, le réseau euclidien, semble d'ailleurs encore plus puissant ; en cette année 2013 a été proposé par Sanjam Garg, Craig Gentry et Shai Halevi [GGH13] une construction cryptographique dite multi-linéaire (contrairement aux couplages qui ne sont que bi-linéaires), et qui permet d'encoder dans le chiffrement lui-même n'importe quelle règle régissant les autorisations de déchiffrement, pour peu qu'elle soit exprimable par une formule calculable.

Chiffrement Homomorphe Une autre direction de recherche ouvrant de nouvelles applications est le chiffrement dit homomorphe, c'est-à-dire un chiffrement qui préserve certaines structures arithmétiques entre les clairs et les chiffrés. L'exemple le plus simple est en fait le chiffrement RSA [RSA78] vu plus haut. Deux chiffrés $c_1 = x_1^e \bmod N, c_1 = x_2^e \bmod N$ peuvent être multipliés, et le résultat vérifie $c_1 \cdot c_2 = (x_1 \cdot x_2)^e \bmod N$, c'est-à-dire que le produit de deux chiffrés est un chiffré valide du produit des deux messages clairs. Dans beaucoup de contextes, notamment en présence d'attaques actives, cette propriété est considérée comme une faiblesse, car la préservation de structure peut permettre à un attaquant d'extraire de l'information ; des contre-mesures brisant ces structures sont ajoutées.

Cependant, cette structure peut aussi être un atout considérable pour de nouveaux scénarios cryptographiques, car elle autorise un tiers à effectuer des opérations sur des données chiffrées sans qu'il soit pour autant capable capable de les déchiffrer. Un exemple d'application est le vote électronique : chaque

vote est chiffré puis publié, on additionne le contenu de tous les bulletins sans les déchiffrer ; calcul qui peut être publiquement vérifié pour éviter certaines fraudes. C'est seulement ce résultat final qui sera déchiffré, révélant uniquement le résultat de l'election, mais pas les votes individuels.

Outre RSA qui autorise la multiplication, d'autres schémas comme celui de Pascal Paillier [Pai99] publié en 1999 permette l'addition. Il faudra attendre les travaux de Craig Gentry [Gen09] en 2009 pour voir apparaître un schéma crédible de chiffrement qui autorise à la fois l'addition et la multiplication ; et le fait de pouvoir conjointement utiliser ces deux opérations autorise en fait d'effectuer n'importe quelle opération sans déchiffrer les données ! Cette fois encore, c'est la structure de certains réseaux euclidiens qui permet une telle construction. Cela rend par exemple possible pour deux personnes d'optimiser un partage équitable de ressources, sans révéler à l'autre ses propres préférences ; ou encore – paradoxe ultime ! – d'effectuer une recherche sur internet, sans que le moteur de recherche n'apprenne le contenu de notre requête.

Ces derniers résultats furent d'abord très théoriques, dans la mesure où les calculs et les tailles des messages chiffrés semblaient astronomiques ; mais les améliorations successives pourraient permettre à une telle technologie d'être appliquée dans un futur proche.

1.2 Outils mathématiques et informatiques

La cryptographie asymétrique s'appuie sur de nombreux concepts mathématiques, et d'informatique théorique. En particulier, les preuves de sécurité se font donc selon le formalisme mathématique, et la notion de difficulté provient de l'informatique théorique.

La notion de difficulté La notion d'algorithme, ou de calcul automatisé à été formalisée par Turing, grâce à son modèle mathématique d'ordinateur, *la machine de Turing*. Sans entrer dans les détails techniques, la machine de Turing est constituée d'un ruban (suite de cases dans lesquels peuvent être écrits, lus et effacés des symboles), une tête de lecture/écriture (se deplaçant sur le ruban pour effectuer ces opérations) ainsi qu'un ensemble d'états et de règles de transitions sur ces états en interaction avec la tête de lecture/écriture. Bien qu'ils soient aujourd'hui bien plus complexes, les ordinateurs entrent dans ce modèle, au sens où tout ordinateur peut être transcrit en une telle machine. La thèse de Church-Turing (énoncé plus métaphysique que mathématique) stipule que tout processus calcul physique rentre dans ce modèle.

Muni de ce modèle, Turing définit les fonctions mathématiques calculables comme les fonctions f pour lesquelles il existe une machine de Turing, qui démarre avec une valeur x codée sur le ruban, et qui s'arrête après un nombre d'étapes (temps) fini, ait écrit $f(x)$ codé sur ce même ruban. Il démontre au passage, qu'il existe des fonctions qui ne sont pas calculables [2]. Par la suite, la théorie s'intéressera à la notion de complexité, c'est-à-dire, pour une fonction calculable, combien d'étapes faut-il à une machine de Turing pour la calculer. C'est ainsi que sont définies un grand nombre de classes de complexités (parfois appelées zoo) ; deux classes sont particulièrement interessantes. La classe P, des fonctions dont le calcul peut être effectué en un nombre d'étapes au plus polynomial en la taille de l'entrée, et la classe NP (pour Non-deterministe Polynomial) des fonctions dont le résultat peut être *verifié* en temps polynomial. La question de savoir si ces deux classes sont égales ou différentes est le plus important problème d'informatique théorique à nos jours ; il siège auprès de la conjecture de Riemann dans la liste des 7 problèmes du millénaire de l'institut Clay.

De plus, dans la classe NP certains problèmes sont dits NP-complets, ce qui signifie qu'ils sont au moins aussi durs que tous les autres problèmes NP, "A est au moins aussi dur que B" signifiant que s'il existe un algorithme polynomial pour le problème A alors il existe un algorithme polynomial pour le problème B. Autrement dit, si un seul problème NP-complet admet un algorithme en temps polynomial, alors tous les autres problèmes NP aussi, c'est-à-dire $\mathsf{P} = \mathsf{NP}$. Étant donné que l'avis majoritaire est que $\mathsf{P} \neq \mathsf{NP}$, de tels problèmes sont considérés comme durs, et des instances suffisamment larges seront impossibles à résoudre en pratique. Les premiers résultats de NP-complétude ont été établis par Cook et Levin [Coo71], la technique de preuve est appelée réduction. Pour montrer que A est au moins aussi dur que B, on montre qu'on peut transformer efficacement une instance du problème B en une instance de A, de telle façon que la réponse à l'instance de A soit la même que (ou puisse se transformer efficacement

2. La technique de sa preuve s'appuie sur le paradoxe de Russell, ou paradoxe du barbier de Séville : si le barbier de Séville rase tous les gens qui ne se rasent pas eux-mêmes, le barbier se rase-t-il ? Plus précisement, il définit la fonction f (qui correspond au fait de se raser soi-même), qui à un programme x associe 1 s'il s'arrête en temps fini, 0 sinon. Si f était calculable, alors on pourrait construire le programme x (qui correspond au barbier) suivant : le programme x demande le calcul de $f(x)$, si le résultat est 0 alors il s'arrête, sinon il entre dans une boucle infinie.

en) la réponse à B. Ainsi, un algorithme pour résoudre A s'il existait, se transformerait en un algorithme résolvant B.

Preuves de Sécurité En ce qui concerne la cryptographie, nous nous appuyons sur le même type d'argumentation pour démontrer la difficulté de casser un schéma cryptographique. Cependant nous devons pour le moment nous contenter de notions de difficulté plus faible, car nous ne connaissons que très peu de primitives cryptographiques se réduisant à des problèmes NP-complets. Ainsi, nous considérerons difficiles les problèmes dont nous pensons qu'ils ne sont pas résolubles en temps polynomial, c'est-à-dire des problèmes dont on pense qu'ils ne sont pas dans P [3]. Bien sûr, en pratique nous nous intéresserons plus précisément au temps approximatif des meilleurs algorithmes connus.

Les preuves de sécurité sont cependant des réductions plus complexes que celle de NP-complétude, car la notion de sécurité d'un schéma cryptographique avancé ne peut se résumer au simple calcul d'une fonction f sur une entrée x. En effet, surtout dans les modèles d'attaques actifs, un attaquant peut interagir avec les différents participants, et obtenir à son gré, des réponses à des requêtes de son choix. Ainsi, la première étape d'une preuve de sécurité consiste à définir le modèle d'attaque, c'est-à-dire les règles d'interaction entre lui et les autres parties, ainsi qu'une condition énonçant si son attaque est réussie (victoire) ; un tel modèle est appelé *jeu*. La preuve de sécurité consistera ensuite en une réduction de ce jeu à un autre problème supposé difficile, idéalement non-interactif (c'est-à-dire au calcul d'une fonction supposée difficile).

Enfin, contrairement à la NP-complétude, qui est une notion de difficulté en pire cas, on s'intéresse pour la cryptographie à la difficulté dans le cas moyen. Autrement dit, la NP-complétude garantit qu'il existe des instances du problème qui vont être difficile à résoudre, mais il est possible pour certains problèmes qu'une instance choisi au hasard ne soit pas aussi difficile. Par exemple, le problème de la factorisation n'est pas aussi difficile pour un entier N choisit au hasard, qui en moyenne aura de nombreux petits facteurs faciles à trouver, que pour un entier $N = pq$ ou p et q sont deux grands nombres premiers vérifiants certaines bonnes conditions. Ainsi, en cryptographie, il faut obtenir la difficulté en moyenne d'un problème, c'est-à-dire que le problème reste dure avec très forte probabilité pour une clef choisie au hasard, quitte à, comme pour RSA restreindre l'espace des instances.

Algorithmes quantiques et ordinateur quantique Entre la classe P et la classe NP s'intercale une autre classe, QP, qui se définit comme l'ensemble des problèmes solubles en temps polynomial sur *un ordinateur quantique*. Précisément, on définit un modèle de calcul dont les états et les opérations sont les axiomes de la mécanique quantique. D'un point de vue théorique, cela permet de stocker dans des registres mémoires des superpositions d'un nombre exponentiel de valeurs, et d'effectuer des opérations sur tous ces états superposés à la fois. Cela ne correspond cependant pas à faire les calculs en parallèle sur un nombre exponentiel de machines ; car une fois des calculs effectués sur une superposition d'états, il n'est possible d'extraire qu'une petite fraction de l'information calculée.

Il existe des problèmes pour lesquels on connaît des algorithmes polynomiaux sur machine quantique mais pas sur machine classique ; c'est notamment le cas de la factorisation et du logarithme discret, grâce à l'algorithme de Schor [Sho97]. Pour d'autres problèmes, on ne connaît pas d'algorithmes plus efficaces sur machine quantique que classique. Dans l'état actuel des connaissances, on sait que $\mathsf{P} \subset \mathsf{QP} \subset \mathsf{NP}$, et l'on peut raisonnablement croire que chacune de ces inégalités est stricte.

Parmi les différents problèmes que nous allons voir pour baser la cryptographie, il en existe certains pour lesquels on ne connaît pas d'algorithmes quantiques efficaces. C'est notamment le cas des problèmes sur les réseaux euclidiens, qualité souvent mise en avant par la communauté. Cet argument est non négligeable, mais limité. Pour certains, il s'agit d'un fait absolument rassurant concernant leur securité. Il est cependant délicat de prétendre qu'il n'existera jamais d'algorithme quantique polynomial pour tel ou tel problème (à moins que le problème ne soit NP-complet) dans la mesure où l'algorithmique quantique est très différente de l'algorithmique classique, et qu'assez peu de chercheurs y sont vraiment familiers.

Il est cependant peu probable que l'ordinateur quantique apparaisse du jour au lendemain ; et il n'est pas impossible que les limites de la physique le rendent irréalisable quelles que soient nos avancées technologiques. Ainsi, l'existence d'un algorithme quantique est avant tout un fait inquiétant sur la structure d'un problème, potentiellement symptôme de l'existence d'algorithmes classiques sous-exponentiels voire quasi-polynomiaux.

3. Plus précisément aux problèmes qui ne sont pas dans BPP, classe qui inclut les algorithmes probabilistes qui résolvent un problème donné avec bonne probabilité en temps polynomial

En d'autres termes, là où la NP-complétude apporterait une réponse quasi-définitive à la sécurité d'un schéma, l'absence d'algorithmes quantiques n'est qu'un argument relatif ; et les risques liés à l'existence d'algorithmes quantiques ne sont pas uniquement conditionnés par l'avénement de l'ordinateur quantique.

1.2.1 Les problèmes utilisés en cryptographie

La factorisation Le problème de la factorisation est sans doute le problème le plus connu pour ses applications en cryptographie, c'est sur lui qu'est "basé" le premier schéma de chiffrement à clef publique, RSA [RSA78], ainsi que son dual, le schéma de signature RSA. Comme nous l'avons détaillé auparavant, on ne connaît pas pour RSA de réduction assurant que ces schémas sont aussi durs que la factorisation elle-même. Cependant, le schéma de Rabin bénéficie lui d'une telle garantie de sécurité. Au-delà du simple chiffrement, il existe aussi un IBE (schéma de chiffrement basé sur l'identité), proposé par Cocks en 2001 [Coc01] ; mais aucune construction à gestion fine d'accès n'est connue à ce jour. De même, dans la direction du chiffrement homomorphe, RSA est naturellement homomorphe pour la multiplication, et la variante proposée par Paillier [Pai99] est elle additivement homomorphe.

Bien que RSA ait ouvert la voie de la cryptographie asymétrique, ces techniques ne sont plus considérées aujourd'hui comme idéales à bien des égards. Du côté de la sécurité d'abord, car les meilleures attaques connues tournent en temps $\approx 2^{\ell^{1/3}}$ pour des nombres premiers de taille ℓ, obligeant à prendre des tailles de clefs cubiques en le paramètre de sécurité $\ell = \mathcal{O}(\lambda^3)$. Grâce au choix d'un petit exposant e, le chiffrement (ou la vérification de signature) se fait en $\tilde{\mathcal{O}}(\lambda^3)$ et relativement efficace en pratique, le déchiffrement (ou la signature) est lui beaucoup plus lent, en $\tilde{\mathcal{O}}(\lambda^6)$. Au-delà de la complexité théorique, les opérations effectués sont très peu efficaces sur des petites architectures. Enfin, les garanties théoriques de sécurité sont assez faibles ; cependant la confiance en ce schéma avec les paramètres actuels est plutôt bonne, du fait que ce schéma a été cryptanalysé depuis maintenant plus de 35 ans. Les cryptanalyses actuelles se concentrent maintenant sur des modèles d'attaques plus puissants, notamment lorsque la clef secrète est partiellement révélée, par exemple grâce au canaux auxiliaires (mesure de consommation du processeur) voire à l'injection de faute.

Le logarithme discret La cryptographie basée sur le logarithme discret offre une plus grande variété d'instanciation ; en effet le logarithme discret peut se définir sur n'importe quel groupe abélien fini (idéalement un groupe cyclique). Essentiellement, les groupes utilisés sont les groupes multiplicatifs $\mathbb{Z}_p^*$ du corps fini $\mathbb{F}_p$ pour p premier ; il s'agit d'un groupe abélien fini d'ordre $p-1$, il est donc isomorphe au groupe additif $\mathbb{Z}_{p-1}$, cependant on ne sait efficacement calculer cet isomorphisme que dans un sens : $f : x \in \mathbb{Z}_{p-1} \mapsto g^x \in \mathbb{Z}_p^*$ pour g un générateur de $\mathbb{Z}_p^*$. Pour un bon choix de p, on ne sait pas inverser efficacement cet isomorphisme. Cependant, les attaques sur ces groupes sont du même ordre que celle de la factorisation ; cela n'offre donc pas de gain de performance pour un même niveau de sécurité.

Plus récemment a été introduit un autre type de groupe issu de la géométrie algébrique : les courbes elliptiques. Pour un même ordre p, il existe un grand nombre de courbes elliptiques sur des corps finis ayant cet ordre ; et pour certaines d'entre elles, on ne connaît aucun "raccourci" pour le logarithme discret ; c'est-à-dire que les meilleurs algorithmes connus sont génériques, et à peine meilleurs que la recherche exhaustive (Baby-Step-Giant-Step et ρ-Pollard). Autrement dit, pour ces courbes on peut choisir des tailles de clef proportionnelles au paramètre de sécurité, précisement deux fois ce paramètre. Cela permet donc des systèmes beaucoup plus compacts, mais pas extrêmement plus rapide que RSA, car les opérations sont plus complexes. Cette cryptographie à base de courbes offre beaucoup de souplesse – et c'est une des raisons de son succès – car la construction de protocole se fait en "boîte noire" ; la communauté a pu ainsi séparer les problèmes de comment construire des courbes efficaces et sûres d'un côté ; et de comment les utiliser pour construire de nouveaux cryptosystèmes de l'autre.

Le logarithme discret en presence de couplages Les couplages (de Weil ou de Tate) ont d'abord été introduits pour la cryptanalyse. Pour certaines courbes elliptiques, on peut définir un couplage – une fonction bilinéaire – c'est-à-dire une fonction e non triviale verifiant $e(g^a, h^b) = e(g,h)^{ab}$ qui se calcule efficacement. Bien qu'il soit difficile de résoudre le problème du logarithme discret sur ces courbes, la sécurité d'un protocole tel que l'échange de clefs Diffie-Hellman serait mis en défaut, car il devient possible de distinguer la clef échangée d'une clef aléatoire. Cependant, Antoine Joux, Puis Dan Boneh et Matthew Franklin ont proposé d'exploiter ces propriétés algébriques pour construire de nouveaux cryptosystèmes inaccessibles auparavant. C'est ainsi qu'en 10 ans, un très grand nombre de cryptosystèmes à gestion fine de droit d'accès ont vu le jour.

Cependant, la construction de courbes sûres avec couplage reste un exercice délicat, et les opérations de couplages sont encore plus côuteuses que les opérations de groupe sur une courbe. Certains choix faits pour des raisons d'efficacité se sont récemment avérés fatals aux nouveaux algorithmes de logarithme discret [BGJT13] en temps quasi-polynomial.

Le problème du sac-à-dos La factorisation, et le problème du logarithme discret ont le défaut majeur de ne pas être liés à des problèmes fondamentaux de la théorie de la complexité. Idéalement, pour se garantir de toute attaque (en supposant que $\mathsf{P} \neq \mathsf{NP}$) on souhaiterait construire des cryptosystèmes basés sur des problèmes NP-complets. Cependant, les premiers problèmes NP-complets mis en evidence étaient de nature très combinatoire, alors que la cryptographie asymétrique semble nécessiter des objets plus algébriques.

Il existe cependant un problème d'énoncé simple, nommé problème du sac-à-dos qui s'avère être NP-complet, et ayant des caractéristiques prometteuses pour la cryptographie. Il s'énonce ainsi : étant donné un ensemble de n objets, de poids respectifs $p_1, \ldots, p_n$, et un poids total t ; trouver un sous-ensemble K de ses objets dont le poids total est t. Autrement dit, trouver un vecteur binaire $(b_1 \ldots b_n)$ tel que $\sum b_i p_i = t$. Une propriété intéressante de ce problème, et très utile pour la cryptographie asymétrique, est que la version décisionnelle de ce problème est aussi dure que la version de recherche ; c'est-à-dire qu'il est aussi difficile de simplement deviner s'il existe un tel vecteur $(b_1 \ldots b_n)$ que de trouver un tel vecteur. La réduction est assez simple ; s'il existe un algorithme pour la décision ; on fait une hypothèse sur b_n, et on la teste en utilisant l'algorithme de décision. Précisement, on teste l'existence d'une solution en enlevant le dernier élement p_n, si c'est le cas on sait alors que l'on peut choisir $b_n = 0$, sinon on doit avoir $b_n = 1$; et on continue récursivement en prenant le nouveau poids total $t' = t - b_n p_n$.

Ainsi, Merkle et Hellman [MH78] proposent en 1978 un schéma basé sur certaines instances de ce problème. Malheureusement, pour rendre le déchiffrement possible, ils se voient obligés d'utiliser des instances très particulières. Ces instances s'avéreront faibles et Diffie et Shamir proposeront une attaque quatre ans plus tard [SD82].

Les problèmes de réseaux euclidiens De façon très simplifiée, les problèmes de réseaux euclidiens sont des versions généralisées du problème du sac à dos ; les poids scalaires $p_1 dots p_n, t$ sont remplacés par des vecteurs, et l'on cherche une combinaison lineaire entière, pas forcément binaire, mais ne serait-ce qu'à petits coefficients. Une section entière (Sect. 1.3) est vouée à la présentation des réseaux euclidiens, les problèmes associés et leur interprétation géométrique.

les problèmes de codes correcteur d'erreur Un code correcteur d'erreur est un espace vectoriel sur un corps fini, muni d'un sous-ensemble de points appelés mots de code ; ainsi que d'une métrique. Ils servent à détecter voire corriger les erreurs de transmission sur un canal bruité. L'émetteur n'est censé envoyer que des mots de code, ainsi si une partie de l'information est modifiée pendant la transmission, il est fort probable que le message reçu ne soit pas un mot du code. L'objet de cette théorie est de concevoir des codes qui permettent de retrouver le mot d'origine en supposant que la transmission n'a provoqué qu'un petit nombre d'altérations. Ce sujet essentiel à l'informatique à été très étudié, et on a montré notamment qu'un code linéaire choisi aléatoirement (donc très mal conçu) rend ce problème NP-complet. Cela a donné lieu à un certain nombre de cryptosystèmes, mais pour un niveau de sécurité convenable, ils sont pour l'instant assez peu efficaces.

Il est cependant intéressant de noter que les problèmes de réseaux et les problèmes de code ont un formalisme assez similaire ; la différence essentielle est la métrique utilisée ; les codes se basent sur la métrique de Hamming (nombre de coordonnées faussées dans le message, indépendamment des fautes elles-mêmes), tandis que les problèmes de réseaux s'appuient généralement sur la norme euclidienne.

Les systèmes d'equations multivariés Pour conclure, nous mentionnons la cryptographie multivariée, qui cherche à s'appuyer sur la difficulté de résoudre des systèmes d'équations polynomiales à plusieurs variables. L'une des techniques consiste à cacher un système d'équations polynomiales facile (formes triangulaires) en le transformant selon des applications linéaires gardées secrètes. Il y a cependant peu d'arguments théoriques de sécurité (peu de garanties que ces instances sont suffisamment dures), et certaines tentatives de choix de paramètres instanciation se sont avérées trop agressives ; ouvrant la porte à des attaques certes exponentielles en théorie, mais faisables en pratique.

1.3 Les Réseaux euclidiens

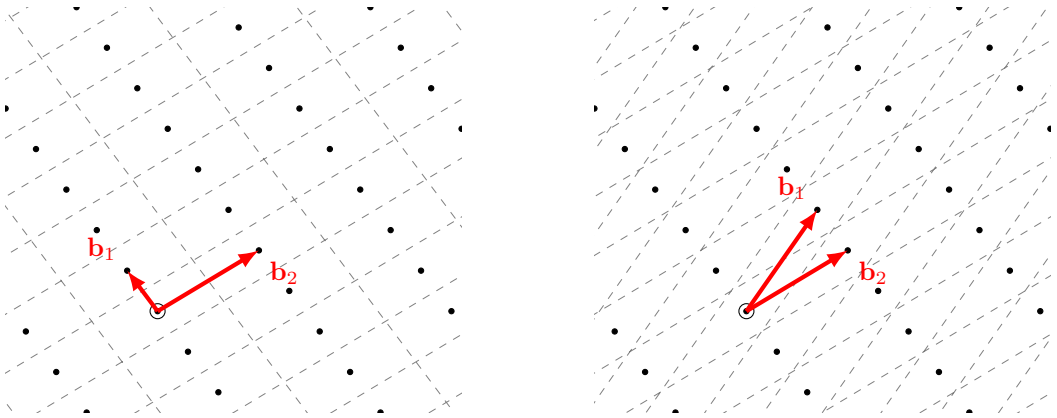
Réseaux, bases et pavages En termes mathématiques, un réseau euclidien est un sous-groupe discret d'un espace vectoriel muni de la norme euclidienne (c'est-à-dire la notion usuelle de distance vérifiant le théorème de Pythagore). De manière informelle, les réseaux euclidiens sont des ensembles de points de l'espace qui suivent un arrangement "régulier". Plus précisément, les réseaux sont des groupes : ils ont une origine $\mathbf{0}$, ils sont stables par addition et par soustraction ; autrement dit, pour un réseau L, pour tous points du réseau $\mathbf{x}, \mathbf{y} \in L$ on a $\mathbf{x} + \mathbf{y} \in L$ et $\mathbf{x} - \mathbf{y} \in L$. Ils sont de plus discrets, c'est-à-dire que tout point est isolé, mais par les propriétés de groupe il suffit que l'origine $\mathbf{0}$ soit isolée, c'est-à-dire qu'il existe un rayon r tel qu'aucun point du réseau à part l'origine ne soit à distance inférieure à r de cette origine. La plus grande valeur possible d'un tel r est appelé premier minimum du réseau, noté $\lambda_1(L)$, autrement défini par $\lambda_1(L) = \min_{\mathbf{x} \in L \setminus \{\mathbf{0}\}} \|\mathbf{x}\|$. La figure 1.1 en donne deux exemples.

FIGURE 1.1 – Exemples de réseaux

(a) Le réseau orthogonal canonique $\mathbb{Z}^2$ (b) Un réseau L quelconque

Les réseaux sont l'analogue discret des espaces vectoriels réels, en particulier ils ont des bases, c'est-à-dire un ensemble de vecteurs $\mathbf{b}_1 \dots \mathbf{b}_n$ tel que tout point du réseau s'écrive de façon unique comme une combinaison entière des vecteurs $\mathbf{b}_1 \dots \mathbf{b}_n$. Un réseau admet un nombre infini de bases différentes (excepté les réseaux de dimension 1). À chacune de ces bases, on peut associer un pavage de l'espace selon un parallélépipède associé à la base, comme montré en figure 1.2.

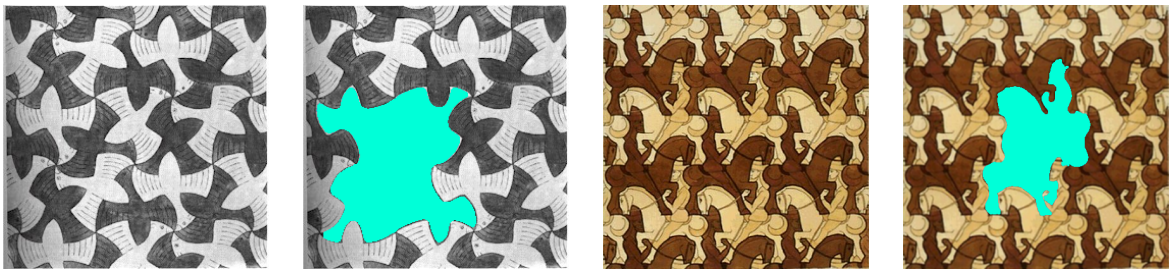
FIGURE 1.2 – Bases d'un réseau et pavages parallélépipédiques



(a) Une bonne base de L (b) Une moins bonne base de L

Cependant, les réseaux admettent d'autres pavages réguliers comme Escher le démontrait dans grand nombre de ses œuvres. De telles figures pavant l'espace selon le réseau L sont appelées domaines fondamentaux du réseau L.

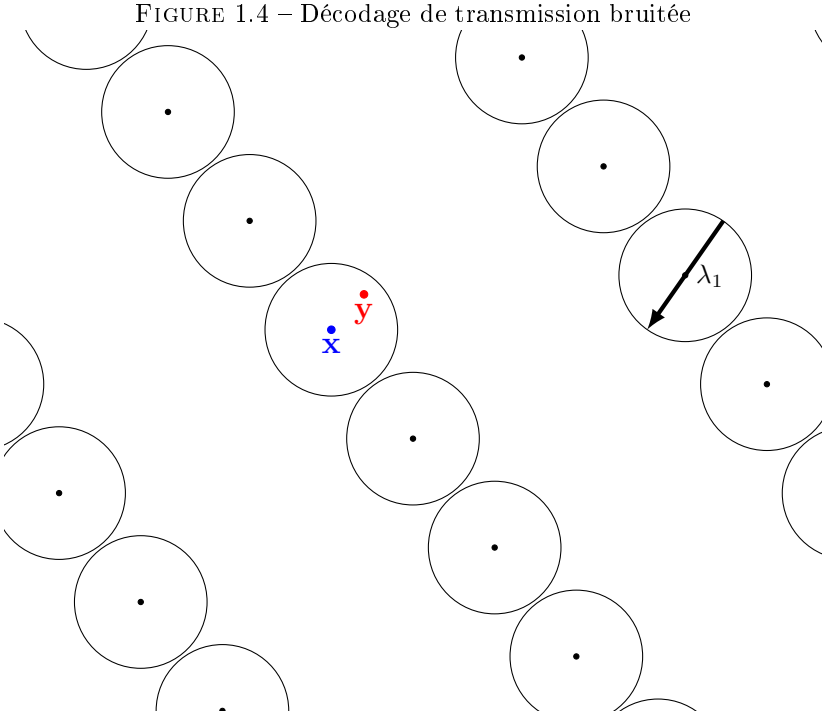
FIGURE 1.3 – Oeuvres d'Escher et domaines fondamentaux associés



Parmi tous les domaines fondamentaux se distingue la cellule de Voronoi, définie comme l'ensemble des points plus prêts de l'origine que de tous les autres points du réseau.

Codage, décodage et correction d'erreur Au-delà de leurs nombreuses applications en mathématique pure, les réseaux euclidiens s'avèrent très utiles en informatique pratique, en particulier pour les communications numériques au travers de canaux analogiques. Pour transmettre un message numérique (un élément d'un ensemble fini) on lui associe un point $\mathbf{x} \in L$ dans un réseau L fixé à l'avance. Les coordonnées de ce point sont cependant dans un espace continu, et ce sont ces coordonnées que l'on transmet à travers le canal analogique. À l'autre bout du canal, le signal reçu risque fort d'être perturbé, ainsi le récipiendaire effectue une mesure $\mathbf{y} = \mathbf{x} + \mathbf{e}$ où $\mathbf{e}$ est une petite erreur. Cependant, si on a la garantie que $\mathbf{e}$ est petit, alors il n'y a mathématiquement pas d'ambiguïté sur $\mathbf{x}$. Précisément, si $\|\mathbf{e}\| \leq \lambda_1(L)/2$, alors il n'existe qu'un seul point du réseau L possible pour $\mathbf{x}$ connaissant la valeur $\mathbf{y}$. Géométriquement, cela est vrai car les boules de diamètre $\lambda_1(L)$ centrées en les points du réseau ne s'intersectent pas.

FIGURE 1.4 – Décodage de transmission bruitée



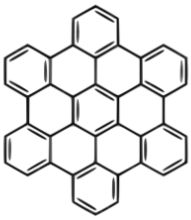
Empilements compacts et constante d'Hermite La figure 1.3 démontre ce principe. Cependant nous pouvons observer qu'une large portion du plan n'est pas couverte par les disques de décodage. Pour optimiser la quantité d'information transmise et la capacité de correction d'erreur, on souhaite minimiser cette zone inutile. C'est une des raisons pour lesquelles on s'intéresse aux empilements optimaux de sphères, c'est-à-dire les réseaux pour lesquels la densité

$$\Delta_n = \mathrm{Vol}\left(\frac{\lambda_1(L)}{2} \cdot \mathfrak{B}_n\right) \Big/ \mathrm{Vol}(L) = \left(\frac{\lambda_1(L)}{2}\right)^n \cdot \frac{\mathrm{Vol}(\mathfrak{B}_n)}{\mathrm{Vol}(L)}$$

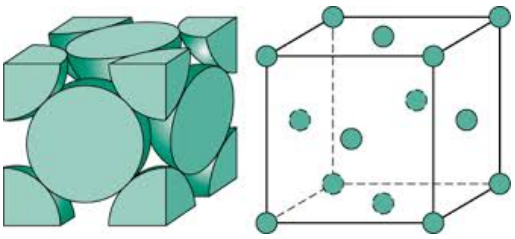
est maximale ($\mathfrak{B}_n$ dénotant la boule unité en dimension n). De façon équivalente, Hermite définit la constante éponyme $\gamma_n = \max_L \lambda_1(L)^2/\operatorname{Vol}(L)^{2/n}$ où le maximum est pris sur tous les réseaux de dimension n. La densité maximale d'un réseau de dimension n est alors de $\Delta_n^{\max} = \frac{\operatorname{Vol}(\mathfrak{B}_n)}{2^n} \cdot \gamma_n^{n/2}$. Déterminer $\Delta^{\max}$ et trouver les réseaux qui atteignent $\Delta^{\max}$ s'appelle le problème d'empilement compact des sphères en dimension n.

En deux dimensions, l'empilement optimal est atteint par le réseau hexagonal, connu depuis toujours par les abeilles pour optimiser l'espace dans leur ruche. En trois dimensions, le réseau optimal est le réseau nommé *cubique face centrée*, que l'on retrouve dans l'arrangement des atomes de nombreux métaux et cristaux. Au-delà, de trois dimensions, ce problème n'a été résolu que pour $n \leq 8$ et pour $n = 24$.

FIGURE 1.5 – Empilements optimaux en dimensions 2 et 3



(a) Structure atomique d'une couche de graphène (réseau hexagonal)



(b) Structure atomique du diamant, du sel *etc.* (réseau cubique face centré)

Retrouver un proche vecteur Même pour des dimensions n dont on connaît un empilement compact de sphères, il est difficile de parvenir à une correction d'erreur optimale en pratique. En effet, bien que, comme montré en figure 1.3, il n'y ait mathématiquement qu'une seule solution à $\mathbf{y} = \mathbf{x} + \mathbf{e}$ connaissant $\mathbf{y}$, avec $\mathbf{x} \in L$ et $\mathbf{e}$ petit, trouver une telle solution $\mathbf{x}$ peut s'avérer algorithmiquement coûteux. Plus généralement, on s'intéresse étant donné un point $\mathbf{y}$ quelconque à trouver un point proche $x \in L$, idéalement le plus proche. Il existe des algorithmes rapides (polynomiaux, voire quasi-linéaires) qui permettent de trouver une bonne approximation du vecteur le plus proche, étant donné une bonne base. Plus précisément étant donné une base $\mathbf{B}$, l'algorithme de Babai permet, étant donné un point $\mathbf{y}$ de trouver $\mathbf{x} \in L$ tel que $\mathbf{x} - \mathbf{y} \in \mathcal{P}(\mathbf{B})$, où $\mathcal{P}(\mathbf{B})$ dénote le parallélépipède engendré par $\mathbf{B}$ – comme illustré en figure 1.2. Rappelons que par invariance du volume des domaines fondamentaux, tous ces parallélépipèdes ont le même volume ; ainsi ceux qui sont les plus proches des cubes garantissent de trouver des vecteurs plus proches que des parallélépipèdes longs et fins. Autrement dit, une bonne base d'un réseau pour résoudre ce problème est une base dont les vecteurs sont à peu près tous de la même longueur et les plus orthogonaux possibles.

Trouver des bonnes bases, un problème difficile De fait, la définition mathématique des réseaux euclidiens et leur étude, remonte à bien avant ces applications au télécommunications. Ainsi, Gauss définira les bases réduite en deux dimensions, et proposera même un algorithme pour trouver de telles bases. Précisément, une base $\mathbf{b}_1, \mathbf{b}_2$ de L est dite *réduite*, si, le premier vecteur $\mathbf{b}_1$ réalise le minimum du réseau, c'est-à-dire si $\|\mathbf{b}_1\| = \lambda_1(L)$, et si $|\langle \mathbf{b}_1, \mathbf{b}_2 \rangle| \leq 1/2 \|\mathbf{b}_1\|^2$. Cette deuxième condition, appelée "size-reduction" s'obtient facilement en appliquant à $\mathbf{b}_2$ la transformation $\mathbf{b}_2 \leftarrow \mathbf{b}_2 - \left\lfloor \langle \mathbf{b}_1, \mathbf{b}_2 \rangle / \|\mathbf{b}_1\|^2 \right\rceil$. L'algorithme de Gauss consiste simplement à appliquer cette transformation, puis inverser les vecteurs $\mathbf{b}_1$ et $\mathbf{b}_2$, et répéter ces deux opérations jusqu'à convergence.

Hermite donnera une notion de réduction pour toute dimension $n \geq 1$; et de ces définitions peuvent se déduire des algorithmes pour obtenir des bases réduites à partir de bases quelconques. Cependant ces algorithmes ont un temps d'exécution exponentiel en la dimension. En 1981, van Emde-Boas [vEB81] démontrera que trouver que trouver le plus court vecteur d'un réseau est un problème NP-difficile, et donc que de trouver des bases réduites est aussi difficile. Un an plus tard, Arjen Lenstra, Hendrik Lenstra and et László Lovász [LLL82] proposeront une relaxation de la réduction selon Hermite, et démontreront que de telle base réduites sont calculables en temps polynomial grâce à l'algorithme éponyme LLL ; mais contrairement à la réduction d'Hermite, une telle base ne contient pas nécessairement le plus court vecteur du réseau. Il est cependant garantit que la base contient une approximation exponentiel du plus court vecteur, c'est-à-dire un vecteur de longueur au plus $2^n \cdot \lambda_1(L)$ (la constante 2 peux être remplacée par d'autres valeurs $c > 4/3$).

Une question naturelle est alors de se demander quelle qualité d'approximation du plus proche vecteur (ou quelle qualité de base réduite) est atteignable en temps polynomial. Pour de petits facteurs d'approximation, on sait montrer que ces problèmes sont toujours NP-difficiles ; mais même au de là, on conjecture qu'il n'est pas possible d'obtenir en temps polynomial des approximations de meilleure qualité asymptotique que celle de l'algorithme LLL.

La cryptographie à base de réseaux Il n'est cependant pas impossible de construire des réseaux dont on connait de bonnes bases ; ce qui est difficile, c'est, étant donné un réseau, en trouver une bonne base. Ainsi, il aisé est pour quiconque de choisir une bonne base puis de considérer le réseau qu'elle engendre, et de publier une mauvaise base de ce réseau. De cette façon, tout le monde peux s'accorder sur un réseau, mais seul son créateur est capable d'effectuer les opérations de décodage correctement dans ce réseau ; de la même manière que pour la construction d'un module RSA, tout le monde connait le module Z_N où $N = pq$, mais seul le créateur du module, ayant choisis p et q, est capable d'inverser des exposants. C'est ainsi que se construit la cryptographie à base de réseaux.

Bien qu'elle ne fut pas accompagnée de réduction de sécurité, la première construction cryptographique du genre est due à Goldreich, Goldwasser et Halevi [GGH97]. Pour le chiffrement, ils proposent, étant donné une mauvaise base d'un réseau, de coder un message en choisissant un point aléatoire du réseau, et en le bruitant par le message à chiffrer ; déchiffrer le message revient à résoudre un problème de correction d'erreur comme en figure 1.3. Ainsi, le récipiendaire légitime, qui connais une bonne base du réseau peux déchiffrer le message grâce à l'algorithme de Babai. Mais ne connaissant que des mauvaises bases, déchiffrer ce message sera difficile pour les attaquants.

Cette première tentative [GGH97] se révélera plutôt faible ; mais en parallèle, un cryptosystème concret NTRUEncrypt [HPS98] est proposé. Malgré l'absence de preuve de sécurité et 15 ans de cryptanalyse, il n'a toujours pas été cassé, et il demeure un des schémas de chiffrement à clef publique les plus rapides. Viendront par la suite des cryptosystèmes à la sécurité prouvée, en particulier grâce aux travaux fondateurs d'Ajtai [Ajt96, Ajt99],definnissant le problème SIS (Short Integer Solution), et établissant sa difficulté dans le cas moyens en ne supposant que la difficulté dans le pire cas pour de certains problèmes de réseaux. Une deuxième étape importante sera l'introduction du problème "learning with error" (LWE) par Regev [Reg05], qui ouvrera la voie d'un cryptographie (en théorie) très efficace, avec de nouvelles garanties de sécurité, et offrant de plus en plus de fonctionnalités.

1.4 Sujet de thèse : Les signatures fondées sur les réseaux

En plus du schéma de chiffrement a clef publique, l'article [GGH97] propose un schéma de signature "dual". Les paires de clefs sont similaires à celles du chiffrement ; c'est-à-dire que le signataire choisit une base courte $\mathbf{B}$, en déduit le réseau engendré $L = \mathcal{L}(\mathbf{B}) \subset \mathbb{R}^n$ (de rang plein), et publie une base quelconque $\mathbf{P}$ de ce même réseau ($\mathcal{L}(\mathbf{P}) = L$). Avant d'être signé, le message m est d'abord haché, et le hash est interprété comme un point de l'espace $\mathbb{R}^n$: $\mathbf{h} = H(m) \in \mathbb{R}^n$. Le signataire utilise ensuite sa base courte, pour retrouver un point du réseau $\mathbf{s} \in L$ proche de $\mathbf{h}$; précisément en utilisant l'algorithme de Babai, il trouvera l'unique vecteur $\mathbf{s} \in L$ tel que $\mathbf{s} - \mathbf{h}$ appartienne au parallélépipède $\mathcal{P}(\mathbf{B})$ (comme en figure 1.2). Pour vérifier la validité d'une signature $\mathbf{s}$ associée à un message m, on vérifie simplement que $\mathbf{s}$ appartient au réseau L (ce qui est aisé connaissant une base quelconque $\mathbf{P}$), et que $\mathbf{s}$ est proche de $\mathbf{h} = H(m)$, c'est-à-dire que $\|\mathbf{s} - H(m)\|$ est petit.

Un adversaire voulant falsifier une signature pour un message m doit donc, ne connaissant qu'une mauvaise base $\mathbf{P}$ du réseau L, retrouver un point proche du point aléatoire $\mathbf{h} = H(m)$; cette tâche s'avère difficile sans la connaissance d'une bonne base du réseau L telle que $\mathbf{B}$.

Quelques années plus tard, les auteurs de NTRUEncrypt proposent un schéma de signature [HNHGSW03], NTRUSign suivant le même procédé ; mais l'utilisation de réseaux cycliques et q-aire offre de bien meilleures performances ; des paramètres pratiques sont proposés. Cependant, aucun de ces deux schémas n'offrent de preuve de sécurité. En effet, falsifier une signature uniquement à partir de la base publique $\mathbf{P}$ est un problème difficile ; mais ces deux schémas ne sont pas *zero-knowledge* (ou *à divulgation nulle de connaissance*) ; c'est-à-dire que la distribution des signatures n'est pas indépendante de la clef secrète. Ainsi, après avoir vu un certain nombre de signatures, l'adversaire dispose d'autres informations que simplement la base $\mathbf{P}$: chaque signature laisse ainsi fuiter une information liée à la clef secrète. Pour casser ce schéma de signature, une approche est donc d'essayer de reconstituer la clef secrète à partir de cette fuite d'information.

1.4.1 Attaques

Une telle attaque à été montée par Nguyen et Regev [NR06] ; l'idée essentielle est de caractériser la distribution du vecteur $\mathbf{v} = \mathbf{s} - H(m)$; l'utilisation de l'algorithme de Babai impose que ce vecteur appartienne au parallélépipède $\mathcal{P}(\mathbf{B})$; et modélisant le vecteur $H(m)$ comme étant aléatoire dans $\mathbb{R}^n$; on obtient que $\mathbf{v}$ est uniformément aléatoire dans ce parallélépipède $\mathcal{P}(\mathbf{B})$. En s'appuyant sur des techniques d'analyse statistique, ils démontrent qu'il est possible d'apprendre ce parallélépipède en temps polynomial ayant accès à un nombre suffisant (polynomial) de tels vecteurs aléatoires $\mathbf{v}$. Ils démontrent ainsi que le schéma de signature [GGH97] est cassable en théorie ; ainsi que certaines versions NTRUSign [HNHGSW03]. En pratique, après seulement 400 signatures, cette attaque est capable de retrouver la clef secrète utilisée.

Cependant, d'autres versions de NTRUSign incluent une *contremesure* contre ces attaques statistiques ; la raison étant qu'un an plus tôt, l'attaque de Gentry et Szydlo [GS02] avait utilisé des fuites similaires pour casser le schéma NSS [HPS01], ancêtre de NTRUSign. La contremesure consiste à perturber le point cible $\mathbf{h} = H(m)$ avant l'application de l'algorithme de Babai. Précisément, ce vecteur est perturbé en appliquant une première fois l'algorithme de Babai, en utilisant une base $\mathbf{B}'$ choisie indépendamment du réseau $L = \mathcal{L}(\mathbf{B})$.

Contributions Dans le chapitre 5 nous analysons cette contremesure. Nous modélisons la nouvelle distribution du vecteur $\mathbf{v} = \mathbf{s} - H(m)$ comme la convolution des parallélépipèdes $\mathcal{P}(\mathbf{B})$ et $\mathcal{P}(\mathbf{B}')$; le domaine de cette distribution étant un *zonotope*. Nous montrons que la méthode employée par Nguyen et Regev [NR06] est susceptible d'être généraliser pour apprendre ces zonotopes. Il reste cependant quelques obstacles pour prouver formellement que notre nouvelle attaque est correcte ; mais nos expériences sont positives. En utilisant 5000 signatures, nous sommes en mesure de retrouver l'intégralité de la clef secrète.

Nous étudions de plus une autre contremesure proposée plus récemment [HWH08], qui procède à une déformation du parallélépipède de Babai ; un peu à la manière des pavages d'Escher (figure 1.3). Nous montrons que cette contremesure particulière n'est pas plus sûre que la précédente, et prouvons en théorie comme en pratique que la correction de l'attaque originale [NR06] n'est pas affectée par cette contremesure particulière. Il semble cependant intéressant de considérer la sécurité potentielle de cette approche. Nous proposons ainsi un modèle général pour analyser ces techniques de déformation, ainsi qu'une approche générale pour s'attaquer à de telles contremesures. Nous montrons la validité de notre approche sur un exemple n'ayant pas les faiblesses de la proposition de [HWH08]. Nous ne sommes pas en mesure d'attaquer en toute généralité ces techniques de déformation ; mais notre analyse suggère que cette approche est très risquée. On peut aussi voir cette analyse comme un ensemble de conditions nécessaires pour qu'une déformation soit potentiellement sûre.

1.4.2 Analyse

Intuitivement, une approche pour éviter les attaques précédentes lors d'une signature serait d'ajouter de l'aléa dans le procédé de signature ; cependant les tentatives de contremesure [HNHGSW03, HWH08] montrent qu'on ne peut pas se contenter de méthodes heuristiques de randomisation. Pour se prémunir, de façon prouvablement sûre de toute attaque par apprentissage, il est nécessaire de rendre la distribution $\mathcal{D}$ de $\mathbf{v} = \mathbf{s} - H(m)$ indépendante de la clef secrète. Le problème est donc de trouver une distribution $\mathcal{D}$ de petits vecteurs, telle qu'on puisse produire efficacement des différences $\mathbf{s} - H(m)$ qui suivent la distribution $\mathcal{D}$ avec $\mathbf{s} \in L$, étant donné n'importe quelle bonne base $\mathbf{B}$ de L et une cible $H(m)$. Ainsi, la distribution $\mathcal{D}$ est indépendante de la base $\mathbf{B}$: aucune information ne fuite.

Une telle distribution sera proposée par Gentry, Peikert et Vaikuntanathan [GPV08], précisément une distribution Gaussienne discrète. Les distributions gaussiennes continues sont des objets centraux en théorie des probabilités ; en particulier à cause du théorème central limite (la moyenne d'un grand nombre de valeurs aléatoires suit une distribution proche d'une gaussienne, quelle que soit la distribution de départ) ; mais aussi pour leurs propriétés géométriques (si les coordonnées (x, y) d'un vecteur $\mathbf{v}$ suivent deux mêmes gaussiennes indépendantes alors la distribution de $\mathbf{v}$ est similaire dans toutes les directions). Leur version discrète dans les réseaux avait déjà été joué un rôle essentiel dans les travaux de Klein [Kle00], ainsi que ceux de Micciancio et Regev [MR04, Reg05]. L'article [GPV08] démontre que l'algorithme de Klein [Kle00] – une version randomisée de l'algorithme de Babai – permet d'utiliser de façon sûre une base courte comme une trappe pour le problème de trouver des vecteurs proches dans un réseau. Une application directe est une version prouvablement sûre des schémas de signature présentés précédemment : en remplaçant l'algorithme de Babai par celui de Klein, les attaques par apprentissages deviennent impossibles et l'on peut prouver que falsifier des signatures est aussi dur que de trouver des vecteurs

courts dans un réseau. Ils montrent de plus que cette trappe peux être utilisée pour construire d'autres types de schémas, en particulier un IBE (schéma de chiffrement basé sur l'identité). Cet article sera suivi de nombreuses constructions utilisant cette même trappe, pour construire des IBE hiérarchiques, et autres primitives de chiffrement à fine structure d'accès [CHKP10, ABB10a, AFV11, Boy13]. Des améliorations d'efficacité on aussi été proposées pour cette trappe [Pei10,MP12], dans certains cas au prix de compromis sur la qualité (largeur minimale de la gaussienne).

Contributions Tous les algorithmes proposés [Kle00, GPV08, Pei10, MP12] nécessitent pour certaines étapes l'utilisation de nombres à virgule flottante pour approximer des nombres réels, ou au moins de fractions avec de très grandes opérandes. Cependant, aucun de ces articles ne précise quelle précision flottante est requise pour assurer la correction de la distribution de sortie. Le chapitre 6 propose une telle analyse. En premier lieu, nous montrons qu'une mise en œuvre directe de l'arithmétique flottante n'apporte pas de gain asymptotique : la précision des flottants se doit d'être linéaire en la dimension pour garantir la sécurité, menant à une complexité totale de $\tilde{\mathcal{O}}(n^3)$, où n désigne la dimension du réseau. Cependant, nous montrons que la complexité de ces algorithmes peut tomber à $\tilde{\mathcal{O}}(n^2)$ en les rendant *paresseux*, voire jusqu'à $\tilde{\mathcal{O}}(n)$ dans certains cas utiles en cryptographie. De plus notre analyse est concrète et pratique : pour des paramètres typiques, nos algorithmes paresseux effectuent la plupart de leurs opérations flottantes en double-précision telle que définie par les standards IEEE, et disponibles nativement sur de nombreuses architectures de processeurs modernes.

Il existe cependant d'autres situations en cryptographie qui nécessitent des gaussiennes discrètes, mais sur des réseaux très simples : $\mathbb{Z}^n$. C'est utile en particulier comme sous-programme de l'algorithme de Klein et ses variantes, mais aussi et surtout pour implémenter le schéma de signature sans trappe proposé par Lyubashevsky [Lyu12] ; schéma que nous améliorerons dans le dernier chapitre. Idéalement, pour cette tâche plus simple, nous souhaiterions des algorithmes efficaces sans flottants, notamment pour des implémentations sur des petites architectures telles que les cartes à puce. Nous développons dans le chapitre 7 de tels algorithmes. Précisément, les méthodes standards nécessitent soit des opérations flottantes, soit de larges tables pré-calculées ; nos algorithmes ne nécessitent pas d'opérations flottantes, et des tables beaucoup plus petites ($\mathcal{O}(\log \sigma)$ contre $\mathcal{O}(\sigma)$ pour une largeur σ de la gaussienne). Nos algorithmes sont basées sur une technique, appelée tirage avec rejet ; permettant de corriger une distribution dont le défaut est connu et calculable.

En utilisant ces mêmes techniques de rejet, nous montrons enfin qu'il est en fait possible d'effectuer un echantionnage similaire a celui de Klein pour une efficacite asymptotique similaire a nos algorithmes paresseux du chapitre 6, mais sans recourir a la virgule flottante ; ces resultats sont prometteurs mais necessitent encore une analyse plus concrète.

1.4.3 Optimisations

Malgré les améliorations apportées aux fonctions à trappes basées sur les réseaux [GPV08, Pei10, MP12, DN12a] l'efficacité des schémas à base de signature utilisant ces trappes laissent à désirer ; pour des applications plus avancées telles que les IBE, de telles performances seraient acceptables, et ces progrès sont significatifs ; mais pour construire des signatures, le paradigme de Fiat-Shamir, qui ne requièrt pas de fonction à trappe, pourrait s'avérer plus efficace. Ainsi, Lyubashevsky et Micciancio proposent en 2008 un schéma d'identification sans trappe basé sur les réseaux, ne requérant que des opérations simples, et des distributions uniformes sur des cubes [LM08]. Au cœur de cette construction, se trouve une étape de rejet qui permet d'éviter toute fuite d'information sur la clef. Précisément, la clef induit une translation sur la distribution des signatures, et la procédure de rejet permet de cacher cette translation. Cependant, pour que la probabilité de rejet ne soit pas trop élevée, le schéma ne s'appuie sur des vecteurs relativement longs, ce qui à un impact négatif sur la sécurité et l'efficacité.

En appliquant la transformation de Fiat-Shamir, ce schéma est transformé un schéma de signature [LM08, Lyu09] ; les performances en pratiques sont encore décevantes par rapport à ce que les réseaux euclidien semblent promettre en termes d'efficacité asymptotique ; en particulier les signatures font près de 50 kilobits. Cependant d'autres contributions viendront améliorer ces performances. La première, proposée dans [Lyu12] consiste à remplacer les distributions uniformes sur des cubes par des distributions gaussiennes ; asymptotiquement, cela améliore par un facteur $O(\sqrt{n})$ la longueur des signatures, où n désigne la dimension du réseau. En combinant cette technique avec un nouvel argument sur la génération des clefs secrètes, Lyubashevsky obtient des signatures de 14 kilobits. Les questions d'implémentation efficace restent cependant en suspens. D'autres travaux [GLP12], s'intéressant à l'implémentation en hardware efficace de ce genre de signature, préfèrent se baser sur la version avec des

distributions uniformes ; au prix d'efforts importants d'optimisation, ils obtiennent des signatures de 9 kilobits.

Contributions Nous proposons de nouvelles améliorations théoriques et pratiques à cette série de schémas de signature sans trappe [Lyu09, Lyu12, GLP12], détaillées dans le chapitre 8. D'un point de vue théorique, nous proposons l'utilisation de Gaussienne bimodale, qui permet de réduire la longueur des signatures par un facteur asymptotique $O(\sqrt{\lambda})$ par rapport au schéma [Lyu12], où λ désigne le paramètre de sécurité ; en pratique cela donne un facteur compris entre 12 et 24 pour les paramètres proposés. Pour ce faire, nous devons effectuer des calculs modulo $2q$ plutôt que q, mais la réduction de sécurité se fait à partir d'un problème de réseau modulo q ; de façon surprenante, cette nouvelle technique permet un preuve de sécurité plus simple et plus performante.

En ce qui concerne les instanciations pratiques, nous procédons à une analyse pragmatique, en s'appuyant par exemple sur des bornes ad-hoc pour les longueurs de produits matrices-vecteurs. Nous appliquons d'autres techniques connues comme la compression de Huffman pour les distributions gaussiennes, et une autre technique adaptée de l'article [GLP12]. Enfin, nous faisons usage de réseaux similaires à ceux de NTRUEncrypt, qui contiennent des vecteurs très courts ce qui améliore d'autant notre construction. Nous détaillons les techniques connues de cryptanalyse, et tentons de donner des mesures précises de la sécurité de notre schéma.

Nous obtenons des signatures de 5 à 6 kilobits selon les versions, pour un niveau de sécurité contre les attaque connues de 128 bits (contre moins de 80 bits pour les propositions de [GLP12, Lyu12]). De plus, notre proposition est appuyée par une implémentation, qui s'avère concurrencer favorablement des schémas de signatures standardisés tel que RSA ou ECDSA ; précisément, la vitesse de signature est comparable à celle d'ECDSA, mais la vérification est *dix fois plus rapide* pour notre schéma (comparée à l'implémentation de `openssh`). Enfin, toutes les distributions nécessaires à l'implémentation de ce schéma peuvent être efficacement implémentée sur petites architectures, grâce aux algorithmes développés dans le chapitre 7.

1.5 Publications

Les résultats de cette thèses on fait l'objet de trois publications.

- *Learning a Zonotope and More : Cryptanalysis of NTRUSign Countermeasures*, co-signé avec P. Nguyen et publié à Asiacrypt 2012 [DN12b]. Les resutats de cet article sont introduits en section 1.4.1 "Attaques", et detaillés dans le chapitre 5.
- *Faster Gaussian Lattice Sampling using Lazy Floating-Point Arithmetic*, co-signé avec P. Nguyen et publié à Asiacrypt 2012 [DN12a]. Les resutats de cet article sont introduits en section 1.4.2 "Analyse", et detaillés dans le chapitre 6.
- *Lattice Signature and Bimodal Gaussian*, co-signé avec A. Durmus, T. Lepoint et V. Lyubashevsky et publié à Crypto 2013 [DDLL13]. Les résultats de cet articles sont introduits en sections 1.4.2 "Analyse" et 1.4.3 "Optimisations", et présentés dans les chapitres 7 et 8.

Les résultats préliminaires du chapitre 7, section 7.2 n'ont pas encore fait l'objet d'une publication. Deux autres publications ont vu le jour durant cette thèse :

- *Anonymity from Asymmetry : New Constructions for Anonymous HIBE*, résultat des travaux de stage de Master 1 encadré par D. Boneh, et publié à CT-RSA 2010 [Duc10]. Ces travaux n'ont pas de liens directe avec ce sujet de thèse.
- *Ring-LWE in Polynomial Rings* co-signé avec A. Durmus, publié à PKC 2012 [DD12]. Ces travaux traitent de la simplification des résultats sur le problème LWE dans des anneaux de polynômes.

Chapter 2

Mathematical and Cryptography Preliminaries

I couldn't help but overhear, probably because I was eavesdropping.

2.1 Notation

Sets. The symbol $\mathbb{N}$ denotes the set of positive integers, $\mathbb{Z}$ the ring of integers, and $\mathbb{Z}_q$ the ring of integers modulo q. For any integer q, we identify the ring $\mathbb{Z}_q$ with the interval $[-q/2, q/2) \cap \mathbb{Z}$ whenever necessary; for example when considering the norm of a vector $\mathbf{v} \in \mathbb{Z}_q$, one should interpret $\mathbf{v}$ as a vector fro $([-q/2, q/2) \cap \mathbb{Z})^n$.

The set of binary $\{0,1\}$ is denoted by $\mathbb{B}$, and the set of ternary numbers $\{-1,0,1\}$ by $\mathbb{T}$; and $\mathbb{B}_w^n$ (resp. $\mathbb{T}_w^n$) denotes the set of binary vectors (resp. ternary vectors) of length n and Hamming weight w (*i.e.* vectors with exactly w out of n non-zero entries).

The symbol $\mathbb{R}$ denotes the set of real numbers and $\mathbb{C}$ the set of complex numbers.

Linear Algebra. In the vectorspace $\mathbb{R}^n$ vectors will be denoted in bold $\mathbf{x} = (x_1 \dots x_n) \in \mathbb{R}^n$, and should be considered as row vectors (except in chapter 8 were they are column vectors). Matrices are denoted by uppercase bold letters, and $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_n]$ is denotes the $m \times n$ matrices whose n rows are the row vectors $\mathbf{b}_i \in \mathbb{R}^m$ (and in the column-vector notation, $\mathbf{B} = (\mathbf{b}_1 \dots \mathbf{b}_n)$ denotes the $n \times m$ matrix whose columns are the colum vectors $\mathbf{b}_i \in \mathbb{R}^m$).

We write $\mathbf{Id}_n$ to be the identity matrix of dimension n. The group of $n \times n$ invertible matrices with real coefficients will be denoted by $\mathcal{GL}_n(\mathbb{R})$ and $\mathcal{O}_n(\mathbb{R})$ will denote the subgroup of orthogonal matrices. The transpose of a matrix M will be denoted by M^t, and M^{-t} will mean the inverse of the transpose.

For any matrix $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_n]$ we denote $\mathbf{B}_{[k]}$ the matrix formed by the k-th first vectors of M $\mathbf{M}_{[k]} = [\mathbf{b}_1 \dots \mathbf{b}_k]$.

The notation $\mathbf{B}^\star = [\mathbf{b}_1^\star \dots \mathbf{b}_n^\star]$ refers to the Gram-Schmidt orthogonalization of $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_n]$ as in definition 3.8.

Vector norms and Balls. The canonical inner-product over $\mathbb{R}^n$ will be denoted $\langle \cdot, \cdot \rangle$ and defined by

$$\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{n} x_i y_i.$$

The associated euclidean norm (also called ℓ_2-norm), noted $\|\cdot\|$ is defined as :

$$\|\mathbf{x}\| = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle} = \sqrt{\sum_{i=1}^{n} x_i^2},$$

more generally, for $p \in (0, \infty)$, the ℓ_p norm is defined as :

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

and finally the ℓ_∞ norm as :

$$\|\mathbf{x}\|_\infty = \max_{i=1}^{n} |x_i| .$$

The set $\mathbb{S}_n$ denotes the unit sphere of $\mathbb{R}^n$ for the Euclidean norm $\|\cdot\| : \mathbb{S}_n = \{\mathbf{x} \in \mathbb{R}^n | \, \|\mathbf{x}\| = 1\}$, while $\mathfrak{B}_n$ denotes the unit open ball $\mathfrak{B}_n = \{\mathbf{x} \in \mathbb{R}^n | \, \|\mathbf{x}\| < 1\}$.

Matrix Norms and singular values For a matrix $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_n]$, the notation $\|\mathbf{B}\|$ denotes the maximal norm of the row vectors of $\mathbf{B}$: $\|\mathbf{B}\| = \max_i \|\mathbf{b}_i\|$. The spectral norm of a matrix will be noted $\|\mathbf{B}\|_s$, it is defined as $\|\mathbf{B}\|_s = \max_{\mathbf{x}\neq 0} \frac{\|\mathbf{xB}\|}{\|\mathbf{x}\|} = \max_{\mathbf{x}\in\mathbb{S}_n} \|\mathbf{xB}\|$.

The singular values $s_1 \dots s_n$ of a matrix $\mathbf{B}$ are the (decreasing ordered) square roots of the eigenvalues of Gram matrix of $\mathbf{B}$, that is $s_i = \sqrt{\lambda_i(\mathbf{BB}^T)}$. The first singular values match the spectral norm : $s_1(\mathbf{B}) = \|\mathbf{B}\|_s$, and if $\mathbf{B}$ is an invertible square matrix, $s_1(\mathbf{B}^{-1}) = s_n(\mathbf{B})^{-1}$.

For any matrix $\mathbf{B}$, one has $\|\mathbf{B}\|_s \geq \|\mathbf{B}\|$, and the equality holds when $\mathbf{B}$ has orthogonal rows.

Polytopes. If $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_n]$ is a matrix with n linearly independent row vectors, then $\mathcal{P}(\mathbf{B})$ denote the parallelepiped $\{\sum_{i=1}^n w_i \mathbf{b}_i : w_i \in [-1/2, 1/2)\}$; if moreover $\mathbf{B}$ is an orthogonal matrix (if $\mathbf{B} \in \mathcal{O}_n(\mathbb{R})$), then $\mathcal{P}(\mathbf{B})$ is said to be an hypercube. More generally, for any matrix $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_n]$, we define the zonotope $\mathcal{Z}(\mathbf{B}) = \{\sum_{i=1}^n w_i \mathbf{b}_i : w_i \in [-1/2, 1/2)\}$ which is a parallelepiped if and only if $\mathbf{B}$ has linearly independent row vectors. In some contexts, we will use $\mathcal{Z}(\mathbf{B})$ as a distribution a particular over the set $\mathcal{Z}(\mathbf{B})$ (def) ; distribution that will not be uniform except in the parallelepiped case (see definition 5.1).

For a lattice L, $\mathcal{V}(L)$ will denote the Voronoï cell of L (see property 3.4).

Rounding. The notations $\lceil x \rfloor$ and $\mathfrak{F}(x)$ denote respectively the closest integer to x and the (smallest) fractional part of x, so that $\lceil x \rfloor + \mathfrak{F}(x) = x$ with $\lceil x \rfloor \in \mathbb{Z}$ and $\mathfrak{F}(x) \in [-1/2, 1/2)$. Naturally, $\lceil \mathbf{b} \rfloor$ and $\mathfrak{F}(\mathbf{b})$ denotes the operation applied to all the coordinates of $\mathbf{b}$.

Distributions. If X is a random variable, we denote by $\mathbb{E}[X]$ its expectation. For any set S, we denote by $\mathcal{U}(S)$ the uniform distribution over S, when applicable. If $\mathcal{D}$ is a distribution over $\mathbb{R}^n$, its *covariance* is the $n \times n$ symmetric positive matrix $\mathrm{Cov}(\mathcal{D}) = \mathbb{E}_{\mathbf{x}\leftarrow\mathcal{D}}[\mathbf{x}^t\mathbf{x}]$. The notation $\mathcal{D} \oplus \mathcal{D}'$ denotes the convolution of two distributions, that is the distribution of $\mathbf{x}+\mathbf{y}$ where $\mathbf{x} \leftarrow \mathcal{D}$ and $\mathbf{y} \leftarrow \mathcal{D}'$ are sampled independently. Furthermore, we denote by $\mathcal{D} \cdot B$ the distribution of $\mathbf{x}B$ where $\mathbf{x} \leftarrow \mathcal{D}$.

We recall that a Bernoulli distribution $\mathcal{B}_c$ assigns 1 (True) with probability $c \in [0, 1]$ and 0 (False) with probability $1 - c$. Overloading the notation, for the sake of convenience, we will denote by $\mathcal{B}_c$ both the distribution and a generic variable that follows that distribution independently of all others (thus we may write $\mathcal{B}_a \oplus \mathcal{B}_b = \mathcal{B}_{a+b-2ab}$).

Additionally, $D_{\mathbb{R},\sigma,c}$ will denote the Gaussian distribution (or Normal Distribution) over $\mathbb{R}$, of variance σ^2 and average c (see definition 3.11). This definition extends to $\mathbb{R}^n$, $D_{\mathbb{R}^n,\sqrt{\Sigma},\mathbf{c}}$, for a symmetric semi-definite matrix Σ as the co-variance matrix of the Gaussian, and $\mathbf{c}$ as its center. Finally, we will also use the discrete version of those distributions, noted $D_{\mathbb{Z},\sigma,c}$ or $D_{L,\sqrt{\Sigma},\mathbf{c}}$ for a discrete set $L \subset \mathbb{R}^n$ (usually for L a lattice).

For a distribution $\mathcal{D}$ over $\mathbb{R}^n$, the moments function are defined in definition 2.4 by :

$$\mathrm{mom}_{\mathcal{D},k}(\mathbf{v}) = \mathbb{E}_{\mathbf{x}\leftarrow\mathcal{D}}\left[\langle \mathbf{v}, \mathbf{x}\rangle^k\right].$$

Differentials. Let f be a function from $\mathbb{R}^n$ to $\mathbb{R}$. The *gradient* of f at $\mathbf{w} \in \mathbb{R}^n$ is denoted by $\nabla f(\mathbf{w}) = (\frac{\partial f}{\partial x_1}(\mathbf{w}), \dots, \frac{\partial f}{\partial x_n}(\mathbf{w}))$. The *Hessian matrix* of f at $\mathbf{w} \in \mathbb{R}^n$ is denoted by $\mathrm{H}\, f(\mathbf{w}) = (\frac{\partial^2 f}{\partial x_i \partial x_j}(\mathbf{w}))_{1\leq i,j\leq n}$.

2.2 Statistical notions

2.2.1 Entropy

Entropy is a notion that captures the unpredictability of a random variable. First, Shannon Entropy measures the quantity of information carried by the outcome of a random variable :

Definition 2.1 (Shannon Entropy) *Let $\mathcal{P}$ be a distribution over a common countable set Ω. The shannon Entropy H of $\mathcal{P}$ is defined as :*

$$H(\mathcal{P}) = -\sum_{i\in\Omega} \mathcal{P}(i) \log \mathcal{P}(i).$$

Unless otherwise specified we will consider this definition with log in base 2, that is quantity of information is measured in bits. It is an additive measure :

Fact 2.1 (Additivity of Shannon Entropy) *Let $\mathcal{P}, \mathcal{Q}$ be independent distributions over some countable set Ω. Then*

$$H(\mathcal{P} \times \mathcal{Q}) = H(\mathcal{P}) + H(\mathcal{Q})$$

This measures provides theoretical bounds on the average compressibility of a data-stream of independent outcomes of a distribution. For our purpose, the most important fact is that almost optimal compression can be achieved for distribution with support of polynomial size :

Theorem 2.2 (Huffman Coding) *For any random variable X over a finite support S, there exist an injective prefix-free code $C\colon S \to \{0,1\}^*$ such that :*

$$H(X) \leq \mathbb{E}\left[|C(X)|\right] < H(X) + 1$$

where $|y|$ denotes the length of the bit-string $y \in \{0,1\}^*$. Additionally, this encoding is computable in polynomial time given the of the probability distribution table of X.

Moreover, by packing several independent variables $X_1, \ldots, X_k$, we can decrease the overhead to less than $1/k$.

Next, the Min-Entropy measures the quality of the best possible guess of the outcome, in log-scale.

Definition 2.2 (Min-Entropy) *Let $\mathcal{P}$ be a distribution over a common countable set Ω. The Min-Entropy H_∞ of $\mathcal{P}$ is defined as :*

$$H_\infty(\mathcal{P}) = -\max_{i \in \Omega} \log \mathcal{P}(i).$$

Once again, we will use $\log_2$ unless otherwise specified; therefore in a cryptographic arguments H_∞ measure the security level in bits of the hardness to guess exactly, and without additional information, the outcome of a random variable.

Min-Entropy is also an additive measure, and both Shannon and Min-Entropy are non-increasing under deterministic function, that is

Fact 2.3 (Non-increasing of Entropy) *Let $\mathcal{P}$ be a distribution over some countable set Ω. Then for any function f of domain Ω, we have :*

$$H(f(\mathcal{P})) \leq H(\mathcal{P}) \quad \text{and} \quad H_\infty(f(\mathcal{P})) \leq H_\infty(\mathcal{P}).$$

Equality holds when f is injective over the support of $\mathcal{P}$.

2.2.2 Statistical distance

A simple but useful notion for security proofs is the following :

Definition 2.3 (Statistical Distance) *Let $\mathcal{P}$ and $\mathcal{Q}$ be two distributions over a common countable set Ω. The statistical distance Δ between $\mathcal{P}$ and $\mathcal{Q}$ is defined as :*

$$\Delta(\mathcal{P}; \mathcal{Q}) = \frac{1}{2} \sum_{i \in \Omega} \left|\mathcal{P}(i) - \mathcal{Q}(i)\right|.$$

The definition naturally extends to infinite domains like $\mathbb{R}^n$ by replacing the sum with an integral.

Apart from being a distance (that is, it verifies the triangular inequality), the essential properties of the statistical distance are : it is additive and never increases under deterministic functions.

proposition 2.4 (Sub-additivity of Statistical Distance) *Let $\mathcal{P}_0, \mathcal{P}_1, \mathcal{Q}_0, \mathcal{P}_1$ be independent distributions over some countable set Ω. Then*

$$\Delta(\mathcal{P}_0 \times \mathcal{P}_1 \| \mathcal{Q}_0 \times \mathcal{Q}_1) \leq \Delta(\mathcal{P}_0 \| \mathcal{Q}_0) + \Delta(\mathcal{P}_1 \| \mathcal{Q}_1)$$

proposition 2.5 (Non-increasing of Statistical Distance) *Let $\mathcal{P}, \mathcal{Q}$ be independent distributions over some countable set Ω. Then for any function f*

$$\Delta(f(\mathcal{P}) \| f(\mathcal{Q})) \leq \Delta(\mathcal{P} \| \mathcal{Q}).$$

Equality holds when f is injective over the support of $\mathcal{P}$.

This second property lets one argue that changing a small part of a protocol by something that is statistically close, (statistical distance is negligible) only change negligibly the statistical behavior of the overall adversary-challenger interaction. The triangular inequality lets one sum up all the steps of an hybrid argument.

2.2.3 Leftover Hash Lemma

A distribution D is said to be ε-uniform if its statistical distance from the uniform distribution is at most ε. Let X and Y be finite sets. A family $\mathcal{H}$ of hash functions from X to Y is said to be *pairwise-independent* if for all distinct $x, x' \in X$, $\Pr_{h \leftarrow H}[h(x) = h(x')] = 1/|Y|$.

Lemma 2.6 (Leftover Hash Lemma [HILL99]) *Let $\mathcal{H}$ be a family of pairwise-independent hash functions from X to Y. Suppose that $h \leftarrow \mathcal{H}$ and $x \leftarrow X$ are chosen uniformly and independently. Then, $(h, h(x))$ is $\frac{1}{2}\sqrt{|Y|/|X|}$-uniform over $\mathcal{H} \times Y$.*

2.2.4 Rejection Sampling Lemma

We now state a general rejection sampling lemma that will be used throughout the thesis. This lemma is usefull when one is given access to a distribution of density g and wish to have sample from a dsitrbution of density f, as soon as f and g are close in the following sense.

Lemma 2.7 (Rejection Sampling) *Let $\mathcal{V}$ and $\mathcal{W}$ be an arbitrary set, and $h\colon V \to \mathbb{R}$ and $f\colon \mathcal{W} \to \mathbb{R}$ be probability distributions. If $g_v\colon \mathcal{W} \to \mathbb{R}$ is a family of probability distributions indexed by $v \in \mathcal{V}$ with the property that there exists a $M \in \mathbb{R}$ such that*

$$\forall v \in \mathcal{V}, \forall \mathbf{z} \in \mathbb{Z}^m, M \cdot g_v(\mathbf{z}) \geq f(\mathbf{z}) \ ,$$

then, the output distribution of the following two algorithms is identical :

1. *$v \leftarrow h$, $z \leftarrow g_v$, output $(\mathbf{z}, v)$ with probability $f(\mathbf{z})/\big(M \cdot g_v(\mathbf{z})\big)$.*
2. *$v \leftarrow h$, $z \leftarrow f$, output $(\mathbf{z}, v)$ with probability $1/M$.*

2.2.5 Momentum Analysis

The moments of a distribution are quantities that summarize geometric properties of a distribution in a vector space.

Definition 2.4 (k^{th}-order moment) *The k^{th}-order moment of a distribution $\mathcal{D}$ over $\mathbb{R}$ is the qunatity :*

$$\mathrm{mom}_{\mathcal{D},k} = \mathbb{E}_{x \leftarrow \mathcal{D}}\left[x^k\right] .$$

To study a multi-dimensional distribution, we extend this definition by defining the moment according to a direction given by a vector $\mathbf{v}$. Precisely, the k^{th}-order moment of a distribution $\mathcal{D}$ over $\mathbb{R}^n$ is the function $\mathrm{mom}_{\mathcal{D},k} : \mathbb{R}^n \to \mathbb{R}$ defined by :

$$\mathrm{mom}_{\mathcal{D},k}(\mathbf{v}) = \mathbb{E}_{\mathbf{x} \leftarrow \mathcal{D}}\left[\langle \mathbf{v}, \mathbf{x} \rangle^k\right] .$$

Note that for any distribution, $\mathrm{mom}_{\mathcal{D},k}$ is a k-linear form, that is : $\mathrm{mom}_{\mathcal{D},k}(\lambda \mathbf{v}) = \lambda^k \cdot \mathrm{mom}_{\mathcal{D},k}(\mathbf{v})$; in particular $\mathrm{mom}_{\mathcal{D},k}$ is fully defined by its value over the sphere $\mathbb{S}_n$.

The first order moment is a linear form, and it is the dual of the average of the distribution, *i.e.* $\mathrm{mom}_{\mathcal{D},1}(\mathbf{v}) = \langle \mathbb{E}_{\mathbf{x} \leftarrow \mathcal{D}}[\mathbf{x}], \mathbf{v} \rangle$. The distribution is said to be centered if this function is null, that is if $\mathbb{E}_{\mathbf{x} \leftarrow \mathcal{D}}[\mathbf{x}] = \mathbf{0}$.

The second order moment is a bilinear form, whose associated matrix is the covariance matrix $\mathrm{Cov}(\mathcal{D})$:

$$\mathrm{mom}_{\mathcal{D},2}(\mathbf{v}) = \mathbf{v} \cdot \mathrm{Cov}(\mathcal{D}) \cdot \mathbf{v}^t \quad \text{where } \mathrm{Cov}(\mathcal{D}) \stackrel{\text{def}}{=} \mathbb{E}_{\mathbf{x} \leftarrow \mathcal{D}}\left[\mathbf{x}^t \cdot \mathbf{x}\right]$$

Geometrically speaking, the second order moment of a centered distribution can be interpreted as an ellipsoidal approximation. When this ellipsoid is a sphere, that is when $\mathrm{Cov}(\mathcal{D})$ is proportional to the identity matrix, or equivalently if $\mathrm{mom}_{\mathcal{D},2}(\mathbf{v})$ only depends on $\|\mathbf{v}\|$, the distribution is said to be isotropic.

2.3 Provable Security

A security proof, or security reduction relates the security of a scheme to a simpler hypothesis, as the security of another scheme or a hardness assumption. It is for now impossible to prove without assumption the security of any public key protocol, it would at least require to settle the million dollars problem P versus NP ; even then, there is still no cryptographic construction that relates to a known $\mathsf{NP} - \mathsf{hard}$ problem.

Despite of the presence of assumptions, security proofs are very useful : they reduce the question of the hardness of a complicated, usually interactive problem to simpler problems. This new problem may have been intensively studied and widely believed to be hard (that is, not in P).

Security Games Before starting any proof, one must start by formally defining what *security* means : this definition is called the *adversarial model* or the *security game*. Precisely, one models the attacker $\mathcal{A}$ as an interactive Turing machine, specifies a challenger $\mathcal{C}$ and how they interact, including victory condition for the attacker. There is usually not a unique possible definition for a given scheme or protocol, and this gives rise to hierarchies of security notions for every type of schemes.

Hybrid Arguments While certain schemes have direct reductions to a hardness assumption, it is often useful to split the proof in several pieces. This is done by defining a sequence of games, starting from the initial security game, and slightly changing its rules until the game is vacuously impossible to win. The proof goes through by showing that each game is hard to distinguish from the previous one, making the initial problem hard to distinguish from the final one, and therefore hard to win. When showing that two successive game are hard to distinguish, one can resort to a hardness assumption or a statistical security argument.

The Random Oracle Model The Random Oracle Model (ROM) is a powerful tool to analyze the security of a scheme, especially helpful during security proof. It models the fact that a certain hash function H behave so randomly, that there is essentially no other way for an attacker to predict anything on some output $H(x)$ on a chosen input x, but by fully computing it. We exploit this behavior by forcing the attacker to commit and reveal the value x before obtaining $H(x)$, which would be a fresh random value. In other words, the Random Oracle Model, consist of assuming that some function H can only be computed in a black-box way ; therefore what happens in the black-box may be modified (in reasonable ways) during a security proof ; for example by obtaining the exact value of x, and/or to inject a problem instance within the value $H(x)$.

The ROM is a powerful model, and it has been shown that security in the ROM doesn't always imply security in the real world : some pathological case can be proved secure in the ROM, but no single instantiation of the function H by a computable function can make them secure. Therefore, part of the literature focus on constructing schemes that are provably secure even without the ROM. However, such schemes are usually far from practical, unlike the most efficient ones proved secure in the ROM ; and except for the pathologically constructed ones, no scheme has shown any weakness when instantiated with any reasonable cryptographic hash function, such as SHA-2 or SHA-3.

2.4 Basic Public-Key Primitives

We will now give the definitions of the main public-key primitives, their correctness property and the various security property that one may require from them.

2.4.1 Public-Key Encryption

A public-key encryption (PKE) scheme is defined by a triplet of PPT algorithms $(\mathbf{KeyGen}, \mathbf{Enc}, \mathbf{Dec})$ where :

$\mathbf{KeyGen} : \mathbb{Z} \to \mathcal{SK} \times \mathcal{PK}$ takes a security parameter λ and output a key pair
$\mathbf{Enc} : \mathcal{PK} \times \mathcal{M} \to \mathcal{C}$ takes a public-key and a message, and output a ciphertext
$\mathbf{Dec} : \mathcal{PK} \times \mathcal{SK} \times \mathcal{C} \to \mathcal{M}$ takes a secret-key and a cipher, and output a message.

The notations $\mathcal{SK}, \mathcal{PK}, \mathcal{M}$ and $\mathcal{C}$ denotes respectevely the set of secret-keys, public-keys, messages and ciphertexts.

It is correct if encrypting a message then decrypting it with a valid key pair returns the original message, namely

Definition 2.5 (Correctness of a PKE scheme) *A public-key encryption scheme is said correct if, for any message $m \in \mathcal{M}$ the following experiment :*

$$(sk, pk) \leftarrow \mathbf{KeyGen}(\lambda), \quad c \leftarrow \mathbf{Enc}(pk, m), \quad m' \leftarrow \mathbf{Dec}(pk, sk, c)$$

yields to $m = m'$ except with probability negligible in λ.

For public-key encryption, a commonly desired security notion is semantic security, *a.k.a.* indistinguishability under chosen plaintext attack (IND-CPA), defined by the following security game :

Definition 2.6 (IND-CPA security game and advantage) *The IND-CPA security game $G^{\text{IND-CPA}}_{\mathbf{PKE}}$ is defined as a protocol between the challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:*

– *the challenger* $\mathcal{C}$ *runs* $(sk, pk) \leftarrow \mathbf{PKE.KeyGen}(\lambda)$ *and sends* pk *to the adversary* $\mathcal{A}$
– *the adversary* $\mathcal{A}$ *chooses two messages* m_0 *and* m_1 *and sends them to* $\mathcal{C}$
– $\mathcal{C}$ *chooses a uniform random bit* b *and encrypt one of the two message accordingly :*

$$c \leftarrow \mathbf{PKE.Enc}(pk, m_b)$$

– $\mathcal{A}$ *sends a guess* b' *to the challenger*
– $\mathcal{C}$ *outputs* 1 *if the guess was correct, that is if* $b = b'$, 0 *otherwise*

The advantage of a IND-CPA adversary $\mathcal{A}$ *against this game is defined as :*

$$\mathrm{Adv}^{\text{IND-CPA}}_{\mathbf{PKE}}(\mathcal{A}) = \Pr\left[G^{\text{IND-CPA}}_{\mathbf{PKE}}(\mathcal{A}) = 1\right] - 1/2$$

Definition 2.7 (IND-CPA security) *A public-key encryption scheme* **PKE** *is said to be IND-CPA secure if for any adversary* $\mathcal{A}$ *that runs in probabilistic polynomial time (PPT) in the security parameter* λ*, its advantage* $\mathrm{Adv}^{\text{IND-CPA}}_{\mathbf{PKE}}(\mathcal{A})$ *is negligible in* λ*.*

2.4.2 Signatures

A public-key signature scheme (SS) is defined by a triplet of algorithms $(\mathbf{KeyGen}, \mathbf{Sign}, \mathbf{Verif})$ where :

$\mathbf{KeyGen} : \mathbb{Z} \to \mathcal{SK} \times \mathcal{PK}$ takes a security parameter λ and output a key pair
$\mathbf{Sign} : \mathcal{PK} \times \mathcal{SK} \times \mathcal{M} \to \mathfrak{S}$ takes a secret-key and a message, and output a signature
$\mathbf{Verif} : \mathcal{PK} \times \mathcal{M} \times \mathfrak{S} \to \{0, 1\}$ takes a public-key, a message and a signature, and output a boolean.

The notations $\mathcal{SK}, \mathcal{PK}, \mathcal{M}$ and $\mathfrak{S}$ denotes respectevely the set of secret-keys, public-keys, messages and signatures. It is correct if signing a message then verifying it with a valid key pair is accept, namely

Definition 2.8 (Correctness of a signature scheme) *A signature scheme is said correct if, for any message* $m \in \mathcal{M}$ *the following experiment :*

$$(sk, pk) \leftarrow \mathbf{KeyGen}(\lambda), \quad \sigma \leftarrow \mathbf{Sign}(sk, m), \quad b \leftarrow \mathbf{Verif}(pk, m, \sigma)$$

yields to $b = 1$ *except with probability negligible in* λ*.*

The desired security notion for signature scheme is the strong unforgeability under chosen message attack (SU-CMA) defined by the following security game :

Definition 2.9 (SU-CMA security game and advantage) *The SU-CMA security game* $G^{\text{SU-CMA}}_{\mathbf{SS}}$ *is defined as a protocol between the challenger* $\mathcal{C}$ *and an adversary* $\mathcal{A}$ *:*
– *the challenger* $\mathcal{C}$ *runs* $(sk, pk) \leftarrow \mathbf{SS.KeyGen}(\lambda)$ *and sends* pk *to the adversary* $\mathcal{A}$
– *the adversary adaptively chooses messages* $m_1 \dots m_k$*, and the challenger responds to each message with the signature* $\sigma_i = \mathbf{SS.Sign}(pk, sk, m)$
– *the adversary sends a message and a forgery* (m^*, σ^*) *to the challenger*
– *the challenger outputs* 1 *if* $\mathbf{SS.Verif}(pk, m^*, \sigma^*) = 1$*, and if* $(m^*, \sigma^*) \neq (m_i, \sigma_i)$ *for all* $i = 1 \dots k$ *; and* 0 *otherwise*

The advantage of a SU-CMA adversary $\mathcal{A}$ *against the signature scheme* **SS** *is defined as :*

$$\mathrm{Adv}^{\text{SU-CMA}}_{\mathbf{SS}}(\mathcal{A}) = \Pr\left[G^{\text{SU-CMA}}_{\mathbf{SS}}(\mathcal{A}) = 1\right]$$

Definition 2.10 (SU-CMA security) *A signature scheme* **PKE** *is said to be SU-CMA secure if for any adversary* $\mathcal{A}$ *that runs in probabilistic polynomial time (PPT) in the security parameter* λ*, its advantage* $\mathrm{Adv}^{\text{SU-CMA}}_{\mathbf{SS}}(\mathcal{A})$ *is negligible in* λ*.*

Remark. A weaker notion of security for signature scheme is the weak unforgeability, where the attacker is not allowed to output a forgery for a message that has already been signed by the challenger, even if he provides a different signature for the same message; formally, we replace $(m^*, \sigma^*) \neq (m_i, \sigma_i)$ by $m^* \neq m_i$ in the last step of the previous game.

2.5 New Functionalities

We will now present the main ideas of advanced cryptosystems, functional encryption, and homomorphic encryption and signatures.

2.5.1 Identity Based Encryption, Functional Encryption

Identity-Based Encryption The concept of identity-based encryption (IBE) was invented by Adi Shamir in 1984, but the first construction was published in 2001 by Boneh and Franklin [BF03]. This removes the need for storing individual public-key, and therefore the need for a public-key infrastructure; in other words, one's public key can simply be his name, or his email address. However it still requires an authority to generate and distribute private-keys, which means there is a key escrow. More importantly, IBE is the first step towards a fine-grained control structure enforced by encryption, namely HIBE and functional encryption described below.

An IBE scheme is defined by a tuple of four PPT algorithms (**MasterKeyGen**, **KeyGen**, **Enc**, **Dec**), where :

MasterKeyGen : $\mathbb{Z} \to \mathcal{MSK} \times \mathcal{MPK}$ takes a security parameter λ and output a master key pair
KeyGen : $\mathcal{MSK} \times \mathcal{I} \to \mathcal{SK}$ takes a secret-key and a message, and output an individual private key
Enc : $\mathcal{MPK} \times \mathcal{I} \times \mathcal{M} \to \mathcal{C}$ takes a public-key, an identity and a message, and output a cipher
Dec : $\mathcal{MPK} \times \mathcal{I} \times \mathcal{SK} \times \mathcal{C} \to \mathcal{M}$ takes an individual secret-key, the associated identity, a ciphertext and output a message.

in addition to the standard encryption sets The notations $\mathcal{SK}, \mathcal{M}$ and $\mathcal{C}$, we have The notations $\mathcal{MSK}, \mathcal{MPK}$ and $\mathcal{I}$ denotes respectevely the set of master secret-keys, master public-keys (also called public parameters) and identities.

Definition 2.11 (Correctness of an IBE scheme) *An IBE is said correct if, for any message $m \in \mathcal{M}$ and identity $i \in \mathcal{I}$ the following experiment :*

$$(msk, mpk) \leftarrow \textbf{MasterKeyGen}(\lambda), sk_i \leftarrow \textbf{KeyGen}(msk, i), c \leftarrow \textbf{Enc}(pk, i, m), m' \leftarrow \textbf{Dec}(mpk, i, sk_i, c)$$

yields to $m = m'$ except with probability negligible in λ.

We will omit any formal security game definition, there are many and they can be found in the literature [BF03].

Remark. An interesting comment is that the role of **KeyGen** resembles a signing algorithm : an individual public-key is somehow a signature of the associated identity; and many instantiation follows this heuristic idea, due to Naor.

Hierarchical Identity Based Encryption The first extension of IBE is its hierarchical variant, that offers multiple levels of authority; one of the advantage is finer grained sandboxing, decentralized key distribution, and as we will see later a simple solution to user revocation. Precisely, identities are now sequences of bounded length, forming a tree whose root is the empty sequence. Each node of this tree is an authority that inherits the decryption power of all its children, and the power to generate key for them and only them. This structure naturally provides an access policy following typical hierarchical structure of companies or government.

Example All the email addresses of some company have the form `name@dpt.companyname.com`. for various values of `name` and `department`. The head of the company generates the master secret key, that allow decryption of all emails written to any employees `*@*.companyname.com`. To each head of department (say `it`), the authority can delegate a keys that will allow decryption of emails to any IT member `*@it.companyname.com`, and the head of IT can generate key for each of its employee, `alice@it.companyname.com` and `bob@it.companyname.com`.

It is also very simple to include revocation in such a system, by including the date (or a time period) at the top-level of the identity sequence; everyday secret-keys should be redistributed hierarchically, but the public-key remains constant, the sender just needs to include a time-stamp as a parameter to the encryption function; without worrying about the list of revoked users.

Functional Encryption One frustrating limitation of the previous example, is that there there is no way to encrypt a message for both `alice@it.companyname.com` and `bob@it.companyname.com` at the same time. In other term, decryption policy only relies in the key, not the encryption process. Such a feature was developed, and called wildcarded IBE; many similar features exists as well, as fuzzy IBE.

The general framework is called functional encryption, or attribute-based encryption; each key is associated with a set of attributes, and each ciphertext to a predicate over those attributes, that determines the decryption policy. The world of pairing has built various sets of possible policies, however seems limited to policies encodable by linear constraints; the thesis of Mike Hamburg [Ham11] provides a general framework for many of those systems.

Lattice-based cryptography has first mimicked many such constructions, but recent results indicate that this linear barrier may be broken, unlocking arbitrary policies to be implemented [GGH13]; using the structures of algebraic lattices.

2.5.2 Homomorphic Encryption and Signatures

Homomorphic Encryption A homomorphic encryption scheme is a PKE scheme (**KeyGen**, **Enc**, **Dec**), together with a fourth primitive **Eval**, that takes as input a public-key pk, the description of a function $f \in \mathcal{F}$, several ciphertexts $c_1, \ldots, c_n$, and outputs a ciphertext c^*. The correction condition is that if $c_i = \mathbf{Enc}(pk, m_i)$, $c^* = \mathbf{Eval}(pk, f, c_1, \ldots, c_n)$, then $\mathbf{Dec}(c^*) = f(m_1, \ldots, m_n)$ with overwhelming probability. Several schemes have natural homomorphic properties, such as RSA which is homomorphic for the set of multivariate monomial function $\mathcal{F} = \{(x_1, \ldots, x_n) \mapsto \prod x_i^{v_i} | v_i \in \mathbb{Z}\}$, or Paillier encryption scheme for the set of linear function $\mathcal{F} = \{(x_1, \ldots, x_n) \mapsto \sum v_i x_i | v_i \in \mathbb{Z}\}$,

Also, the introduction of pairing naturally gives rise to homomorphic encryption for bilinear functions; but lattices have recently provided fully homomorphic encryption (FHE), that is the set of function that can be evaluated homomorphically is the set of all efficiently computable functions. The initial breakthrough was done by Craig Gentry [Gen09] using non-standard assumptions, but many follow-up work have improved upon it, both in term of efficiency and security assumption. For example, it is now possible to base FHE on the standard LWE assumption [BV11]; and on the efficiency side, it is now possible to evaluate AES block cipher homomorphically in a reasonable amount of time [GHS12].

Homomorphic Signature In a symmetric manner, one may want similar homomorphisms on signatures. While for encryption the goal is to allow delegation on encrypted data, the goal of homomorphic signature is to provide certified results of a function on certified data; and for efficiency (and non triviality) we want everyone to be able to check such certified results without requiring access to the data themselves.

For example, with linearly homomorphic signatures, it is possible, from a certified data-set of revenues to produce a certified value of the mean value, but not much more. Once again, before recent results of lattice-based cryptography, one was limited to linear operations; but lattices now provide homomorphic signatures for bounded degree polynomials [BF11]; it is now possible to extract small certification of a wider range of statistical measure over large certified data-sets.

Chapter 3

Geometry of numbers Preliminaries

ΑΓΕΩΜΕΤΡΗΤΟΣ · ΜΗΔΕΙΣ · ΕΙΣΙΤΩ

3.1 Lattices

3.1.1 Definitions

Definition 3.1 (Lattice) *A lattice $L \subset \mathbb{R}^m$ is a discrete subgroup of the vector space $\mathbb{R}^m$; that is L is a non-empty subset of $\mathbb{R}^m$ such that :*

- *for any $\mathbf{x}, \mathbf{y} \in L$, $\mathbf{x} - \mathbf{y} \in L$ (L is a group)*
- *$\mathbf{0}$ is isolated in L that is, there exists a radius r such that $r \cdot \mathfrak{B}$, the centered ℓ_2 open ball of radius r, contains no other lattice point than $\mathbf{0}$: $r \cdot \mathfrak{B} \cap L = \{\mathbf{0}\}$.*

The dimension m of the vector space containing the lattice is called the embedding dimension of the lattice L. The first minima $\lambda_1(L) > 0$ of a lattice is the largest radius r as above : $\lambda_1(L) = \sup\{r \in \mathbb{R} : r \cdot \mathfrak{B} \cap L = \{\mathbf{0}\}\} = \min_{\mathbf{x} \in L \setminus \{\mathbf{0}\}} \|\mathbf{x}\|$.

Definition 3.2 (Sublattice) *If $L, L' \subset \mathbb{R}^m$ are both lattices, we say that L' is a sublattice of L if $L' \subset L$. If L is n-dimensional, then L' is n'-dimensional for some $n' \leq n$, and $\lambda_1(L') \geq \lambda_1(L)$.*

Lattice are therefore discrete analogues of vector spaces. Both structures share some properties. First, one may define the dimension of a lattice as the dimension of the vector space it spans

Definition 3.3 (Dimension of a Lattice) *The dimension of a lattice $L \subset \mathbb{R}^m$ is the dimension $n \leq m$ of the vector space it spans :* $\mathrm{Span}_{\mathbb{R}}(L)$.

Secondly there exists a notion of basis for lattices :

Definition 3.4 (Basis of a Lattice) *A basis $\mathbf{B} = \{\mathbf{b}_1 \dots \mathbf{b}_n\}$ of a lattice $L \subset \mathbb{R}^m$, is a finite set of linearly independent vectors in $\mathbb{R}^m$ whose $\mathbb{Z}$-span is exactly the lattice. Seeing $\mathbf{B}$ as a matrix whose row vectors are the $\mathbf{b}_i$'s, $\mathbf{B}$ is a basis if its vectors are linearly independent and if*

$$\mathbb{Z}^n \cdot \mathbf{B} = L \qquad \text{where } \mathbb{Z}^n \cdot \mathbf{B} = \{\mathbf{z} \cdot \mathbf{B} : \mathbf{z} \in \mathbb{Z}^n\} = \mathcal{L}(B)$$

proposition 3.1 (Existence of Basis and uniqueness up to units) *For any lattice $L \subset \mathbb{R}^m$ of dimension n,*

- *there exists at least one basis $\mathbf{B}$ of cardinality n*
- *If $\mathbf{B}$ and $\mathbf{B}'$ are both basis of L, there exists an integer matrix $\mathbf{U} \in \mathbb{Z}^{n \times n}$ such that $\mathbf{UB} = \mathbf{B}'$ and $\det(\mathbf{U}) = \pm 1$.*

While the uniqueness property is rather easy to demonstrate, proving existence requires additional tools. Our approach is based on the notion of Fundamental Domain, detailed in the next section 3.1.2. We defer the proof to Sect. 3.1.3.

3.1.2 Basis and Fundamental Domains

The following property can be interpreted geometrically as follows : if $\mathbf{B}$ is a basis of a lattice L, then the parallelepiped $\mathcal{P}(\mathbf{B})$ tiles the space with respect to L, that is, the sets $\mathbf{x}+\mathcal{P}(\mathbf{B})$ for $\mathbf{x} \in L$ covers the whole vector space $\mathrm{Span}_{\mathbb{R}}(L)$, and those shifted sets do not overlaps.

proposition 3.2 *If $\mathbf{B}$ is a basis of a lattice $L = \mathcal{L}(\mathbf{B}) \subset \mathbb{R}^m$, then any $\mathbf{x} \in \mathrm{Span}_{\mathbb{R}}(L)$ can be written uniquely as $\mathbf{x} = \mathbf{v} + \mathbf{w}$ with $\mathbf{v} \in L$ and $\mathbf{w} \in \mathcal{P}(\mathbf{B})$ where $\mathcal{P}(\mathbf{B})$ denotes the parallelepiped spanned by $\mathbf{B}$: $\mathcal{P}(\mathbf{B}) = \{\sum_{i=1}^{n} w_i \mathbf{b}_i : w_i \in [-1/2, 1/2)\}$. We denote $\mathbf{x} \bmod \mathbf{B}$ the unique $\mathbf{w}$ as above.*

Proof: One can uniquely write $\mathbf{x} = \sum x_i \mathbf{b}_i$ for $x_i \in \mathbb{R}$ since $\mathbf{B}$ is a $\mathbb{R}$-basis of the vector space $\mathrm{Span}_{\mathbb{R}}(L)$. For the existence, simply choose $\mathbf{v} = \sum \lfloor x_i \rceil \mathbf{b}_i$ and $\mathbf{w} = \sum (x_i - \lfloor x_i \rceil)\mathbf{b}_i$. For uniqueness, decompose $\mathbf{v} = \sum v_i \mathbf{b}_i$ where $v_i \in \mathbb{R}$ and $\mathbf{w} = \sum w_i \mathbf{b}_i$ for $w_i \in [-1/2, 1/2)$. Since the vectors of $\mathbf{B}$ are linearly independent, $\mathbf{x} = \mathbf{v} + \mathbf{w}$ implies $x_i = v_i + w_i$ for all indexes i; the only decomposition of x_i as a sum of an integer and a real in $[-1/2, 1/2)$ is indeed $v_i = \lfloor x_i \rceil$ and $w_i = x_i - \lfloor x_i \rceil$. □

This property can be restated using the notion of fundamental domain, stating that if $\mathbf{B}$ is a basis of L, then $\mathcal{P}(\mathbf{B})$ is a fundamental domain of L. Informally, a fundamental domain of a lattice L, is a set that tiles whole vector space $\mathrm{Span}_{\mathbb{R}}(L)$.

Definition 3.5 (Fundamental Domain) *For a lattice L, a measurable set $\mathcal{F} \subset \mathrm{Span}_{\mathbb{R}}(L)$ is called a fundamental domain of L if $\bigcup_{\mathbf{x}\in L} \mathcal{F} + \mathbf{x} = \mathrm{Span}_{\mathbb{R}}(L)$ and if the union of those interiors $\bigcup_{\mathbf{x}\in L} \mathcal{F}^\circ + \mathbf{x}$ is disjoint. Equivalently, $\mathcal{F}$ is a fundamental domain if any $\mathbf{y} \in \mathrm{Span}_{\mathbb{R}}(L)$ can be written as a sum $\mathbf{y} = \mathbf{v} + \mathbf{w}$ where $\mathbf{v} \in L$ and $\mathbf{w} \in \mathcal{F}$ and if this decomposition is unique for any $\mathbf{y} \notin \bigcup_{\mathbf{x}\in L} \partial\mathcal{F} + \mathbf{x}$.*

Note that we have not yet prove the existence of basis so we cannot yet derive existence of measurable fundamental domains. Using the axiom of choice, one may indeed build a fundamental domain of L, but it might not be measurable, and lacks geometric interpretation. For the rest of this section, we will focus our study on convex fundamental domail to avoid technicalities related to Lesbuegues measure

Fact 3.3 (Measurability of a convex and its frontier) *If $\mathcal{C} \subset \mathbb{R}^n$ is convex, then $\mathcal{C}$ is measurable, and its frontier is also negligeable, that is*

$$\mathrm{Vol}(\mathcal{C}) = \mathrm{Vol}(\mathcal{C}^\circ) = \mathrm{Vol}(\bar{\mathcal{C}}) \text{ and } \mathrm{Vol}(\partial\mathcal{C}) = 0$$

The existence of the later can be establish using the notion of Vornoï cell, and more importantly this fundamental domain is convex, measurable, and has non-zero volume.

proposition 3.4 (Definition and property of the Voronoï Cell) *For a n-dimensional lattice $L \subset \mathbb{R}^m$, the Voronoï Cell is $\mathcal{V}(L)$ defined as the set of all points of $\mathrm{Span}_{\mathbb{R}}(L)$ that are closer to the origin than to any other lattice point, namely $\mathcal{V}(L) = \{\mathbf{x} \in \mathrm{Span}_{\mathbb{R}}(L) : \forall \mathbf{y} \in L, \|\mathbf{x}\| \leq \|\mathbf{x} - \mathbf{y}\|\}$. It has the following properties :*

- *$\mathcal{V}(L)$ is convex, and therefore measurable*
- *the n-volume of $\mathcal{V}(L)$ is non zero : $\mathrm{Vol}(\mathcal{V}(L)) \geq \left(\frac{\lambda_1(L)}{2}\right)^n \mathrm{Vol}(\mathfrak{B}_n) > 0$*
- *$\mathcal{V}(L)$ is a fundamental domain of L*

Proof: Note that $\mathcal{V}(L)$ can be written as the intersection of $\mathrm{Span}_{\mathbb{R}}(L)$ with all the half-spaces $H_{\mathbf{x}} = \{\mathbf{v} : \langle \mathbf{v}, \mathbf{x} \rangle \leq 1/2\}$ for non zero $\mathbf{x} \in L$. All those sets are convex, so is $\mathcal{V}(L)$.

Now, because lattice are discrete, $\mathbf{0}$ is isolated, that is all non zero vectors $\mathbf{x}$ of L have norm greater than $\lambda_1(L) > 0$. This implies that all half-spaces $H_{\mathbf{x}}$ contain the centered open ball $\frac{\lambda_1(L)}{2} \cdot \mathfrak{B}_n$ of radius $\lambda_1(L)/2$. Therefore $\frac{\lambda_1(L)}{2} \cdot \mathfrak{B}_n \subset \mathcal{V}(L)$ and $\mathrm{Vol}(\mathcal{V}(L)) \geq \mathrm{Vol}(\frac{\lambda_1(L)}{2} \cdot \mathfrak{B}_n) = \left(\frac{\lambda_1(L)}{2}\right)^n \mathrm{Vol}(\mathfrak{B}_n)$.

Finally, for the existence of the fundamental domain, we start by noting that $\bigcup_{\mathbf{x}\in L} \mathcal{V}(L) + \mathbf{x} = \mathrm{Span}_{\mathbb{R}}(L)$. Indeed, for any $\mathbf{y} \in \mathrm{Span}_{\mathbb{R}}(L)$, let $\ell = \inf_{\mathbf{x}\in L} \|\mathbf{x} - \mathbf{y}\|$ be the distance of L from the lattice. Because L is discrete, this infimum is reached at some $\mathbf{x} \in L$, and by definition $\mathbf{y} \in \mathcal{V}(L) + \mathbf{x}$.

Last, if $\mathbf{y} \in \mathcal{V}^\circ(L) + \mathbf{x}$ for some $\mathbf{x} \in L$, there exists $\epsilon > 0$ such that for any vector $\mathbf{e}$ of norm $\|\mathbf{e}\| \leq \epsilon$ we have $\mathbf{y} + \mathbf{e} \in \mathcal{V}(L) + \mathbf{x}$; this implies that for any $\mathbf{x}' \in L$, $\|\mathbf{y} - \mathbf{x}'\| \geq \|\mathbf{y} - \mathbf{x}\| + \epsilon$; in other words $\mathbf{y} \notin \mathcal{V}(L) + \mathbf{x}'$, which concludes the proof.

□

An essential geometric property of fundamental domains, is that they all share the same volume, as stated below. The geometric interpretation of the proof is the following : tiles L according to a first fundamental domain $\mathcal{F}$, and cut a second fundamental domain $\mathcal{F}'$ in pieces according to this tiling. Then, those pieces can be shifted in such a way that their union is a non intersecting covering of $\mathcal{F}$.

proposition 3.5 (Volume of Fundamental Domains) *For any n-dimensional lattice L, there exists a positive real, denoted $\mathrm{Vol}(L) > 0$ such that any fundamental domain let $\mathcal{F} \subset \mathrm{Span}(L)$ has n-volume $\mathrm{Vol}(L)$.*

Proof: Let $\mathcal{F}, \mathcal{F}'$ be two convex fundamental domain. One rewrite $\mathcal{F}' = \bigcup_{\mathbf{x}\in L} \mathcal{F}' \cap (\mathcal{F} + \mathbf{x})$ and $\mathcal{F} = \bigcup_{\mathbf{x}\in L} \mathcal{F} \cap (\mathcal{F}' + \mathbf{x})$. Because those union are disjoint (but for intersections of zero measure), we have

$$\mathrm{Vol}(\mathcal{F}) = \sum_{\mathbf{x}\in L} \mathrm{Vol}(\mathcal{F} \cap (\mathcal{F}' + \mathbf{x})) = \mathrm{Vol}(\sum_{\mathbf{x}\in L} (\mathcal{F} - \mathbf{x}) \cap \mathcal{F}') = \mathrm{Vol}(\mathcal{F}').$$

□

From the same kind of argument, we can also proof the following fact.

proposition 3.6 (Volume of sub-Fundamental Domains) *For any n-dimensional lattice L, if $\mathcal{S}$ is a measurable set such that $\bigcup_{\mathbf{x}\in L} \mathcal{S} + \mathbf{x}$ is a disjoint union, $\mathrm{Vol}(\mathcal{S}) \leq \mathrm{Vol}(L)$.*

Our last tool is the following property, stating that sublattices have larger volume than the original lattice.

proposition 3.7 (Volume of a sublattice) *If L' is a sublattice of $L \in \mathbb{R}^m$, and if both L, L' have the same dimension n, then $\mathrm{Vol}(L') \geq \mathrm{Vol}(L)$.*

Proof: Because $L' \subset L$, we have $\mathrm{Span}_{\mathbb{R}}(L') \subset \mathrm{Span}_{\mathbb{R}}(L)$, and because they have the same dimension, $\mathrm{Span}_{\mathbb{R}}(L') = \mathrm{Span}_{\mathbb{R}}(L)$. Now consider the Voronoï cells of L and L'; recall that $\mathcal{V}(L)$ can be written as the intersection of $\mathrm{Span}_{\mathbb{R}}(L)$ with all the half-spaces $H_{\mathbf{x}} = \{\mathbf{v} : \langle \mathbf{v}, \mathbf{x} \rangle \leq 1/2\}$ for non zero $\mathbf{x} \in L$. This directly implies that

$$\mathcal{V}(L) \subset \mathcal{V}(L'), \quad \text{therefore } \mathrm{Vol}(L) = \mathrm{Vol}(\mathcal{V}(L)) \leq \mathrm{Vol}(\mathcal{V}(L')) = \mathrm{Vol}(L').$$

□

The notion of volume is a key notion to evaluate the number of lattice point in a set. While there exists many theorems giving bounds, in a heuristic context, such as evaluating the complexity of algorithm on random lattices, on may simply rely on the Gaussian Heuristic.

Heuristic 3.1 (Gaussian Heuristic) *For a full rank lattice L, and a measurable set $\mathcal{S}$, the expected number of lattices point in $\mathcal{S}$ is*

$$\#(L \cap \mathcal{S}) \approx \mathrm{Vol}(\mathcal{S}) / \mathrm{Vol}(L)$$

It is easy to build counterexample to such heuristic, yet if the set $\mathcal{S}$ is reasonable, (*e.g.* convex, or spherical), theorem such as Minkowski's one (Thm. 3.14), or Blichfeldt lemma [Bli] provides formal statements.

Such a heuristic provides an estimation of the first minimum of a random lattice :

$$\lambda_1(L) \approx r = \left(\frac{\mathrm{Vol}(L)}{\mathrm{Vol}(\mathfrak{B})} \right)^{1/n} \approx \frac{\mathrm{Vol}(L)^{1/n} \cdot \sqrt{n}}{\sqrt{2\pi e}}.$$

Indeed, in a centered ball of radius $r \cdot \alpha$ ($\alpha > 0$) we expect to find about α^n lattice points; for $\alpha < 1$ we except to have less than one lattice point (the origin $\mathbf{0}$), and for $\alpha > 1$ we expect many points. This heuristic can also be made formal for properly defined distributions of random lattices.

3.1.3 Proof of the Existence and Uniqueness of Basis (Property 3.1)

Let n be the dimension of $\mathrm{Span}_{\mathbb{R}}(L)$, and let $\mathbf{B} = \{\mathbf{b}_1 \dots \mathbf{b}_n\}$ be a set of n linearly independent vectors of L. Consider the parallelepiped $\mathcal{P}(\mathbf{B})$ and its intersection with L : $S = L \cap \mathcal{P}(\mathbf{B})$. Clearly, $\mathbf{0} \in S$, therefore there are two cases, either $S = \{\mathbf{0}\}$, that is the parallelepiped spanned by $\mathbf{B}$ contain no non-trivial point, and $\mathbf{B}$ will be a basis ; either there exists another matrix $\mathbf{B}'$ of linearly independent vector of L that span a parallelepiped of volume at least twice as small. We conclude by showing that the volume $\mathcal{P}(\mathbf{B})$ for matrices of n linearly independent vectors of L is lower bounded by the volume of L, which is strictly positive ; therefore, starting from an arbitrary such matrix $\mathbf{B}$, one falls in the basic case after finitely many iterations of the induction case.

Base Case : $S = \{\mathbf{0}\}$. In this case we will prove that $\mathbf{B}$ is indeed a basis of L. Let $\mathbf{v} \in L$ be an arbitrary point of the lattice. Because $\mathbf{B}$ $\mathbb{R}$-spans the same vector-space than L, $\mathbf{v}$ can be written as $\mathbf{v} = \sum_{i=1}^{n} v_i \mathbf{b}_i$ for real values v_i. Set $\mathbf{w}$ as $\mathbf{w} = \sum_{i=1}^{n} \lfloor v_i \rceil \mathbf{b}_i$, which belongs to L. Therefore $\mathbf{v} - \mathbf{w} \in L$, and $\mathbf{v} - \mathbf{w} \sum_{i=1}^{n} (v_i - \lfloor v_i \rceil) \cdot \mathbf{b}_i$ belongs to $\mathcal{P}(\mathbf{B})$ since $(v_i - \lfloor v_i \rceil) \in [-1/2, 1/2)$. We conclude that $\mathbf{v} - \mathbf{w} \in S$, that is $\mathbf{v} = \mathbf{w}$ *i.e.* $v_i \in \mathbb{Z}$ for all indexes i : any $\mathbf{v} \in L$ belongs to the $\mathbb{Z}$-span of $\mathbf{B}$.

Induction Case : there exists a non-zero vector $\mathbf{w} \in S$. Write $\mathbf{w} = \sum_{i=1}^{n} w_i \mathbf{b}_i$ for $w_i \in [-1/2, 1/2)$; and let j be an index such that $w_j \neq 0$. Without loss of generality, we can assume that $j = 1$. Now, one construct the new matrix $\mathbf{B}' = \{\mathbf{w}, \mathbf{b}_2, \dots \mathbf{b}_n\}$. It can be expressed in terms of $\mathbf{B}$ as

$$\mathbf{B}' = \mathbf{M} \cdot \mathbf{B} \qquad \text{where } \mathbf{M} = \begin{pmatrix} w_1 & w_2 & \dots & w_n \\ 0 & & & \\ \vdots & & \mathbf{Id}_{n-1} & \\ 0 & & & \end{pmatrix}$$

The volume of the parallelepiped $\mathcal{P}(\mathbf{B}')$ is :

$$\begin{aligned} \mathrm{Vol}(\mathcal{P}(\mathbf{B}')) &= \sqrt{\det(\mathbf{B}' \cdot \mathbf{B}'^t)} = \sqrt{\det(\mathbf{M} \cdot \mathbf{B} \cdot \mathbf{B}^t \cdot \mathbf{M}^t)} \\ &= \sqrt{\det(\mathbf{M}) \cdot \det(\mathbf{B} \cdot \mathbf{B}^t) \cdot \det(\mathbf{M}^t)} = \det(\mathbf{M}) \cdot \mathrm{Vol}(\mathcal{P}(\mathbf{B})) \end{aligned}$$

It remains to note that $\det(\mathbf{M}) = w_1 \in [-1/2, 1/2)$ to conclude that $\mathbf{B}'$ is also a set of n linearly independent vector of L, such that

$$\mathrm{Vol}(\mathcal{P}(\mathbf{B}')) \leq \frac{1}{2} \mathrm{Vol}(\mathcal{P}(\mathbf{B}))$$

.

Proof of existence. We want to show that, if $\mathbf{B}$ is a set of linearly independent vectors of L, then $\mathrm{Vol}(\mathcal{P}(\mathbf{B})) \geq \mathrm{Vol}(L) > 0$. We just have to notice that $\mathbf{B}$ is a basis of $\mathcal{L}(\mathbf{B})$ which is a sublattice of L, thus, $\mathrm{Vol}(\mathcal{P}(\mathbf{B})) = \mathrm{Vol}(\mathcal{L}(\mathbf{B})) \geq \mathrm{Vol}(L)$ according to property 3.7.

Uniqueness. Let $\mathbf{B}$ and $\mathbf{B}'$ be bases of L. In particular, any vector $\mathbf{b} \in \mathbf{B}$ belongs to L, therefore, it can be written as a linear combination of vectors of $\mathbf{B}'$: there exists an integer matrix $\mathbf{U} \in \mathbb{Z}^{n \times n}$ such that $\mathbf{B} = \mathbf{U} \cdot \mathbf{B}'$. Similarly, there exists $\mathbf{U}' \in \mathbb{Z}^{n \times n}$ *s.t.* $\mathbf{B}' = \mathbf{U}' \cdot \mathbf{B}$. We obtain $\mathbf{B} = \mathbf{U} \cdot \mathbf{U}' \cdot \mathbf{B}$, which implies $\mathbf{U} \cdot \mathbf{U}' = \mathbf{Id}_n$ because $\mathbf{B}$ has linearly independent vectors. We conclude that $\det(\mathbf{U}) \cdot \det(\mathbf{U}') = 1$; which implies $\det(\mathbf{U}) = \det(\mathbf{U}') = \pm 1$ since $\det(\mathbf{U})$ and $\det(\mathbf{U}')$ are integers.

3.1.4 Duality

Definition 3.6 (The dual of a Lattice) *The dual of a lattice L, noted $\hat{L}$, is the set of all points in* $\mathrm{Span}_{\mathbb{R}}(L)$ *that have integer scalar product with all vectors of L, that is*

$$\hat{L} = \{\mathbf{x} \in \mathrm{Span}_{\mathbb{R}}(L) : \forall \mathbf{v} \in L, \langle \mathbf{x}, \mathbf{v} \rangle \in \mathbb{Z}\}.$$

proposition 3.8 (The dual of a Lattice is a Lattice) *For any lattice L, its dual $\hat{L}$ is a lattice.*

Proof: The dual of $L \subset \mathbb{R}^m$ is obviously a subgroup of $\mathrm{Span}_{\mathbb{R}}(L)$. Now, for the discreteness, let $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_n]$ be a basis of L and let $\hat{\mathbf{v}}$ be non-zero point of the dual; this implies that there exists some i such that $\langle \hat{\mathbf{v}}, \mathbf{b}_i \rangle \neq 0$, and by definition of the dual we must have $\langle \hat{\mathbf{v}}, \mathbf{b}_i \rangle \in \mathbb{Z}$. In particular $|\langle \hat{\mathbf{v}}, \mathbf{b}_i \rangle| \geq 1$, which implies $\|\hat{\mathbf{v}}\| \geq 1/\|\mathbf{b}_i\| \geq 1/\min_j \|\mathbf{b}_j\|$. □

Note that this proof implies a bound on $\lambda_1(\hat{L})$ in term of the best basis of L. This is the basis of many important results in geometry of numbers, known as Transference Theorems, relating the successive minima of $\hat{L}$ to those of L.

proposition 3.9 (Dual basis) *If $\mathbf{B} \in \mathbb{Z}^{n \times m}$ is a basis of the lattice $L \subset \mathbb{R}^m$ of rank n, then $\hat{\mathbf{B}} = (\mathbf{B}\mathbf{B}^t)^{-1}\mathbf{B}$ is a basis of $\hat{L}$, with the equality $\hat{\mathbf{B}} = \mathbf{B}^{-t}$ if L is full rank.*

In particular, L and $\hat{L}$ have the same dimension n, and inverse volumes $\mathrm{Vol}(\hat{L}) = \mathrm{Vol}(L)^{-1}$

One of the use of dual lattice arises when summing a function over all lattice points; the Poisson formula then gives a link with the Fourier transform of that function over the dual lattice.

Definition 3.7 (Fourier Transform) *The Fourier Transform is defined for continuous functions of $\mathbb{R}^n \to \mathbb{C}$ that are absolutely integrable (that is $\int_{\mathbf{x} \in \mathbb{R}^n} |f(\mathbf{x})|\, d\mathbf{x} < \infty$) as follows*

$$\hat{f}(\mathbf{y}) = \int_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x}) \cdot e^{-2\pi \imath \langle \mathbf{x}, \mathbf{y} \rangle} d\mathbf{x}$$

where $\imath$ denotes the canonical imaginary square root of -1.

Theorem 3.10 (Poisson Summation Formula For Lattices) *For any continuous absolutely integrable function $f : \mathbb{R}^m \to \mathbb{C}$, and any lattice $L \subset \mathbb{R}^m$, we have*

$$\sum_{\mathbf{x} \in L} f(\mathbf{x}) = \frac{1}{\mathrm{Vol}(L)} \cdot \sum_{\mathbf{y} \in \hat{L}} \hat{f}(\mathbf{y}).$$

3.1.5 Gram-Schmidt Orthogonalization (GSO)

The Gram-Schmidt orthogonalization (GSO) is an algorithm that transforms any basis $\mathbf{B}$ of a vector space to an orthogonal basis $\mathbf{B}^\star$ of the same vector space. Yet, if $\mathbf{B}$ is a basis of a lattice L, $\mathbf{B}^\star$ is not necessarily a basis of the same lattice since, all the $\mathbf{b}_i^\star$'s may not belong to L for any index $i > 1$; in general, and unlike vector spaces, lattices do not admit orthogonal base. Yet the Gram-Schmidt of a lattice basis remain a useful object. In particular, when it will come to using basis a to approximate the closest vector problem, the GSO can provide a better approximation.

Definition 3.8 (Gram-Schmidt Orthogonalization (GSO)) *Let $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_n] \in \mathbb{R}^{n \times m}$ be a matrix. The Gram-Schmidt Orthogonalization (GSO) $\mathbf{B}^\star = [\mathbf{b}_1^\star \dots \mathbf{b}_n^\star] \in \mathbb{R}^{m \times n}$ is defined as follows :*

$$\begin{aligned} \mathbf{b}_1^\star &= \mathbf{b}_1 \\ \mathbf{b}_i^\star &= \mathbf{b}_i - \pi_{\mathrm{Span}_{\mathbb{R}}(\{\mathbf{b}_1 \dots \mathbf{b}_{i-1}\})}(\mathbf{b}_i) = \pi_{\{\mathbf{b}_1 \dots \mathbf{b}_{i-1}\}^\perp}(\mathbf{b}_i) \end{aligned}$$

Note that this recursive definition implies that for any $k \leq n$, $[\mathbf{b}_1^\star \dots \mathbf{b}_k^\star]$ is the GSO of $[\mathbf{b}_1 \dots \mathbf{b}_k]$, in other words $(\mathbf{B}_{[k]})^\star = (\mathbf{B}^\star)_{[k]} = \mathbf{B}^\star_{[k]}$.

proposition 3.11 (Iwasawa Decomposition) *For any $n \times m$ real matrix $\mathbf{B}$, there exists a unique decomposition $\mathbf{B} = \mu\mathbf{D}\mathbf{Q}$, where $\mu = (\mu_{i,j})$ is an $n \times n$ lower-triangular matrix with unit diagonal, $\mathbf{D}$ an n-dimensional positive diagonal matrix and $\mathbf{Q}$ an $n \times m$ matrix with orthonormal row vectors.*

The GSO verifies $\mathbf{B}^\star = \mathbf{D}\mathbf{Q}$; additionally the diagonal $\mathbf{D}$ matrix verifies $D_{i,i} = \|\mathbf{b}_i^\star\|$ and the transition matrix μ satisfies

$$\mu_{i,j} = \langle \mathbf{b}_i\, , \mathbf{b}_j^\star \rangle / \|\mathbf{b}_j^\star\|^2$$

.

An interesting property of the GSO $\mathbf{B}^\star$, is that, even if it isn't a basis of the lattice $\mathcal{L}(\mathbf{B})$, the parallelepiped it spans is still a fundamental domain of $\mathcal{L}(\mathbf{B})$. In fact we can even prove a slightly more general statement :

proposition 3.12 *If* $\mathbf{B}$ *is a basis of an* n*-dimensional lattice* $L = \mathcal{L}(\mathbf{B}) \subset \mathbb{R}^m$ *and if* $\mathbf{B}^\star = \mu^{-1}\mathbf{B}$ *is the GSO of* $\mathbf{B}$*, then* $\mathcal{P}(\mathbf{B}^\star)$ *is a fundamental domain.*

While the proof is standard and elementary, we believe it is worth going through, the main reason is that we can actually deduce an efficient algorithm from this proof, called Babai's nearest plane algorithm, that we will see later.

Proof: We proceed by induction. Recall from the definition of GSO that $(\mathbf{B}_{[k]})^\star = (\mathbf{B}^\star)_{[k]}$. For a one dimensional lattice $\mathbf{B} = [\mathbf{b}]$, we have $\mathbf{B}^\star = \mathbf{B}$, so $\mathcal{P}(\mathbf{B}^\star) = \mathcal{P}(\mathbf{B})$ is a fundamental domain of $\mathcal{L}(\mathbf{B})$.

Now, consider an n-dimensional lattice $\mathcal{L}(\mathbf{B})$. By induction, $\mathcal{P}(\mathbf{B}^\star_{[n-1]})$ is a fundamental domain of $\mathcal{L}(\mathbf{B}_{[n-1]})$. Let $\mathbf{x}$ be an arbitrary vector in $\mathrm{Span}_{\mathbb{R}}(L)$, it can be written as $\mathbf{x}' + x_n\mathbf{b}_n^\star$ where $\mathbf{x} \in \mathrm{Span}_{\mathbb{R}}(\mathbf{B}_{[n-1]})$ and $x_n \in \mathbb{R}$. Set $x'_n = \lfloor x_n \rceil \in \mathbb{Z}$, and rewrite

$$\mathbf{x} = \underbrace{\mathbf{x}' + (x_n - x'_n)(\mathbf{b}_n - \mathbf{b}_n^\star)}_{\mathbf{v} \,\in\, \mathrm{Span}_{\mathbb{R}}(\mathbf{B}_{[k-1]})} + \underbrace{x'_n\mathbf{b}_n}_{\in \mathcal{L}(\mathbf{b}_n)} + \underbrace{(x_n - x'_n)\mathbf{b}_n^\star}_{\in \mathcal{P}(\mathbf{b}_n^\star)}.$$

Note that $\mathbf{v} = \mathbf{x}' + (x_n - x'_n)(\mathbf{b}_n - \mathbf{b}_n^\star)$ belongs to $\mathrm{Span}_{\mathbb{R}}(\mathbf{B}_{[k-1]})$, therefore, by induction it can be written as $\mathbf{v} = \mathbf{y} + \mathbf{z}$ where $\mathbf{y} \in \mathcal{L}(\mathbf{B})$ and $\mathbf{z} \in \mathcal{P}(\mathbf{B}^\star_{[n-1]})$. We conclude on existence by checking that $\mathbf{x} = (\mathbf{v} + x'_n\mathbf{b}_n) + (\mathbf{z} + (x_n - x'_n)\mathbf{b}_n^\star)$, and that $(\mathbf{v} + x'_n\mathbf{b}_n) \in \mathcal{L}(\mathbf{B})$ and $(\mathbf{z} + (x_n - x'_n)\mathbf{b}_n^\star) \in \mathcal{P}(\mathbf{B}^\star)$.

For uniqueness, consider two decompositions of $\mathbf{x} = \mathbf{y} \cdot \mathbf{B} + \mathbf{z} \cdot \mathbf{B}^\star = \mathbf{y}' \cdot \mathbf{B} + \mathbf{z}' \cdot \mathbf{B}^\star$ $\mathbf{y}, \mathbf{y}' \in \mathbb{Z}^n$ and $\mathbf{z}, \mathbf{z} \in [-1/2, 1/2)^n$. Consider the quantity $\langle \mathbf{x}, \mathbf{b}_n^\star \rangle$, because $\mathbf{b}_n^\star$ is orthogonal to all $\mathbf{b}_i^\star$ and $\mathbf{b}_i^\star$, we have :

$$\langle \mathbf{x}, \mathbf{b}_n^\star \rangle = y_n \langle \mathbf{b}_n, \mathbf{b}_n^\star \rangle + z_n \langle \mathbf{b}_n^\star, \mathbf{b}_n^\star \rangle = y'_n \langle \mathbf{b}_n, \mathbf{b}_n^\star \rangle + z'_n \langle \mathbf{b}_n^\star, \mathbf{b}_n^\star \rangle .$$

It remains to note that $\langle \mathbf{b}_n, \mathbf{b}_n^\star \rangle = \langle \mathbf{b}_n^\star, \mathbf{b}_n^\star \rangle = \|\mathbf{b}_n^\star\|^2$, to deduce $y_n = y'_n$ and $z_n = z'_n$. To conclude, it remains to apply the induction hypothesis on $\mathbf{x} - (y_n\mathbf{b}_n + z_n\mathbf{b}_n^\star)$ which belongs to $\mathcal{L}(\mathbf{B}_{[n-1]})$. □

In particular, this gives a relation between the GSO of a basis and the volume of a lattice its spans, namely

Corollary 3.13 *If* $\mathbf{B}$ *is the basis of a* n*-dimensional lattice* $L \subset \mathbb{R}^m$*, and* $\mathbf{B}^\star$ *is its GSO, then*

$$\mathrm{Vol}(L) = \prod_{i=1}^{n} \|\mathbf{b}_i^\star\|$$

.

Proof: Simply note that $\mathrm{Vol}(L) = \mathrm{Vol}(\mathcal{P}(\mathbf{B}^\star))$ because $\mathcal{P}(\mathbf{B}^\star)$ is fundamental domain, and that $\mathrm{Vol}(\mathcal{P}(\mathbf{B}^\star)) = \prod_{i=1}^n \|\mathbf{b}_i^\star\|$ because $\mathbf{b}_i^\star$ are orthogonal. □

3.1.6 Lattice Sphere Packing, Hermite's constant

To restate the question of lattice sphere packings, Hermite introduced the following constant

Definition 3.9 (Hermite's constant) *Hermite's constant* γ_n *in dimension* n *is defined as the supremum of* $\lambda_1(L)^2/\mathrm{Vol}(L)^{2/n}$ *over all lattices* L *of rank* n*.*

It is not hard to realize that a lattice reaching this supremum is optimal for the Lattice Sphere Packing problem ; indeed for any lattice, the largest radius possible for sphere packing is $r = \lambda_1(L)/2$, and the supremum density is given by

$$\Delta_n = \mathrm{Vol}(r \cdot \mathfrak{B}_n)/\mathrm{Vol}(L) = \left(\frac{\lambda_1(L)}{2}\right)^n \cdot \mathrm{Vol}(\mathfrak{B}_n)/\mathrm{Vol}(L) = \frac{\mathrm{Vol}(\mathfrak{B}_n)}{2^n} \cdot \gamma_n^{n/2}$$

The determination of this constant has been a challenge to mathematicians since the very definition of lattices. Today, this constant is known only for very few dimension n, namely for $1 \leq n \leq 8$ ann $n = 24$.

dimension n	2	3	4	5	6	7	8	24
γ_n	$2/\sqrt{3}$	$2^{1/3}$	$\sqrt{2}$	$8^{1/5}$	$(64/3)^{1/6}$	$64^{1/7}$	2	4

The first upper bound is from hermite and is exponential :$\gamma_n \leq \gamma_2^{n-1}$. Minkowski establishes the first linear bound as a corollary of its fundamental Theorem :

Theorem 3.14 (Minkowski's Convex Body Theorem) *Let L be a full-rank lattice of $\mathbb{R}^n$ and let $\mathcal{S}$ be a set, convex, symmetric with respect to the origin $\mathbf{0}$, and of measure $\mathrm{Vol}(\mathcal{S}) > 2^n \mathrm{Vol}(L)$. Then, $\mathcal{S}$ contains a non zero lattice point : $\mathcal{S} \cap L \setminus \{\mathbf{0}\} \neq \emptyset$.*

Proof: Let $L' = 2L = \{2\mathbf{x} : \mathbf{x} \in L\}$, one easily establish that L' is a lattice and its volume is $\mathrm{Vol}(L') = 2^n \mathrm{Vol}(L)$. Because $\mathrm{Vol}(\mathcal{S}) > \mathrm{Vol}(L')$, fact 3.6 states that the union $\bigcup_{\mathbf{x}\in L'} \mathcal{S}+\mathbf{x}$ is not disjoint, that is there exists $\mathbf{x} \in L' \setminus \{\mathbf{0}\}$ such that $\mathcal{I} = \mathcal{S} \cap (\mathbf{x}+\mathcal{S}) \neq \emptyset$. We will prove that $\mathbf{x}' = \mathbf{x}/2 \in L \setminus \{\mathbf{0}\}$ is contained in $\mathcal{S}$. Notice that the intersection $\mathcal{I}$ is convex (as an intersection of convex), symmetric with respect to $\mathbf{x}'$, and non-empty. Therefore it contain its symmetry center $\mathbf{x}' : \mathbf{x}' \in \mathcal{S} \cap L \setminus \{\mathbf{0}\}$. □

We deduce the bound as a corollary :

Corollary 3.15 *For any lattice L of rank $n \geq 1$, we have*

$$\lambda_1(L)^n \cdot \mathrm{Vol}(\mathfrak{B}_n) \leq 2^n \mathrm{Vol}(L).$$

This implies that, for any $n \geq 1$, the Hermite's constant γ_n is bounded by

$$\gamma_n \leq \left(\frac{4}{\mathrm{Vol}(\mathfrak{B}_n)}\right)^{2/n} \leq 1 + n/4$$

Note that the bound obtained on $\lambda_1(L)$ is only a factor two worst than its intuitive value suggested by the Gaussian Heuristic (Heuristic 3.1).

3.1.7 Successive Minima

Definition 3.10 *Let $L \subset \mathbb{R}^m$ be a lattice of rank n. Then for all $1 \leq i \leq n$ the i-th minimum of the lattice L, $\lambda_i(L)$ is defined as the minimum of $\max\{\|\mathbf{v}_j\|\}$ where the set $\{\|\mathbf{v}_j\|\}$ runs over all sets of i linearly independent vectors of L. Equivalently, $\lambda_i(L)$ is the minimal radius r such that $L \cap r \cdot \mathfrak{B}_n$ spans an i-dimensional vector space.*

Note that by definition, the successive minima are increasing : $\lambda_1(L) \leq \lambda_2(L) \leq \cdots \leq \lambda_n(L)$. Those minima somehow inform us on the best possible basis of a lattice, that is, for any basis B, $\|\mathbf{b}_1\| \geq \lambda_1(L)$, and if this bound is reached, then $\|\mathbf{b}_2\| \geq \lambda_2(L)$ *etc.* However, there might not exist such an optimal basis; precisely there always exists a linearly independent family $\mathbf{b}_1 \dots \mathbf{b}_n$ reaching all this minima simultaneously, but they might not form a basis. Counterexamples exist for any dimension $n \geq 4$.

3.2 Discrete Gaussian Distributions

In continuous probability theory, Gaussian distributions play a very central role. One dimensional Gaussian are extremely natural because of the central limit Theorem, stating that the average of n independent samples from some distribution $\mathcal{D}$ over $\mathbb{R}$ has for limit distribution a Gaussian distribution. In many dimensions, they provide another interesting property : if $\mathbf{x} = (x_1 \dots x_n)$ is a random vector such that all coordinates x_i are sampled independently from the same centered Gaussian distribution, then the distribution of $\mathbf{x}$ has rotational symmetry (it is invariant under the orthogonal group $\mathcal{O}_n(\mathbb{R})$), that is the density probability at $\mathbf{x}$ only depends on $\|\mathbf{x}\|$.

One out of the many applications of Gaussian distribution in computer science is anti-aliasing for image processing; it consists of applying a convolution by a bi-dimensional Gaussian to a computer-generated image to smooth out pixel-related artifacts. This convolution process is usually deterministic for screen display, however, some printing techniques actually randomize it to render fifty shades of grey using only dots with a unique shade of black.

Besides the good geometrical and algebraic properties of Gaussians, their importance in lattice theory, and in lattice-based cryptography, comes from this smoothing property : Gaussians are good at hiding lattices cell. Conceptually, the only difference with the image processing application is that anti-aliasing usually considers pixels (pic-cells), that are cells of the two-dimensional lattice $\mathbb{Z}^2$, while we aim to hide the cells of arbitrary lattices.

3.2.1 Continuous Gaussian : Definition and properties

Definition 3.11 (Continuous Gaussian Distribution) *The one-dimensional continuous Gaussian distribution, of center $c \in \mathbb{R}$ and variance $\sigma^2 > 0$, noted $D_{\mathbb{R},\sigma,c}$ is the distribution over $\mathbb{R}$ of density probability at $x \in \mathbb{R}$*

$$D_{\mathbb{R},\sigma,c}(x) = \frac{1}{\sigma \cdot \sqrt{2\pi}} \cdot e^{-\frac{(x-c)^2}{2\sigma^2}}.$$

The continuous Gaussian distribution in dimension n, of center $c \in \mathbb{R}^n$ and covariance $\Sigma \in R^{n\times n}$ for a symmetric positive definite matrix $\Sigma > 0$, denoted by $D_{\mathbb{R},\sqrt{\Sigma},\mathbf{c}}$ is the distribution over $\mathbb{R}$ of density probability at $x \in \mathbb{R}$

$$D_{\mathbb{R}^n,\Sigma,\mathbf{c}}(\mathbf{x}) = \frac{1}{\sqrt{\det(\Sigma)} \cdot (2\pi)^{n/2}} \cdot e^{-\frac{1}{2}(\mathbf{x}-\mathbf{c})\Sigma^{-1}(\mathbf{x}-\mathbf{c})^t}.$$

If $\Sigma = \sigma^2 \cdot \mathbf{Id}_n$ for some $\sigma > 0$, then the previous distribution is said to be spherical, and can be denoted by $D_{\mathbb{R},\sigma,c}$; its probability density being

$$D_{\mathbb{R}^n,\sigma,\mathbf{c}}(\mathbf{x}) = \frac{1}{\left(\sigma\sqrt{2\pi}\right)^n} \cdot e^{-\frac{\|\mathbf{x}-\mathbf{c}\|^2}{2\sigma^2}}.$$

proposition 3.16 (Linear transformations of Gaussians) *Let $\Sigma_1, \Sigma_2 \in \mathbb{R}^{n\times n}$ be two symmetric positive definite matrices, and let $\mathbf{B} \in \mathbb{R}^{k\times n}$ be a non-singular matrix. If $\mathbf{x}_1$ and $\mathbf{x}_2$ are drawn independently according to $D_{\mathbb{R},\sqrt{\Sigma_1},\mathbf{c}_1}$ and $D_{\mathbb{R},\sqrt{\Sigma_2},\mathbf{c}_2}$, then*

- *$\mathbf{x}_1 \cdot \mathbf{B} \in \mathbb{R}^k$ follows the distribution $D_{\mathbb{R}^n,\sqrt{\Sigma_1},\mathbf{c}_1} \cdot \mathbf{B} = D_{\mathbb{R}^k,\sqrt{\mathbf{B}^t\Sigma_1\mathbf{B}},\mathbf{c}_1\cdot\mathbf{B}}$.*
- *$\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2 \in \mathbb{R}^n$ follows the distribution $D_{\mathbb{R}^n,\sqrt{\Sigma_1+\Sigma_2},\mathbf{c}_1+\mathbf{c}_2}$*

Note that the second statement can in fact be seen as a particular case of the first, by considering $(\mathbf{x}_1|\mathbf{x}_2)$ as being drawn from $D_{\mathbb{R}^{2n},\sqrt{\Sigma}}$ with $\Sigma = \left(\begin{smallmatrix}\Sigma_1 & \mathbf{0}\\ \mathbf{0} & \Sigma_2\end{smallmatrix}\right)$ under the transformation matrix $\mathbf{B} = (\mathbf{Id}_n|\mathbf{Id}_n)^t \in \mathbb{R}^{2n\times n}$.

3.2.2 Discrete Gaussian

Definition 3.12 *The (unnormalized) weight of Gaussian distribution of parameter $\sigma \in \mathbb{R}$ and center $c \in \mathbb{R}$ at $x \in \mathbb{R}$ is defined by $\rho_{\sigma,c}(x) = \exp\left(\frac{(x-c)^2}{2\sigma^2}\right)$, and more generally for a positive definite symmetric matrix $\Sigma \in \mathbb{R}^{n\times n}$ by*

$$\rho_{\sqrt{\Sigma},\mathbf{c}}(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x}-\mathbf{c})\Sigma^{-1}(\mathbf{x}-\mathbf{c})^t}.$$

We extend ρ to any countable set S, by letting $\rho_{\sqrt{\Sigma},\mathbf{c}}(S) = \sum_{x\in S}\rho_{\sigma,\mathbf{c}}(x)$ provided that this sum converges, which is the case for any subset S of any lattice[1] *. Finally, we omit the center $\mathbf{c}$ when it is zero : $\rho_{\sqrt{\Sigma}} = \rho_{\sqrt{\Sigma},\mathbf{0}}$, and extend the notation $\rho_{\sigma,\mathbf{c}} = \rho_{\sqrt{\Sigma},\mathbf{c}}$ for some scalar $\sigma > 0$ if $\Sigma = \sigma^2\mathbf{Id}_n$*

The discrete Gaussian distribution over $\mathbb{Z}$ is defined by the probabilities $D_{\mathbb{Z},\sigma,c}(x) = \rho_{\sigma,c}(x)/\rho_{\sigma,c}(\mathbb{Z})$ for any $x \in \mathbb{Z}$, and more generally, over a lattice L by

$$D_{L,\sqrt{\Sigma},\mathbf{c}}(\mathbf{x}) = \rho_{\sqrt{\Sigma},\mathbf{c}}(\mathbf{x})/\rho_{\sigma,\mathbf{c}}(L) \text{ for any } \mathbf{x} \in L.$$

Remark While the parameter σ still impacts the standard deviation of a discrete Gaussian, unlike the continuous Gaussian distribution $D_{\mathbb{R},\sigma,c}$, the distribution $D_{\mathbb{Z},\sigma,c}$ may have a different variance $\sigma'^2 \neq \sigma^2$; yet for large enough value of σ (precisely when $\sigma \geq \eta_\iota(\mathbb{Z})$ where $\eta_\iota(\mathbb{Z})$ is the smoothing parameter of $\mathbb{Z}$, as defined below) we will have $\sigma' \approx \sigma$. This naturally extends to multi-dimensional Gaussians over arbitrary lattices.

3.2.3 The Smoothing Parameter

In our earlier discussion, we made a heuristic explanation of the anti-aliasing property of Gaussian distributions. Obviously, the quality of that anti-aliasing depends of the chosen standard-deviation parameter σ for the Gaussian, the larger the smoother. Yet their will be drawbacks with using a Gaussian with too large deviation; in this anti-aliasing scenario, our original picture would get too blurry. Therefore, one would like to minimize the standard deviation to reach a certain level of smoothness. This is quantified by the smoothing parameter.

1. Start with noting that $(\mathbf{x}-\mathbf{c})\Sigma^{-1}(\mathbf{x}-\mathbf{c})^t > s\|\mathbf{x}-\mathbf{c}\|$ for some constant $s > 0$. Then note that the number of points $\mathbf{x} \in L$ such that $\|\mathbf{x}\| < \ell$ is $\mathcal{O}(\ell^n)$; conclude using the fact that $\int_{x>0} x^n e^{-sx^2} dx < \infty$.

Definition 3.13 (Smoothing Parameter [MR04]) *For any n-dimensional lattice L and any real $\iota > 0$, the* smoothing parameter *$\eta_\iota(L)$ (see [MR04]) is the smallest real $s > 0$ such that $\rho_{1/\sqrt{2\pi}s}(\hat{L} \setminus \{\mathbf{0}\}) \leq \iota$.*

Let us give the intuition behind this definition by considering the simpler case $L = \mathbb{Z}$. The parameter $\eta_\iota(\mathbb{Z})$ informs us on the minimal deviation σ so that the function $g(x) = \rho_\sigma(\mathbb{Z} + x) = \sum_{y \in \mathbb{Z}} \rho_\sigma(x + y)$ is about ι-close the a constant function. It is easy to see that g is $\mathbb{Z}$-periodic, therefore one may consider its Fourier coefficients c_i for $i \in \mathbb{Z}$. The Fourier transform $\hat{\rho}_\sigma$ of ρ_σ is proportional to $\rho_{1/2\pi\sigma}$; therefore each coefficient c_i is equal to $\rho_{1/\sigma}(i)$ up to a constant factor. The coefficient c_0 is simply the average value of g; while other coefficient measures the variations of g. The bound $\rho_{1/\sqrt{2\pi}s}(\mathbb{Z} \setminus \{\mathbf{0}\}) \leq \iota$ in the previous bound is equivalent to a bound on those coefficients c_i for $i \neq 0$. More formally, we have the following property.

Lemma 3.17 (Implicit in [MR04, Lemma 4.4]) *For any lattice L and $\iota \in (0, 1)$ $\sigma \geq \frac{1}{\sqrt{2\pi}}\eta_\iota(L)$ and $c \in span(L)$, we have :*

$$\sum_{\mathbf{x} \in L} \rho_\sigma(\mathbf{x} - \mathbf{c}) \in \left[\frac{1-\iota}{1+\iota}, 1\right] \cdot \sum_{\mathbf{x} \in L} \rho_\sigma(\mathbf{x})$$

or more compactly

$$\rho_{\sigma,\mathbf{c}}(L) \in \left[\frac{1-\iota}{1+\iota}, 1\right] \cdot \rho_{\sigma,0}(L)$$

Yet this bound is implicit, and computing the exact smoothing parameter of a lattice is in fact a hard problem [2]; the approximation problem complexity has been recently studied [CDLP13]. Nevertheless, for simple lattice such as $\mathbb{Z}^n$ it becomes rather easy to compute, and in general we have the following bound, that is almost tight for lattices $\mathbb{Z}^n$

Lemma 3.18 (Lemma 3.3 of [MR04]) *For any lattice $L \subset \mathbb{R}^m$ of dimension n and any $\iota \in (0, 1]$,*

$$\eta_\iota(L) \leq \sqrt{\ln(2n(1 + 1/\iota))/\pi} \cdot \lambda_n(L).$$

In particular, for any super-logarithmic function $\omega(\log n)$ there exists a negligible function $\iota(n)$ such that $\eta_\iota(L) \leq \sqrt{\omega(\log n)}\lambda_n(L)$

3.2.4 Tailcut

Another property of Gaussian distribution is their rapid decay at infinity. For example, in one dimension, about 99.7% percent of the mass of $D_{\mathbb{R},\sigma}$ is contained in the range $[-3\sigma, 3\sigma]$. In high dimension, this phenomena becomes even stronger, that is the length $\|\mathbf{v}\|$ of a vector $\mathbf{v} \leftarrow D_{\mathbb{R}^n,\sigma}$ is expected to be very close to $\sqrt{n} \cdot \sigma$ for large enough n. We refer to the standard studies on χ^2_n distributions (chi-squared distribution with n degrees of freedom) for the interested reader, present in most Statistic textbooks.

In the case of discrete distributions, one can establish similar facts : they will turn extremely useful for analysis and design as they let us ignore large vectors that would be drawn with negligible probability. The following lemmata are from Banaszczyk.

Lemma 3.19 ([Ban93]) *For any $\sigma > 0$, $c > \sigma/2\pi$, we have :*

$$\sum_{x > c} \rho_\sigma(x) \leq \frac{c\sqrt{e}}{2\sigma} \cdot \rho_\sigma(c)$$

Lemma 3.20 ([Ban93]) *For any $\sigma > 0$ and $\tau > 1/\sqrt{2\pi}$, and any n-dimensional lattice L, and vector $\mathbf{c} \in \mathbb{R}^n$, $\rho_\sigma((L + \mathbf{c}) \setminus \tau\sigma\sqrt{n}\mathfrak{B}) < 2C^n\rho_\sigma(L)$, where $C = \tau\sqrt{2\pi e} \cdot e^{-\pi\tau^2} < 1$, and $\mathfrak{B}$ is the centered unit ball.*

2. Notice that the asymptotic behavior of $\rho_{1/\sqrt{2\pi}s}(\hat{L} \setminus \{\mathbf{0}\})$ for $s \to +\infty$ discloses the first minimum $\lambda_1(\hat{L})$ of $\hat{L}$

3.2.5 Entropy

Another reason to use Gaussian distribution in cryptography, is that for a given standard deviation, the continuous Guassian is the distribution of maximal entropy ; that is, it optimizes the unpredictability of a random vector for a fixed expected length. This is also true for discrete Gaussian; this fact is believably standard, yet we propose our proof, using a convexity argument.

Lemma 3.21 (Distribution of maximal entropy) *Let $\sigma \geq 0$ and let $S \subset \mathbb{R}^n$ be a countable subset of a finite-dimensional vector space such that*

$$\rho_\sigma(S) < \infty, \quad \sum_{\mathbf{x}\in S} \rho_\sigma(\mathbf{x}) \cdot \ln(\rho_\sigma(\mathbf{x})) < \infty \quad \text{and } V = \frac{1}{\rho_\sigma(S)} \sum_{\mathbf{x}\in S} \|x\|^2 \rho_\sigma(x) < \infty.$$

Then, over all distribution of variance V over the support S, $D_{S,\sigma} : \mathbf{x} \in S \mapsto \rho_\sigma(\mathbf{x})/\rho_\sigma(S)$ is the distribution of maximal entropy.

Proof: We recall that for a distribution $\mathcal{P} : S \to [0,1]$, the entropy of $\mathcal{P}$ is defined as $H(\mathcal{P}) = -\sum_{i\in S} \mathcal{P}(i)\cdot\ln(\mathcal{P}(i))$. Since $x \mapsto -x\ln x$ is a convex function over $[0,1]$, the entropy function $H : F \to \mathbb{R}$ is also convex over the convex set F of functions from S to $[0,1]$.

The set D of distribution of variance V is the intersection $F \cap A_0 \cap A_2$ of F with the affine hyperplanes $A_0 = \{f \in (S \mapsto \mathbb{R}) : \sum_{\mathbf{x}\in S} f(\mathbf{x}) = 1\}$ and $A_2 = \{f \in (S \mapsto \mathbb{R}) : \sum_{\mathbf{x}\in S} \|\mathbf{x}\|^2 f(\mathbf{x}) = V\}$. This set is also convex. Since the entropy function is differentiable and convex over the convex set D, is it sufficient to prove that $dH(f)/df$ is zero at $f = D_{S,\sigma}$.

Set df to be a differential element of D, and denote by $df_{\mathbf{x}}$ its differential value at $\mathbf{x} \in S$. Because $D \subset A_0$ and respectively $D \subset A_2$, we have

$$\sum_{\mathbf{x}\in S} df_{\mathbf{x}} = 0 \quad \text{resp.} \sum_{\mathbf{x}\in S} \|\mathbf{x}\|^2 df_{\mathbf{x}} = 0 \tag{3.1}$$

Since $\frac{d}{dx} - x\ln x = -(1+\ln x)$, the differential of the entropy function H at some $f \in D$ is given by

$$H(f+df) - H(f) = -\sum_{\mathbf{x}\in S} (1 - \ln(f(\mathbf{x})))df_{\mathbf{x}}$$

in particular, for $f = D_{S,\sigma}$ one has

$$\begin{aligned} H(D_{S,\sigma} + df) - H(D_{S,\sigma}) &= -\sum_{\mathbf{x}\in S} (1 - \ln\frac{\rho_\sigma(\mathbf{x})}{\rho_\sigma(S)}) df_{\mathbf{x}} \\ &= -(1 - \ln\rho_s(S)) \cdot \underbrace{\sum_{\mathbf{x}\in S} df_{\mathbf{x}}}_{=0 \text{ by } (3.1)} + \sum_{\mathbf{x}\in S} \ln(\rho_s(\mathbf{x})) df_{\mathbf{x}} \\ &= \frac{-1}{2\sigma^2} \sum_{\mathbf{x}\in S} \|\mathbf{x}\|^2 df_{\mathbf{x}} = 0 \end{aligned}$$

where the last equality follows from (3.1). This concludes the proof. □

3.3 Lattices with Algebraic Structure

Definition 3.14 (Lattice over Polynomial-Rings) *Let $\mathcal{R}$ be the ring of polynomials modulo some polynomial $P \in \mathbb{Z}[X]$ of degree b, $\mathcal{R} = \mathbb{Z}[X]/(P(X))$. A lattice L of embedding rank l over $\mathcal{R}$ (a $\mathcal{R}$-lattice) is the set of $\mathcal{R}$-linear combinations of vectors $\mathbf{p}_1 \dots \mathbf{p}_k$ of $\mathcal{R}^l$:*

$$L = \mathcal{L}_{\mathcal{R}}(\{\mathbf{p}_1 \dots \mathbf{p}_k\}) = \mathrm{Span}_{\mathcal{R}}(\{\mathbf{p}_1 \dots \mathbf{p}_k\}) = \left\{ \sum_{i=1}^{k} a_i \mathbf{p}_i : a_i \in \mathcal{R} \right\}$$

Such a lattice can be seen as an integer lattice spanned by the vectors $\mathbf{b}_{i+b(j-1)} = \mathbb{Z}^{lb}(X^{i-1}\mathbf{p}_j(X))$ for $(i,j) \in \{1 \dots b\} \times \{1 \dots k\}$ where $\mathbb{Z}^{lb}(\mathbf{p})$ is the canonical representation of $\mathbf{p}$ in $\mathbb{Z}^{lb}$: each coordinate $p_i \in \mathcal{R}$ is represented by the vector of its coefficients $(c_0 \dots c_{b-1}) \in \mathbb{Z}^b$ so that $p_i(X) = \sum_{i=0}^{b-1} c_i X^i \in \mathcal{R}$.

The simplest example of such lattices are cyclic lattices, that are lattices over $\mathcal{R} = Z[X]/(X^b - 1)$. Seeing them as regular $\mathbb{Z}$ lattices, they are generated by matrices made of $b \times b$ circulant blocks

$$M = \begin{bmatrix} \mathcal{C}(a_{1,1}) & \dots & \mathcal{C}(a_{1,l}) \\ \vdots & & \vdots \\ \mathcal{C}(a_{k,1}) & \dots & \mathcal{C}(a_{k,l}) \end{bmatrix}$$

where $a_{i,j}$ are polynomials of degree b and $\mathcal{C}(a)$ for some polynomial $a = \sum_i a_i X^i \in \mathcal{R}$ denotes the circulant matrix

$$\begin{bmatrix} a_0 & a_1 & \cdots & a_{N-1} \\ a_{N-1} & a_0 & \cdots & a_{N-2} \\ \vdots & \ddots & \ddots & \vdots \\ a_1 & \cdots & a_{N-1} & a_0 \end{bmatrix}$$

Those lattices are interesting to build efficient cryptographic primitives, as they seem to provide as much security as general $n = kb$ dimensional integer lattices (for carefully chosen rings $\mathcal{R}$), while offering compact representations and efficient operations.

3.3.1 Cyclotomic Polynomials, Cyclotomic Field

Definition 3.15 (Cyclotomic polynomials) *The m-th cyclotomic polynomial $\Phi_m(X) \in \mathbb{Z}[X]$ (or simply Φ_m) is the minimal monic polynomial of any primitive m^{th} root of unity. More concretely, let $\zeta_m = e^{2\pi\imath/m}$, we have*

$$\Phi_m(X) = \prod_{k \in \mathbb{Z}_m^*} X - \zeta_m^k$$

The field $\mathbb{Q}(\zeta_m) \simeq \mathbb{Q}[X]/(\Phi_m(X))$ is called the m-th cyclotomic field.

The degree of Φ_m is $b = \phi(m)$, the Euler totient of m; it is an irreducible polynomial. For some values of m, $\Phi_m(X)$ has a simple form

- If m is prime $\Phi_m(X) = 1 + X + \cdots + X^{m-1}$, its degree is $\phi(m) = m - 1$
- If $m = 2^k$ is a power of two $\Phi_m(X) = X^{m/2} + 1$, its degree is $\phi(m) = m/2 = 2^{k-1}$

Note that the polynomial $X^p - 1$ used to build cyclic lattices is not cyclotomic, yet if b is prime it has only two irreducible factors

$$X^p - 1 = \Phi_p(X) \cdot (X - 1)$$

3.3.2 Canonical Embedding and Fourier Transform

The field $\mathbb{Q}(\zeta_m) \simeq \mathbb{Q}[X]/(\Phi_m)$ has exactly $\phi(m)$ embeddings, *i.e.* ring morphisms to $\mathbb{C}$, $(\sigma_k)_{k \in \mathbb{Z}_m^*}$, defined by $\sigma_k : x \mapsto x(\zeta_m^k)$, for $k \in \mathbb{Z}_m^*$. The canonical embedding $\sigma : \mathbb{Q}(\zeta_m) \to \mathbb{C}^{\phi(m)}$ is defined as the direct sum of all the embeddings : $\sigma(x) = \bigoplus_{k \in \mathbb{Z}_m^*} \sigma_k(x)$.

Using the fast Fourier transform (FFT), this allows multiplications of polynomial in time quasilinear in the dimension $\phi(m)$ of the ring. Indeed, for any polynomials $a, b \in Q[X]/(\Phi_m)$, one has $\sigma(a \cdot b) = \sigma(a) \odot \sigma(b)$ where $\odot$ denotes component-wise multiplication. The fast Fourier transform allows one to compute σ and σ^{-1} in quasilinear time [Bri88].

3.3.3 Number Theoretic Transform

In our application, we would rather be interested in computing products of polynomials of $\mathbb{Z}[X]/(\Phi_m)$ modulo some integer q. The general Fourier transform approach requires to deal with continuous complex coefficients, and its implementation require floating-point arithmetic, with approriate precision. Yet, if we are only interested in the result modulo some integer q (in other words perform the product in $\mathbb{Z}_q[X]/(\Phi_m)$) one can resort to the Number Theoretic Transform (NTT). This is possible if there exists m-th primitive roots of unity over the finite field $\mathbb{F}_q$; more concretely the requirement are that q must be prime, and m must be co-prime to q : $m \equiv 1 \bmod q$. The overall algorithm is similar, except we would work with embeddings to $\mathbb{F}_q$: $\sigma_k : x \mapsto x(z_m^k)$ where $z_m \in \mathbb{F}_q$ is a m-th primitive root of the unity over $\mathbb{F}_q$.

3.4 Complexity in Geometry of Numbers

In addition to their fascinating mathematical aspect, lattices are also connected to fundamental questions in complexity theory. Indeed, there are several optimization tasks defined over lattices providing examples of various complexity classes. We will focus on the ones relevant to this thesis; a rich survey can be found in [NV10, Chapters 14 and 15].

3.4.1 Hardness of Exact Problems : SVP and CVP

Probably, the most studied computational problem on lattices is the Shortest Vector Problem (SVP).

Definition 3.16 (The Shortest Vector Problem, SVP) *Given a basis* $\mathbf{B}$ *of a lattice full-rank* $L = \mathcal{L}(\mathbf{B})$*, find one of the shortest non-zero vectors of* L*, that is, find a vector* $\mathbf{v} \in L$ *such that* $\|\mathbf{v}\| = \lambda_1(L)$*. Unless otherwise specified, we consider this problem for the* ℓ_2*-norm.*

The first algorithm for this problem was given by Lagrange [Lag73] (sometimes also attributed to Gauss [Gau01]) for lattices of dimension 2; it is a variant of the **gcd** algorithm. After that, a lot of work was done on various notions of reductions for lattice basis; some of them can be paired with algorithms to find short vectors. Yet, those have running time at least exponential in the dimension n of the lattice. The first hardness result for SVP is due to van Emde-Boas [vEB81], stating that SVP is $\mathsf{NP}-\mathsf{hard}$ for the ℓ_∞ norm; only ten years after the very definition of NP-hardness by Cook and Levin [Coo71]. While it was conjectured that it should be as hard for any ℓ_p norm, it is only in 1998 that it was established for the ℓ_2 norm (under randomized reduction), in the breakthrough work of Ajtai [Ajt98].

Another elementary problem is now to find a closest lattice vector to a given target vector.

Definition 3.17 (The Closest Vector Problem, CVP) *Given a basis* $\mathbf{B}$ *of a lattice* $L = \mathcal{L}(\mathbf{B}) \subset \mathbb{R}^m$*, and a target* $\mathbf{t} \in \mathbb{R}^m$ *find the closest vector of* L *to* $\mathbf{t}$*, that is finding a vector* $\mathbf{v} \in L$ *that minimize* $\|\mathbf{t} - \mathbf{v}\|$*. Unless otherwise specified, we consider this problem for the* ℓ_2*-norm.*

Its NP-hardness was also established in [vEB81]. This problem can be restated as reducing a point modulo L to the fundamental domain $\mathcal{V}(L)$, the Voronoï cell. Interestingly, this problems remains $\mathsf{NP}-\mathsf{hard}$ even if one is given the lattice in advance, allowing arbitrary pre-computation before $\mathbf{t}$ is revealed; this is known as the Closest Vector Problem with Precomputation CVPP. The first algorithm for this CVP is from Kannan [Kan87] and run in deterministic time $n^{\mathcal{O}(n)}$. Randomized single exponential time $O(2^n)$ was reached by using a sieve algorithm [AKS01]; finally, Micciancio and Voulgaris found deterministic single exponential time algorithm for CVP, but also to fully compute the whole Voronoï cell $\mathcal{V}(L)$ [MV10].

Note that all those problems are not known to be in NP (they are $\mathsf{NP}-\mathsf{hard}$ but not known to be $\mathsf{NP}-\mathsf{complete}$); giving a solution does not trivially prove that no shorter solution exists.

3.4.2 Hardness of Approximation Problems : SVP_γ and CVP_γ

Both previous problems have natural approximate versions.

Definition 3.18 (Approximate Shortest Vector Problem, SVP_γ) *For any* $\gamma \geq 1$*, the approximate shortest vector problem* SVP_γ *is as follows : given a basis* $\mathbf{B}$ *of a lattice* $L = \mathcal{L}(\mathbf{B})$*, find a non-zero vector of* L*,* $\mathbf{v} \in L$ *such that* $\|\mathbf{v}\| \leq \gamma\lambda_1(L)$*.*

Definition 3.19 (Approximate Closest Vector Problem, CVP_γ) *For any* $\gamma \geq 1$*, the approximate closest vector problem* CVP_γ *is as follows : given a basis* $\mathbf{B}$ *of a lattice* $L = \mathcal{L}(\mathbf{B})$*, a target point* $\mathbf{t} \in \mathbb{R}^m$*, find vector of* L*,* $\mathbf{v} \in L$ *such that* $\|\mathbf{v} - \mathbf{t}\| \leq \gamma \min_{\mathbf{x}\in L} \|\mathbf{x} - \mathbf{t}\|$

Goldreich et al. [GMSS99] gave a polynomial-time reduction from SVP_γ to CVP_γ for any value of γ; approximating SVP to a factor γ is not harder than approximating CVP to the same factor. Then, Arora *et al.* [ABSS93] proved that CVP_γ is $\mathsf{NP}-\mathsf{hard}$ [3] for $\gamma = 2^{\log^{1-\epsilon} n}$ for small constants $\epsilon > 0$; and the result was improved by Dinur *et al.* [DKRS03].

Even the variant with precomputation is known to be inaproximable; Feige and Micciancio established the NP-hardness of CVPP_γ for $\gamma = \sqrt{5/3}$, later improved by Regev to $\sqrt{3}$ [FM04, Reg03].

Those problems becomes easy when the approximation factor γ becomes exponential in n as we will see in the next section. The hardness of those problems for γ polynomial in n remains an open problem. Since all lattice primitives rely on the hardness of such problems for polynomial approximation factor; those problems are of extreme theoretical and practical interest.

3. Unless there exists quasi-polynomial algorithms for $\mathsf{NP}-\mathsf{hard}$ problems

3.4.3 Problems with Promises : uSVP_γ and BDD_γ

An alternative relaxation of those problems can also come from promise of a gap between the optimal solution and all others.

Definition 3.20 (Unique Shortest Vector Problem, uSVP_γ) *For any $\gamma \geq 1$, the unique shortest vector problem SVP_γ is as follows : given a basis $\mathbf{B}$ of a lattice $L = \mathcal{L}(\mathbf{B})$ such that $\lambda_2(L) \geq \gamma\lambda_1(L)$, find a shortest non-zero vector of L, $\mathbf{v} \in L$ such that $\|\mathbf{v}\| = \lambda_1(L)$.*

Definition 3.21 (Bounded Distance Decoding Problem, BDD_ℓ) *For any $\ell > 0$, the Bounded Distance Decoding Problem is to find a lattice point $\mathbf{c} \in L$ that is at distance at most ℓ from a given target $\mathbf{t}$: $\|\mathbf{t} - \mathbf{v}\| \leq \ell$, assuming that such a solution exists.*

Note that this second problem becomes similar to approximation of CVP if ℓ is greater than the covering radius (in which case the assumption that such solution exist is true for every $\mathbf{t}$). For the hardness results for this problem, we refer to a complete survey by Khot [NV10, Chapter 14] on the matter.

3.4.4 Problems SIS and LWE, Worst-case to Average case Connection

For cryptography purposes, we will use mainly two problems that are connected to lattice problems, namely SIS and LWE.

Definition 3.22 (The Short Integer Solution Problem, SIS) *The Short Integer Solution problem $\mathsf{SIS}_{n,m,q,\beta}$, with m unknowns, $n \leq m$ equations modulo q and norm-bound β is as follows : given a random matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ chosen uniformly, find a non-zero short vector $\mathbf{v} \in \mathbb{Z}_q^m \setminus \{\mathbf{0}\}$ such that $\mathbf{v} \cdot \mathbf{A} = \mathbf{0}$ and $\|\mathbf{v}\| \leq \beta$.*

If the bound β is set too low the problem is vacuously hard ; solutions are expected to exist when the linear application associated to $\mathbf{A}$ is expected to be surjective on the sub-domain of vectors of norms less than β ; for the ℓ_2 that is when $\beta^m \cdot \mathrm{Vol}(\mathfrak{B}_m) \geq q^n$. The natural lattice associated to the problem is the following

Definition 3.23 (The SIS-lattice and its cosets) *For integers parameters $n, m > n, q$, and for a $\mathsf{SIS}_{n,m,q,}$. instance matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, the SIS-lattice lattice associated to $\mathbf{A}$, $\mathcal{L}^{\perp q}(\mathbf{A})$, (or simply $\mathcal{L}^{\perp}(A)$ when q is obvious from the context) is defined as :*

$$\mathcal{L}^{\perp q}(\mathbf{A}) = \{\mathbf{v} \in \mathbb{Z}^m : \mathbf{v} \cdot \mathbf{A} \equiv \mathbf{0} \bmod q\}.$$

In other words, it is the lattice defined by $\mathbf{A}$ as a parity-check matrix. Additionally, if the syndrom $\mathbf{u} \in \mathbb{Z}_q^n$ admits an integral solution to $\mathbf{x} \cdot \mathbf{A} = \mathbf{u}$, then we define $\mathcal{L}_{\mathbf{u}}^{\perp q}(\mathbf{A})$, the coset of $\mathcal{L}^{\perp q}(\mathbf{A})$ of syndrom $\mathbf{u}$ as

$$\mathcal{L}_{\mathbf{u}}^{\perp q}(\mathbf{A}) = \{\mathbf{v} \in \mathbb{Z}^m : \mathbf{v} \cdot \mathbf{A} \equiv \mathbf{u} \bmod q\}.$$

With this definition, SIS can be seen as a version of approximate SVP_γ for a specific distribution of lattice. One may compute the value of γ using the fact that $\mathrm{Vol}(L) = q^n$ with overwhelming probability (there are q^n co-sets of L since $\mathbf{c} \cdot \mathbf{A} \bmod q$ can take q^n values when A has rank n), and the expected value of $\lambda_1(L)$ according to the Gaussian Heuristic.

Still, it is not obvious that this specific distribution does make the SVP_γ hard ; indeed, it is possible that some class of instances makes SVP_γ actually easy. Yet, when introduced in the founding work of Ajtai [Ajt96] for cryptographic purpose, it came with a worst-case to average case reduction ; that is a proof that solving random SIS instance is not easier than solving any instance of certain lattice problems (uSVP, or the Approximate Short Independent Vectors Problem SIVP_γ). This result was later improved and simplified [MR04, GPV08].

Theorem 3.22 ([Ajt96, MR04, GPV08]) *If there exists a Probabilistic Polynomial Time (PPT) algorithm $\mathcal{A}$ that solves $\mathsf{SIS}_{n,m,q,\beta}$ instances with $q \geq 2\beta\sqrt{n}$ with non negligible probability ; then there exists a (PPT) algorithm $\mathcal{B}$ that solves $\mathsf{uSVP}_{2\beta\sqrt{n}}$ and $\mathsf{SIVP}_{2\beta\sqrt{n}}$ on any lattice (*i.e. in the worst case*), of dimension n.*

There exists an inhomogeneous variant of this problem, ISIS

Definition 3.24 (The Inhomogeneous Short Integer Solution Problem, ISIS) *The Inhomogeneous Short Integer Solution problem* $\mathsf{ISIS}_{n,m,q,\beta}$*, with* m *unknown,* $n \leq m$ *equations, modulo* q *and norm-bound* β *is as follows : given a uniformly random matrix* $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ *and a uniform target* $\mathbf{t}$*, find a short vector* $\mathbf{v} \in \mathbb{Z}_q^m \setminus \{\mathbf{0}\}$ *such that* $\mathbf{v} \cdot \mathbf{A} = \mathbf{t}$ *and* $\|\mathbf{v}\| \leq \beta$*.*

It is also proved as hard as worst-case lattice problems [GPV08]. To rephrase both problems, the SIS problem requires one to find a non-zero short vector of $\mathcal{L}^{\perp}(\mathbf{A})$ given a matrix $\mathbf{A}$, and given an additional syndrom $\mathbf{t}$, ISIS requires a short vector of $\mathcal{L}^{\perp}_{\mathbf{u}}(\mathbf{A})$.

The second problem, LWE was popularized by the results of Regev [Reg05], close variants but already appeared in previous works [BFKL94, AD97, Ale03]. It can be seen as instances of BDD for specific lattice and syndrome distribution.

Definition 3.25 (The Learning with Errors Problem, sLWE, search version) *The Learning with Errors Problem, search version,* $\mathsf{sLWE}_{n,m,q,\chi}$*, with* n *unknown,* $m \geq n$ *samples, modulo* q *and with errors distribution* χ *is as follows : for a random secret* $\mathbf{s}$ *uniformly chosen in* $\mathbb{Z}_q^n$*, and given* m *samples of the form* $(\mathbf{a}, b = \langle \mathbf{s}, \mathbf{a} \rangle + e \bmod q)$ *where* $e \leftarrow \chi$ *and* $\mathbf{a}$ *is uniform in* $\mathbb{Z}_q^n$*, recover the secret vector* $\mathbf{s}$*.*

The first property of LWE for cryptographic purpose is the equivalence with decisional version ; the above problem is no easier than the one below when q is polynomial in n

Definition 3.26 (The Learning with Errors Problem, dLWE, decisional version) *The Learning with Errors Problem, decisional version,* $\mathsf{dLWE}_{n,m,q,\chi}$*, with* n *unknown,* $m \geq n$ *samples, modulo* q *and with errors distribution* χ *is as follows : for a random secret* $\mathbf{s}$ *uniformly chosen in* $\mathbb{Z}_q^n$*, and given* m *samples either all of the form* $(\mathbf{a}, b = \langle \mathbf{s}, \mathbf{a} \rangle + e)$ *where* $e \leftarrow \chi$*, or from the uniform distribution* $(\mathbf{a}, b) \leftarrow \mathcal{U}(\mathbb{Z}_q^n \times \mathbb{Z}_q)$ *; decide if the samples comes from the former or the latter case.*

Theorem 3.23 (Decision to Search Reduction for LWE) *For any integers* n *and* m*, any prime* $q \leq \mathrm{poly}(n)$*, and any distribution* χ *over* $\mathbb{Z}_q$*, if there exists a PPT algorithm that solves* $\mathsf{dLWE}_{n,m,q,\chi}$ *with non-negligible probability, then there exists a PPT algorithm that solves* $\mathsf{dLWE}_{n,m',q,\chi}$ *for some* $m' = m \cdot poly(n)$ *with non-negligible probability.*

This results has also been generalized to other smooth modulus [MP12] ; informally we will talk about the LWE problem without distinction between search and decisional version.

To state this problem in term of lattice, consider the matrix $\mathbf{A} = [\mathbf{a}_1 \dots \mathbf{a}_m]$ whose rows are the samples $\mathbf{a}_i$, the vector $\mathbf{b} = \mathbf{s} \cdot \mathbf{A}^t + \mathbf{e}$ where $\mathbf{e} \leftarrow \chi^m$, and define

Definition 3.27 (The LWE-lattice) *For integers parameters* $n, m > n, q$*, and for a* $\mathsf{LWE}_{n,m,q,}$*. instance matrix* $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$*, the* LWE*-lattice associated to the instance* $(\mathbf{A}, \mathbf{b})$*,* $\mathcal{L}^q(\mathbf{A}^t)$*, is defined as :*

$$\mathcal{L}^q(\mathbf{A}^t) = \left\{ \mathbf{v} \in \mathbb{Z}^m : \exists \mathbf{s} \in \mathbb{Z}_q^n s.t. \mathbf{v} \equiv \mathbf{s} \cdot \mathbf{A}^t \bmod q \right\}.$$

In other words, $\mathcal{L}^q(\mathbf{A}^t)$ *is the lattice generated by the column vectors of* $\mathbf{A}$ *and the column vectors of* $q\mathbf{Id}_m$*,* i.e. *the canonical vectors scaled by* q*.*

When $\mathbf{e}$ is small, the problem can now be seen as follows : given a point $\mathbf{b}$ close to a random lattice point $\mathbf{s} \cdot \mathbf{A}^t$, one is asked to recover $\mathbf{s}$ or equivalently the lattice point $\mathbf{s} \cdot \mathbf{A}$; LWE is a variant of BDD for a certain distribution of lattice and error.

As for the SIS problem, the LWE problem was shown to be equivalent to hard lattice problem *in the worst case* such as uSVP_γ or SIVP_γ for centered discrete Gaussian error distribution $\chi = D_{\mathbb{Z},\sigma}$; that is solving LWE instance on average is no easier than solving the hardest instances of certain lattice problems.

Theorem 3.24 (Worst-case to Average-case Connection for LWE [Reg05]) *If there exists a Probabilistic Polynomial Time (PPT) algorithm* $\mathcal{A}$ *that solves* $\mathsf{LWE}_{n,m,q,\chi}$ *for* $\chi = D_{\mathbb{Z},\alpha q}$ *where* $\alpha q > 2\sqrt{n}$ *with non negligible probability ; then there exists a Quantum Polynomial Time algorithm* $\mathcal{B}$ *that solves* uSVP_γ *and* SIVP_γ *on any lattice (*i.e. in the worst case*) of dimension* n *where* $\gamma = \tilde{\mathcal{O}}(n/\alpha)$*.*

Yet, the original reduction [Reg05] was a quantum algorithm ; in certain cases this reduction could be made classical [Pei09, BLP+13]. The limitation to Gaussian distributions was also overcomed in recent works [DMQ13, MP13, AKPW13].

Let us conclude with an additional property of LWE, which is that the problem is not much easier if the coordinates of the secret are drawn from the same distribution as the error.

Definition 3.28 (LWE with Small Secret dLWE′) *The Learning with Errors Problem with small error,* $\mathsf{dLWE}'_{n,m,q,\chi}$*, with* n *unknown,* $m \geq n$ *samples, modulo* q *and with errors distribution* χ *is as follows : for a random secret* $\mathbf{s}$ *drawn from* $-\chi^n$*, and given* m *samples either all of the form* $(\mathbf{a}, b = \langle \mathbf{s}, \mathbf{a} \rangle + e)$ *where* $e \leftarrow \chi$*, or from the uniform distribution* $(\mathbf{a}, b) \leftarrow \mathcal{U}(\mathbb{Z}_q^n \times \mathbb{Z}_q)$ *; decide if the samples comes from the former or the latter case.*

Lemma 3.25 (Reduction from dLWE to dLWE′ [ACPS09]) *If there is an algorithm* $\mathcal{A}$ *that solves* $\mathsf{dLWE}'_{n,m,q,\chi}$ *for* $m \geq n$*, and* $q \geq 2$ *in time* T *with probability* p*, then there exists an algorithm* $\mathcal{B}$ *that solves* $\mathsf{dLWE}_{n,m+n,q,\chi}$ *in time* $T + \mathrm{poly}(n, m, \log(q))$ *with probability* $p - 2^{-\Omega(n))}$*.*

Proof: First, assume we are given samples from $\mathsf{dLWE}_{n,m+n,q,\chi}$: $(\mathbf{A}, \mathbf{b} = \mathbf{sA} + \mathbf{e})$. Except with probability less than $2^{-\Omega(n)}$, there exists a subset of rows of $\mathbf{A}$ that forms an invertible matrix modulo $q \geq 2$. With loss of generality, let us assume that the n first rows of $\mathbf{A}$ form an invertible matrix : $\mathbf{A}^t = (\mathbf{A}_1^t | \mathbf{A}_2^t)$ where $\mathbf{A}_1 \in \mathbb{Z}_q^{n \times n}$; decompose $\mathbf{b}^t = (\mathbf{b}_1^t | \mathbf{b}_2^t)$ as well. It remains to set

$$\begin{aligned} \mathbf{b}' &= \mathbf{b}_2 - \mathbf{b}_1 \cdot \mathbf{A}_1^{-1} \cdot \mathbf{A}_2 \\ &= \mathbf{sA}_2 + \mathbf{e}_2 - (\mathbf{sA}_1 + \mathbf{e}_2) \cdot \mathbf{A}_1^{-1} \cdot \mathbf{A}_2 \\ &= -\mathbf{e}_1 \cdot \mathbf{A}_1^{-1} \cdot \mathbf{A}_2 + \mathbf{e}_2 \end{aligned}$$

and to notice that $(\mathbf{A}_1^{-1} \cdot \mathbf{A}_2, \mathbf{b}')$ forms a properly distributed set of samples from $\mathsf{dLWE}'_{n,m,q,\chi}$ for the secret $-\mathbf{e}_1$. On the other hand, one would verify that if $(\mathbf{A}, \mathbf{b})$ comes from a uniform distribution rather than the legitimate LWE distribution, then applying the same transformation does provide a uniform distribution for $(\mathbf{A}_1^{-1} \cdot \mathbf{A}_2, \mathbf{b}')$. □

3.4.5 Ring Version of LWE

A major issue of the previous problem LWE to base cryptographic primitives is the blow-up factor; for a secret vector $\mathbf{s} \in \mathbb{Z}_q^n$ one require a full vector $\mathbf{a} \in \mathbb{Z}_q^n$ to create one pseudo-random scalar $b \in \mathbb{Z}_q$; used to mask a single bit (or a scalar in $\mathbb{Z}_{q'}$ for $q' = \mathrm{poly}(n)$). This will imply a ciphertext to plaintext size ratio of a least $\mathcal{O}(n)$.

It is therefore tempting make a security/efficiency trade-off, by using vectors $\mathbf{a}_i$ that are related to each others, so a party can disclose just a few of them, and let the other party derive the other $\mathbf{a}_i$'s. Using matrix terminology, this can be seen as restricting the matrix $\mathbf{A}$ to a certain set of structured matrices, Toeplitz matrices for example, or block-circulant matrices. Yet, this restriction invalidates the hardness reduction ; and for certain type of structures, the problem is known to become indeed easy. For example, in the case of block circulant matrix, the decisional LWE problems becomes easy : let $\mathbf{A}$ be square $n \times n$ circulant matrices, the fact that

$$\mathbf{s} \cdot \mathbf{A} \cdot (1 \dots 1)^t = (\sum s_i) \cdot a \text{ where } a = \sum A_{1,i}$$

leads to a simple distinguisher given a few samples $\mathbf{s} \cdot \mathbf{A} + \mathbf{e}$.

Algebrically, the previous attack can be interpreted as follows ; the square circulant matrices $\mathbb{Z}^{n \times n}$ forms a ring that is homomorphic to the ring of polynomials $\mathcal{R} = \mathbb{Z}[X]/(X^n - 1)$. The polynomial $(X^n - 1)$ can be factored as $(X - 1) \cdot (X^{n-1} + \cdots + X + 1)$, and the attack arises from the low degree factor $(X - 1)$; in other words, by the Chinese remainder theorem, the ring $\mathcal{R}$ can be decomposed as $\mathcal{R} = (\mathbb{Z}[X]/(X-1)) \times (\mathbb{Z}[X]/(X^{n-1} + \cdots + X + 1))$. Yet, when the ring we work with can not be factored, no better attacks against those structured-LWE instances are known.

This motivated a new definition of LWE over polynomial rings, ring-LWE by Lyubashevsky, Regev and Peikert [LPR10] ; yet the most general definition requires some algebraic tools that are beyond the scope of this Thesis. We restrict our attention to very specific rings $\mathcal{R} = \mathbb{Z}[X]/(\Phi_n)$ for $n = 2^k$ a power of 2 ; recalling that $\Phi_n = X^{n/2} + 1$ is a cyclotomic (therefore irreducible) polynomial. The interested reader is refered to [LPR10, DD12, LPR13].

Definition 3.29 (Ring-dLWE for the ring $\mathcal{R} = \mathbb{Z}[X]/(\Phi_{2^k})$) *Let* $\mathcal{R}$ *denote the ring* $\mathcal{R} = \mathbb{Z}[X]/(\Phi_n)$ *for* $n = 2^k$ *integer power of* 2*, and* $\mathcal{R}_q = \mathcal{R}/(q\mathcal{R})$ *for some integer* q*. The* $\mathcal{R} - \mathsf{dLWE}_{m,q,\chi}$*,* $m \geq 1$ *samples, modulo* q *and with errors distribution* χ *over* $\mathbb{Z}$ *is as follows : for a random secret* $\mathbf{s} \in \mathcal{R}_q$*, with*

uniform coefficients in $\mathbb{Z}_q$, and given m samples either all of the form $(a, b = a \cdot s + e)$ where $e \in \mathcal{R}_q$ has coefficients drawn from χ, or from the uniform distribution $(a, b) \leftarrow \mathcal{U}(\mathcal{R}^2)$; decide if the samples come from the former or the latter case.

The main result of [LPR10] is a worst-case to average-case reduction to hard lattice problems similar to the regular version of LWE, but only to a subclass of lattices. Similarly to regular LWE, there is a variant of the problem with small secret, which is equivalent to the uniform secret one when the secret coefficients are also drawn from the distribution χ.

3.5 Super-Polynomial Approximation Algorithms

3.5.1 LLL, Finding Short Vectors and Short Basis

The LLL algorithm [LLL82] is an algorithm by Lenstra, Lenstra and Lovász to find short vectors in lattices in polynomial time, but it only guarantees an exponential approximation of the shortest vectors. Nevertheless, this algorithm turned out to have numerous application in cryptanalysis, including for the RSA encryption scheme [Cop97], whose relation to lattices is not straightforward. It also has impact on problems outside the realm of cryptanalysis ; some applications are detailed in a celebratory book [NV10].

The LLL algorithm in fact works with a whole basis, trying to transform it to achieve shortness and near-orthogonality. More precisely

Definition 3.30 (LLL Reduced Basis) *For $\delta \in (1/4, 1]$, a basis $\mathbf{B}$ of a lattice $L = \mathcal{L}(\mathbf{B})$ of dimension n, is said to be δ-LLL-reduced if*

- *it is* size reduced*, that is $|\mu_{i,j}| \leq 1/2$ for any $i > j$ where μ denotes the GSO transformation matrix ;*
- *it verifies Lovász conditions : for all $1 \leq i < n$*

$$\left\|\mathbf{b}^\star_{i+1} + \mu_{i+1,i}\mathbf{b}^\star_i\right\|^2 \geq \delta \left\|\mathbf{b}^\star_i\right\|^2 \text{ or equiv. } \left\|\mathbf{b}^\star_{i+1}\right\|^2 \geq (\delta - \mu^2_{i+1,i}) \left\|\mathbf{b}^\star_i\right\|^2$$

This definition is a relaxation of the Hermite's notion of reduction ; precisely for $\delta = 1$, the condition implies that two consecutive Gram-Schmidt vectors of such a reduced basis forms an optimal basis for the 2-dimensional lattice they span :

$$\forall i < d, \|\mathbf{b}^\star_i\| = \lambda_1(\mathcal{L}(\{\pi_i(\mathbf{b}_i), \pi_i(\mathbf{b}_{i+1})\}) \text{ and } \left\|\mathbf{b}^\star_{i+1}\right\| = \lambda_2(\mathcal{L}(\{\pi_i(\mathbf{b}_i), \pi_i(\mathbf{b}_{i+1})\})$$

where π_i denote the projection orthogonaly to $\text{Span}\{\mathbf{b}_1 \dots \mathbf{b}_{i-1}\}$ as in definition 3.8. This notion of reduction ensures the following qualities of the basis

Theorem 3.26 (Qualities of LLL-reduced basis) *For any $\delta \in (1/4, 1]$ and any $\alpha = 1/(\delta - 1/4)$. If $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_n]$ is a δ-LLL-reduced basis of an n-dimensional lattice L then*

- $\|\mathbf{b}_1\| \leq \alpha^{(n-1)/4} \text{Vol}(L)^{1/n}$
- $\|\mathbf{b}_i\| \leq \alpha^{(n-1)/2} \cdot \lambda_i(L)$ *for all* $i \leq n$
- $\prod_{i=1}^n \|\mathbf{b}_i\| \leq \alpha^{n(n-1)/4} \text{Vol}(L)$

The size reduction notion is quite simple to achieve, it can be seen as an GSO procedure where the coefficients $\mu_{i,j}$ are rounded to the nearest integer. This suggest the following algorithm

Algorithm 1 LLL Algorithm

Input: A basis $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_n]$ of a lattice L, a parameter δ
Output: A δ-LLL-reduced basis of the lattice L

1: Size-reduce $\mathbf{B}$
2: **if** there exists an index j which does not satisfies Lovaśz' condition **then**
3: swap $\mathbf{b}_j$ and $\mathbf{b}_{j+1}$ and return to Step 1
4: **end if**

The correctness is rather obvious, the technicalities comes from proving termination. Termination can indeed be proved in general, but to ensure polynomial running time one needs $\delta < 1$; that is we don't know how to obtain exactly Hermite's reduced basis in polynomial time, but we can approximate it. We refer to the very detailed survey [NV10, Chapter 1] for the analysis and the relation with Hermite's reduction algorithm. Interestingly, the performance in practice are usually much better than what would be expected from Theorem 3.26 ; yet those bounds remains exponential in the dimension n.

3.5.2 SVP Enumeration Algorithm and BKZ

The BKZ algorithm (for Blockwize-Korkine-Zolotarev) can be seen as a generalization of LLL that search for a basis such that each vector $\pi_i(\mathbf{b}_i)$ is a shortest vector of the lattice generated by $\pi_i(\mathbf{b}_i) \dots \pi_i(\mathbf{b}_{i+k-1})$ where π_i denotes the projection orthogonally to $\mathbf{b}_1 \dots \mathbf{b}_{i-1}$; whereas LLL does that for $k = 2$. In other words, LLL is an relaxed algorithmic version of Hermite's inequality : $\gamma_n \leq \gamma_2^{n-1}$; and BKZ an algorithmic version of Mordell's inequality : $\gamma_n \leq \gamma_k^{(n-1)/(k-1)}$.

Rather than the swap instruction, BKZ will use a subroutine that search for the shortest vector in lattice of dimension k; asymptotically, the best known algorithm for this task is the Sieve algorithm from Ajtai, Kummar and Sivakumar [AKS01], that run in time $2^{\mathcal{O}(k)}$; it is therefore essential to keep the block-size k low to run BKZ in polynomial time. In practice however, one would rather use short vector enumeration algorithm from Kannan-Fincke-Pohst [FP83,PS08] (that runs in $2^{\mathcal{O}(d^2)}$ given an LLL-reduced basis), with a pruning technique [SE93, SH95, GNR10] that drastically decreases the practical running time.

3.5.3 Behavior of LLL and BKZ

Experimentally, the basis resulting from LLL and BKZ algorithm follows a profile that depends only on the so called "Hermite factor" δ associated to the reduction algorithm; or equivalently the progression ratio $c \in \mathbb{R}$.

Heuristic 3.2 (Geometric Series Assumption) *For either LLL or BKZ-k for any block-size k, there exists a constant $c > 1$ such that reduced basis verifies $\|\mathbf{b}_i^\star\| / \|\mathbf{b}_{i+1}^\star\| \approx c$ for all indexes i. From volume equality, and setting $\delta = \sqrt{c}$ (the Hermite factor), this implies*

$$\|\mathbf{b}_1^\star\| \approx \delta^n \cdot \mathrm{Vol}(L)^{1/n} \text{ and } \|\mathbf{b}_i^\star\| \approx \|\mathbf{b}_i^\star\| \cdot c^{-i}$$

Typical values are $\delta_{\mathrm{LLL}} = 1.02$ for LLL, and $\delta_{\text{BKZ-20}} = 1.012$ for BKZ-20. An heuristic analysis explaining this behavior for large blocksizes was recently detailed by Chen and Nguyen [CN11], this allowed models and prediction of $\delta_{\text{BKZ-k}}$, and BKZ$-k$ running time for increasing block-size k.

Yet, there are several cases where this heuristic breaks down. First, in the case where the lattice contains an unusually short vector (that is we are in presence of a uSVP_γ instance), this heuristic fails when the shortest vector is actually found. Interestingly, it seems that the same Hermite factor is still a meaningful value, according to the study of Gama and Nguyen [GN08], the unique short vector is expected to be found by a reduction algorithm whenever $\lambda_2(L)/\lambda_1(L) \geq \delta^n$; in other words, in practice LLL will solve uSVP_γ instances for $\gamma \geq \delta_{\mathrm{LLL}}^n$.

Another case where this heuristic partially fails is when the reduction algorithm is given some short vectors as part of the input basis; this will often be the case of the q-ary lattices used in cryptography. In this case, some of those short vectors are untouched and the heuristic only applies to the rest of the basis.

3.5.4 Babai's Algorithm, Finding Close Vectors

The previous algorithms let one finds a lattice basis of reasonable quality; we will now see how such good basis can be used to find close points; that is to solve CVP_γ and BDD_β. Those algorithms were introduced by Babai [Bab86]; they can be seen as algorithmic versions of Properties 3.2 and 3.12, stating that if $\mathbf{B}$ is a basis of $\mathcal{L}$ then $\mathcal{P}(\mathbf{B})$ and $\mathcal{P}(\mathbf{B}^\star)$ are fundamental domains of $\mathcal{L}$; those algorithms actually perform reduction modulo the lattice according to those fundamental domains.

For simplicity, we will only consider full-rank lattices, given by a basis $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_n] \in \mathbb{Z}^{n \times n}$. The first algorithm, called Babai's round-off, simply consists in writing the target $\mathbf{t}$ as an $\mathbb{R}$-linear combination of $\mathbf{B}$, and rounding independently each coordinate; exactly as in the proof of property 3.2.

Algorithm 2 Babai Round-off Algorithm [Bab86]

Input: A Basis $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_n]$ of a lattice L, a target $\mathbf{t}$
Output: A decomposition $\mathbf{t} = \mathbf{v} + \mathbf{w}$ where $\mathbf{v} \in L$ and $\mathbf{w} \in \mathcal{P}(\mathbf{B})$

1: $\mathbf{v} = \lfloor \mathbf{t} \cdot \mathbf{B}^{-1} \rceil \cdot \mathbf{B}$
2: $\mathbf{w} = \mathbf{t} - \mathbf{w}$
3: return $(\mathbf{v}, \mathbf{w})$

The second one starts by writing $\mathbf{t}$ in the orthogonal basis $\mathbf{B}^\star$. At the first loop step, it determines which plane among all the $x\mathbf{b}_n + \mathrm{Span}_{\mathbb{R}}(\mathbf{b}_1 \dots \mathbf{b}_{n-1})$ for $x \in \mathbb{Z}$, is the closest to $\mathbf{t}$; and it continues recursively as in the proof of Property 3.12. We give an equivalent iterative description.

Algorithm 3 Babai's Nearest Plane Algorithm [Bab86]

Input: A basis $\mathbf{B}$ of a full-rank lattice $L \subset \mathbb{R}^n$, a target point $\mathbf{t} : \mathbb{R}^n$.
Pre-computation : The Gram-Schmidt decomposition $\mathbf{B} = \boldsymbol{\mu} \cdot \mathbf{B}^\star$ and the inverse $\mathbf{B}^{\star -1}$ of $\mathbf{B}^\star$
Output: A decomposition $\mathbf{t} = (\mathbf{v} + \mathbf{w})$ where $\mathbf{v} \in L$ and $\mathbf{w} \in \mathcal{P}(\mathbf{B}^\star)$

$\mathbf{v}, \mathbf{z} \leftarrow \mathbf{0}$
$\mathbf{w} \leftarrow \mathbf{t} \cdot \mathbf{B}^{\star -1}$
for $i = n$ downto 1 **do**
  $z_i \leftarrow \lceil w_i \rfloor$
  $\mathbf{w} \leftarrow \mathbf{w} - z_i \cdot \boldsymbol{\mu}_i$
  $\mathbf{v} \leftarrow \mathbf{v} + z_i \cdot \mathbf{b}_i$
end for
return $(\mathbf{v}, \mathbf{t} - \mathbf{v})$

This second algorithm gives better results than the former, intuitively because the radius of $\mathcal{P}(\mathbf{B}^\star)$ is smaller than the one of $\mathcal{P}(\mathbf{B})$. Precisely, this second algorithm will properly solve BDD_β for any $\beta \leq \min \|\mathbf{b}_i^\star\| /2$, since the ball of radius β is included in $\mathcal{P}(\mathbf{B}^\star)$. One may also show that it solves Approximate CVP_γ for $\gamma = \sqrt{n} \cdot \frac{\max \|\mathbf{b}_i^\star\|}{\min \|\mathbf{b}_i^\star\|}$.

Chapter 4

Overview of Lattice Based Cryptography

My work is a game – a very serious game.
Maurits Cornelis Escher.

4.1 Analogies between Lattice and Discrete-log Cryptographic Constructions

4.1.1 Comparison of SIS and ISIS with DL

Definition 4.1 (Discrete Logarithm Problem, DL) *The Discrete Logarithm problem over a cyclic group $\mathbb{G}$ of prime order p, noted multiplicatively, and given a generator g is as follows : for a random $x \in \mathbb{Z}_p$ drawn uniformly, and given $h = g^x$ recover the exponent x.*

The hardness of the discrete logarithm problem can also be stated as the one-wayness of the function $x \mapsto g^x$. Note that this function is a permutation, which won't be the case for the lattices variants.

The direct analogue is the problem ISIS *for certain parameters* : for parameters n, m, q, β, on an instance matrix $\mathbf{A} \in \mathbb{Z}^{m \times n}$, the hardness of $\mathsf{ISIS}_{n,m,q,\beta}$ is equivalent to the one-wayness of the function $f : S \to \mathbb{G}$ (S the bounded subset of $\mathbb{Z}_q^m \cap \beta \cdot \mathfrak{B}_m$, and $\mathbb{G}$ the additive group $\mathbb{Z}_q^n$) defined by $\mathbf{x} \mapsto \mathbf{x} \cdot \mathbf{A}$; assuming that β is large enough so that the output of this function is almost uniform for a uniform input. But as we said, this one-way function is not a permutation : each image have many, perhaps exponentially many pre-images.

While SIS is simply ISIS with the fixed target $\mathbf{0}$, doesn't seem to be equivalent to a one-wayness property. Yet $\mathsf{SIS}_{n,m,q,\beta}$ still implies the one-wayness of $f : \mathbf{x} \in (\mathbb{Z}_q^m \cap \frac{\beta}{2} \cdot \mathfrak{B}_m) \mapsto \mathbf{x} \cdot \mathbf{A}$ (note the $\frac{\beta}{2}$) ; assuming the parameters guarantees that almost each image of f has at least 2 preimage. Indeed, if one were able to break the one-wayness, say on an instance $\mathbf{t} = f(\mathbf{x})$, finding a pre-image $\mathbf{x}'$; then with probability at least $1/2$ we have $\mathbf{x} \neq \mathbf{x}'$ and $\mathbf{x} - \mathbf{x}'$ would be a solution to the $\mathsf{SIS}_{n,m,q,\beta}$ instance $\mathbf{A}$.

4.1.2 Comparison of dLWE with dDH

Definition 4.2 (Decisonal Diffie-Hellman Problem, dDH) *The Decisional Diffie-Hellman problem over a cyclic group $\mathbb{G}$ of prime order p, noted multiplicatively, and given a generator g is as follows : for independent exponents $a, b, c \in \mathbb{Z}_p$ chosen uniformly distinguish between the distributions $\mathcal{D}_{\mathsf{dDH}}$: (g^a, g^b, g^{ab}) and $\mathcal{D}_{\$} : (g^a, g^b, g^c)$.*

In a geometric point of view, the two problems dLWE and dDH are similar in the following sense : dDH requires the challenger to decide if $x_1 = (g, g^a) \in \mathbb{G}^2$ and $x_2 = (g^b, g^c) \in \mathbb{G}^2$ are colinear or independent (and random) ; equivently decide if $\mathrm{Span}_{\mathbb{Z}_p}(\{x_1, x_2\})$ is a strict subspace of $\mathbb{G}^2$. Similarly, LWE requires an attacker to decide if samples $x_i = (\mathbf{a}_i, b_i) \in \mathbb{G} \times \mathbb{Z}_q$ (where $\mathbb{G} = \mathbb{Z}_q^n$) are purely random or if they are all close to a same hyperplane of $\mathbb{G} \times \mathbb{Z}_q$ of codimension 1, namely, $\mathcal{H} = \{(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle)\} : \mathbf{a} \in \mathbb{G}\}$. In other words, dDH is a specific hidden subgroup problem – that can not be harder than the discrete logarithm over this group – while LWE is a noisy variant, that can be hard even on very simple groups such as the additive $\mathbb{Z}_q^n$.

4.2 Lattices Schemes without trapdoors

4.2.1 Encryption from Original LWE

The LWE problem gives rise to a very simple encryption scheme. Precisely, Regev [Reg05] proposed the following : set $q \geq 2$ to be some prime in the range $[n^2, 2n^2]$ and set $m = 2n \log q$, and choose the error distribution $\chi = D_{\mathbb{Z},\sigma} \bmod q$ for $\sigma = o(q/(\sqrt{n}\log^2 n))$. It encrypts one bit μ.

KeyGen(1^n) : Choose $\mathbf{s}$, chosen uniformly in $\mathbb{Z}_q^n$. Draw m LWE samples : $(\mathbf{a}_i, b_i)$ where $\mathbf{a}_i$ are uniform and indepedent in $\mathbb{Z}_q^n$, and $b_i = \langle \mathbf{a}_i, \mathbf{s}\rangle + e_i$ where e_i are drawn independently from χ. Output key pair $\big(sk = \mathbf{s}, pk = \{(\mathbf{a}_i, b_i)\}\big)$

Enc$(pk = \{(\mathbf{a}_i, b_i)\}, \mu \in \{0,1\})$ Choose a random binary vector $\mathbf{t} \in \mathbb{Z}^m$, and output the cipher $c = (\mathbf{a}, b + \lfloor \frac{q}{2} \rfloor \cdot \mu)$ where $\mathbf{a} = \sum t_i \mathbf{a}_i$ and $b = \sum t_i b_i$

Dec$(sk = \mathbf{s}, c = (\mathbf{a}, b))$ Compute $\mu^* = b - \langle \mathbf{a}, \mathbf{s}\rangle$ and output 0 if $\mu^* \in [-q/4, q/4)$; output 1 otherwise.

Correctness (sketch) One easily checks that $\mu^* = \sum t_i e_i + \lfloor \frac{p}{2} \rfloor \cdot \mu$; intuitively the central limit theorem suggest that $\sum t_i e_i$ is very close to a Gaussian distribution of standard deviation $\sqrt{m} \cdot \sigma = o(q/\log n)$ (for a formal argument, one can rely on Hoeffding Bound) ; therefore $|\sum t_i e_i| < q/4$ with overwhelming probability.

CPA security (sketch). First, one would replace the public key using the dLWE hardness assumption, by truly uniform samples $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^{n+1}$. Then, the main ingredient is the leftover hash lemma 2.6 stating that the encryption process (a subset-sum of m random elements of $\mathbb{Z}_q^{n+1}$) produce a vector $(\mathbf{a}|b)$ that is almost uniform and independent from the $(\mathbf{a}_i|b_i)$, when those $(\mathbf{a}_i|b_i)$ comes from the uniform distribution. At this point, the cipher c becomes almost independent from the message μ.

Efficiency. The previous scheme has public key size of $(n+1)m \log_2 q = \tilde{\mathcal{O}}(n^2)$; yet assuming the parties share a random common reference string, it can be taken down to $m \log_2 q = \tilde{\mathcal{O}}(n)$; the private key has size $n \log_2 q = \tilde{\mathcal{O}}(n)$ and ciphertext size $m \log_2 q = \tilde{\mathcal{O}}(n)$ for 1-bit plaintext. Yet, using a variant based on ring-LWE, it is possible to decrease the ciphertext size to $\tilde{\mathcal{O}}(n)$ for a plaintext of n bits and decrease the ciphertext expension factor to polylog(n).

Comparison with El-Gamal Encryption. If this scheme is to be compared with a discrete-logarithm related scheme, the best match is probably the El-Gamal encryption scheme. We recall that El-Gamal security is guaranteed by the decisional-Diffie-Hellman (dDH) hardness assumption.

El-Gamal encryption proceeds as follows :

KeyGen(1^n) : Choose $s \in \mathbb{Z}_p$, chosen uniformly in $\mathbb{Z}_p$. Publish $pk = (g, h) = (g, g^s) \in \mathbb{G}$

Enc$(pk = (g,h), \mu \in \mathbb{G})$: Choose a random scalar $t \in \mathbb{Z}_p$ and publish the ciphertext $c = (g^t, h^t \cdot \mu)$

Dec$(sk = \mathbf{s}, c = (a, b))$: Compute $\mu^* = b/a^s$

To point out the similarities, we will give a geometrical interpretation of both schemes. Let us start with El-Gamal. The public key $pk = (g,h) \in \mathbb{G}^2$ can be seen as a line over the plane $\mathbb{G}^2$; more precisely, given this public key one can generate a uniform random point on this line $\mathcal{L} = \{(g^t, h^t) : t \in \mathbb{Z}_p\}$ for a random $t \in \mathbb{Z}_p$. The ciphertext space can been seen as a partition of p cosets of this line, and each cosets corresponds to one ciphertext : $C_\mu = \{(g^t, h^t) \odot (g^0, \mu) : t \in \mathbb{Z}_p\}$. Last, for decryption, one may see the secret key as a linear form over $\mathbb{G}^2$: $\hat{s} = (a,b) \mapsto b/a^s$, namely an element of the dual $\widehat{\mathbb{G}^2} = (\mathbb{G}^2 \to \mathbb{G})$, and this linear form is null over the "public key line" : $\hat{s}(g^t, h^t) = h^t/g^{st} = g^{st}/g^{st} = g^0$.

Now let us try to give a similar interpretation of Regev's Scheme. Let us first ignore the errors (assume $e = 0$), and pretend for now $m < n$. The public key $\{(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^{n+1}\}$ spans an m-dimensional hyperplane of $\mathbb{Z}_q^{n+1}$, and one can easily samples from that hyperplane by choosing some $(\mathbf{a}, b') = (\sum t_i \mathbf{a}_i, \sum t_i b_i)$. A message is encrypted as a coset of this hyperplane $(0, \lfloor \frac{q}{2} \rfloor \cdot \mu) + (\mathbf{a}, b')$. Last, one can associate a linear for $\hat{s}$ to the secret key : $\hat{s} : \mathbb{Z}^{n+1} \to \mathbb{Z}$ as $\hat{s}(\mathbf{a}, b) = b - \langle \mathbf{s}, \mathbf{a}\rangle$; and such linear form is zero over the public hyperplane.

Still, without the errors, it would be easy for an attacker to recover a correct $\hat{s}$ from the public key, using simple linear algebra, and decrypt messages. Therefore, the scheme in fact gives generators $(a_i, b_i) \in \mathbb{Z}^{n+1}$, that are not exactly cancelled by $\hat{s}$, but still rather small $\hat{s}(a_i, b_i) = e_i$. Therefore, small linear combinations of such generators, remains small under $\hat{s}$, ensuring correct decryption. Geometrically, the hyperplane is replaced by a set of small linear combinations of some generators. Note that one needs

$m > n$ for the security proof, to apply the leftover hash lemma 2.6 ; which is not necessary in El-Gamal because all distribution are trivially uniforms.

4.2.2 Key Exchange

Assuming the parties do have a common reference string parsed as a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, it is possible to re-interpret the previous scheme as a an approximate 1-round key exchange protocol.

Alice		**Bob**
Choose uniform random $\mathbf{t} \in \{0,1\}^m$		Choose uniform random $\mathbf{s} \in \mathbb{Z}_q^n$
Compute $\mathbf{a} = \mathbf{t} \cdot \mathbf{A}^t \in \mathbb{Z}^n$		Calcul $\mathbf{b} = \mathbf{s} \cdot \mathbf{A} + \mathbf{e} \in \mathbb{Z}^m$
	$\xrightarrow{\mathbf{a}}$	
	$\xleftarrow{\mathbf{b}}$	
Compute $k_A = \langle \mathbf{t}, \mathbf{b} \rangle$		Compute $k_B = \langle \mathbf{a}, \mathbf{s} \rangle$

At the end of this protocol, one notices that we have $k_A = k_B + \langle \mathbf{t}, \mathbf{e} \rangle$. Since $\mathbf{t}$ and $\mathbf{e}$ are small, we have $k_A \approx k_B$, yet for any third party that doesn't know anything about random values $\mathbf{t}, \mathbf{s}, \mathbf{e}$, k_A and k_B are computationnally indistinguishable from random by the dLWE assumption. In the encryption scheme, this is used as a one time mask to hide the message; yet for other constructions, one may want to have an exact key agreement ; maybe repeating this several time in parallel to obtain more common random bits (or using the ring-LWE variant). It is in fact possible to do so by only keeping the high bits of k_A and k_B, but to ensure that those high bits will match with overwhelming probability, one must choose parameters so that the relative error $\frac{k_A - k_B}{q} = \frac{\langle \mathbf{t}, \mathbf{e} \rangle}{q}$ is super-polynomially small.

While it is possible that this choice of parameter may still be asymptotically secure, the hardness results of [Reg05] does not hold anymore; and choosing secure parameters in practice would lead to a rather inneficient scheme.

4.2.3 Identification Scheme

The first Identification Scheme which can be proved secure under lattice hardness assumption was published by Lyubashevsky in [Lyu08]. On an the algebraic point of view, it is extremly similar to the Schnorr's identification protocol [Sch90], based on the hardness of the discrete logarithm problem. Yet, a direct analog of the discrete log in linear algebra is just to find an arbitary integer solution of a set of linear equation. To make this problem hard we have to require for short solution (i.e. ISIS). The issue is that Schnorr's scheme use uniform distribution over the group for the secret, the challenge and the proof of identity, and without uniformity information on the secret would leak. To solve this the article [Lyu08] introduced a rejection step in the procedure to avoid this leak. It is a three round protocol (Commit-Challenge-Response).

Prover		**Verifier**
Choose unif. secret key $\mathbf{s} \in \{0,1\}^m$		
Choose unif. $\mathbf{A} = [\mathbf{a}_1; \ldots; \mathbf{a}_m] \in \mathbb{Z}_q^{m \times n}$		
Compute $t = \mathbf{s} \cdot \mathbf{A} = \sum s_i \mathbf{a}_i \bmod q$		
Publish $(\mathbf{A}, \mathbf{t})$ as the public key	$\xrightarrow{\mathbf{A},\mathbf{t}}$	
	Commit	
Choose $\mathbf{y}_i$ unif. in $\{0, 1, \ldots 5m-1\}^m$		
Compute $\mathbf{v} = \mathbf{y} \cdot \mathbf{A} \bmod q$	$\xrightarrow{\mathbf{v}}$	
	Challenge	
	$\xleftarrow{c}$	Choose unif. challenge $c \in \{0,1\}$
	Response	
If $\mathbf{y} + \mathbf{s} \in \mathcal{S}$ or $c = 0$, set $\mathbf{z} = \mathbf{y} + \mathbf{s}$		
Else set $\mathbf{z} = \perp$	$\xrightarrow{\mathbf{z}}$	
		If $\mathbf{z} \neq \perp$ and $\|\mathbf{z}\| \leq \beta = 5m^{3/2}$ and $\mathbf{z} \cdot \mathbf{A} = \mathbf{y} + c \cdot \mathbf{t}$, Accept
		Otherwise, Reject

Security proof (sketch). The set $\mathcal{S}$ is $\{1, \dots 5m-1\}^m$ (note the absence of 0), therefore an honest prover will provide a valid response with probability $1/2 + 1/2(1-1/5m)^m > 3/4$; on the other hand an adversary is not expected to be able to answer with probability much larger than $1/2$. Indeed, knowing the challenge in advance allows him to build a valid commit-response pair (choose $\mathbf{y}$ and commit $\mathbf{v} = \mathbf{y} \cdot \mathbf{A} - c\mathbf{t}$). But now assume that once he commited to some $\mathbf{v}$, he is able to answer properly to both challenge $c = 0$ and $c = 1$ by $\mathbf{z}_0$ and $\mathbf{z}_1$. This implies $(\mathbf{z}_0 - \mathbf{z}_1) \cdot \mathbf{A} = \mathbf{t}$ and $\|\mathbf{z}_0 - \mathbf{z}_1\| \leq 2\beta$: the attacker has solve the ISIS instance $(\mathbf{A}, \mathbf{t})$. The formal proof is based on a rewinding argument to obtain both answers from the attacker, namely the forking lemma of Pointcheval and Stern [PS96].

It just remains to repeat several instances in parallel of this protocol and apply a majority vote to ensure that legitimate prover will almost always be accepted and forger will be rejected.

Rejection Step. Without the rejection step (the verification that $\mathbf{y}+\mathbf{s} \in \mathcal{S}$ when $c = 1$), the security of the scheme would break down : notice that the distribution of the response i-th coordinate $z_i = y_i + c \cdot s_i$ is uniform in $\{cs_i, cs_i + 1, \dots, cs_i + 5m - 1\}$; in particular if $z_i = 0$ this implies $cs_i = 0$ and symetrically $z_i = 5m$ implies $cs_i = 1$. But other values of z_i do not reveal anything, $s_i = 0$ and $s_i = 1$ are both equally likely. In this formal security proof, this allows the simulation algorithm to produce valid identification transcripts, with the proper distribution of the response $\mathbf{z}$ without actually knowing the secret $\mathbf{s}$.

Comparison with Schnorr's Identification The Identification from [Sch90], over a multiplicative cyclic group $\mathbb{G} = \langle g \rangle$ of order p is as follows :

Prover		**Verifier**
Choose unif. secret key $s \in \mathbb{Z}_p$		
Compute $t = g^s \in \mathbb{G}$		
Publish t as the public key	$\xrightarrow{t}$	
	Commit	
Choose y unif. in $\mathbb{Z}_p$		
Compute $v = g^y \in \mathbb{G}$	$\xrightarrow{v}$	
	Challenge	
	$\xleftarrow{c}$	Choose unif. challenge $c \in \mathbb{Z}_p \cap [0, 2^\ell]$
	Response	
Set $z = y - sc \in \mathbb{Z}_p$	$\xrightarrow{z}$	If $v = g^z \cdot t^c \in \mathbb{G}$
		Otherwise, Reject

The security argument (for passive attacks only) relies on the hardness discrete logarithm problem, which as we discuss sooner is also a one-wayness assumption. If an attacker is able to answer to two different challenges c, c' for the same commited value y, then one obtains z and z' that must verify $z - z' = s(c - c')$ where s is the discrete logarithm of t ; knowing c and c' one can recover s. Yet, one does not need to include a rejection step, because the distribution of z is uniform and independent of z because y is uniform over $\mathbb{G}$.

4.2.4 Digital Signatures

In a follow-up work [Lyu09], he showed that it was possible to apply the Fiat-Shamir Transform [FS87] to obtain a secure signature scheme. The idea of the Fiat-Schamir Transform is to replace the interaction with a verifier by a call to a random oracle. Indeed, the role of the verifier for the challenge step is just to choose an unpredictable value *after* the value $\mathbf{y}$ has been commited ; this interaction can be replaced by a request to the random oracle containing both the message to be signed, and the value to be commited $\mathbf{y}$. The scheme was later improved in [GLP12] for practical implementation, and in [Lyu12], showing that replacing the uniform distribution $\{0, 1, \dots 5m-1\}^m$ by a Gaussian Distribution allows asymptotical improvement, namely the verification threshold β can be decreased from $\tilde{\mathcal{O}}(m^{3/2})$ to $\tilde{\mathcal{O}}(m)$; and other parameter improvements by setting the public key as a LWE instance rather than a truly random instance ; alowing $m = 2n$ rather than $m \approx n \log q$.

The final scheme [Lyu12] is as follows ; given a common random matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, the public key is set to $\mathbf{T} = \mathbf{S} \cdot \mathbf{A} \in \mathbb{Z}^{k \times n}$ for a random secret key $\mathbf{S} \in \mathbb{Z}_q^{k,m}$ with small coefficients (say uniform in $\{-d \dots d\}$).

The secret vector $\mathbf{s}$ has been replaced by the matrix $\mathbf{S}$ so to allow the k-th repetition of the protocol to all run at once. The details on the rejection probability will be discussed later on chapter 8, for now, let us

Algorithm 4 Signature Algorithm of [Lyu12]

Input: Message μ, public key $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, secret key $\mathbf{S} \in \mathbb{Z}_q^{n \times m}$, standard deviation σ
Output: A signature $(\mathbf{z}, \mathbf{c})$ of the message μ

1: $\mathbf{y} \leftarrow D_{\mathbb{Z}^m,\sigma}$
2: $\mathbf{c} \leftarrow H(\mathbf{yA} \bmod q, \mu) \in \{0,1\}^k$
3: $\mathbf{z} \leftarrow \mathbf{y} - \mathbf{cS} \in \mathbb{Z}_q^m$
4: **Output**$(\mathbf{z}, \mathbf{c})$ with probability $\rho_\sigma(\mathbf{z}) \Big/ (M \cdot \rho_\sigma(\mathbf{z} + \mathbf{cS}))$ otherwise **restart**

Algorithm 5 Verification Algorithm of [Lyu12]

Input: Message μ, public Key $\mathbf{A} \in \mathbb{Z}_q^n$, signature $(\mathbf{z}, \mathbf{c})$
Output: Accept or Reject the signature

1: **if** $\|\mathbf{z}\| > B_2$ **then** Reject
2: Accept iff $\mathbf{c} = H(\mathbf{z} \cdot \mathbf{A} + \mathbf{c} \cdot \mathbf{T} \bmod q, \mu)$

just say that once again we want the signature vector $\mathbf{z}$ to be independent of the secret key $\mathbf{s}$. Once again, this scheme compares to a discrete-logarithm based scheme, namely Schnorr signatures, that are derived from the identification scheme (in the same article [Sch90]), using the Fiat-Shamir Transform [FS87].

4.3 Lattice Schemes with Trapdoor Basis

4.3.1 Short Basis as Trapdoors

As we've seen in the previous chapter (section 3.5.4), knowing a short basis of a lattice lets one solve problems that would be hard otherwise; by design Babai's algorithms can solve CVP_γ, but it is also possible to solve any instance $(\mathbf{A} \in \mathbb{Z}_q^{m \times n}, \mathbf{t} \in \mathbb{Z}_q^n)$ of $\mathsf{ISIS}_{n,m,q,\beta}$ by doing the following

- Choose an arbitrary solution $\mathbf{v} \in \mathbb{Z}_q^m$ of $\mathbf{v} \cdot \mathbf{A} = \mathbf{t}$
- Use one of Babai's algorithm with a short basis $\mathbf{B}$ of the SIS-lattice $\Lambda^\perp(\mathbf{A})$ on the target $\mathbf{v}$, to obtain a lattice point $\mathbf{v}'$ close to $\mathbf{v}$
- Answer $\mathbf{w} = \mathbf{v} - \mathbf{v}'$ as a short solution to $\mathbf{w} \cdot \mathbf{A} = \mathbf{t}$

For the simple rounding algorithm, the solution $\mathbf{w}$ is guaranteed to lie in $\mathcal{P}(\mathbf{B})$, ensuring a solution of SIS for $\beta = \mathrm{diam}(\mathcal{P}(\mathbf{B})) \leq \sum \|\mathbf{b}_i\|$. Similarly, for the nearest plane algorithm, we will have $\mathbf{w} \in \mathcal{P}(\mathbf{B}^\star)$, and therefore a SIS solution for $\beta = \sqrt{\sum \|\mathbf{b}_i^\star\|^2} \leq \sqrt{n} \cdot \|\mathbf{B}^\star\|$.

Before we go any further, we must mention that such a use of the trapdoor is in fact *insecure*, as proven by the attack of Nguyen and Regev [NR06], still this already gives an idea of how to use lattices for public-key cryptography. In brief, the attack from [NR06] is based on the fact that the distribution of the output $\mathbf{w}$ for a random target $\mathbf{t}$ is uniform in $\mathcal{P}(\mathbf{B})$ (or in $\mathcal{P}(\mathbf{B}^\star)$ for the nearest plane algorithm); in particular it is not independent of the secret key; for this reason, the author of NTRUSign [HNHGSW03] introduced a *heuristic* perturbation technique. Later, Gentry *et al.* [GPV08] showed that Gaussian Sampling provides a *provably secure* solution to this information leakage.

Partial Basis. An additional idea was introduced with various construction of HIBE [CHKP10], reused in [MP12], consisting of using a partial basis, that is a basis of a sub-lattice, of smaller dimension rather than a full basis. Precisely, in presence of a SIS-lattice $\Lambda^\perp(\mathbf{A})$ for $\mathbf{A}^t = [\mathbf{A}_1^t | \mathbf{A}_2^t]$, it is in fact enough to know a short basis of $\Lambda^\perp(\mathbf{A}_2)$ to solve the ISIS instances $(\mathbf{A}, \mathbf{t})$ (note that $\Lambda^\perp(\mathbf{A}_2)$ easily identifies to a sublattice of $\Lambda^\perp(\mathbf{A})$, namely $\Lambda^\perp(\mathbf{A}) \cap \{(\mathbf{0}|\mathbf{x}_2)\}$). Indeed, if $\mathbf{x}_2$ verifies $\mathbf{x}_2 \cdot \mathbf{A}_2 \equiv \mathbf{t} \bmod q$, then $\mathbf{x} = (\mathbf{0}|\mathbf{x}_2)$ verifies $\mathbf{x} \cdot \mathbf{A} \equiv \mathbf{t} \bmod q$. Note that we still need a basis of a sublattice $\Lambda^\perp(\mathbf{A}_2)$ of large enough dimension m_2; if $m_2 < n$, their won't be in general a solution $\mathbf{x}_2$, and even if $m_2 = n$, we will have $\Lambda^\perp(\mathbf{A}_2) = \mathbb{Z}_q^n$, which best basis is $q\mathbf{Id}_n$ and is not short enough for applications. Yet, this idea does simplifies the construction and the use of lattices with trapdoors.

4.3.2 Construction of Lattice with Trapdoors

Heuristic Trapdoor Construction from [GGH97] The first natural idea to build a lattice together with a short basis, is simply to start by choosing the basis $\mathbf{B}$ first, and output the lattice generated by $\mathbf{B}$, $\mathcal{L}(\mathbf{B})$. To reveal the lattice $L = \mathcal{L}(\mathbf{B})$ without revealing the short basis $\mathbf{B}$, one of the very first lattice

based construction, by Goldreich, Goldwasser et Halevi [GGH97] suggested to simply randomize the basis by a multiplication by a random unimodular matrix $\mathbf{T}$. Micciancio then suggested [Mic01] to use the Hermite Normal Form of the basis $\mathbf{B}$; which is unique for the lattice L and computable in polynomial time from any basis of L, it is in some sense the least powerful basis of L; and in particular independent of the secret key $\mathbf{B}$ knowing that $\mathcal{L}(\mathbf{B}) = L$.

Yet, this approach has several drawbacks; on the theoretical side, nothing guarantees that the class of lattice generated in such a way are indeed hard. On the practical side, those lattices are not q-ary; and this may hurt the efficiency of the cryptosystem since one should deal with potentially large integers. The asymptotical security of the encryption scheme based remains open, yet Nguyen was able to break most of the proposed parameters [Ngu99] thanks to a weakness in the form of the error of the BDD instance; the signature scheme was also fully broken [NR06], but because of the use of Babai algorithm that reveals the trapdoor basis, rather than because of a intrinsic weakness in the constructed lattices.

Heuristic trapdoor construction from NTRUSign The NTRU cryptosystems NTRUEncrypt and NTRUSign [HPS98,HNHGSW03,HHGPW10,Con03], also start from a set of short vectors to build a lattice. For efficiency purpose, the construction in fact works in the ring-setting : the matrices they consider are block-circulant, or equivalently the system is based on the ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n - 1)$ for a prime n. For a polynomial $a = \sum_{i=0}^{n-1} a_i X^i \in \mathcal{R}$, let $\mathcal{C}(a)$ denotes the matrix

$$\begin{bmatrix} a_0 & a_1 & \cdots & a_{n-1} \\ a_{n-1} & a_0 & \cdots & a_{n-2} \\ \vdots & \ddots & \ddots & \vdots \\ a_1 & \cdots & a_{n-1} & a_0 \end{bmatrix}$$

One first chooses two polynomials $f, g \in \mathcal{R}$, and define the lattice $L = \mathcal{L}^{\perp q}(\mathbf{A})$ where $\mathbf{A}^t = (\mathcal{C}(-g)^t | \mathcal{C}(f)^t)$, and reveals the Hermite Normal Form $\mathbf{A}' = (\mathcal{C}(-g/f)^t | \mathbf{Id}_n)^t$ of the lattice L. It is easy to see that the row vectors of $(\mathcal{C}(f)|\mathcal{C}(g))$ form a set of n short vectors, and this actually enough to build the encryption scheme NTRUEncrypt, that does not requires a full trapdoor basis. Still, to get a full basis and build the NTRUSign scheme, one finds a solution (F, G) to the equation (over $\mathcal{R} = \mathbb{Z}[X]/(X^n - 1)$) $fG' - gF' = q$ using a resultant operation. Then the vector (F', G') is reduced modulo (f, g) to obtain the rest of the basis (F, G). For their set of parameters, they obtain $\|F\| \approx \|G\| \approx \sqrt{\frac{n}{12}}\|f\|$; but this is very specific to their parameters, especially the choice of a very small modulus q. Indeed, because $\mathbf{B} = \begin{bmatrix} \mathcal{C}(f) & \mathcal{C}(g) \\ \mathcal{C}(F) & \mathcal{C}(G) \end{bmatrix}$ is a basis of L, we have $\det(\mathbf{B}) = \mathrm{Vol}(L) = q^n$, therefore $\|(f|g)\| \cdot \|(F|G)\| \geq q^n$. In particular, any basis $\mathbf{B}$ of that lattice must verify $\|\mathbf{B}\| \geq \|\mathbf{B}^\star\| \geq \sqrt{q}$.

An essential theoretical question is to know if a lattice generated that way is still hard. The lattice is a SIS lattice, more precisely a ring-SIS lattice; yet to get provably hard instance one must check that $-g/f$ follows almost a uniform distribution. Stehlé and Steinfeld recently proved [SS11] that it is the case if the size of the coefficient was larger than $q^{1/2+\epsilon}$, which is essentially optimal (otherwise $\mathcal{C}(-g/f)$ does not contains enough entropy to be uniform over $\mathcal{R}_q$). Nevertheless, in many cases one would be tempted to choose smaller parameters for efficiency reasons; and for reasonably smaller parameters, thoses lattices seems to remain hard.

Trapdoors for Provably Hard Lattices The first construction of a truly random SIS-lattice together with a short basis is due to Miklós Ajtai [Ajt99]; the construction starts with the initial remark that it is rather easy to build a random SIS-lattice together with one short vector. Indeed, let $\bar{\mathbf{A}} = [\mathbf{a}_1 \dots \mathbf{a}_m]$ be an $\mathbb{Z}_q^{m \times n}$ whose row vectors are the $\mathbf{a}_i$'s. Choose $\mathbf{b} = \sum t_i \mathbf{a}_i$ for small randoms independent coordinates t_i (typically uniform in $\{0, 1\}$), and consider the matrix $\mathbf{A} = [\mathbf{a}_1 \dots \mathbf{a}_m, \mathbf{b}]$. For $m > n \log q + 2 \log \frac{1}{2\epsilon}$, by the Leftover Hash Lemma 2.6, the matrix $\mathbf{A}$ is ϵ-uniform on $\mathbb{Z}_q^{m+1 \times n}$; and one knows a short vectors of $L = \mathcal{L}^\perp(\mathbf{A})$, namely $\mathbf{t} = (t_1 \dots t_m, -1)$. It is interesting to note that this vector is expected to be only a constant factor greater than the shortest one; indeed the expected value of $\lambda_1(L)$, according to the Gaussian Heuristic (Heuristic 3.1) is

$$\lambda_1(L) \approx \frac{\mathrm{Vol}(L)^{1/m}\sqrt{m}}{\sqrt{2\pi e}} = q^{n/m}\frac{\sqrt{m}}{\sqrt{2\pi e}} = \mathcal{O}(\sqrt{m}) \text{ and } \|\mathbf{t}\| = \mathcal{O}(\sqrt{m}).$$

Still, the original construction of Ajtai [Ajt99] for a full short basis, rather than a single short vector produce a basis of a lesser quality, namely, the vectors of this basis have size $\tilde{\mathcal{O}}(m^{5/2})$. Ten years later,

this construction was improved by Alwen and Peikert [AP09], shrinking the length of the basis vectors to its optimal, $\mathcal{O}(\sqrt{m})$.

A new construction of trapdoors was given by [MP12] that is much simpler and improves the constant factors of the previous one; it also comes with specialized algorithm to use the trapdoors that should be more efficient in practice than general algorithm. The idea is as follows; first we can generalize the remark of Ajtai presented ealier, and generate a matrix $\mathbf{A}^t = [\bar{\mathbf{A}}^t | -\bar{\mathbf{A}}^t\mathbf{R}^t] \in \mathbb{Z}^{n\times m_1+m_2}$ for a uniformly random $\bar{\mathbf{A}}^t \in \mathbb{Z}^{n\times m_1}$, and say, a random ternary matrix $\mathbf{R}^t \in \mathbb{Z}^{m_1\times m_2}$. If $m_1 > n\log q + 2\log\frac{m_2}{2\epsilon}$, then by the LHL (lemma 2.6), $\mathbf{A}$ is ϵ-uniform; moreover one knows m_2 many short vectors : the rows of $(\mathbf{R}|\mathbf{Id}_n)$. Yet, this partial basis doesn't have the proper form to be sufficient as a trapdoor (such as done in [CHKP10] and presented in the previous section 4.3.1). The construction of [MP12] sligthly modifies this idea as follows : first craft a very simple lattice $\mathcal{L}^{\perp q}(\mathbf{G}^t)$ together with a very good basis $\mathbf{B}$ and set $\mathbf{A}^t = [\bar{\mathbf{A}}^t|\mathbf{G}^t - \bar{\mathbf{A}}^t\mathbf{R}^t] = \mathbf{T}^t\mathbf{A}'^t$ for $\mathbf{A}'^t = [\bar{\mathbf{A}}^t|\mathbf{G}]$ and $\mathbf{T} = \begin{bmatrix} \mathbf{Id} & -\mathbf{R}^t \\ 0 & \mathbf{Id} \end{bmatrix}$. Using the transformation matrix $\mathbf{T}^{-1}$, one can transform SIS instance over $\mathcal{L}^{\perp q}(\mathbf{A})$ to a ISIS instance over $\mathcal{L}^{\perp q}(\mathbf{A}')$; solve it using the partial basis $[\mathbf{0}|\mathbf{B}]$, and multiply it by $\mathbf{T}$ to obtain a solution only $\|\mathbf{T}\|_s \approx \|\mathbf{R}\|_s = \mathcal{O}(\sqrt{m})$ times larger. In some sense this construction (that is provably secure) can be seen as being inspired by the heuristic (and insecure [SD82]) knapsack cryptosystem of Merkle and Hellman [MH78].

4.3.3 Using Lattice Trapdoors

Heuritic Countermeasure to Statistical Leak. As explained earlier, short basis do provide a trapdoor for the ISIS function, thanks to Babai's algorithms; but repeated usage of this algorithm will leak the short basis used as proved by the attack in [NR06]. Yet, NTRUSign [HNHGSW03] includes a perturbation technique to try to prevent this kind of attack, in about half of the parameters proposed for standarization (IEEE P1363.1 [IEE03]). The perturbation consist of adding a random vector to the target $\mathbf{t}$ of Babai algorithm; more precisely this pertubation is chosen deterministically by simply applying Babai algorithm one the target but with a different lattice. All details on the original attack of [NR06] an on this heuristic countermeasure will be detailed in Chapter 5; where we will generalize this attack to tackle this perturbation technique. We will also present an alternative countermeasure of Hu *et al.* [HWH08] and its weakness. In a nutshell, Chapter 5 dismisses heuristic approach to this issue, and justifies the choice for the theoretically sound approach – Gaussian Sampling, as decribed below – despite its practical inconvenients.

Provable Countermeasure : Gaussian Sampling. As we have discussed in section 3.2.2, Gaussian distribution are very good at hiding the geometrical structures of the cells of a lattice. Klein [Kle00] developped an algorithm to sample from such distribution, but his goal was more related to cryptanalysis than to secure construction. The work of Gentry, Peikert and Vaikuntanathan [GPV08] proved that, given a short basis $\mathbf{B}$ of a lattice L this algorithm was able to sample a distribution ι-close to $D_{L,\sigma,\mathbf{c}}$ for arbitrary center $\mathbf{c}$, and for σ as small as $\eta_\iota(\mathbb{Z})\cdot\|\mathbf{B}^\star\|$. In fact, this algorithm is a randomized variant of Babai's nearest plane algorithm; and it lets one find a vector of L close to $\mathbf{c}$, at a distance of about $\sqrt{n}\sigma$. The essential difference with the non randomized nearest plane algorithm is that the output distribution is (almost) independent of the Basis : for a random input $\mathbf{c} \in \mathbb{Z}^n$, the output $\mathbf{w}$ is such that $\mathbf{c}+\mathbf{w} \in L$ and it follows $D_{\mathbb{Z}^n,\sigma}$ that is independent of the basis, unlike the uniform distribution over $\mathcal{P}(\mathbf{B}^\star)$ produced by the nearest plane algorithm.

Alternative algorithms have been proposed later. Peikert [Pei10] proposed an efficiency/quality trade-off, by basing his Gaussian Sampler on the simple rounding algorithm rather than the nearest plane algorithm. Technically, this avoid dealing with Gram-Schmidt orthogonalization and floating-point operations, allows better parallelization, and preserves quasi-linear running time in the Ring-Setting. Yet the Gaussian Distribution obtain is not spherical anymore, but discrete analogue of property 3.16 allows one to correct it to spherical by adding a perturbation computed off-line. Still this offline computation has algorithmic cost equivalent to the first Gaussian Sampler [Kle00, GPV08]; in brief, most of the work is moved to an offline phase (that can be computed knowing the short basis $\mathbf{B}$ of L, but does not require the target point) at the cost of increasing the standard deviation from $\sigma = \eta_\iota(\mathbb{Z})\cdot\|\mathbf{B}^\star\|$ to $\sigma = \eta_\iota(\mathbb{Z})\cdot\|\mathbf{B}\|_s$.

Last, Micciancio and Peikert developped a specialized algorithm for their new trapdoor construction [MP12]; the main idea was presented sooner in section 4.3.1, combined with the perturbation technique of [Pei10]. This is believably the most efficient trapdoor for provably hard random lattices.

Yet, all those three algorithms requires floating-point operations either in the online phase or the offline phase; and running those algorithm with to low precision may hurt the security by leaking partial

information about the short basis. In Chapter 6 we will study precisely what are the precision requirement for those floating-point operations. Additionnally we will study several algorithmic improvements allowing asymptotical speedup of $\tilde{\Theta}(n)$ for all those algorithm; mainly by introducing and analyzing a so-called lazyness technique.

4.3.4 Provably Secure Signatures from Lattice Trapdoors

The first Hash-Then-Sign signature (which is the alternative signature paradigm to Fiat-Shamir [FS87] for signature in the ROM) provably based on lattice problem was build by [GPV08] using Gaussian Sampling. The first application is the following signature scheme based on the hardness of SIS.

Signer		**Verifier**
	Key Pair Set-up	
Choose a unif. matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ with a short basis $\mathbf{B}$ of $\mathcal{L}^{\perp q}(\mathbf{A})$		
Send the public key $pk = \mathbf{A}$	$\xrightarrow{\mathbf{A}}$	
	Sign	
Hash the message to $\mathbb{Z}_q^n : \mathbf{t} = H(\mu)$		
Sample a unif. random solution $\mathbf{v} \in \mathbb{Z}_q^m$ of $\mathbf{v} \cdot A = \mathbf{t}$		
Sample $\mathbf{w} \leftarrow D_{L,\sigma,\mathbf{v}}$ using $\mathbf{B}$		
Output the signature $\mathbf{s} = \mathbf{v} - \mathbf{w}$	$\xrightarrow{\mathbf{s}}$	
	Verification	
		Verifies that $\|\mathbf{s}\| \leq 2\sqrt{n}\sigma$ and that $\mathbf{s} \cdot \mathbf{A} = H(\mu)$

One easily check correctness by noting that $\mathbf{w} \in L$, that is, $\mathbf{w} \cdot \mathbf{A} = \mathbf{0}$, therefore $\mathbf{s} \cdot \mathbf{A} = \mathbf{t} = H(\mu)$; moreover the expected length of $\mathbf{s} = \mathbf{w} - \mathbf{v}$ is $\sigma \cdot \sqrt{n}$ (more formally, one may apply lemma 3.20).

Security proof (sketch) of SU-CMA security under hardness of SIS. The simulator is given a SIS instance $\mathbf{A}$, that is sent as the public key. Upon random oracle or signature queries on message m_i, the simulator choose $\mathbf{s} \leftarrow D_{\mathbb{Z}^m,\sigma,\mathbf{0}}$, and program the random oracle $H(\mu_i) = \mathbf{t}_i$ where $\mathbf{t}_i = \mathbf{s}_i \cdot \mathbf{A}$. If σ is large enough, this guarenties that this distribution of $(\mathbf{s}, H(\mu))$ is ϵ-close to the one sampled by the real signature protocol; using the smoothing lemma 3.17 and the LHL 2.6. Last, when the attacker provides a forgery for a message μ_j, it is a short vector $\mathbf{s}^*$ that verifies $\mathbf{s}^* \cdot \mathbf{A} = \mathbf{t}_j$; because their are many short solutions to this equation (formally because the conditional min-entropy of $\mathbf{s}_j \leftarrow D_{Z^m,\sigma,\mathbf{0}}$ knowing $\mathbf{t}_j$ is large), it is very likely that $\mathbf{s}^* \neq \mathbf{s}_j$. Therefore, $\mathbf{x} = \mathbf{s} - \mathbf{s}^*$ is a short solution (of length at most $4\sigma\sqrt{n}$) to $\mathbf{x} \cdot \mathbf{A} = \mathbf{0}$; that is, the simulation has a solution to the SIS instance.

Remark. One could also make a security proof based on ISIS, by guessing on which random oracle query is the attacker going to forge a message; yet this proof wouldn't be tight, loosing a factor N_{RO}, the number of RO queries. One can see a parallel with other Hash-Then-Sign signatures; in particular the Rabin signature scheme based on the factorization problem [Rab79b]. The fact that the trapdoored one way function has *several pre-image* for each image does avoid loosing a factor N_{RO}, because any forgery can be used as a solution to the underlying problem; on the other hand, schemes as the RSA signatures or Schnorr's signatures does not admit tight proofs because the trapdoored function is a permutation. Let us quickly describe Rabin's Scheme and its security proof.

Signer		**Verifier**
	Key Pair Set-up	
Choose two prime numbers p, q		
Send the public key $N = pq$	$\xrightarrow{N}$	
	Sign	
Hash the message to $QR(\mathbb{Z}_N) : t = H(\mu)$		
Choose s randomly among the 4 sol. of $s^2 \equiv t \bmod N$ using the CRT decomposition $\mathbb{Z}_n \simeq \mathbb{Z}_p \times \mathbb{Z}_q$	$\xrightarrow{s}$	
	Verification	
		Verifies that $s^2 = H(\mu)$

In the security proof, upon RO queries on m_i the simulator first chooses some s_i and program $H(\mu_i) = s_i^2$. Without the knowledge of s_i, a forger will output an s' that is neither s_j nor $-s_j$ with probability 1/2. We have $s^2 - s'^2 = H(\mu_j) - H(\mu_j) \equiv 0 \bmod N$ that can be rewritten as $(s_j + s')(s_j - s') \equiv 0 \bmod N$ and with probability 1/2 neither $s_j + s'$ and $s_j - s'$ is zero modulo N : the simulator has found a non-trivial factor of N. And the analogy goes further ; one could directly base the proof of Rabin's signature on the quadratic reduosity problem by embedding the challenge in the right RO query, but this would lead, as for the signature of [GPV08] based on ISIS, to the loss of a factor N_{RO} in the security proof.

4.3.5 Lattice Based IBE and Beyond

In addition to Hash-Then-Sign Signatures, the article [GPV08] also build a provably secure IBE, by combining the previous Signatures with the a dual version of LWE-encryption [Reg05] presented in section 4.2.1. The scheme is as follows

MasterKeyGen(1^λ) : Generate a uniform random matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ together with a short basis of $\Lambda^{\perp q}(A)$. Publish $\mathbf{A}$ as the master public key, and keep $\mathbf{B}$ as the master secret key.

KeyGen$(\mathbf{B}, I)$: Hash the identity I to obtain the identity's public key $\mathbf{u}_I = H(I)$. Extract the secret key $\mathbf{s}_I$ as a short solution to $\mathbf{s}_I$ to $\mathbf{s}_i \cdot A = \mathbf{u}_I$ (using Gaussian Sampling).

Enc$(\mathbf{A}, I, \mu \in \{0,1\})$: Hash the identity I to get obtain the identity's public key $\mathbf{u}_I = H(I)$. Choose a uniform random vector $\mathbf{v} \in \mathbb{Z}_q^n$, and an error vector $(\mathbf{x}, x) \leftarrow \chi^{m+1}$. Output the cipher $(\mathbf{p}, c = \langle \mathbf{u}_I, \mathbf{v} \rangle + x + \mu \cdot \lfloor \frac{q}{2} \rfloor)$ where $\mathbf{p} = \mathbf{v} \cdot \mathbf{A}^t + \mathbf{x}$.

Dec$(\mathbf{s}_I, (\mathbf{p}, c))$ Decrypt the message as $\mu' = c - \langle \mathbf{s}_I, \mathbf{p} \rangle$.

The correctness is easily established by noting that $\langle \mathbf{s}_I, \mathbf{p} \rangle = \mathbf{s}_I \cdot \mathbf{A} \cdot \mathbf{v}^t + \langle \mathbf{s}_I, \mathbf{x} \rangle = \langle \mathbf{u}_I, \mathbf{v} \rangle + \langle \mathbf{s}_I, \mathbf{x} \rangle$ which is close to $\langle \mathbf{u}_I, \mathbf{v} \rangle$ for appropriate distribution χ. The security proof combines the techniques we have seen so far in this section. We refer to the original construction for details [GPV08].

This system was then extended to a hierachical IBE [CHKP10], using the partial basis technique : the main key pair is a lattice $\mathcal{L}^{\perp q}(\mathbf{A})$ together with a short basis $\mathbf{B}$. Public key of the first level have form $\mathbf{A}_{I_1}^t = [\mathbf{A}^t | \mathbf{H}_{I_1}^t]$ where $\mathbf{H}_{I_1}^t = H_1(I_1)$ is a $\mathbb{Z}_q^{m \times n}$ random matrix; using the partial basis the main authority can extract short vectors $\mathbf{x}$ of $\mathcal{L}^{\perp q}(\mathbf{A}_{I_1})$ as $\mathbf{x} = (\mathbf{x}_0 | \mathbf{x}_1)$ by first choosing a small random $\mathbf{x}_1$ and then finding a small $\mathbf{x}_0$ such that $\mathbf{x}_0 \cdot \mathbf{A}^t = -\mathbf{x}_1 \cdot \mathbf{H}_{I_1}$. By extracting enough such short vectors, one can derive a short basis of $\mathcal{L}^{\perp q}(\mathbf{A}_{I_1})$; using Gaussian Sampling, one can ensure that this key derivation process is independent of the may secret key $\mathbf{B}$. The second level of identity's public key has the form $\mathbf{A}_{I_1.I_2}^t = [\mathbf{A}^t | \mathbf{H}_{I_1}^t | \mathbf{H}_{I_2}^t]$, and the secret key for $I_1.I_2$ can be derived the same way from the secret key of I_1. Many improvements have been proposed since this original proposal by Agrawal, Boneh and Boyen [ABB10a, ABB10b, Boy10, Boy13].

Note that in this HIBE system, each key extraction increase the length of the basis by a factor at least $O(\sqrt{m})$, thus limiting the depth of the system. In practice one would seek to optimize at least the hidden constant of the key extraction process, *i.e.* the quality of the Gaussian Sampling algorithm.

Chapter 5

Learning Attacks against NTRUSign Countermeasures

Learn all you can from the mistakes of others. You won't have time to make them all yourself.
Alfred Sheinwold – Bridge player

Résumé

Ce chapitre reprend de façon plus détaillée les résultats de l'article *Learning a Zonotope and More : Cryptanalysis of NTRUSign Countermeasures*, co-signé avec P. Nguyen et publié à Asiacrypt 2012.

Il y a un intérêt croissant pour la cryptographie basée sur les réseaux. D'un point de vue pratique, il n'y a cependant qu'un seul schéma de signature qui soit compétitif avec les schémas standards : NTRUSign, conçu en 2003. La version basique de NTRUSign à été cassée par Nguyen et Regev en 2006 : il est possible de retrouver la clé secrète à partir d'environ 400 signatures. Cependant, des contre-mesures ont été proposées pour réparer le schéma, telles que la méthode de perturbation utilisée dans la candidature à la standardisation de NTRUSign ainsi que la technique de déformation de Hu *et al.* dans IEEE Trans. Inform. Theory en 2008. Prétendument, ces deux contre-mesures étaient résistantes à l'attaque NR. Nous montrons dans ce chapître que, de façon surprenante, ces affirmations sont fausses en revisitant l'attaque NR par descente de gradient : cette attaque se révèle bien plus puissante qu'attendue, et peut casser en pratique ces deux contre-mesures. Plus précisément, nous expliquons pourquoi l'algorithme de Nguyen et Regev permettant l'apprentissage statistique d'un parallélépipède peux être heuristiquement modifiée pour apprendre des objets plus complexes, telle que des zonotopes et des parallélépipèdes déformés.

Concrètement, nous sommes en mesure de récupérer des clés privées NTRUSign en quelques heures, en utilisant 8000 signatures pour la version originale NTRUSign-251 du schéma tel que soumis à la standardisation IEEE P1363 en 2003, ou 5000 signatures pour les nouveaux paramètres proposés en 2010.

Abstract

This chapter is a detailed version of the article *Learning a Zonotope and More : Cryptanalysis of NTRUSign Countermeasures*, coauthored with P. Nguyen and published at Asiacrypt 2012.

There is growing interest in lattice cryptography, but from a practical point of view, only one lattice signature scheme is competitive with standard signatures : NTRUSign, designed in 2003. The basic version of NTRUSign was broken by Nguyen and Regev in 2006 : one can efficiently recover the secret key from about 400 signatures. However, countermeasures have been proposed to repair the scheme, such as the perturbation used in NTRUSign standardization proposals, and the deformation proposed by Hu *et al.* at IEEE Trans. Inform. Theory in 2008. These two countermeasures were claimed to prevent the NR attack. Surprisingly, we show that these two claims are incorrect by revisiting the NR gradient-descent attack : the attack is much more powerful than previously expected, and breaks both countermeasures in practice. More precisely, we explain why the Nguyen-Regev algorithm for learning a parallelepiped can heuristically be modify to learn more complex objects, such as zonotopes and deformed parallelepipeds. As a concrete application, we recover the NTRUSign secret key in a few hours, using 8,000 signatures for the original NTRUSign-251 scheme with one perturbation submitted to IEEE P1363 in 2003, or 5,000 signatures for the latest 80-bit-security parameter set proposed in 2010.

5.1 Introduction

Since the the field started with the seminal work of Ajtai [Ajt96] back in 1996, cryptography based on hard lattice problems has benefited from significant progress in the past few years. But from a practical point of view, very few lattice schemes can really compete with standardized schemes for now. This is especially true in the case of signature schemes, for which there is arguably only one realistic lattice alternative : NTRUSign [HNHGSW03], which is an optimized instantiation of the Goldreich-Goldwasser-Halevi (GGH) signature scheme [GGH97] using the compact lattices introduced in NTRU encryption [HPS98] and whose performances are comparable with ECDSA. By comparison, signatures have size beyond 10,000 bits (at 80-bit security level) for the most efficient provably-secure lattice signature scheme known, namely the recent scheme of Lyubashevsky [Lyu12].

However, NTRUSign has no provable-security guarantee. In fact, the GGH signature scheme and its simplest NTRUSign instantiation were broken at EUROCRYPT '06 by Nguyen and Regev [NR06], who presented a polynomial-time key-recovery attack using a polynomial number of signatures : in the case of NTRUSign, 400 signatures suffice in practice to disclose the secret key within a few hours. In the GGH design, a signature is a lattice point which is relatively close to the (hashed) message. Clearly, many lattice points could be valid signatures, but GGH selects one which is closely related to the secret key : each message–signature pair actually discloses a sample almost uniformly distributed in a secret high-dimensional parallelepiped. The NR attack works by learning such a parallelepiped : given a polynomial number of samples of the form $\sum_{i=1}^{n} x_i \mathbf{b}_i$ where the x_i's are picked uniformly at random from $[-1/2, 1/2]$ and the secret vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^n$ are linearly independent, the attack recovers the parallelepiped basis $(\mathbf{b}_1, \ldots, \mathbf{b}_n)$, by finding minima of a certain multivariate function, thanks to a well-chosen gradient descent. The NR attack motivated the search of countermeasures to repair NTRUSign :

- The very first countermeasure already appeared in half of the parameter choices of NTRU's IEEE P1363.1 standardization proposal [IEE03], the other half being broken by NR. It consists of applying the signature generation process twice, using two different NTRU lattices, one of which being kept secret : here, the secret parallelepiped becomes the Minkowski sum of two secret parallelepipeds, which is a special case of zonotopes. This slows down signature generation, and forces to increase parameters because the signature obtained is less close to the message than previously. However, no provable security guarantee was known or even expected. In fact, heuristic (theoretical) attacks have been claimed by both the designers of NTRUSign [HNHGSW03] and more recently by Malkin *et al.* [MPSW], but both are completely impractical : the most optimistic estimates [HGP+, MPSW] state that they both require at least 2^{60} signatures, and naturally, none of these attacks have been fully implemented. Yet, as a safety precaution, the designers of NTRUSign [HGP+] only claim the security of NTRUSign with perturbation up to 1 million signatures in [HGP+]. Still, breaking this countermeasure was left as an open problem in [NR06].
- In 2008, Hu, Wang and He [HWH08] proposed a simpler and faster countermeasure in IEEE Trans. Inform. Theory, which we call IEEE-IT, where the secret parallelepiped is deformed. Again, the actual security was unknown.
- Gentry, Peikert and Vaikuntanathan [GPV08] proposed the first provably secure countermeasure for GGH signatures, by using a randomized variant [Kle00] of Babai's nearest plane algorithm. However, this slows down signature generation significantly, and forces to increase parameters because the signatures obtained are much less close to the message than previously. As a result, the resulting signature for NTRUSign does not seem competitive with classical signatures : no concrete parameter choice has been proposed.

Our Results. We revisit the Nguyen-Regev gradient-descent attack to show that it is much more powerful than previously expected : in particular, an optimized NR attack can surprisingly break in practice both NTRU's perturbation technique [HGP+] as recommended in standardization proposals [IEE03, HHGPW10], and the IEEE-IT countermeasure [HWH08]. For instance, we can recover the NTRUSign secret key in a few hours, using 8,000 signatures for the original NTRUSign-251 scheme with one perturbation submitted to IEEE P1363 standardization in 2003, or only 5,000 signatures for the latest 80-bit-security parameter set [HHGPW10] proposed in 2010. These are the first successful experiments fully breaking NTRUSign with countermeasures, and it seems to even work with a constant number of perturbations. We also develop a more general attack than NR to attack natural generalizations of the IEEE-IT countermeasure [HWH08]. The warning is clear : our work strongly suggests to dismiss all GGH/NTRUSign countermeasures which are not supported by some provable security guarantee.

Our work sheds new light on the NR attack. The original analysis of Nguyen and Regev does not

apply to any of the two NTRUSign countermeasures, and it seemed *a priori* that the NR attack would not work in these cases. We show that the NR attack is much more robust than anticipated, by extending the original analysis of the Nguyen-Regev algorithm for learning a parallelepiped, to tackle more general objects such as zonotopes (to break the NTRUSign perturbation countermeasure) or deformed parallelepipeds (to break the IEEE-IT countermeasure). For instance, in the zonotope case, the parallelepiped distribution $\sum_{i=1}^{n} x_i \mathbf{b}_i$ is replaced by $\sum_{i=1}^{m} x_i \mathbf{v}_i$ where $\mathbf{v}_1, \ldots, \mathbf{v}_m \in \mathbb{R}^n$ are secret vectors with $m \geq n$. The key point of the NR attack is that all the local minima of a certain multivariate function are connected to the directions $\mathbf{b}_i$'s of the secret parallelepiped. We show that there is somewhat a similar (albeit more complex) phenomenon when the parallelepiped is replaced by zonotopes or deformed parallelepipeds : there, we establish the existence of local minima connected to the secret vectors spanning the object, but we cannot rule out the existence of other minima. Yet, the attack works very well in practice, as if there were no other minima.

5.2 Background and Notation

We recall the definitions of Zonotopes and Parallelepipeds, and the zonotopic distribution.

Definition 5.1 (Zonotopes and Parallelepipeds) *By zonotope, one usually means the Minkowski sum of finitely many segments of the form* $[0,1]\mathbf{v}$. *Here, we use a slightly different definition by replacing* $[0,1]$ *with* $[-1,1]$ *: for an arbitrary* $m \times n$ *row matrix* $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_m]$, *the* zonotope *spanned by* $\mathbf{V}$ *is the set* $\mathcal{Z}(\mathbf{V}) = \{\sum_{i=1}^{m} x_i \mathbf{v}_i, -1 \leq x_i \leq 1\}$.

The zonotopic distribution denoted by $\mathcal{D}_{\mathcal{Z}(\mathbf{V})}$ *is the convolution distribution over* $\mathcal{Z}(\mathbf{V})$ *obtained by picking independently each* x_i *uniformly at random from* $[-1,1]^n$ *: in other words,* $\mathcal{D}_{\mathcal{Z}(\mathbf{V})} = \mathcal{U}([-1,1]^m) \cdot \mathbf{V}$, *which in general is not the uniform distribution over* $\mathcal{Z}(\mathbf{V})$. *When the context is clear, we may note* $\mathcal{Z}(\mathbf{V})$ *for the zonotopic distribution.*

However, in the particular case $\mathbf{V} \in \mathcal{GL}_n(\mathbb{R})$, $\mathcal{Z}(\mathbf{V})$ *is simply the* parallelepiped $\mathcal{P}(\mathbf{V})$ *spanned by* $\mathbf{V}$, *and* $\mathcal{D}_{\mathcal{P}(\mathbf{V})}$ *is equal to the uniform distribution over* $\mathcal{P}(\mathbf{V})$.

5.2.1 The GGH Signature Scheme

The GGH scheme [GGH97] works with a lattice L in $\mathbb{Z}^n$. The secret key is a basis $\mathbf{B} \in \mathbb{Z}^{n \times n}$, with very short row vectors (their entries are polynomial in n). Following [Mic01], the public key is the Hermite normal form (HNF) of L. The messages are hashed onto a "large enough" subset of $\mathbb{Z}^n$, for instance a large hypercube. Let $\mathbf{m} \in \mathbb{Z}^n$ be the hash of the message to be signed. The signer applies Babai's round-off CVP approximation algorithm (Alg. 2 from [Bab86]) to get a lattice vector close to $\mathbf{m}$:

$$\mathbf{s} = \lfloor \mathbf{m}\mathbf{B}^{-1} \rceil \mathbf{B}, \tag{5.1}$$

so that $\mathbf{s} - \mathbf{m} \in \mathcal{P}(\mathbf{B})$. To verify the signature $\mathbf{s}$ of $\mathbf{m}$, one would first check that $\mathbf{s} \in L$ using the public basis B, and compute the distance $\|\mathbf{s} - \mathbf{m}\|$ to check that it is sufficiently small.

5.2.2 NTRUSign

Basic scheme. NTRUSign [HGP+] is an instantiation of GGH using the compact lattices from NTRU encryption [HPS98], which we briefly recall : we refer to [HGP+, Con03] for more details. In the former NTRU standards [Con03] proposed to IEEE P1363.1 [IEE03], $N = 251$ and $q = 128$. Let $\mathcal{R}$ be the ring $\mathbb{Z}[X]/(X^N - 1)$ whose multiplication is denoted by $*$. One computes a quadruplet $(f, g, F, G) \in \mathcal{R}^4$ such that $f * G - g * F = q$ in $\mathcal{R}$ and f is invertible mod q, where f and g have 0–1 coefficients (with a prescribed number of 1), while F and G have slightly larger coefficients, yet much smaller than q. This quadruplet is the NTRU secret key. Then the secret basis is the following $(2N) \times (2N)$ block-wise circulant matrix :

$$\mathbf{B} = \begin{bmatrix} \mathcal{C}(f) & \mathcal{C}(g) \\ \mathcal{C}(F) & \mathcal{C}(G) \end{bmatrix} \text{ where } \mathcal{C}(a) \text{ denotes } \begin{bmatrix} a_0 & a_1 & \cdots & a_{N-1} \\ a_{N-1} & a_0 & \cdots & a_{N-2} \\ \vdots & \ddots & \ddots & \vdots \\ a_1 & \cdots & a_{N-1} & a_0 \end{bmatrix},$$

and f_i denotes the coefficient of X^i of the polynomial f. Thus, the lattice dimension is $n = 2N$. Due to the special structure of $\mathbf{B}$, a single row of $\mathbf{B}$ is sufficient to recover the whole secret key. Because f is chosen invertible mod q, the polynomial $h = g/f \mod q$ is well-defined in $\mathcal{R}$: this is the NTRU public

key. Its fundamental property is that $f * h \equiv g \mod q$ in $\mathcal{R}$. The polynomial h defines the following (natural) public basis of the lattice :

$$\begin{bmatrix} I_n & \mathcal{C}(h) \\ 0 & qI_n \end{bmatrix},$$

which implies that the lattice volume is q^N.

The messages are assumed to be hashed in $\{0,\ldots,q-1\}^{2N}$. Let $\mathbf{m}$ be such a hash. We write $\mathbf{m} = (\mathbf{m}_1, \mathbf{m}_2)$ with $\mathbf{m}_i \in \{0,\ldots,q-1\}^N$. It is shown in [HGP+] that the vector $(\mathbf{s},\mathbf{t}) \in \mathbb{Z}^{2N}$ which we would obtain by applying Babai's round-off CVP approximation algorithm to $\mathbf{m}$ using the secret basis R can be alternatively computed using convolution products involving $\mathbf{m}_1$, $\mathbf{m}_2$ and the NTRU secret key (f,g,F,G). In practice, the signature is simply $\mathbf{s}$ and not $(\mathbf{s},\mathbf{t})$, as $\mathbf{t}$ can be recovered from $\mathbf{s}$ thanks to $\mathbf{h}$. We described the basic NTRUSign scheme [HGP+], as used in half of the parameter choices of the former NTRU standards [Con03].

Perturbations. The second half of parameter choices of NTRU standards [Con03] use perturbation techniques [HGP+, Con03, HHGP+05] to strengthen the security of NTRUSign. Those techniques are described later in Section 5.2.4. But there is a second change : instead of the standard NTRU secret key, one uses the so-called *transpose basis*, which is simply R^t, then the public basis remains the same, except that one defines the public key as $h = F/f = G/g \mod q$ rather than $h = g/f \mod q$.

New parameters. In the latest NTRU article [HHGPW10], new parameters for NTRUSign have been proposed. These include different values of (N,q) and a different shape for f and g : the coefficients of f and g are now in $\{0,\pm 1\}$, rather than $\{0,1\}$ like in [HGP+]. But the scheme itself has not changed. To distinguish both versions, we call NTRUSign-2003 the version of [HGP+,Con03], and NTRUSign-2010 the latest version [HHGPW10].

5.2.3 The Nguyen-Regev Attack

We briefly recall the Nguyen-Regev attack [NR06], using a slightly different presentation. The NR attack solves the following idealized problem :

Problem 5.1 (The Hidden Parallelepiped Problem or HPP) *Let* $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_n] \in \mathcal{GL}_n(\mathbb{R})$ *and let* $\mathcal{P}(\mathbf{V}) = \{\sum_{i=1}^n x_i\mathbf{v}_i : x_i \in [-1/2, 1/2]\}$ *be the parallelepiped spanned by* $\mathbf{V}$. *The input to the HPP is a sequence of* poly(n) *independent samples from the uniform distribution* $\mathcal{D}_{\mathcal{P}(\mathbf{V})}$. *The goal is to find a good approximation of the rows of* $\pm\mathbf{V}$.

In practice, instead of samples from $\mathcal{D}_{\mathcal{P}(\mathbf{V})}$, the attack uses $(\mathbf{s}-\mathbf{m})$ for all given message-signature pairs $(\mathbf{m},\mathbf{s})$: this distribution is heuristically close to $\mathcal{D}_{\mathcal{P}(\mathbf{V})}$ where $\mathbf{V}$ is the secret basis. To recover rows of $\mathbf{V}$, the attack simply rounds the approximations found to integer vectors. The NR attack has two stages : morphing and minimization.

Morphing the Parallelepiped into a Hypercube. The first stage of the NR attack is to transform the hidden parallelepiped into a hidden hypercube (see Alg. 6), using a suitable linear transformation L. It is based on the following elementary lemma [NR06, Lemmas 1 and 2] :

Lemma 5.1 *Let* $\mathbf{V} \in \mathcal{GL}_n(\mathbb{R})$ *and denote by* $\mathbf{G} \in \mathcal{GL}_n(\mathbb{R})$ *the symmetric positive definite matrix* $\mathbf{V}^t\mathbf{V}$. *Then :*

- $\mathrm{Cov}(\mathcal{D}_{\mathcal{P}(\mathbf{V})}) = \mathbf{G}/12$.
- *If* $\mathbf{L} \in \mathcal{GL}_n(\mathbb{R})$ *satisfies* $\mathbf{L}\mathbf{L}^t = \mathbf{G}^{-1}$ *and we let* $\mathbf{C} = \mathbf{V}\mathbf{L}$, *then* $C \in \mathcal{O}_n(\mathbb{R})$ *and* $\mathcal{D}_{\mathcal{P}(\mathbf{V})} \cdot L = \mathcal{D}_{\mathcal{P}(C)}$.

Algorithm 6 Isotropize$(\mathcal{X})$: Morphing a Parallelepiped into a Hybercube

Input: A set $\mathcal{X}$ of vectors $\mathbf{x} \in \mathbb{R}^n$ sampled from the uniform distribution $\mathcal{D}_{\mathcal{P}(\mathbf{V})}$ over a parallelepiped.
Output: A matrix $\mathbf{L}$ such that $\mathcal{D}_{\mathcal{P}(\mathbf{V})} \cdot L$ is close to $\mathcal{D}_{\mathcal{P}(C)}$ for some $C \in \mathcal{O}_n(R)$.

1: Compute an approximation G of $\mathbf{V}^t\mathbf{V}$ using the set $\mathcal{X}$, using $\mathrm{Cov}(\mathcal{D}_{\mathcal{P}(\mathbf{V})}) = \mathbf{V}^t\mathbf{V}/3$ (see Lemma 5.1).

2: Return $\mathbf{L}$ such that $\mathbf{L}\mathbf{L}^t = \mathbf{G}^{-1}$

This stage is exactly (up to some scaling) the classical preprocessing used in independent component analysis to transform the covariance matrix into the identity matrix :

Lemma 5.2 *Let* $\mathbf{G}$ *be the covariance matrix of a distribution* $\mathcal{D}$ *over* $\mathbb{R}^n$. *If* $\mathbf{L} \in \mathcal{GL}_n(\mathbb{R})$ *satisfies* $\mathbf{L}\mathbf{L}^t = \mathbf{G}^{-1}$, *then* $\mathrm{Cov}(\mathcal{D}\cdot\mathbf{L}) = \mathbf{Id}_n$.

Learning a Hypercube. The second stage of the NR attack is to solve the hidden hypercube problem, using minimization with a gradient descent (see Alg. 7). Nguyen and Regev [NR06] showed that for any $\mathbf{V} \in \mathcal{O}_n(\mathbb{R})$, if $\mathcal{D}$ denotes the distribution $\mathcal{D}_{\mathcal{P}(\mathbf{V})}$:

- There are exactly $2n$ local minima for the function $\text{mom}_{\mathcal{D},4}(\mathbf{w}) = \mathbb{E}_{\mathbf{x}\leftarrow\mathcal{D}}[\langle \mathbf{x}, \mathbf{w}\rangle^4]$ over the unit sphere $\mathbb{S}_n$: they are located at $\pm\mathbf{v}_1, \cdots, \pm\mathbf{v}_n$, and they are global minima.
- It is possible to find all minima of $\text{mom}_{\mathcal{D},4}(\cdot)$ over $\mathbb{S}_n$ in random polynomial time, using Alg. 7 with parameter $\delta = 3/4$, thanks to the nice shape of $\text{mom}_{\mathcal{D},4}(\cdot)$. Alg. 7 is denoted by $\textbf{Descent}(\mathcal{X}, \mathbf{w}, \delta)$ which, given a point $\mathbf{w} \in \mathbb{S}_n$, performs a suitable gradient descent using the sample set $\mathcal{X}$, and returns an approximation of some $\pm\mathbf{v}_i$.

Algorithm 7 Descent$(\mathcal{X}, \mathbf{w}, \delta)$: Solving the Hidden Hypercube Problem by Gradient Descent

Input: A set $\mathcal{X}$ of samples from the distribution $\mathcal{D}_{\mathcal{P}(\mathbf{V})}$ where $\mathbf{V} \in \mathcal{O}_n(\mathbb{R})$, a vector $\mathbf{w}$ chosen uniformly at random from $\mathbb{S}_n$ and a descent parameter δ.
Output: An approximation of some row of $\pm\mathbf{V}$.

1: Compute an approximation $\mathbf{g}$ of the gradient $\nabla \text{mom}_{\mathbf{V},4}(\mathbf{w})$ using $\mathcal{X}$.
2: Let $\mathbf{w}_{new} = \mathbf{w} - \delta\mathbf{g}$.
3: Divide $\mathbf{w}_{new}$ by its Euclidean norm $\|\mathbf{w}_{new}\|$.
4: **if** $\text{mom}_{\mathbf{V},4}(\mathbf{w}_{new}) \geq \text{mom}_{\mathbf{V},4}(\mathbf{w})$ where the moments are approximated using $\mathcal{X}$ **then**
5: **return** the vector $\mathbf{w}$.
6: **else**
7: Replace $\mathbf{w}$ by $\mathbf{w}_{new}$ and go back to Step 1.
8: **end if**

The whole NR attack is summarized by Alg. 8.

Algorithm 8 SolveHPP$(\mathcal{X})$: Learning a Parallelepiped [NR06]

Input: A set $\mathcal{X}$ of vectors $\mathbf{x} \in \mathbb{R}^n$ sampled from $\mathcal{D}_{\mathcal{P}(\mathbf{V})}$, where $\mathbf{V} \in \mathcal{GL}_n(\mathbb{R})$
Output: An approximation of a random row vector of $\pm\mathbf{V}$

1: $L \leftarrow \textbf{Isotropize}(\mathcal{X})$ using Alg. 6
2: $\mathcal{Y} \leftarrow \mathcal{X} \cdot L$
3: Pick $\mathbf{w}$ uniformly at random from $\mathbb{S}_n$
4: Compute $\mathbf{r} \leftarrow \textbf{Descent}(\mathcal{Y}, \mathbf{w}, \delta) \in \mathbb{S}_n$ using Alg. 7 : use $\delta = 3/4$ in theory and $\delta = 0.7$ in practice.
5: Return $\mathbf{r}\mathbf{L}^{-1}$

Shrinking the number of NTRUSign-signatures. In practice, the NR attack requires a polynomial number of signatures, but it is possible to decrease this amount by a linear factor experimentally [NR06], using the following symmetry of NTRU lattices. We define the NTRUSign symmetry group, denoted $\mathfrak{S}_N^{\text{NTRU}}$, as the group spanned by the application $\sigma \in \mathcal{O}_n(\mathbb{R}) : (x_1, \ldots, x_N | y_1, \cdots, y_N) \mapsto (x_2, \ldots x_N, x_1 | y_2, \cdots y_N, y_1)$. If L is the NTRU lattice, then $\sigma(L) = L$. Additionnaly $(\sigma(\mathbf{m}), \sigma(\mathbf{s}))$ follows the same distribution as $(\mathbf{m}, \mathbf{s})$ for a random message $\mathbf{m}$ and signature $\mathbf{s}$. So, any pair $(\mathbf{m}, \mathbf{s})$ gives rise to N parallelepiped samples. This technique also allows a N-factor speedup for covariance computation, which is the most time consuming part of the attack.

5.2.4 Countermeasures

NTRUSign perturbation : Summing Parallelepipeds. Roughly speaking, these techniques perturbates the hashed message $\mathbf{m}$ before signing it with the NTRU secret basis. More precisely, the hashed message $\mathbf{m}$ is first signed using a second NTRU secret basis (of another NTRU lattice, which is kept secret), and the resulting signature is then signed as before. Heuristically, the effect on the sample distribution of the transcript is as follows : if R and R' are the two secret bases, the distribution of $\mathbf{s} - \mathbf{m}$ becomes the convolution $\mathcal{P}(R) \oplus \mathcal{P}(R')$, *i.e.* a natural distribution over the Minkowski sum of the two parallelepipeds obtained by adding the uniform distributions of both parallelepipeds.

IEEE-IT perturbation : Parallelepiped Deformation. Hu *et al.* [HWH08] suggested another approach to secure NTRUSign in the journal IEEE Trans. IT. Their definition are specific to NTRUSign-bases, but it can be generalized to GGH, and we call this technique "Parallelepiped deformation".

Let $\delta : [-1/2, 1/2)^n \to \mathbb{Z}^n$ be a function, possibly secret-key dependent. The signature generation (5.1) is replaced by :

$$\mathbf{s} = \Big(\lceil \mathbf{m}\mathbf{B}^{-1} \rfloor + \delta\big(\mathfrak{F}\left(\mathbf{m}\mathbf{B}^{-1}\right)\big) \Big)\mathbf{B} \tag{5.2}$$

If δ outputs small integer vectors, then the signature $\mathbf{s}$ is still valid. The associated deformation function is $d_\delta(\mathbf{x}) = \mathbf{x} + \delta(\mathbf{x})$. The sample distribution of $\mathbf{s} - \mathbf{m}$ is deformed in the following way : $d_\delta(\mathcal{U}^n) \cdot \mathbf{B}$ where $d_\delta(\mathcal{U}^n)$ denotes the distribution of $\mathbf{x} + \delta(\mathbf{x})$ with $\mathbf{x} \leftarrow \mathcal{U}^n$.

In [HWH08], the deformation δ_{IEEE} for a NTRUSign secret key (f, g, F, G) is as follows :

- Let $U \subset [N]$ be the set of indexes u such that the u-th entry of $f + g + F + G$ is 1 modulo 2, and let $A = \#U$. On the average, $A \approx N/2$, and it is assumed that $A \geq 25$, otherwise a new secret key must be generated.
- Let $1 \leq u_1 < u_2 < \cdots < u_A \leq N$ be the elements of U. For $i \notin [A]$, u_i denotes $u_{(i \bmod A)}$.
- Let the input of δ_{IEEE} be the concatenation of two vectors $\mathbf{x}, \mathbf{y} \in [-1/2, 1/2)^N$. Then the i-th entry of $\delta_{\text{IEEE}}(\mathbf{x}|\mathbf{y})$ is :

$$\left[\delta_{\text{IEEE}}(\mathbf{x}|\mathbf{y})\right]_i = \begin{cases} 0 & \text{if } i \notin U \\ s(x_{u_j}, y_{u_j}, y_{u_{j+1}}, y_{u_{j+3}}, y_{u_{j+7}}, y_{u_{j+12}}) & \text{if } i = u_j \end{cases}$$

$$\text{where } s(a_0, \ldots, a_5) = \begin{cases} 1 & \text{if } a_i < 0 \text{ for all } i \\ -1 & \text{if } a_i > 0 \text{ for all } i \\ 0 & \text{otherwise} \end{cases}$$

Gaussian Sampling. Gentry *et al.* [GPV08] described the first provably secure countermeasure : Gaussian sampling. In previous schemes, the distribution of $\mathbf{s} - \mathbf{m}$ was related to the secret key. In [GPV08], the distribution becomes independent of the secret key : it is some discrete Gaussian distribution, which gives rise a to a security proof in the random-oracle model, under the assumption that finding close vectors is hard in the NTRU lattice. Unfortunately, this countermeasure is not very competitive in practice : the sampling algorithm [Kle00] is much less efficient than NTRUSign generation, and the new signature is less close to the message, which forces to increase parameters.

5.3 Learning a Zonotope : Breaking NTRUSign with Perturbations

This section is organized as follows. In Sect. 5.3.1, we introduce the hidden zonotope problem (HZP), which is a natural generalization of the hidden parallelepiped problem (HPP), required to break NTRUSign with perturbations. In Sect. 5.3.2, we explain why the Nguyen-Regev HPP algorithm (Alg. 8) can heuristically solve the HZP, in cases that include NTRUSign, provided that Step 5 is slightly modified. Yet, the approximations obtained by the algorithm are expected to be worse than in the non-perturbed case, so we present in Section 5.3.3 a folklore meet-in-the-middle algorithm for BDD in NTRU lattices. Finally, in Sect. 5.3.4, we present experimental results with our optimized NR attack which show that NTRUSign with one (or slightly more) perturbation(s) is completely insecure, independently of the type of basis. In particular, we completely break the original NTRUSign proposed to IEEE P1363 standardization [Con03] : only one half of the parameter sets was previously broken in [NR06].

5.3.1 The Hidden Zonotope Problem

Assume that one applies $k - 1$ NTRUSign perturbations as a countermeasure, which corresponds to k NTRUSign lattices $L_1, \ldots, L_k$ (with secret bases $\mathbf{B}_1, \ldots, \mathbf{B}_k$) where only L_k is public. One signs a hashed message $\mathbf{m} \in \mathbb{Z}^n$ by computing $\mathbf{s}_1 \in L_1$ such that $\mathbf{s}_1 - \mathbf{m} \in \mathcal{P}(\mathbf{B}_1)$, then $\mathbf{s}_2 \in L_2$ such that $\mathbf{s}_2 - \mathbf{s}_1 \in \mathcal{P}(\mathbf{B}_2)$, ..., and finally $\mathbf{s}_k \in L_k$ such that $\mathbf{s}_k - \mathbf{s}_{k-1} \in \mathcal{P}(\mathbf{B}_k)$. It follows that $\mathbf{s}_k$ is somewhat close to $\mathbf{m}$, because $\mathbf{s}_k - \mathbf{m}$ is in the Minkowski sum $\mathcal{P}(\mathbf{B}_1) + \mathcal{P}(\mathbf{B}_2) + \cdots + \mathcal{P}(\mathbf{B}_k)$, which is a zonotope spanned by $\mathbf{B}_1, \ldots, \mathbf{B}_k$. And heuristically, the distribution of $2(\mathbf{s}_k - \mathbf{m})$ is the convolution of all the k uniform distributions $\mathcal{D}_{\mathcal{P}(\mathbf{B}_i)}$.

In other words, similarly to the perturbation-free case, an attacker wishing to recover the secret key of a GGH-type signature scheme using perturbations using a polynomial number of signatures is faced with the following problem with $m = kn$:

Problem 5.2 (The Hidden Zonotope Problem or HZP) *Let $m \geq n$ be positive integers, and $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_m]$ be an $m \times n$ row matrix of rank n. The input to the HZP is a sequence of* $\text{poly}(n, m)$

independent samples from $\mathcal{D} = \mathcal{D}_{\mathcal{Z}(\mathbf{V})}$ over $\mathbb{R}^n$, which is the convolution distribution over the zonotope $\mathcal{Z}(\mathbf{V}) = \{\sum_{i=1}^m x_i \mathbf{v}_i, -1 \leq x_i \leq 1\}$ spanned by $\mathbf{V}$. The goal is to find a good approximation of the rows of $\pm\mathbf{V}$.

Here, we assume $\mathbf{V}$ to have rank n, because this is the setting of NTRUSign with perturbation, and because the HPP is simply the HZP with $m = n$.

5.3.2 Extending the Nguyen-Regev Analysis to Zonotopes

Here, we study the behavior of the original Nguyen-Regev algorithm for learning a parallelepiped (**SolveHPP**$(\mathcal{X})$, Alg. 8) on a HZP instance, that is, when the secret matrix $\mathbf{V}$ is not necessarily square, but is an arbitrary $m \times n$ matrix of rank n with $m \geq n$. To do this, we need to change the analysis of Nguyen and Regev [NR06], and we will have to slightly change Alg. 8 to make the attack still work : Alg. 9 is the new algorithm. Recall that the input distribution $\mathcal{D}_{\mathcal{Z}(\mathbf{V})}$ is formed by $\sum_{i=1}^m x_i \mathbf{v}_i$ where the x_i's are uniformly chosen in $[-1, 1]$. We study how the two stages of the NR attack behave for $\mathcal{D}_{\mathcal{Z}(\mathbf{V})}$.

Isotropizing Zonotopes. We start with a trivial adaptation of Lemma 5.1 to zonotopes :

Lemma 5.3 *Let $\mathbf{V}$ be an $m \times n$ matrix over $\mathbb{R}$ of rank n. Let $\mathbf{G}$ be the symmetric definite positive matrix $\mathbf{V}^t\mathbf{V}$. Then :*
- $\mathrm{Cov}(\mathcal{D}_{\mathcal{Z}(\mathbf{V})}) = \mathbf{G}/12$.
- *If $\mathbf{L} \in \mathcal{GL}_n(\mathbb{R})$ satisfies $\mathbf{L}\mathbf{L}^t = \mathbf{G}^{-1}$ and we let $\mathbf{C} = \mathbf{V}\mathbf{L}$, then $\mathbf{C}^t\mathbf{C} = \mathbf{Id}_n$ and $\mathcal{D}_{\mathcal{Z}(\mathbf{V})} \cdot \mathbf{L} = \mathcal{D}_{\mathcal{Z}(\mathbf{C})}$.*

Lemma 5.3 shows that if we apply **Isotropize**$(\mathcal{X})$ (Alg. 6) to samples from $\mathcal{D}_{\mathcal{Z}(\mathbf{V})}$ (rather than $\mathcal{D}_{\mathcal{P}(\mathbf{V})}$), the output transformation $\mathbf{L}$ will be such that $\mathcal{D}_{\mathcal{Z}(\mathbf{V})} \cdot \mathbf{L}$ is close to $\mathcal{D}_{\mathcal{Z}(\mathbf{C})}$ for some $m \times n$ matrix $\mathbf{C}$ such that $\mathbf{C}^t\mathbf{C} = \mathbf{Id}_n$.

In other words, the effect of Step. 2 in **SolveHPP**$(\mathcal{X})$ (Alg. 8) is to make the zonotope matrix $\mathbf{V}$ have orthonormal columns : $\mathbf{V}^t\mathbf{V} = \mathbf{Id}_n$. The following lemma gives elementary properties of such matrices, which will be useful for our analysis :

Lemma 5.4 *Let $\mathbf{V}$ be an $m \times n$ row matrix $[\mathbf{v}_1, \ldots, \mathbf{v}_m]$ such that $\mathbf{V}^t\mathbf{V} = \mathbf{Id}_n$. Then :*
- $\|\mathbf{w}\|^2 = \sum_{i=1}^m \langle \mathbf{w}, \mathbf{v}_i \rangle^2$ *for all* $\mathbf{w} \in \mathbb{R}^n$.
- $\|\mathbf{v}_i\| \leq 1$ *for all* $1 \leq i \leq m$.
- $\sum_{i=1}^m \|\mathbf{v}_i\|^2 = n$ *and* $\mathbb{E}_{\mathbf{x} \leftarrow \mathcal{U}(\mathbb{S}_n)}(\|\mathbf{x}\mathbf{V}^t\|^2) = n/m$.

Proof: The first claim follows from $\mathbf{V}^t\mathbf{V} = \mathbf{Id}_n$. By taking $\mathbf{w} = \mathbf{v}_i$, we obtain $\|\mathbf{v}_i\| \leq 1$. Because $\mathbf{V}^t\mathbf{V}$ and $\mathbf{V}\mathbf{V}^t$ have the same trace, we have $\sum_{i=1}^m \|\mathbf{v}_i\|^2 = n$. And we deduce the final claim from $\mathrm{Cov}(\mathcal{U}(\mathbb{S}_n)) = \frac{1}{n}\mathbf{Id}_n$. □

Learning an isotropic Zonotope. Nguyen and Regev [NR06] used the target function

$$\mathrm{mom}_{\mathcal{D},4}(\mathbf{w}) = \mathbb{E}_{\mathbf{x} \leftarrow \mathcal{D}}[\langle \mathbf{x}, \mathbf{w} \rangle^4]$$

for $\mathbf{w} \in \mathbb{S}_n$, $\mathcal{D} = 2 \cdot \mathcal{D}_{\mathcal{P}(\mathbf{V})}$ and $\mathbf{V} \in \mathcal{O}_n(\mathbb{R})$ to recover the hidden hypercube. We need to study this function when $\mathcal{D}$ is the zonotope distribution $\mathcal{D} = \mathcal{D}_{\mathcal{Z}(\mathbf{V})}$ to recover the hidden zonotope. Let us recall that the *gradient* of f at $\mathbf{w} \in \mathbb{R}^n$ is denoted by $\nabla f(\mathbf{w}) = (\frac{\partial f}{\partial x_1}(\mathbf{w}), \ldots, \frac{\partial f}{\partial x_n}(\mathbf{w}))$. and the *Hessian matrix* of f at $\mathbf{w} \in \mathbb{R}^n$ is denoted by $\mathrm{H}f(\mathbf{w}) = (\frac{\partial^2 f}{\partial x_i \partial x_j}(\mathbf{w}))_{1 \leq i,j \leq n}$. Nguyen and Regev [NR06] gave elementary formulas for $\mathrm{mom}_{\mathcal{D},4}$ and $\nabla \mathrm{mom}_{\mathcal{D},4}$ when $\mathcal{D} = \mathcal{D}_{\mathcal{P}(\mathbf{V})}$ and $\mathbf{V} \in \mathcal{O}_n(\mathbb{R})$, which can easily be adapted to the zonotope distribution $\mathcal{D}_{\mathcal{Z}(\mathbf{V})}$ if $\mathbf{V}^t\mathbf{V} = \mathbf{Id}_n$, as follows :

Lemma 5.5 *Let $\mathbf{V}$ be a $m \times n$ matrix over $\mathbb{R}$ such that $\mathbf{V}^t\mathbf{V} = \mathbf{Id}_n$, and $\mathcal{D}$ be the convolution distribution $2 \cdot \mathcal{D}_{\mathcal{Z}(\mathbf{V})}$ over the zonotope spanned by $\mathbf{V}$. Then, for any $\mathbf{w} \in \mathbb{R}^n$:*

$$\mathrm{mom}_{\mathcal{D},4}(\mathbf{w}) = \frac{1}{3}\|\mathbf{w}\|^4 - \frac{2}{15}\sum_{i=1}^m \langle \mathbf{v}_i, \mathbf{w} \rangle^4$$

$$\nabla \mathrm{mom}_{\mathcal{D},4}(\mathbf{w}) = \frac{4}{3}\mathbf{w} - \frac{8}{15}\sum_{i=1}^m \langle \mathbf{v}_i, \mathbf{w} \rangle^3 \mathbf{v}_i \quad \textit{if} \quad \mathbf{w} \in \mathbb{S}_n$$

Corollary 5.6 *Under the same hypotheses as Lemma 5.5, the minima over* $\mathbb{S}_n$ *of the function* $\mathrm{mom}_{\mathcal{D},4}(\mathbf{w})$ *are the maxima (over* $\mathbb{S}_n$*) of* $f(\mathbf{w}) = \sum_{i=1}^m f_{\mathbf{v}_i}(\mathbf{w})$ *where* $f_{\mathbf{v}}(\mathbf{w}) = \langle \mathbf{v}, \mathbf{w} \rangle^4$ *is defined over* $\mathbb{R}^n$*.*

In [NR06, Lemma 3], Nguyen and Regev used Lagrange multipliers to show that when $\mathbf{V} \in \mathcal{O}_n(\mathbb{R})$, the local minima of $\mathrm{mom}_{\mathcal{D}_{\mathcal{P}(\mathbf{V})},4}$ were located at $\pm\mathbf{v}_1, \ldots, \pm\mathbf{v}_n$, and these minima are clearly global minima. However, this argument breaks down when $\mathbf{V}$ is a rectangular $m \times n$ matrix of rank n such that $\mathbf{V}^t\mathbf{V} = \mathbf{Id}_n$. To tackle the zonotope case, we use a different argument, which requires to study each function $f_{\mathbf{v}_i}(\mathbf{w}) = \langle \mathbf{v}_i, \mathbf{w} \rangle^4$ individually :

Lemma 5.7 *Let* $\mathbf{v} \in \mathbb{R}^n$ *and* $f_{\mathbf{v}}(\mathbf{w}) = \langle \mathbf{v}, \mathbf{w} \rangle^4$ *for* $\mathbf{w} \in \mathbb{R}^n$ *. Then :*

1. *The gradient and Hessian matrix of* $f_{\mathbf{v}}$ *are* $\nabla f_{\mathbf{v}}(\mathbf{w}) = 4 \langle \mathbf{w}, \mathbf{v} \rangle^3 \cdot \mathbf{v}$ *and* $\mathrm{H}\, f_{\mathbf{v}}(\mathbf{w}) = 12 \langle \mathbf{w}, \mathbf{v} \rangle^2 \cdot \mathbf{v}^t\mathbf{v}$*.*
2. *There are only two local maxima of* $f_{\mathbf{v}}$ *over* $\mathbb{S}_n$*, which are located at* $\pm\mathbf{v}/\|\mathbf{v}\|$*, and their value is* $\|\mathbf{v}\|^4$*.*
3. *The local minima of* $f_{\mathbf{v}}$ *over* $\mathbb{S}_n$ *are located on the hyperplane orthogonal to* $\mathbf{v}$*, and their value is 0.*
4. *The mean value of* $f_{\mathbf{v}}$ *over* $\mathbb{S}_n$ *is* $3\|\mathbf{v}\|^4/(n(n+2))$*.*

Proof: Claim 1 is trivial. Claim 2 and 3 use Lagrange multipliers : extrema are reached when the gradient $\nabla f_{\mathbf{v}}(\mathbf{w})$ is colinear with with $\mathbf{w}$, that is when $\mathbf{w}$ is colinear with $\mathbf{v}$ (locally maximizing $f_{\mathbf{v}}$) or when $\langle \mathbf{w}, \mathbf{v} \rangle = 0$ (locally minimizing $f_{\mathbf{v}}$). Now, assume without loss of generality that $\|\mathbf{v}\| = 1$, and consider the random variable $X = \langle \mathbf{v}, \mathbf{w} \rangle^2$ where $\mathbf{w}$ is a point chosen uniformly at random from the unit sphere. It is known that X follows a distribution $\mathrm{Beta}(1/2, (n-1)/2)$, and therefore has mean $\mu = 1/n$ and variance $V = 2(n-1)/[(n+2)n^2]$. It follows that X^2 has mean $V + \mu^2 = 3/[(n+2)n]$. □

This already gives a different point of view from Nguyen and Regev in the special case where $\mathbf{V} \in \mathcal{O}_n(\mathbb{R})$: for all $1 \leq j \leq n$, $\mathbf{v}_j$ is a local maximum of $f_{\mathbf{v}_j}$ and a local minimum of $f_{\mathbf{v}_i}$ for all $i \neq j$ because $\mathbf{v}_i \perp \mathbf{v}_j$; and therefore $\pm\mathbf{v}_1, \ldots, \mathbf{v}_n$ are local extrema of $\mathrm{mom}_{\mathcal{U}\cdot V,4}$.

In the general case where $\mathbf{V}$ is an $m \times n$ matrix such that $\mathbf{V}^t\mathbf{V} = \mathbf{Id}_n$, our main result provides a sufficient condition on $\mathbf{V}$ which guarantees that a given direction $\mathbf{v}_j/\|\mathbf{v}_j\|$ is close to a local minimum of $\mathrm{mom}_{\mathcal{D}_{\mathcal{Z}(\mathbf{V})},4}$:

Theorem 5.8 (Local Minima for Zonotopes) *Let* $\mathbf{V}$ *be a* $m \times n$ *matrix over* $\mathbb{R}$ *such that* $\mathbf{V}^t\mathbf{V} = \mathbf{Id}_n$*. Assume that there is* $\alpha \geq 1$ *such that* $\mathbf{V}$ *is* α*-weakly-orthogonal, that is, its* m *rows satisfy for all* $i \neq j$ *:*

$$|\langle \mathbf{v}_i, \mathbf{v}_j \rangle| \leq \alpha \|\mathbf{v}_i\| \|\mathbf{v}_j\| / \sqrt{n}.$$

Let $1 \leq j \leq m$ *and* $0 < \varepsilon < 1/\sqrt{2}$ *such that :*

$$\varepsilon \|\mathbf{v}_j\|^4 > 6 \left(\frac{\alpha}{\sqrt{n}} + \varepsilon \right)^2 \varepsilon + \frac{4}{\|\mathbf{v}_j\|^3} \| \sum_{i \neq j} \langle \mathbf{v}_j, \mathbf{v}_i \rangle^3 \mathbf{v}_i \| \tag{5.3}$$

which holds in particular if

$$\|\mathbf{v}_j\| \geq \frac{2\sqrt{\alpha}}{n^{1/12}} \quad \textit{and} \quad \varepsilon = \frac{5\alpha^3}{\sqrt{n}\|\mathbf{v}_j\|^4} < 1/\sqrt{2}.$$

Then, over the unit sphere, the function $\mathrm{mom}_{\mathcal{D}_{2\cdot\mathcal{Z}(\mathbf{V})},4}$ *has a local minimum at some point* $\mathbf{m}_j \in \mathbb{S}_n$ *such that* $\mathbf{m}_j$ *is close to the direction of* $\mathbf{v}_j$*, namely :*

$$\left\langle \mathbf{m}_j, \frac{\mathbf{v}_j}{\|\mathbf{v}_j\|} \right\rangle > 1 - \frac{\epsilon^2}{2} \quad \textit{and} \quad \left\| \mathbf{m}_j - \frac{\mathbf{v}_j}{\|\mathbf{v}_j\|} \right\| \leq \epsilon.$$

And the local minimum $\mathrm{mom}_{\mathcal{D}_{\mathcal{Z}(\mathbf{V})},4}(\mathbf{m}_j)$ *discloses an approximation of* $\|\mathbf{v}_j\|$*, namely :*

$$\left| \mathrm{mom}_{\mathcal{D}_{\mathcal{Z}(\mathbf{V})},4}(\mathbf{m}_j) - \left(\frac{1}{3} - \frac{2\|\mathbf{v}_j\|^4}{15} \right) \right| \leq \frac{2}{15} \left(\varepsilon^4 + 4\varepsilon^3 + 6\varepsilon^2 + 4\varepsilon + m \left(\varepsilon + \frac{\alpha}{\sqrt{n}} \right)^4 \right).$$

Before giving a detailed proof, let us provide some intuition. Let $\mathbf{d}_i = \mathbf{v}_i / \|\mathbf{v}_i\| \in \mathbb{S}_n$ for all $1 \leq i \leq m$. The direction $\mathbf{d}_j$ is a local maximum of $f_{\mathbf{v}_j}$ over $\mathbb{S}_n$. On the other hand, $f_{\mathbf{v}_i}(\mathbf{d}_j)$ is very small for all $i \neq j$ by weak orthogonality. This suggests that $\mathbf{d}_j$ should be very close to a local maximum of the whole sum $\sum_{i=1}^m f_{\mathbf{v}_i}(\mathbf{d}_j)$, provided that the local maximum $\|\mathbf{v}_j\|^4$ of $f_{\mathbf{v}_j}$ is somewhat larger than $\sum_{i \neq j} f_{\mathbf{v}_i}(\mathbf{d}_j)$. The proof thus relies on a careful look at the neighborhood of $\mathbf{d}_j$, using second-order Taylor-Lagrange approximations.

Theorem 5.9 (Vectorial Taylor-Lagrange Theorem) *If f is twice differentiable over some open set $\Omega \subseteq \mathbb{R}^n$, and that the second-order derivative of f is continuous over Ω, then for all $(\mathbf{a}, \mathbf{b}) \in \Omega^2$ such that Ω contains the segment $[\mathbf{a}, \mathbf{b}]$, there exists $\theta \in (0,1)$ such that :*

$$f(\mathbf{b}) - f(\mathbf{a}) = \langle \nabla f(\mathbf{a}), \mathbf{b} - \mathbf{a} \rangle + \frac{1}{2}(\mathbf{b} - \mathbf{a})\, \mathrm{H}\, f(\mathbf{a} + \theta(\mathbf{b} - \mathbf{a}))(\mathbf{b} - \mathbf{a})^t.$$

Proof: We keep using the notations defined above. Let $\mathcal{B} = \{\mathbf{w} \in \mathbb{S}_n : \|\mathbf{w} - \mathbf{d}_j\| < \varepsilon\}$ be the open ball of $\mathbb{S}_n$ of radius ε and centered at $\mathbf{d}_j$. Notice that for all $\mathbf{w} \in \mathbb{S}_n$:

$$\|\mathbf{w} - \mathbf{d}_j\|^2 = \|\mathbf{w}\|^2 + \|\mathbf{d}_j\|^2 - 2\langle \mathbf{w}, \mathbf{d}_j \rangle = 2(1 - \langle \mathbf{w}, \mathbf{d}_j \rangle).$$

Therefore $\mathcal{B} = \left\{\mathbf{w} \in \mathbb{S}_n : \langle \mathbf{d}_j, \mathbf{w} \rangle > 1 - \varepsilon^2/2\right\}$ and :

- The topological closure of $\mathcal{B}$ is $\bar{\mathcal{B}} = \left\{\mathbf{w} \in \mathbb{S}_n : \langle \mathbf{d}_j, \mathbf{w} \rangle \geq 1 - \varepsilon^2/2\right\}$.
- The boundary of $\mathcal{B}$ is $\partial\mathcal{B} = \left\{\mathbf{w} \in \mathbb{S}_n : \langle \mathbf{d}_j, \mathbf{w} \rangle = 1 - \varepsilon^2/2\right\}$.

Recall that $f = \sum_{i=1}^m f_{\mathbf{v}_i}$. We will prove the following property :

$$\forall \mathbf{w} \in \partial\mathcal{B}, f(\mathbf{w}) < f(\mathbf{d}_j), \tag{5.4}$$

which allows to conclude the proof of Th. 5.8. Indeed, by continuity, the restriction of f to $\bar{\mathcal{B}}$ has a global maximum at some point $\mathbf{m}_j \in \bar{\mathcal{B}}$. And (5.4) implies that $\mathbf{m}_j \notin \partial\mathcal{B}$, therefore $\mathbf{m}_j \in \mathcal{B}$. Thus, $\mathbf{m}$ is a global maximum of f over the open set $\mathcal{B}$: in other words, $\mathbf{m}_j$ is a local maximum of f, and therefore a local minimum of $\mathrm{mom}_{\mathcal{D},4}$. Furthermore, by definition of $\mathcal{B}$, we have : $\|\mathbf{m_j} - \mathbf{d}_j\| < \varepsilon$ and $\langle \mathbf{d}_j, \mathbf{m}_j \rangle > 1 - \varepsilon^2/2$. And the final inequality follows from :

$$\mathrm{mom}_{\mathcal{D},4}(\mathbf{m}_j) - \left(\frac{1}{3} - \frac{2\|\mathbf{v}_j\|^4}{15}\right) = \frac{2}{15}\left(\langle \mathbf{v}_j, \mathbf{d}_j \rangle^4 - \langle \mathbf{v}_j, \mathbf{m}_j \rangle^4 - \sum_{i \neq j} \langle \mathbf{v}_i, \mathbf{m_j} \rangle^4\right).$$

We now prove (5.4). Let $\mathbf{w} \in \partial\mathcal{B}$. To show $f(\mathbf{d}_j) - f(\mathbf{w}) > 0$, we decompose f as $f = f_{\mathbf{v}_j} + \sum_{i \neq j} f_{\mathbf{v}_i}$. On the one hand, the first term is :

$$f_{\mathbf{v}_j}(\mathbf{d}_j) - f_{\mathbf{v}_j}(\mathbf{w}) = \|\mathbf{v}_j\|^4 - (1 - \frac{\varepsilon^2}{2})^4 \|\mathbf{v}_j\|^4 = \left(\frac{4\varepsilon^2}{2} - \frac{6\varepsilon^4}{4} + \frac{4\varepsilon^6}{8} - \frac{\varepsilon^8}{16}\right)\|\mathbf{v}_j\|^4 \geq \varepsilon^2 \|\mathbf{v}_j\|^4 \tag{5.5}$$

because $\varepsilon < 1/\sqrt{2}$. On the other hand, the second term can be bounded by the Taylor-Lagrange formula, which states that there exists $\theta \in (0,1)$ such that :

$$\sum_{i \neq j} (f_{\mathbf{v}_i}(\mathbf{w}) - f_{\mathbf{v}_i}(\mathbf{d}_j)) = \left\langle \sum_{i \neq j} \nabla f_{\mathbf{v}_i}(\mathbf{d}_j), \mathbf{w} - \mathbf{d}_j \right\rangle + \frac{1}{2}(\mathbf{w} - \mathbf{d}_j) \sum_{i \neq j} \mathrm{H}\, f_{\mathbf{v}_i}(\mathbf{d}_j + \theta(\mathbf{w} - \mathbf{d}_j))(\mathbf{w} - \mathbf{d}_j)^t \tag{5.6}$$

Let $\mathbf{g} = \sum_{i \neq j} \nabla f_{\mathbf{v}_i}(\mathbf{d}_j) = 4 \sum_{i \neq j} \langle \mathbf{d}_j, \mathbf{v}_i \rangle^3 \mathbf{v}_i$ by Lemma 5.7. We have :

$$\left| \left\langle \sum_{i \neq j} \nabla f_{\mathbf{v}_i}(\mathbf{d}_j), \mathbf{w} - \mathbf{d}_j \right\rangle \right| \leq \varepsilon \|\mathbf{g}\|. \tag{5.7}$$

Now, let $\mathbf{c} = \mathbf{d}_j + \theta(\mathbf{w} - \mathbf{d}_j)$. By Lemma 5.7 :

$$\mathrm{H}\left(\sum_{i \neq j} f_{\mathbf{v}_i}\right)(\mathbf{c}) = 12 \sum_{i \neq j} \langle \mathbf{v}_i, \mathbf{c} \rangle^2 \mathbf{v}_i^t \mathbf{v}_i,$$

which is a symmetric positive matrix. We have :

$$\langle \mathbf{v}_i, \mathbf{c} \rangle^2 = \langle \mathbf{v}_i, \mathbf{d}_j + \theta(\mathbf{w} - \mathbf{d}_j) \rangle^2 \leq \alpha^2/n + \theta^2\varepsilon^2 + 2\alpha\theta\varepsilon/\sqrt{n} \leq (\alpha/\sqrt{n} + \varepsilon)^2.$$

We deduce the following inequalities between positive matrices :

$$\mathrm{H}\left(\sum_{i \neq j} f_{\mathbf{v}_i}\right)(\mathbf{c}) \leq 12(\alpha/\sqrt{n} + \varepsilon)^2 \sum_{i \neq j} \mathbf{v}_i^t \mathbf{v}_i \leq 12(\alpha/\sqrt{n} + \varepsilon)^2 \sum_{i=1}^m \mathbf{v}_i^t \mathbf{v}_i = 12(\alpha/\sqrt{n} + \varepsilon)^2 \mathbf{Id}_n$$

Hence :

$$\left|(\mathbf{w}-\mathbf{d}_j)\sum_{i\neq j}\mathrm{H}\,f_{\mathbf{v}_i}(\mathbf{d}_j+\theta(\mathbf{w}-\mathbf{d}_j))(\mathbf{w}-\mathbf{d}_j)^t\right| \leq 12(\alpha/\sqrt{n}+\varepsilon)^2\|\mathbf{w}-\mathbf{d}_j\|^2 \leq 12(\alpha/\sqrt{n}+\varepsilon)^2\varepsilon^2. \quad (5.8)$$

Collecting (5.5), (5.6), (5.7) and (5.8), we obtain :

$$f(\mathbf{d}_j)-f(\mathbf{w}) \geq \varepsilon^2\|\mathbf{v}_j\|^4-\|\mathbf{g}\|\varepsilon-6(\alpha/\sqrt{n}+\varepsilon)^2\varepsilon^2 = \left(\varepsilon\|\mathbf{v}_j\|^4-\|\mathbf{g}\|-6(\alpha/\sqrt{n}+\varepsilon)^2\varepsilon\right)\varepsilon,$$

which is >0 by (5.3).

To conclude, it remains to prove that (5.3) is satisfied when $\|\mathbf{v}_j\| \geq \frac{2\sqrt{\alpha}}{n^{1/12}}$ and $\varepsilon = \frac{5\alpha^3}{\sqrt{n}\|\mathbf{v}_j\|^4} < 1/\sqrt{2}$. We first bound the gradient : $\|\mathbf{g}\| \leq 4\sum_{i\neq j}|\langle\mathbf{d}_j,\mathbf{v}_i\rangle|^3\|\mathbf{v}_i\|$, where $|\langle\mathbf{d}_j,\mathbf{v}_i\rangle| \leq \|\mathbf{v}_i\|\alpha/\sqrt{n}$ by weak-orthogonality. Since $\|\mathbf{v}_i\| \leq 1$ and $\sum_{i=1}^m\|\mathbf{v}_i\|^2 = n$ by Lemma 5.4 :

$$\|\mathbf{g}\| \leq 4\sum_{i\neq j}\|\mathbf{v}_i\|^4\alpha^3/n^{3/2} \leq 4\sum_{i\neq j}\|\mathbf{v}_i\|^2\alpha^3/n^{3/2} \leq 4\alpha^3/\sqrt{n}$$

Thus :

$$\varepsilon\|\mathbf{v}_j\|^4-\|\mathbf{g}\|-6(\alpha/\sqrt{n}+\varepsilon)^2\varepsilon \geq \varepsilon\|\mathbf{v}_j\|^4-\frac{4\alpha^3}{\sqrt{n}}-6(\alpha/\sqrt{n}+\varepsilon)^2\varepsilon.$$

Now, notice that $\alpha \geq 1$ and $\|\mathbf{v}_j\| \leq 1$ (by Lemma 5.4) imply that $\alpha/\sqrt{n} \leq \varepsilon$. And $\varepsilon\|\mathbf{v}_j\|^4 = \frac{5\alpha^3}{\sqrt{n}}$ by definition of ε. Hence :

$$\varepsilon\|\mathbf{v}_j\|^4-\|\mathbf{g}\|-6(\alpha/\sqrt{n}+\varepsilon)^2\varepsilon \geq \frac{\alpha^3}{\sqrt{n}}-6(2\varepsilon)^2\varepsilon = \frac{\alpha^3}{\sqrt{n}}-24\varepsilon^3,$$

which is >0 if and only if $\frac{\alpha}{n^{1/6}} > 24^{1/3}\varepsilon$. By assumption, $\|\mathbf{v}_j\| \geq \frac{2\sqrt{\alpha}}{n^{1/12}}$, therefore $\|\mathbf{v}_j\|^4 \geq \frac{16\alpha^2}{n^{1/3}}$:

$$\varepsilon = \frac{5\alpha^3}{\sqrt{n}\|\mathbf{v}_j\|^4} < \frac{5\alpha^3 n^{1/3}}{\sqrt{n}16\alpha^2} = \frac{5\alpha}{16n^{1/6}} < \frac{\alpha}{24^{1/3}n^{1/6}}.$$

□

Th. 5.8 states that under suitable assumptions on $\mathbf{V}$ (which we will discuss shortly), if $\|\mathbf{v}_j\|$ is not too small, then the secret direction $\mathbf{v}_j/\|\mathbf{v}_j\|$ is very close to a local minimum of $\mathrm{mom}_{\mathcal{D}_{\mathcal{Z}(\mathbf{V})},4}$, whose value discloses an approximation of $\|\mathbf{v}_j\|$, because it is $\approx \frac{1}{3}-\frac{2}{15}\|\mathbf{v}_j\|^4$. This suggests **SolveHZP**$(\mathcal{X})$ (Alg. 9) for learning a zonotope : **SolveHZP**$(\mathcal{X})$ is exactly **SolveHPP**$(\mathcal{X})$ (Alg. 8) in which Step 5 of **SolveHPP**$(\mathcal{X})$ has been replaced by Step 5, in order to take into account that $\|\mathbf{v}_j\|$ is no longer necessarily equal to 1, but can fortunately be approximated by the value of the local minimum.

Algorithm 9 SolveHZP$(\mathcal{X})$: Learning a Zonotope

Input: A set $\mathcal{X}$ of vectors $\mathbf{x} \in \mathbb{R}^n$ sampled from $\mathcal{D}_{\mathcal{Z}(\mathbf{V})}$, where $\mathbf{V}$ is an $m \times n$ matrix of rank n.
Output: An approximation of some row vector of $\pm\mathbf{V}$.

1: $L \leftarrow$ **Isotropize**$(\mathcal{X})$ using Alg. 6
2: $\mathcal{X} \leftarrow \mathcal{X}\cdot\mathbf{L}$
3: Pick $\mathbf{w}$ uniformly at random from $\mathbb{S}_n$
4: Compute $\mathbf{r} \leftarrow$ **Descent**$(\mathcal{X},\mathbf{w},\delta) \in \mathbb{S}_n$ using Alg. 7 : use $\delta = 3/4$ in theory and $\delta = 0.7$ in practice.
5: Return $\lambda\mathbf{r}L^{-1}$ where $\lambda = ((\frac{1}{3}-\mathrm{mom}_{\mathcal{X},4}(\mathbf{r}))\frac{15}{2})^{1/4}$

First, we discuss the value of α in Th. 5.8 . Note that weak-orthogonality is a natural property, as shown by the following basic result :

Lemma 5.10 *Let $\mathbf{v} \in \mathbb{S}_n$ and denote by X the random variable $X = \langle\mathbf{v},\mathbf{w}\rangle^2$ where $\mathbf{w}$ has uniform distribution over $\mathbb{S}_n$. Then X has distribution Beta$(1/2,(n-1)/2)$, $\mathbb{E}(X) = \frac{1}{n}$, $\mathbb{E}(X^2) = \frac{3}{n(n+2)}$, $\mathbb{E}(X^3) = \frac{15}{n(n+2)(n+4)}$ and more generally : $\mathbb{E}(X^k) = \frac{k-1/2}{n/2+k-1}\mathbb{E}(X^{k-1})$.*

By studying more carefully the Beta distribution, it is possible to obtain strong bounds. For instance, Ajtai [Ajt06, Lemma 47] showed that for all sufficiently large n, if $\mathbf{v} \in \mathbb{S}_n$ is fixed and $\mathbf{w}$ has uniform distribution over $\mathbb{S}_n$, then $|\langle \mathbf{v}, \mathbf{w} \rangle| \leq (\log n)/\sqrt{n}$ with probability $\geq 1 - \frac{1}{n^{(\log n)/2-1}}$. Since the probability is subexponentially close to 1, this implies that if $m = n^{O(1)}$ and we assume that all the directions $\mathbf{v}_i/\|\mathbf{v}_i\|$ are random, then $\mathbf{V}$ is $(\log n)$-weakly orthogonal with probability asymptotically close to 1.

This gives strong evidence that, if $m = n^{O(1)}$, the assumption on $\mathbf{V}$ in Th. 5.8 will be satisfied for $\alpha = \log n$. We can now discuss the remaining assumptions. If $\alpha = \log n$, we may take any index j such that $\|\mathbf{v}_j\| \geq \Omega(1/n^{13})$: in particular, if $\|\mathbf{v}_j\| = \Omega(1)$, we may take $\varepsilon = O(\log^3 n)/\sqrt{n}$. And higher values of α can be tolerated, as while as $\alpha = o(n^{1/6})$. Now recall that $\sum_{i=1}^{m} \|\mathbf{v}_i\|^2 = n$, thus $\max_i \|\mathbf{v}_i\| \geq \sqrt{n/m}$ and $\|\mathbf{v}_i\|$ is on the average $\sqrt{n/m}$. In particular, if the number of perturbations is constant, then $m = O(n)$ and $\max_i \|\mathbf{v}_i\| \geq \Omega(1)$, therefore Th. 5.8 applies to at least one index j, provided that $\alpha = o(n^{1/6})$. In fact, one can see that the result can even tolerate slightly bigger values of m than $\Theta(n)$, such as $m = o(n^{7/6}/\log n)$.

While Th. 5.8 explains why **SolveHZP**$(\mathcal{X})$ (Alg. 9) can heuristically solve the HZP, it is not a full proof, as opposed to the simpler parallelepiped case. The obstructions are the following :

- First, we would need to prove that the distance is sufficiently small to enable the recovery of the original zonotope vectors, using an appropriate BDD solver. Any error on $\mathbf{v}_j/\|\mathbf{v}_j\|$ is multiplied by $L^{-1}\|\mathbf{v}_j\|$. In [NR06], the error on $\mathbf{v}_j$ could be made polynomially small for any polynomial, provided that the number of samples was (polynomially) large enough. But ε cannot be chosen polynomially small for any arbitrary polynomial in Th. 5.8.
- Second, we would need to prove that **Descent**$(\mathcal{X}, \mathbf{w}, \delta)$ (Alg. 7) finds a random local minimum of $\mathrm{mom}_{\mathcal{D}_{\mathcal{Z}(\mathbf{V})},4}$ in polynomial time, even in the presence of noise to compute $\mathrm{mom}_{\mathcal{D}_{\mathcal{Z}(\mathbf{V})},4}$. Intuitively, this does not seem unreasonable since the function $\mathrm{mom}_{\mathcal{D}_{\mathcal{Z}(\mathbf{V})},4}$ is very regular, but it remains to be proved : we do however provide a heuristic explanation in Section 5.3.5.
- Finally, we would need to prove that there are no other local minima, or at least, not too many of them.

Regarding the third obstruction, it is easy to prove the following weaker statement, which implies that global minima of $\mathrm{mom}_{\mathcal{D}_{\mathcal{Z}(\mathbf{V})},4}$ over the unit sphere are close to some direction $\mathbf{v}_j/\|\mathbf{v}_j\|$:

Lemma 5.11 *Let $\mathbf{V}$ be a $m \times n$ matrix over $\mathbb{R}$ such that $\mathbf{V}^t\mathbf{V} = \mathbf{Id}_n$, and $\mathcal{D}$ be the distribution $\mathcal{D}_{\mathcal{Z}(\mathbf{V})}$. Let $\mathbf{w}$ be a global maximum of $f(\mathbf{w}) = \sum_{i=1}^{m} f_{\mathbf{v}_i}(\mathbf{w})$ over $\mathbb{S}_n$. Then there exists $j \in \{1, \ldots, m\}$ such that :*

$$\frac{1}{m^{1/4}} < \frac{|\langle \mathbf{v}_j, \mathbf{w} \rangle|}{\|\mathbf{v}_j\|} \leq 1.$$

We discuss the open problem of overcoming those obtrusction in the conclusion (Sect. 5.6) of that chapter.

5.3.3 Meet-in-the-Middle Error Correction Algorithm

In this section, we present Odlyzko's attack [HGSW03], with a few tweaks of our own to adapt to the specificity of our problem. Absolute values and norms in $\mathbb{Z}_q$ and $\mathbb{Z}_q^n$ are defined on the representative of the class modulo q that lies in $\{-\lceil q/2 \rfloor \ldots \lceil (q+1)/2 \rfloor\}$.

Overview For the original NR attack on non-perturbed NTRUSign, the results of the gradient descent converge exactly to the row vectors of $\mathbf{V}$, thus, with enough samples, the exact row vectors could be recovered by rounding each coordinate to the nearest integer. This is no longer the case here : in fact, even with an infinite number of samples, the rounding may give close but different vectors. Furthermore, our experiments show that perturbation slow down convergence, therefore error recovery should significantly decrease the required amount of samples. It is therefore important to have a different method to correct errors in the output of the gradient descent.

Consider a vector $\mathbf{w}$ returned by the descent, whose rounding is $(\mathbf{F}'||\mathbf{G}')$, and assume it is close to $\mathbf{b}_{N+1} = (\mathbf{F}|\mathbf{G})$. The error-correction problem can be stated as follows : given an integer vector $(\mathbf{F}'|\mathbf{G}')$, find small integer vectors $\epsilon_F, \epsilon_G \in \mathbb{Z}^N$ such that $(\mathbf{F}' + \epsilon_F|\mathbf{G}' + \epsilon_G)$ belongs to the public NTRUSign lattice. Given the public key $\mathbf{h}$, this equivalent to finding a small ϵ_G such that $\mathbf{h} * \epsilon_G - (\mathbf{F}' - \mathbf{h} * \mathbf{G}')$ is small (or symmetrically with ϵ_F). This problem is in fact a non-homogeneous variant of directly finding the secret key $(\mathbf{f}, \mathbf{g})$ from $\mathbf{h}$, which has already been studied for NTRU encryption : Odlyzko discovered a meet-in-the-middle attack described in [HGSW03], which was significantly improved by Howgrave-Graham [HG07] using lattice reduction. These attacks are exponential in $\|\mathbf{f}\|_1$ and dictate

the parameter sets of NTRUSign. It seems that both attacks are compatible with a non-homogeneous equation, however for ease of implementation, we only considered the original Odlyzko attack.

Assuming the error $(\epsilon_F|\epsilon_G)$ is ternary (*i.e* coefficients in $\{-1,0,1\}$), this attack runs in time and memory $\approx (2N)^{e/2}$ where $e = \|\epsilon_F\|_1$. Our tweaked version of the MiM attack can in fact deal with a non-ternary vector ϵ_G if $\|\epsilon_G\|_\infty$ is small enough.

Algorithm Description In this section, we describe Odlyzko's attack [HGSW03], with a few tweaks (described below) of our own to adapt to the specificity of our problem. Absolute values and norms in $\mathbb{Z}_q$ and $\mathbb{Z}_q^n$ are defined on the representative of the class modulo q that lies in $\{-\lceil q/2\rfloor \dots \lceil (q+1)/2\rfloor\}$.

For a binary vector $\mathbf{c} \in \{0,1\}^d$, $\bar{\mathbf{c}}$ denotes the entry-wise complementary of $\mathbf{c}$.

Problem 5.3 (Error correction problem) *Given the public key* $\mathbf{h}$ *such that* $\mathbf{g} = \mathbf{h} * \mathbf{f}$*, error bounds* $e, w \in \mathbb{N}$*,* $w < q/2$ *and a set of potential approximations* $\mathcal{A} = \{(\mathbf{f}'_i|\mathbf{g}'_i) : i \in [m]\}$ *of* f *and* g*, such that for at least one index* i*, the approximation satisfies the following :*

- $\epsilon_{\mathbf{f}} = \mathbf{f} - \mathbf{f}'_i$ *is ternary and* $\|\epsilon_{\mathbf{f}}\|_1 \leq e$*,*
- $\epsilon_{\mathbf{g}} = \mathbf{g} - \mathbf{g}'_i$ *satisfies* $\|\epsilon_{\mathbf{g}}\|_\infty \leq 2w+1$ *;*

recover $\mathbf{f}$ *and* $\mathbf{g}$*.*

Tweaks Our generalized problem definition already introduced two new parameters, namely w and the set of approximation $\mathcal{A}$. To solve it, we introduce another parameter $d \in \mathbb{N}$, corresponding to the depth of the code. This new parameter d helps us to control the explosion of the number of codes when w grows.

The original Odlyzko attack [HGSW03] is obtained with the following setting : $w = 0, d = n, \mathcal{A} = \{\mathbf{0}\}$.

Codes We consider that the parameters $w < q/2$ and d are fixed for the rest of the section. Let

$$\mathcal{R}_0 = \{(-w \mod q) \dots (\lceil q/2\rceil + w) \mod q\} \text{ and } \mathcal{R}_1 = \{-x \mod q : x \in \mathcal{R}_0\}.$$

Those sets satisfy the following property :

$$\forall x_1, x_2 \in \mathbb{Z}_q, |x_1 + x_2| \leq 2w+1 \Rightarrow ((x_1 \in \mathcal{R}_0 \wedge x_2 \in \mathcal{R}_1) \text{ or } (x_1 \in \mathcal{R}_1 \wedge x_2 \in \mathcal{R}_0)). \tag{5.9}$$

The set of codes $\mathcal{C}(\mathbf{v})$ of a vector $\mathbf{v} \in \mathbb{Z}_q^n$ is the set of binary vectors $\mathbf{c}$ of length d that satisfy :

$$\forall i \in [d], c_i = 0 \Rightarrow v_i \in \mathcal{R}_0 \text{ and } c_i = 1 \Rightarrow v_i \in \mathcal{R}_1.$$

It is designed to satisfy the following :

$$\forall \mathbf{v}, \mathbf{w} \in \mathbb{Z}_q^n, \|\mathbf{v} + \mathbf{w}\| \leq 2w+1 \Rightarrow \exists \mathbf{c_v} \in \mathcal{C}(\mathbf{v}), \exists \mathbf{c_w} \in \mathcal{C}(\mathbf{w}) \; s.t. \; c_{\mathbf{v}} = \bar{c_{\mathbf{w}}}$$

or equivalently, $\|\mathbf{v} + \mathbf{w}\| \leq 2w+1$ only if there exists a code of $\mathbf{v}$ and a code of $\mathbf{w}$ that are complementary. However, the converse is not true (even for $d = n$) : those codes are designed to discriminate pairs that cannot be solutions, but false positives can happen.

Finally, it should be noted that the average number of codes of a random vector $\mathbf{v}$, $\#\mathcal{C}(\mathbf{v})$ is $\left(1 + \frac{4w+2}{q}\right)^d$. And there is an algorithm computing the set $\mathcal{C}(\mathbf{v})$ in time $O(d \cdot \#\mathcal{C}(\mathbf{v}))$, which we denote by $\mathbf{Codes}(w, d, \mathbf{v})$.

Algorithm The algorithm makes use of labelled boxes $B[\mathbf{c}]$ where $\mathbf{c}$ is a code, to store vectors ; this is usually implemented using hash-tables. In practice, to save memory, the vectors $\epsilon_{\mathbf{f}}$ can be compressed because they are quite sparse : one can limit to storing the indexes of 1 and -1 for example. The description of algorithm **MiMErrorRecov** is given below (Alg. 10).

Correctness and Efficiency While the correction of this algorithm is straightforward considering the termination condition, its efficiency is not obvious. For the case $w = 0$, we can refer to the original analysis of [HGSW03]. Having as input a set $\mathcal{A}$ instead of running our algorithm on each approximation one after another improves practical efficiency, as well as theoretical efficiency by a factor $\#\mathcal{A}$ when the star-product is implemented without Fast-Fourier Transform, without increasing memory.

Once again, the empirical statement that this algorithm terminates in a reasonable time for the approximation provided is sufficient for our purposes.

Algorithm 10 MiMErrorRecov$(\mathcal{A}, \mathbf{h}, e, w, d, q)$: Meet-in-the-Middle Error Correction

Input: A set of potential approximations $\mathcal{A}$, parameters $e, w, d, q \in \mathbb{N}$
Output: A vector $(\mathbf{f}, \mathbf{g})$ such that $\mathbf{g} = \mathbf{h} * \mathbf{f}$, and that is close (as in Problem 5.3) to one vector of $\mathcal{A}$

$\mathcal{K} \leftarrow \{\mathbf{k}_i = \mathbf{h} * \mathbf{f}'_i - \mathbf{g}'_i : (\mathbf{f}'_i | \mathbf{g}'_i) \in \mathcal{A}\}$
while true **do**
  let $\epsilon_{\mathbf{f}}$ be a random ternary vector s.t. $\|\epsilon_{\mathbf{f}}\|_1 \leq \lceil e/2 \rceil$
  $\epsilon_{\mathbf{g}} = h * \epsilon_{\mathbf{f}}$
  $\mathcal{C} \leftarrow$ **Codes**$(w, d, \epsilon_{\mathbf{g}})$
  for $c \in \mathcal{C}$ **do**
    Store $\epsilon_{\mathbf{f}}$ in $B[\mathbf{c}]$
  end for
  for $\mathbf{k}_i \in \mathcal{K}$ **do**
    $\mathcal{C} \leftarrow$ **Codes**$(w, d, \epsilon_{\mathbf{g}} + \mathbf{k}_i)$
    for $c \in \mathcal{C}$ **do**
      for $\epsilon'_{\mathbf{f}} \in B[\bar{\mathbf{c}}]$ **do**
        $\epsilon'_{\mathbf{g}} \leftarrow \mathbf{h} * \epsilon'_f$
        if $\|\epsilon_{\mathbf{f}} + \epsilon'_{\mathbf{f}}\|_\infty \leq 1$ and $\left\|\mathbf{k}_i + \epsilon_{\mathbf{g}} + \epsilon'_{\mathbf{g}}\right\|_\infty \leq 2w + 1$
          then return $(\mathbf{f}'_i + \epsilon_{\mathbf{f}} + \epsilon'_{\mathbf{f}} \mid \mathbf{g}'_i + \mathbf{k}_i + \epsilon_{\mathbf{g}} + \epsilon'_{\mathbf{g}})$
      end for
    end for
  end for
end while

5.3.4 Experiments

We now report on experiments with the attack performed on NTRUSign, with n up to 502. Our experiments are real-world experiments using signatures of uniformly distributed messages.

Conditions of Th. 5.8 Our discussion following Th. 5.8 suggested that the matrix $\mathbf{V}$ should be heuristically weakly-orthogonal for $\alpha = \log n$. In practice, we may in fact take $\alpha \approx 5$ for both types of NTRUSign secret bases.

Regarding the norms $\|\mathbf{v}_i\|$ after morphing, we experimentally verified that $\|\mathbf{v}_i\| \approx \sqrt{1/k}$ where k is the number of perturbations for NTRUSign transposed bases (see Table 5.3.4), as expected by $\sum_{i=1}^{m} \|\mathbf{v}_i\|^2 = n$. But for the so-called standard bases, the situation is a bit different : half of the $\|\mathbf{v}_i\|$'s are very small, and the remaining half are close to $\sqrt{2/k}$. This can be explained by the fact that standard bases are unbalanced : half of the vectors is much shorter than the other vectors.

For a constant number of perturbations, we experimentally verified that $\|\mathbf{g}\| = O(1/n)$ with a small constant ≤ 4 (see 5.3.4), where $\mathbf{g} = \frac{4}{\|\mathbf{v}_j\|^3} \| \sum_{i \neq j} \langle \mathbf{v}_j, \mathbf{v}_i \rangle^3 \mathbf{v}_i \|$ is the "gradient" appearing in the conditions of Th. 5.8.

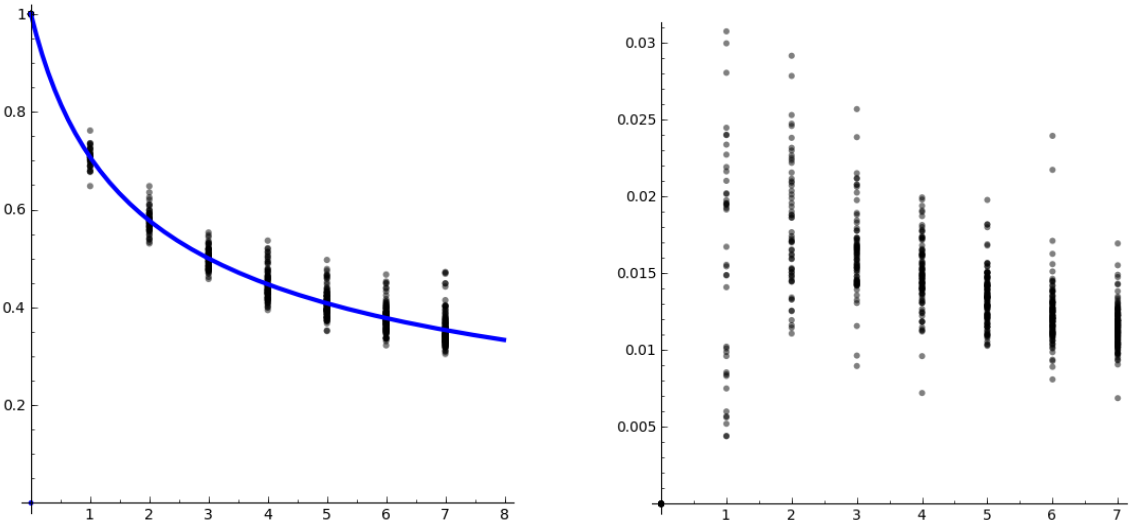
Thus, the conditions of Th. 5.8 are experimentally verified for a small number of perturbations : for all vectors $\mathbf{v}_j$'s in the case of transposed bases, and for half of the vectors $\mathbf{v}_j$'s in the case of standard bases.

Modifications to the original NR attack. We already explained that the original NR algorithm **SolveHPP**$(\mathcal{X})$ (Alg. 8) had to be slightly modified into **SolveHZP**$(\mathcal{X})$ (Alg. 9).

However, because Th. 5.8 states that the secret direction might be perturbed by some small ε, we also implemented an additional modification : instead of the elementary BDD algorithm by rounding, we used in the final stage a special BDD algorithm tailored for NTRU lattices, which is a tweaked version of Odlyzko's meet-in-the-middle attack on NTRU described in [HGSW03]. Details were given in Section. 5.3.3.

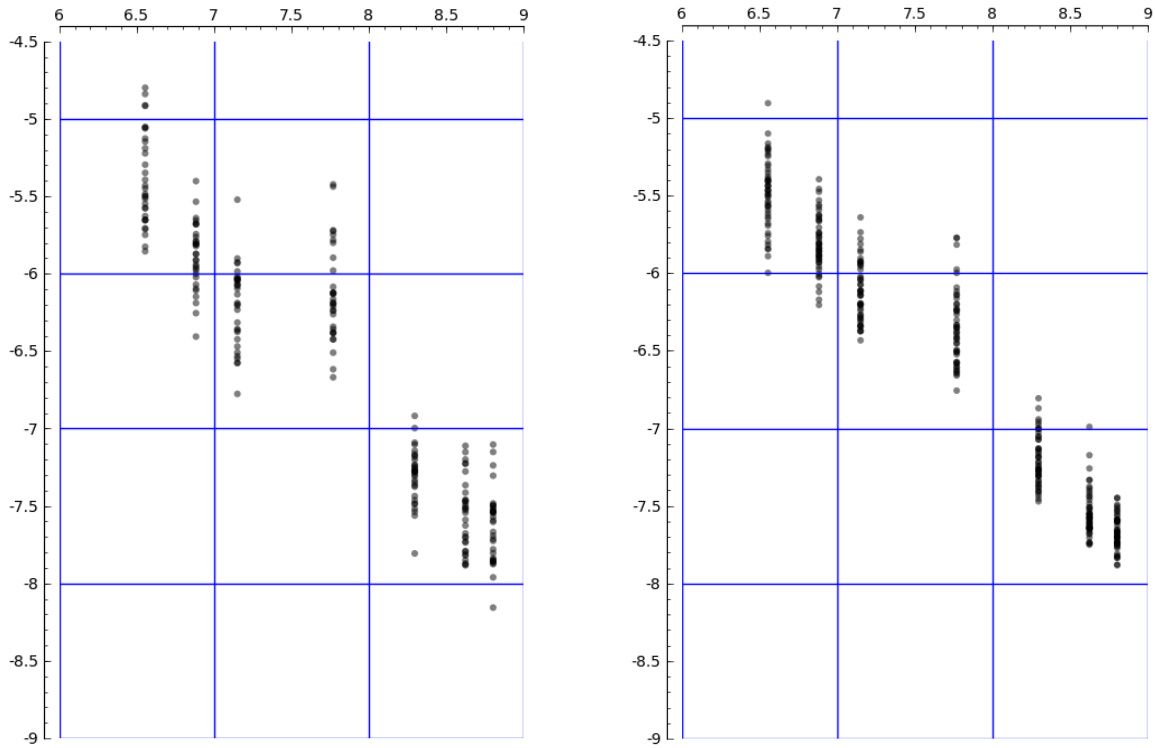
Practical cryptanalysis. We first applied successfully the optimized NR-attack on the original NTRUSign-251 scheme with one perturbation (which corresponds to a lattice dimension of 502), as initially submitted to the IEEE P1363 standard : about 8,000 signatures were sufficient to recover the secret key, which should be compared with the 400 signatures of the original attack [NR06] when there was no perturbation. This means that the original NTRUSign-251 scheme [HGP+] is now completely broken.

TABLE 5.1 – Evolution of the norms $\|\mathbf{v}_j\|$ (on the left) and $\|\mathbf{g}\|$ (on the right), as the number of perturbations increases.



The following graphics gives measured norms $\|\mathbf{v}_j\|$ (left) and $\|\mathbf{g}\|$ (right) for transposed NTRUSign bases, from 0 to 8 perturbations (*i.e.* $k = 1 \dots 9$) measured over 10 secret-key generations for each k, in dimension $n = 94$.
Recall that $\sum_{i=1}^{m} \|\mathbf{v}_i\|^2 = n$, so on average, we expect $\|\mathbf{v}_i\| \approx \sqrt{n/m} = 1/\sqrt{k}$. This theoretical estimate is confirmed by our experiments for transposed NTRUSign bases.

TABLE 5.2 – Norms $\|\mathbf{g}\|$ as a function of the dimension n, in $\log_2$-$\log_2$ scale, for $k = 2$ (left) and $k = 3$ (right)



The following graphics give measured norms $\|\mathbf{g}\|$ for transposed NTRUSign bases with respectively 1 (on the left) and 2 (on the right) perturbation, as the dimension n increases from 94 to 502, on a $\log_2$-$\log_2$ scale, measured over 10 secret key generations for each dimension.
On each graphics, the slope seems to be -1 :1, which means that for a constant number of perturbations $\|\mathbf{g}\|$ decreases as $\Theta(1/n)$.

Furthermore, we performed additional experiments for varying dimension and number of perturbations, for the parameters proposed in the latest NTRU article [HHGPW10], where transposed bases are used. Table 1 summarizes the results obtained : each successful attack took less than a day, and the MiM error recovery algorithm ran with less than 8Gb of memory.

TABLE 5.3 – Experiments with the generalized NR-attack on the latest NTRUSIGN parameters [HHGPW10]

Security level : dimension n	Toy : 94	80-bit : 314	112-bit : 394	128-bit : 446
0 perturbation	300 :(0,1)	400 :(0,1)	400 :(0,1)	600 :(0,1)
1 perturbation	**1000 :(1,2)**	**5000 :(0,1)**	**4000 :(0,1)**	**4000 :(0,0)**
2 perturbations	10000 :(5,3)	12000 :(0,2)		
3 perturbations	12000 :(5,4)			
4 perturbations	100000 :(0,1)			

In this table, each non-empty cell represents a successful attack for a given transposed basis (the column indicates the security level and the dimension) and number of perturbations (row). These cells have the form s : $(e = \|\epsilon_F\|_1, w = \|\epsilon_G\|_\infty)$ where s is the number of signatures used by the learning algorithm, and where $(\epsilon_F|\epsilon_G)$ is the error vector of the best approximation given by a descent. The running time of our MiM-Algorithm is about $(n/2)^{\lceil e/2 \rceil + 1}$ for such small w.

Our experiments confirm our theoretical analysis, which suggested that NTRUSIGN with a constant number of perturbations is insecure, but we see that the number of signatures required increases with the number of perturbations.

5.3.5 Heuristical Argument for the Convergence of the Descent

In this section, we heuristically explain why **Descent**$(\mathcal{X}, \mathbf{w}, \delta)$ (Alg. 7) converges to a point which is in some sense close to a secret direction $\mathbf{v}_j/\|\mathbf{v}_j\|$, for the descent parameter $\delta = 3/4$.

We assume that all gradients are computed exactly without any error. Similarly to [NR06], we note that $\delta = 3/4$ and Lemma 5.5 imply that Step 2 in Algorithm 7 performs

$$\mathbf{w}_{new} = \frac{2}{5} \sum_{i=1}^{m} \langle \mathbf{w}, \mathbf{v}_i \rangle^3 \mathbf{v}_i.$$

The vector is then normalized in Step 3.

Consider the linear transformation $\mathbf{V}^t$, which maps any $\mathbf{w} \in \mathbb{R}^n$ to $\mathbf{w}\mathbf{V}^t \in \mathbb{R}^m$. By Lemma 5.4, the image of $\mathbb{S}_n$ by V_t is included in $\mathbb{S}_m$. For any vector $\mathbf{y} \in \mathbb{R}^m$, we denote by $\mathbf{y}^{[3]} \in \mathbb{R}^m$ the vector obtained by raising to the power 3 all the coordinates of $\mathbf{y}$. Then the equation can be rewritten as :

$$\mathbf{w}_{new} = \frac{2}{5}(\mathbf{w}V^t)^{[3]}\mathbf{V},$$

which implies

$$\mathbf{w}_{new}V^t = \frac{2}{5}(\mathbf{w}\mathbf{V}^t)^{[3]}\mathbf{V}\mathbf{V}^t.$$

This suggests to consider the vector $\mathbf{y} = \mathbf{w}\mathbf{V}^t \in \mathbb{S}_m$. At Step 2, we have $\mathbf{y}_{new} = \frac{2}{5}\mathbf{y}^{[3]}\mathbf{V}\mathbf{V}^t$, and this vector is normalized in Step 3 to belong to $\mathbb{S}_m$. Alternatively, we may write $\mathbf{y}_{new} = \frac{\mathbf{y}^{[3]}}{\|\mathbf{y}^{[3]}\|}\mathbf{V}\mathbf{V}^t$ and its normalization will be the same as before. In other words, the original descent over $\mathbb{S}_n$ can alternatively be viewed as a descent over $\mathbb{S}_m$ with the following iteration :

$$\mathbf{y} = \text{normalize}\left(\frac{\mathbf{y}^{[3]}}{\|\mathbf{y}^{[3]}\|}\mathbf{V}\mathbf{V}^t\right).$$

In the hypercube case, we have $\mathbf{V}\mathbf{V}^t = \mathbf{Id}_n$ and it is easy to see that each iteration increases the largest coordinate of $\mathbf{y}$ (in absolute value), compared to all others. In the zonotope case, there seems to be a similar but more complex phenomenon.

Let $1 \leq i \leq m$ be such that $|y_i| = \max_{1 \leq j \leq m} |y_j|$. If $\mathbf{y}$ is chosen uniformly at random from $\mathbb{S}_m$ each $\mathbb{E}(y_j^2)$ is equal to $1/m$. But it follows from order statistics of the Gaussian distribution that $|y_i|\sqrt{m}$ grows to infinity, and the gap $\min_{j \neq i} |y_i|/|y_j|$ with other coordinates is at least $1 + \Omega(1/\log m)$. After

the transformation $\mathbf{y} = \frac{\mathbf{y}^{[3]}}{\|\mathbf{y}^{[3]}\|}$, y_i is still the largest coordinate of $\mathbf{y}$ in absolute value, and the gap has increased. Let us see what happens after multiplication by $\mathbf{V}\mathbf{V}^t$: let $\mathbf{z} = \mathbf{y}\mathbf{V}\mathbf{V}^t$. By Lemma 5.4, we (heuristically) expect $\|\mathbf{z}\|$ to be close to $\sqrt{n/m}$. Now consider the coordinate z_i of $\mathbf{z}$:

$$z_i = \sum_{j=1}^{m} y_j \langle \mathbf{v}_j, \mathbf{v}_i \rangle = y_i \|\mathbf{v}_i\|^2 + \sum_{j \neq i} y_j \langle \mathbf{v}_j, \mathbf{v}_i \rangle .$$

Therefore :

$$|z_i - y_i \|\mathbf{v}_i\|^2| = \left| \sum_{j \neq i} y_j \langle \mathbf{v}_j, \mathbf{v}_i \rangle \right| .$$

This sum can be viewed as an inner product of two vectors $\mathbf{y}'$ (formed by y_j for $j \neq i$) and $\mathbf{v}'$ (formed by all $\langle \mathbf{v}_j, \mathbf{v}_i \rangle$ for $j \neq i$). And if the angle between $\mathbf{y}'$ and $\mathbf{v}'$ is random, we expect $|\langle \mathbf{y}', \mathbf{v}' \rangle|$ to be less than $O(\|\mathbf{y}'\| \times \|\mathbf{v}'\|/\sqrt{m})$ with high probability (by Lemma 5.10). By definition :

$$\|\mathbf{y}'\|^2 = \|\mathbf{y}\|^2 - y_i^2 = 1 - y_i^2,$$

and

$$\|\mathbf{v}'\|^2 = \left(\sum_{j=1}^{m} \langle \mathbf{v}_j, \mathbf{v}_i \rangle^2 \right) - \langle \mathbf{v}_i, \mathbf{v}_i \rangle^2 = \|\mathbf{v}_i\|^2 - \langle \mathbf{v}_i, \mathbf{v}_i \rangle^2$$

by Lemma 5.4. Therefore :

$$\|\mathbf{y}'\| = \sqrt{1 - y_i^2}$$
$$\|\mathbf{v}'\| = \|\mathbf{v}_i\| \sqrt{1 - \|\mathbf{v}_i\|^2}$$

Hence, we heuristically expect that :

$$z_i = y_i \|\mathbf{v}_i\|^2 + O\left(\frac{\|\mathbf{v}_i\|}{\sqrt{m}} \right),$$

where the left-hand term $y_i \|\mathbf{v}_i\|^2$ dominates if as expected $\|\mathbf{v}_i\| \approx \sqrt{n/m}$. We also expect $z_{i'} \approx y_{i'} \|\mathbf{v}_{i'}\|^2$ for the few other indices i' such that $|y_{i'}|$ is much bigger than $1/\sqrt{m}$. For the remaining coordinates z_j, we expect $|z_j| \approx \sqrt{n}/m$ because heuristically $\|\mathbf{z}\| \approx \sqrt{n/m}$. It follows that z_i is still expected to be the largest coordinate in absolute value. Hence, after sufficiently many iterations, we expect one coordinate of $\mathbf{y}$ to dominate all the others.

5.4 Learning a Deformed Parallelepiped

5.4.1 Breaking the IEEE-IT Countermeasure

In this section, we show that the deformation suggested in [HWH08] is unlikely to prevent the NR attack [NR06]. More generally, we show that the NR attack heuristically still works if the deformation is only *partial*, which means that it preserves at least one of the canonical axes, namely :

Definition 5.2 (Partial Deformation) *A deformation δ is* partial *if there exists at least one index i such that :*
- *for all* $\mathbf{x} \in [-1/2, 1/2)^n$, $[\delta(\mathbf{x})]_i = 0$
- $\delta(\mathbf{x})$ *is independent of* x_i : $(\forall j \neq i, x_j = y_j) \Rightarrow \delta(\mathbf{x}) = \delta(\mathbf{y})$

Such an index i is said to be ignored by the deformation δ.

From its definition in Sect. 5.2.4, it is clear that δ_{IEEE} is partial, it ignores exactly all index $i \notin U$. For the following the details of the construction of the deformation δ_{IEEE} are of no importance, except for the fact that U is a non-empty set.

Our main result is the following

Theorem 5.12 *Let δ be a partial deformation, and i be an index ignored by δ. Let $\mathcal{D} = 2 \cdot d_\delta(\mathcal{U}^n)$ and $\mathbf{M} \in \mathcal{GL}_n(\mathbb{R})$ be an invertible matrix and $\mathbf{G} = \text{Cov}(\mathcal{D} \cdot \mathbf{M})$. Let $\mathbf{L}$ be such that $\mathbf{L}\mathbf{L}^t = \mathbf{G}^{-1}$. Then $\mathbf{r} = \frac{1}{\sqrt{3}} \cdot \mathbf{m}_i \mathbf{L}$ is a local minimum of $\text{mom}_{4,\mathcal{D}'}(\cdot)$ over the unit sphere, where $\mathcal{D}' = \mathcal{D} \cdot \mathbf{M} \cdot \mathbf{L}$.*

Proof: First, we can change the ordering to ensure that $i = 1$. By definition of partiality, for $\mathbf{x} \leftarrow \mathcal{D}$, we have that x_1 is follows the uniform distribution $\mathcal{U}_1^1$ and is and independent of x_j for $j \geq 2$. Thus, the covariance $\mathbf{G}_0 = \mathrm{Cov}(\mathcal{D})$ has the following form :

$$\mathbf{G}_0 = \frac{1}{3}\left[\begin{array}{c|c} 1 & 0 \cdots 0 \\ \hline 0 & \\ \vdots & \mathbf{G}'_0 \\ 0 & \end{array}\right] \text{ and let } \mathbf{L}_0 = \sqrt{3}\left[\begin{array}{c|c} 1 & 0 \cdots 0 \\ \hline 0 & \\ \vdots & \mathbf{L}'_0 \\ 0 & \end{array}\right]$$

where $\mathbf{L}'_0$ is a square root of ${\mathbf{G}'_0}^{-1}$. We have $\mathbf{L}_0\mathbf{L}_0^t = \mathbf{G}^{-1}$, thus $\mathbf{L}_0$ is a square root of $\mathbf{G}_0^{-1}$. By definition, we have $(\mathbf{ML})^t\mathbf{G}_0\mathbf{ML} = \mathbf{Id}_n$ and $\mathbf{L}_0^t\mathbf{G}_0\mathbf{L}_0 = \mathbf{Id}_n$, thus the two distributions $\mathcal{D}'$ and $\mathcal{D}_0 = \mathcal{D} \cdot \mathbf{L}_0$ are orthonormal and equivalent. Hence, there exists an orthogonal matrix $\mathbf{Q}$ such that : $\mathbf{ML} = \mathbf{L}_0\mathbf{Q}$.

We write $\mathbf{r} = (1/\sqrt{3}\ \ 0\ \ \ldots\ \ 0) \cdot \mathbf{ML} = 1/\sqrt{3} \cdot \mathbf{e}_1 \cdot \mathbf{ML}$. This lets us verify that $\|\mathbf{r}\| = 1$ since $\mathbf{r} = 1/\sqrt{3} \cdot \mathbf{e}_1\mathbf{L}_0\mathbf{Q} = \mathbf{e}_1\mathbf{Q}$, where $\mathbf{e}_1$ is unitary and $\mathbf{Q}$ is orthogonal.

We define $\alpha : \mathbf{r}^\perp \to \mathbb{S}_n$, the function that maps the hyperplane of $\mathbb{R}^n$ orthogonal to $\mathbf{r}$ to the unit sphere by $\alpha(\mathbf{v}) = \frac{\mathbf{r}+\mathbf{v}}{\|\mathbf{r}+\mathbf{v}\|}$. It is a local homeomorphism such that $\alpha(\mathbf{0}) = \mathbf{r}$. Thus, it suffices to prove that $\mathbf{0}$ is a local minimum of $\beta = mom_{4,\mathcal{D}} \circ \alpha$.

We note that any vector $\mathbf{v} \in \mathbf{r}^\perp$ and $\mathbf{v}' = \mathbf{v}(\mathbf{ML})^t$ we have $v'_1 = 0 : v'_1 = \langle \mathbf{v}', \mathbf{e}_1 \rangle = \langle \mathbf{v}(\mathbf{ML})^t, \mathbf{e}_1 \rangle = \langle \mathbf{v}, \mathbf{e}_1\mathbf{ML} \rangle = \langle \mathbf{v}, \sqrt{3}\mathbf{r} \rangle = 0$.

With the same notations, for $\mathbf{x} \leftarrow \mathcal{D}$, we have $\langle \mathbf{r}, \mathbf{x} \cdot \mathbf{ML} \rangle = \sqrt{3}x_1$; this is independent from $\langle \mathbf{v}, \mathbf{x} \cdot \mathbf{ML} \rangle = \langle \mathbf{v}', \mathbf{x} \rangle$. This independence leads to :

$$\begin{aligned} \beta(\mathbf{v}) \quad = \frac{1}{\|\mathbf{r}+\mathbf{v}\|^4}\Big(& \mathrm{mom}_{4,\mathcal{D}'}(\mathbf{r}) + 4\,\mathrm{mom}_{3,\mathcal{D}'}(\mathbf{r}) \cdot \mathrm{mom}_{1,\mathcal{D}'}(\mathbf{v}) \\ & + 6\ \mathrm{mom}_{2,\mathcal{D}'}(\mathbf{r}) \cdot \mathrm{mom}_{2,\mathcal{D}'}(\mathbf{v}) \\ & + 4\,\mathrm{mom}_{1,\mathcal{D}'}(\mathbf{r}) \cdot \mathrm{mom}_{3,\mathcal{D}'}(\mathbf{v}) + \mathrm{mom}_{4,\mathcal{D}'}(\mathbf{v})\Big) \end{aligned}$$

Also, $\langle \mathbf{r}, \mathbf{x} \cdot \mathbf{ML} \rangle = \langle (1/\sqrt{3} \cdot \mathbf{e}_1\mathbf{L}_0\mathbf{Q}, \mathbf{x}\mathbf{L}_0\mathbf{Q} \rangle = \langle \mathbf{e}_1, \mathbf{x}\mathbf{L}_0 \rangle = \sqrt{3}x_1$ follows a symmetric distribution, therefore odd-order moments of $\mathcal{D}'$ are null at $\mathbf{r}$. Additionally $\mathcal{D}'$ is orthonormal thus for any vector $\mathbf{a}$, $mom_{2,\mathcal{D}'}(\mathbf{a}) = \|\mathbf{a}\|^2$.

$$\beta(\mathbf{v}) \geq \frac{\mathrm{mom}_{4,\mathcal{D}'}(\mathbf{r}) + 6\,\|\mathbf{v}\|^2}{(1 + \|\mathbf{v}\|^2)^2}$$

It remains to prove that the function $\gamma : x \mapsto \frac{c+6x^2}{(1+x^2)^2} - c$ is positive on some open interval containing 0 for $c = \mathrm{mom}_{4,\mathcal{D}'}(\mathbf{r})$. A routine calculation shows this is true as long as $c < 3$.

$$c = \mathbb{E}_{x \leftarrow \mathcal{D}}\left[\langle \mathbf{r}, \mathbf{x} \cdot \mathbf{ML} \rangle^4\right] = \mathbb{E}_{x \leftarrow \mathcal{D}}\left[(\sqrt{3}x_1)^4\right] = 9/5 < 3$$

□

While this is a strong theoretical argument supporting why the NR attack still works, it is not a full proof, for similar reasons to the zonotope case (see the previous section) : there may be other minima, and we did not prove that the gradient descent efficiently finds minima. A rigorous proof as the one from [NR06] is beyond the scope of this thesis.

Experimental results The attack was run, using 300,000 signatures, to recover the secret key in 80-bit, 112-bit and 128-bit NTRUSign security level settings, and each run led to a secret key recovery, in about two days. No other local minimum was found. Though the samples no longer belong to a set stable by NTRU symmetry group $\mathfrak{S}_N^{\mathrm{NTRU}}$, we may still try to apply the symmetry trick, to multiply the number of samples by N, like in [NR06]. This modifies the distribution of the sample to the average of its orbit : $\mathfrak{S}_N^{\mathrm{NTRU}}(\mathcal{D}) = \sigma(\mathbf{x}) : \mathbf{x} \leftarrow \mathcal{D}, \sigma \leftarrow \mathcal{U}(\mathfrak{S}_N^{\mathrm{NTRU}})$. It turns out that applying the attack on such an averaged distribution leads once again to descents converging to some basis vectors : in fact, by symmetry, all of them are equally likely. The attack used 2,000 signatures, and ran in less than an hour, on the same basis.

Intuitively, this averaging strongly reduces the co-dependence between the coordinates of $\mathbf{x} \leftarrow \mathcal{D}_\sigma$, making the resulting distribution much closer to a parallelepiped than $\mathcal{D}$.

5.5 A Generic Attack against Public Deformations

In this section, we try to extend the NR attack to tackle more general deformation techniques, in the case where the deformation function is publicly known, *i.e.* does not depend on the private key. While this attack does not target any specific proposal, it casts a different light on the NR attack. Rather than a pratical attack claim, the goal of this section is to argue that deformation techniques are likely to be insecure if not properly design : this study may serve as a criterion to assess the security provided by a given deformation.

Additional definitions Two distributions $\mathcal{D}$ and $\mathcal{D}'$ over $\mathbb{R}^n$ are said to be *equivalent* if there exists an invertible matrix $\mathbf{B}$ such that $\mathcal{D}' = \mathcal{D} \cdot \mathbf{B}$: this relation is symmetric, transitive and reflexive. A distribution $\mathcal{D}$ is said to be *isotropic* (resp. *unit-isotropic*) if its covariance matrix is proportional (resp. equal) to the identity matrix : $\mathrm{Cov}(\mathcal{D}) = \lambda I_n$ for some $\lambda \in \mathbb{R}$. Note that for any $w \in \mathbb{R}^+$, $\mathcal{U}_w^n$ is isotropic, and unit-isotropic if $w = \sqrt{3}$. If two distributions $\mathcal{D}$ and $\mathcal{D}'$ are equivalent and $\mathrm{Cov}(\mathcal{D}) = \mathrm{Cov}(\mathcal{D}')$, then there exists an orthogonal matrix $\mathbf{Q} \in \mathcal{O}_n(\mathbb{R})$ such that $\mathcal{D}' = \mathcal{D} \cdot \mathbf{Q}$.

A distribution $\mathcal{D}$ is *unambiguous* if for any invertible matrix $\mathbf{M} \in \mathcal{GL}_n(\mathbb{R})$, $\mathbf{M}$ is uniquely defined by $\mathcal{D} \cdot \mathbf{M}$ up to row permutation and sign change; or equivalently, if $\mathcal{D} \cdot \mathbf{M} = \mathcal{D} \cdot \mathbf{M}'$ implies that $[\mathbf{M}; -\mathbf{M}]$ and $[\mathbf{M}'; -\mathbf{M}']$ share the same rows.

Generalized Problem Statement Due to the generality of the problem, we do not claim any formal result on this generalized algorithm. We make the informal assumption that we can find a polynomial set of reference direction that span the whole space $\mathbb{R}^n$ on the orthonormalized distribution $\mathcal{D}$, via gradient descents. The problem we wish to solve is the following :

Problem 5.4 (The Generalized Hidden Parallelepiped Problem or GHPP) *Let $\mathcal{D}$ be an unambiguous distribution, with an efficient and publicly known sampler. Let $\mathbf{M} \in \mathcal{GL}_n(\mathbb{R})$ be a secret matrix. Given access to an oracle sampling from $\mathcal{D}_1 = \mathcal{D} \cdot \mathbf{M}$, recover an approximation of one of the rows of $\mathbf{M}$.*

In other context, this problem is also called learning a linear transformation. Special instances have been solved in [FJK96], when the entries of $\mathbf{x} \leftarrow \mathcal{D}$ are mutually independent, but this condition is not verified by the deformed distribution of [HWH08], nor by its generalization of Section 5.5.3.

5.5.1 Overview of the Attack

Isotropization of the two distributions. The first issue is that the reference distribution $\mathcal{D}$ may not be orthogonal. However, it can be transformed into a distribution $\mathcal{D}_0 = \mathcal{D} \cdot \mathbf{L}_0$ where $\mathbf{L}_0^{-1}$ is a square root of $G_0 = \mathrm{Cov}(\mathcal{D})$. Since $\mathcal{D}$ is public, and efficiently computable, it is easy to recover an approximation of G_0.

Like in the NR attack, we may also apply this to the samples of $\mathcal{D}'$, in order to obtain $\mathbf{L}$ and $\mathcal{D}' = \mathcal{D}_1 \cdot \mathbf{L} = \mathcal{D} \cdot \mathbf{ML}$ where $\mathcal{D}'$ is orthonormal.

As $\mathcal{D}_0$ and $\mathcal{D}'$ are orthonormal and equivalent, there exist an orthogonal matrix such that $\mathcal{D}' = \mathcal{D}_0\mathbf{Q}$. It is assumed that $\mathcal{D}$ is unambiguous, thus $\mathbf{M} = \mathbf{L}_0\mathbf{Q}\mathbf{L}^{-1}$ (up to row reordering, and row signs).

Equations on Q. As $\mathbf{L}$ and $\mathbf{L}_0$ are known, it only remains to find the orthogonal matrix $\mathbf{Q}$.

In the original parallelepiped case, the NR attack can be explained by the following : since $\mathcal{D}$ is orthogonal, one may take $\mathbf{L}_0 = \alpha I_n$; and in this case, guessing one row $\mathbf{o}_i$ of $\mathbf{Q}$ directly leads to a row of $\mathbf{M}$. To generalize this part of the algorithm, we now need to find the matrix $\mathbf{Q}$ entirely, because of multiplication on the left by a non-diagonal matrix.

We now run the fourth-moment gradient descent on both distributions $\mathcal{D}_0$ and $\mathcal{D}''$. For now, let us simply assume that the fourth-moment minima found over $\mathcal{D}_0$ are a set $\mathcal{M} = \{\mathbf{m}_1, \ldots, \mathbf{m}_n\}$ of n independents vectors of $\mathbb{R}^n$. Note that the moments of any order of a distribution verify : $\mathrm{mom}_{\mathcal{D}\cdot\mathbf{Q},i}(\mathbf{v}) = \mathrm{mom}_{\mathcal{D},i}(\mathbf{v} \cdot \mathbf{Q})$, thus the results of the gradient descent on $\mathcal{D}'$ is an approximation of $\mathcal{M}' = \mathcal{M} \cdot \mathbf{Q}$. If we guess the proper ordering $\mathbf{m}'_1 \ldots \mathbf{m}'_n$ such that $\mathbf{m}_i\mathbf{Q} = \mathbf{m}'_i$ for all i, then we have enough equations to fully recover $\mathbf{Q}$.

Sorting the equations by graph isomorphism. Still, there are exponentially many orderings to try. However we can reduce the search space thanks to the following property : since the matrix $\mathbf{Q}$ is orthonormal, the associated transformation preserves scalar products. This means that we can reduce our search space to orderings verifying $\langle \mathbf{m}'_i, \mathbf{m}'_j \rangle = \langle \mathbf{m}_i, \mathbf{m}_j \rangle$ for all i, j. If this restricted search space has polynomial size and can be efficiently enumerated, it remains to try them all.

This can be seen as a graph (with labeled edges) isomorphism problem : let $\mathcal{G}(\mathcal{S})$ denote the graph whose vertices are the elements of $\mathcal{S}$, and where each edge (s_0, s_1) is labelled by $\langle \mathbf{s}_0, \mathbf{s}_1 \rangle$. The search space is the set of all isomorphisms between $\mathcal{G}(\mathcal{M})$ and $\mathcal{G}(\mathcal{M}')$. While the theoretical hardness of graph isomorphism is still an open question, it is folklore that "random" instances of reasonable size can be easy in practice. So the main problem is the number of such isomorphisms, or equivalently the number of automorphisms of $\mathcal{G}(\mathcal{M})$. In some case, it might be necessary to work modulo the symmetries of $\mathcal{D}$ to shrink the size of this automorphism group.

In practice, it is necessary to weaken the labelling of the graph because of statistical errors.

Target function. In general, there is no reason to use fourth-order moments as a target function to minimize. In fact, one may choose any function of the form $F : \mathbf{v} \mapsto \mathbb{E}_{x \leftarrow \mathcal{D}} [f(\langle \mathbf{x}, \mathbf{v} \rangle)]$, to minimize or maximize, while still preserving the set equation $\mathcal{M} \cdot \mathbf{Q} = \mathcal{M}'$.

Additional vector sorting information. It is possible to add additional information on each result of the descent, to reduce the ambiguity of the graph, by labeling those vertices by the value of $F_{\mathcal{D}}$ at that point, or any other function verifying $F_{\mathcal{D}\cdot\mathbf{Q}}(\mathbf{v} \cdot \mathbf{Q}) = F_{\mathcal{D}}(\mathbf{v})$. Another example could be information about the landscape around the minimum/maximum, like the eigenvalues of the Hessian matrix of $F_{\mathcal{D}}$ at $\mathbf{v}$.

5.5.2 Attack Description

Compass and Graph of distribution. Let $\mathfrak{L}$ denote a set of labels which includes real numbers : $\mathbb{R} \subset \mathfrak{L}$.

Definition 5.3 (Compass) *Let $\mathfrak{C}$ be a function that maps distributions over $\mathbb{R}^n$ to a labeled sets of direction $\mathfrak{C} : (\mathbb{R}^n \to \mathbb{R}^+) \to \mathcal{P}(\mathbb{S}_n \times \mathfrak{L})$. $\mathfrak{C}$ is said to be an (n-dimensional) compass, if for all distributions, and all orthogonal matrices $\mathbf{Q} \in \mathcal{O}_n(\mathbb{R})$:*

$$(\mathbf{v}, \ell) \in \mathfrak{C}(\mathcal{D}) \Leftrightarrow (\mathbf{v} \cdot \mathbf{Q}, \ell) \in \mathfrak{C}(\mathcal{D} \cdot \mathbf{Q})$$

For our purpose, we will of course require that the compass is efficiently computable. Thus we will restrict the choice of the compass to the following class : let $F_{\mathcal{D}} : \mathbf{v} \mapsto \mathbb{E}_{x \leftarrow \mathcal{D}} [f(\langle \mathbf{x}, \mathbf{v} \rangle)]$ for some efficiently computable function $f : \mathbb{R} \to \mathbb{R}$, such that f', the derivative of f is also known and efficiently computable. Let $\ell_{\mathcal{D}} : \mathbb{S}_n \to \mathfrak{L}$ be an efficiently computable labelling function. The associated compass is :

$$\mathfrak{C}_{f,\ell}(\mathcal{D}) = \{(\mathbf{v}, \ell_{\mathcal{D}}(\mathbf{v})) : \mathbf{v} \text{ is a local minima of } F_{\mathcal{D}} \text{ over } \mathbb{S}_n\}$$

During the rest of this section, we consider that f and ℓ are fixed. We assume the existence of an efficient algorithm **ApproxCompass**$(\mathcal{S}, f, f', \ell)$, that given a set $\mathbb{S}$ of sample from $\mathcal{D}$, compute an approximation of $\mathfrak{C}_{f,\ell}(\mathcal{D})$. In practice, the implementation proceeds by gradient descent starting at a random point. Assuming that there is only a polynomial number of minima, and that the set of samples is large enough to find minima, this should be efficient. Once again, we have no formal claim for those properties, and in practice one should adopt an empirical approach to verify that it works.

Definition 5.4 (Labeled Graph) *Let $\mathcal{M} \subset \mathbb{S}_n \times \mathfrak{L}$ be a labeled set of directions. The graph $\mathcal{G}(\mathcal{M})$ is defined by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where :*

- *The set of vertices is : $\mathcal{V} = \{\mathbf{v} : (\mathbf{v}, \ell) \in \mathcal{M}\}$*
- *The set of edges is : $\mathcal{E} = \mathcal{E}_{diag} \cup \mathcal{E}_{angle}$ where $\mathcal{E}_{diag} = \{(\mathbf{v}, \mathbf{v}, \ell) : (\mathbf{v}, \ell) \in \mathcal{M}\}$ and $\mathcal{E}_{angle} = \{(\mathbf{v}, \mathbf{w}, \langle \mathbf{v}, \mathbf{w} \rangle) : (\mathbf{v}, \ell), (\mathbf{w}, \ell') \in \mathcal{M}\}$.*

Clearly, there exists an efficient algorithm to construct $\mathcal{G}(\mathcal{M})$, which we will denote by **BuildGraph**$(\mathcal{M})$.

Theorem 5.13 *Let $\mathcal{D}$ be a distribution over $\mathbb{R}^n$ and let $\mathfrak{C}$ be an n-dimensional compass. Then, for every orthogonal matrix $\mathbf{Q} \in \mathcal{O}_n(\mathbb{R})$, the graphs $\mathcal{G}(\mathfrak{C}(\mathcal{D}))$ and $\mathcal{G}(\mathfrak{C}(\mathcal{D} \cdot \mathbf{Q}))$ are graph-isomorphic.*

Algorithm definition. The following algorithm requires an additional parameter n_S : a number of samples from the reference distribution. We assume that we have an algorithm **EnumGraphIso**$(\mathcal{G}, \mathcal{G}')$ which, given two graphs $\mathcal{G}, \mathcal{G}'$ as input, returns the set $\mathfrak{S}$ of all graph-isomorphisms from $\mathcal{G}$ to $\mathcal{G}'$; this is a list of bijections from the set of vertices of $\mathcal{G}$ to the set of vertices of $\mathcal{G}'$. In practice, any constraint solver might do well, or to ensure the polynomial running time, one might use an ad-hoc algorithm, as in for our Example of Section 5.5.3.

We will also use **SolveLinSystem**$(\mathcal{S})$ which, given a set of pairs $(\mathbf{v}, \mathbf{w}) \in (\mathbb{R}^n)^2$ as input, returns, if is well-defined, the solution $\mathbf{M} \in \mathcal{M}_n(\mathbb{R})$ to the linear system $\{\mathbf{v} = \mathbf{w} \cdot \mathbf{M} : (\mathbf{v}, \mathbf{w}) \in \mathcal{S}\}$.

Algorithm 11 LearnDeformedPar$(\mathcal{D}, \mathcal{S})$

Input: An efficiently sampleable distribution $\mathcal{D}$, a large enough pool of samples $\mathcal{S}$ from distribution $\mathcal{D} \cdot \mathbf{M}$

Output: A set of matrices containing (an approximation of) $\mathbf{M}$

1: $\mathcal{S}_0 \leftarrow \{\mathbf{v}_i \in \mathbb{R}, i \in [n_S]\}$ where $\mathbf{v}_i \leftarrow \mathcal{D}$
2: $\mathbf{L}_0 \leftarrow$ **Isotropize**$(\mathcal{S}_0)$
3: $\mathbf{L} \leftarrow$ **Isotropize**$(\mathcal{S})$
4: $\mathfrak{C}_0 \leftarrow$ **ApproxCompass**$(\mathcal{S}_0 \cdot \mathbf{L}_0, f, f', \ell)$
5: $\mathfrak{C} \leftarrow$ **ApproxCompass**$(\mathcal{S} \cdot \mathbf{L}, f, f', \ell)$
6: $\mathcal{G}_0 \leftarrow$ **BuildGraph**$(\mathfrak{C}_0)$
7: $\mathcal{G} \leftarrow$ **BuildGraph**$(\mathfrak{C})$
8: **for** $\sigma \in$ **EnumGraphIso**$(\mathcal{G}, \mathcal{G}_0)$ **do**
9: $\quad \mathcal{EQ} \leftarrow \{(m, \sigma(m)) : m \in \mathcal{M}_0\}$
10: $\quad \mathbf{Q} \leftarrow$ **SolveLinSystem**$(\mathcal{EQ})$
11: $\quad$ output $\mathbf{L}_0\mathbf{Q}\mathbf{L}^{-1}$
12: **end for**

To clarify all uninstantiated parameters and functions in this algorithm, we provide a fully detailed example in the following section.

5.5.3 Application on a Toy Example

Considering the weakness of the proposal from [HWH08] described in section 5.4, it seems interesting to test our generalized attack on a minimally repaired version of this deformation.

In all this section, we take the deformation δ similar to δ_{IEEE} described in Sect. 5.2.4, replacing the definition of the set U by $U = [N]$. The resulting deformation is not partial anymore.

The chosen Compass is $\mathfrak{C}_{f,f}$ with $f : x \mapsto x^4$, that is the set of $\mathbf{v} \in \mathbb{S}_n$ that are local minima of the fourth-order moment, labelled by the value of the fourth-order moment itself.

The following experiments were done on an 80-bit NTRUSign secret basis, that is $N = 157$. The given constants may vary with the dimension, but the structure of the graph is still the same.

Compass and Graph, Experimental result. We apply the isotropization process to $\mathcal{D} = d_\delta(U^n)$ and obtain the matrix $\mathbf{L}_0$ and $\mathcal{D}_0 = \mathcal{D} \cdot \mathbf{L}_0$ which is istropic.

The gradient descent from random points finds $2n$ minima, that goes by pair $\mathbf{m}$ and $-\mathbf{m}$, since the distribution is invariant by $-I_n$. To simplify the compass and its graph, we forget one vector of each pair, making a choice that keeps all scalar product positive, (this particular case allows it, but it is not possible in general).

To get a reference ordering, we set $\mathbf{m}_i$ to be the closest vector to $\mathbf{e}_i\mathbf{L}_0$ among $\mathcal{M}$. The compass is as follows :

$$\mathfrak{C}_{f,f} = \{(\mathbf{m}_i, c_0) : i \in [N]\} \cup \{(\mathbf{m}_{i+N}, c_1) : i \in [N]\}$$

where $c_0 \approx 0.236$ and $c_1 \approx 0.200$. Now, for the edges of the graph : $\forall i < j$

$$\begin{cases} \langle \mathbf{m}_i, \mathbf{m}_j \rangle = C \approx 0.035 & \text{if } \ i \leq N < j \text{ and } \ (i - j \mod N) \in \{0, 1, 3, 7, 12\} \\ |\langle \mathbf{m}_i, \mathbf{m}_j \rangle| \leq 0.01 & \text{otherwise} \end{cases}$$

In practice, those constants converge well with 3 000 samples (and Transcript Augmentation). This ensure that the attack should succeed with that many samples ; however it has not been fully implemented yet.

Isomorphism Enumeration of the Compass Graph. We are now given the same graph, but with unlabeled edges. The problem is to recover those labels.

We first use the diagonal edge to identify the subset $\mathcal{M}_0 = \{\mathbf{m}_i : i \in [N]\}$: those are the only vertices that have an edge to themselves labelled by c_0, and similarly $\mathcal{M}_1 = \{\mathbf{m}_{i+N} : i \in [N]\}$ with c_1.

Now, we try find the proper order in $\mathcal{M}_0$, by considering the neighbourhood of each point : let $N_k^C(\mathbf{m})$ be the set of points at distance k of $\mathbf{m}$ when considering only edges labeled by C. We have

$$\begin{aligned} N_2^C(\mathbf{m}_i) &= \{\mathbf{m}_{i+l_1-l_2 \mod N} : l_1, l_2 \in \{0,1,3,7,12\}\} \\ &= \{\mathbf{m}_{i+l \mod N} : l \in \pm\{0,1,2,3,4,5,6,7,9,11,12\}\} \\ N_4^C(\mathbf{m}_i) &= \{\mathbf{m}_{i+l_1+l_2 \mod N} : l_1, l_2 \in \pm\{0,1,2,3,4,5,6,7,9,11,12\}\} \\ &= \{\mathbf{m}_{i+l \mod N} : l \in \{-24 \dots 24\}\} \end{aligned}$$

Thus, if $N \geq 51$,

$$\#(N_4^C(\mathbf{m}_i) \cap N_4^C(\mathbf{m}_j)) = 48 \Leftrightarrow i - j = \pm 1 \mod N \tag{5.10}$$

Thanks to this equivalence, if a vertex is known to be labeled by $\mathbf{m}_i$, one may recover the set of 2 vertices that should be labeled by $\mathbf{m}_{i-1}$ and $\mathbf{m}_{i+1}$.

Now, let us take a vector in $\mathcal{M}_0$; and assume it is $\mathbf{m}_1$ (there is N possible choices). Use (5.10) to find $\mathbf{m}_2$ among the 2 possible choices. By induction, use (5.10) on $\mathbf{m}_i$ to find $\mathbf{m}_{i+1}$, choosing the one that is different from $\mathbf{m}_{i-1}$.

Now that $\mathcal{M}_0$ is fully ordered, it is easy to order $\mathcal{M}_1$: $\mathbf{m}_{N+i}$ is the only vertex connected to all vertices $\mathbf{m}_{i-l}$ for $l \in \{0,1,3,7,12\}$, by C-labelled edges.

In total, we have made one binary choice, and one choice among N, thus, the search space has size $2N$. It is efficiently enumerable, since measuring distance and enumerating neighbourhood can be done in polynomial time.

Symmetry discussion. One may note that the distribution $\mathcal{D}$ has much more symmetries : a subgroup of size 2^n corresponding to reflexions along any canonical vector $\mathbf{e}_i$. This subgroup has been "quotiented" from the graph by forgetting one element of each pair $\mathbf{m}, -\mathbf{m}$. The second subgroup is $\mathfrak{S}_N^{\mathrm{NTRU}}$, which has size N. It corresponds to the choice of $\mathbf{m}_0$ among $\mathcal{M}_0$: the choice of $\mathbf{m}_0$ is not important.

However, the graph has an additional symmetry, that sends the coordinate $\mathbf{m}_i$ to $\mathbf{m}_{N-i}$ for $i \in [N]$ and $\mathbf{m}_{i+N}$ to $\mathbf{m}_{2N-i}$ for $i \in \{N+1 \dots 2N\}$. This symmetry is not induced by the symmetries of $\mathcal{D}$; this means that only one choice out of two for $\mathbf{m}_2$ will lead to the desired isomorphism.

5.6 Conclusion and Open Problems

For a conclusion, we discuss some open problems related to the results presented in this chapter.

Full Proof of the Attack. A frustrating aspect of this work, is that we were not yet able to prove the correctness and the efficiency of our generalized algorithm ; that is proving that the descent does converge towards one of the minima of interest ; our arguments remain heuristic, but are supported by experiments. In particular, we observed that there were never more than $2m$ (precisely, m opposite pairs) different points to whichthe descent converges to. Leaving the convergence speed problem apart, the natural question is to count, for an orthogonal zonotope $\mathcal{Z} \subset \mathbb{R}^n$, the number of local minima of the function $\mathrm{mom}_{\mathcal{Z},4}$ over the unit sphere $\mathbb{S}_n$. The problem can be restated geometrically as follows

Open Problem 5.1 *Let $\mathcal{H} \subset \mathbb{R}^m$ be an hyperplane of dimension n. How many local maxima does the ℓ_4 norm $\mathbf{x} \mapsto \|\mathbf{x}\|_4 = \sqrt[4]{\sum x_i^4}$ has over the domain $\mathcal{H} \cap \mathbb{S}_m$, the intersection of the hyperplane $\mathcal{H}$ with the ℓ_2-unit ball of $\mathbb{R}^m$?*

One extreme case, $m = n$ (that is $\mathcal{H} = \mathbb{R}^m$), is already covered by the work of Nguyen and Regev, for which we always have exactly $2m$ local maxima, that are actually also global maxima. On the opposite side, it is not hard to prove that, if $n = 2$ and for any $m \geq 2$, then there are no more than 4 strict local maxima ; in this case the function is studied over a circle, and a simple Fourier decomposition argument leads to this result. In between, the question has resisted our efforts. Another interesting point is that, if we replace the ℓ_4 norm by the ℓ_∞ norm ; then it becomes easy again to prove that there is no more than $2m$ local maxima over $\mathcal{H} \cap \mathbb{S}_m$: the sphere $\mathbb{S}_m$ can easily be splitted into $2m$ convex domain on which $\|\cdot\|_\infty$ is a convex function.

Considering that the attack works in practice, this question might seem anectdotic ; yet ongoing research on the provable security side showed that such zonotopic distribution also arise in other contexts ; we therefore believe than any theoretical result improving our understanding of those distributions would be valuable.

Limits of the Attack. From our experiments, we have noticed that the attacks does succeed for more than one perturbation, yet the amount of samples needed and the running time seems to grow quite rapidely with the number of perturbations. The more perturbation, the less deep the maxima, the less steep the basin, and the slowest convergence. At the limit $m \to \infty$, by the central limit theorem, we expect the distribution to be very close to a Gaussian, and the moments becomes very close to constant over the unit sphere. A stronger limit can be suggested by an algebraic argument. We try to recover a matrix in $\mathbf{B} \in \mathbb{R}^{m \times n}$ from the function $\mathrm{mom}_{\mathcal{Z}(\mathbf{B})}$; which is an n-variate polynomial of order 4 ; that is a space of dimension less than n^4. Therefore, if $m > n^3$ (that is, for more than n^2 perturbations), the set of matrices $\mathbf{B}' \in \mathbb{R}^{m \times n}$ such that $\mathrm{mom}_{\mathcal{Z}(\mathbf{B}'),4} = \mathrm{mom}_{\mathcal{Z}(\mathbf{B}),4}$ will be in general (that is except in singular cases) a manifold of dimension strictly greater than 0 ; in other words there will be infinetly many solutions considering only 4^{th} order moments.

Our work doesn't completly rules out the practical security of the perturbation technique, but it suggest that one would need a large amout of perturbation to avoid any statistical attacks.

Security without Gaussian Sampling ? The attack of Section 5.5 has many necessary conditions to succeed, but it seems hard to guarantee that there are no choice of Compass that solves it, if the distribution is unambiguous.

On the contrary, an ambiguous distribution is likely to make the problem hard : if the symmetry group $\mathrm{Sym}(\mathcal{D})$ is much larger than $\mathrm{Sym}(U^n)$, many matrices $\mathbf{M} \in \mathcal{M}_n(\mathbb{R})$ are valid, and they form some orbit of a large symmetry group.

One may recover a valid $\mathbf{M} \in \mathcal{M}_n(\mathbb{R})$ matrix, in this orbit, in the cryptographic context, he then needs to finds the true secret key among the orbit. His only chance of success would use the fact that in the cryptographic context, the actual secret key matrix has integer entries ; this might be possible considering the techniques of [GS02].

At the very limit of this approach is the Gaussian Sampling : if we choose $\mathcal{D}$ to be a spheric Gaussian distribution (or any spheric distribution) ; $\mathrm{Sym}(\mathcal{D})$ is the whole orthogonal group, and that prove that no more than the gram matrix of $\mathbf{M}$ is revealed. At this point it seems that it makes more sense to use Klein's algorithm [Kle00, GPV08], or the Gaussian Perturbation of Peikert [Pei10] (especially in the light of the attack of [GS02]) that provably hide entirely the matrix $\mathbf{M}$ However Gaussian-Sampling algorithm are still one order of magnitude slower than desired, and provide not-so-close vectors.

Chapter 6

Discrete Gaussian Sampling with Floating Point Arithmetic

Résumé

Ce chapitre reprend de façon plus détaillé les résultats de l'article *Faster Gaussian Lattice Sampling using Lazy Floating-Point Arithmetic*, co-signé avec P. Nguyen et publié à Asiacrypt 2012.

Pour être prouvablement sûre, et en particulier empêcher toute attaque par apprentissage tel que présentée dans le chapitre précédent, de nombreuses primitives cryptographiques à base de réseau nécessitent un algorithme efficace pour tirer des points aléatoires d'un réseau selon une distribution Gaussienne. Tous les algorithmes connus pour cette tâche ont recours à un moment ou un autre à des opérations arithmétiques sur de longs entiers. Dans ce chapitre, nous étudions à quel point ces tirages aléatoires peuvent être accélérés par l'utilisation de l'arithmétique flottante. En premier lieu, nous montrons qu'une mise en oeuvre directe de l'arithmétique flottante n'apporte pas de gain asymptotique : la précision des flottants se doit d'être linéaire en la dimension pour garantir la sécurité, menant à une complexité totale de $\tilde{\mathcal{O}}(n^3)$, où n désigne la dimension du réseau. Cependant, nous montrons que la complexité de ces algorithmes peut tomber à $\tilde{\mathcal{O}}(n^2)$ en les rendant *paresseux*, voir jusqu'à $\tilde{\mathcal{O}}(n)$ dans certains cas utiles en cryptographies. De plus notre analyse est concrète et pratique : pour des paramètres typiques, nos algorithmes paresseux effectuent la plupart de leurs opérations flottantes en double-precision définie par la norme IEEE.

Il est cependant fort probable que les résultats presentés dans le chapitre suivant (section 7.2) puissent ameliorer ces travaux, et qu'ils soit possible, en utilisant des techniques de rejet, d'atteindre une efficacité asymptotique comparable, sans recourir a l'arithmetique flottante.

Abstract

This chapter is a detailed version of the article *Faster Gaussian Lattice Sampling using Lazy Floating-Point Arithmetic*, coauthored with P. Nguyen published at Asiacrypt 2012.

To be provably secure, and in particular prevent the learning attacks such as the one presented in the previous chapter, many lattice cryptographic primitives require an efficient algorithm to sample lattice points according to some Gaussian distribution. All algorithms known for this task require long-integer arithmetic at some point, which may be problematic in practice. We study how much lattice sampling can be sped up using Foating-point arithmetic. First, we show that a direct Floating-point implementation of these algorithms does not give any asymptotic speedup : the Floating-point precision needs to be greater than the security parameter, leading to an overall complexity $\tilde{\mathcal{O}}(n^3)$ where n is the lattice dimension. However, we introduce a laziness technique that can signifficantly speed up these algorithms. Namely, in certain cases such as NTRUSign lattices, *laziness* can decrease the complexity to $\tilde{\mathcal{O}}(n^2)$ or even $\tilde{\mathcal{O}}(n)$ in certain cryptographic relevant cases. Furthermore, our analysis is concrete practical : for typical parameters, most of the Floating-point operations only require the double-precision IEEE standard.

It is quite possible that the results presented in the next chapter (section 7.2) could subsume this work, and that one could, using rejection techniques, reach similar complexity without the need for floating-point arithmetique.

6.1 Introduction

In the previous chapter we shown that it is essential to avoid leaking statistical information related to the secret key; doing so is yet much more complicated in the field of lattice-based cryptography than cryptography over finite groups, as RSA related schemes or elliptic curves based schemes. Indeed, for finite field, it is usually possible to avoid leakage by making the distribution uniform over the group; for lattice-based cryptography however, the legitimate party needs to output *small elements*, therefore forbidding uniform distribution.

In the GGH and NTRUSign signature schemes, the secret short basis was used as a trapdoor to solve approx-CVP problem, and the various NTRUSign countermeasures were added in an attempt to make the statistical leakage unusable. It is only much later than a provable solution will be found, namely Gaussian Sampling; in 2008 Gentry *et al.* [GPV08] prove that drawing solutions to the approx-CVP problem according to a Discrete Gaussian leaks no information about the geometry of the lattice. An algorithm to do so (we will call them Gaussian Sampler) already existed [Kle00] due to Klein; it is essentially a variant of Babai nearest plane algorithm that replaces deterministic rounding to the closest integer steps by randomized rounding to one of the closest integers.

This kind of trapdoors rapidly became standard for lattice-based constructions [CHKP10, ABB10a, AFV11, Boy13], allowing signatures but also many extensions such as IBE, HIBE ... Yet, the efficiency (running time ($\tilde{\mathcal{O}}(n^3)$)) of Klein's algorithm is not satisfactory for cryptographic purposes, and alternative algorithms were proposed [Pei10, MP12] ; those algorithms are related to Babai round-off algorithm and therefore have better efficiency, but worse quality (the outputted CVP solution is longer). Yet, all these algorithms require long-integer arithmetic to deal with rational numbers at some point, which may be problematic in practice. Although the descriptions usually mention that one can replace these real numbers by approximations with sufficiently high precision, which guarantees efficiency in an asymptotic sense, the practical impact is unclear : no article in the lattice cryptography literature seems to specify exactly which precision one should take, and how operations will be performed exactly. This was not an issue when lattice-based cryptography was considered to be mostly of theoretical interest, but recent works [MR09, Pei10, LP11, RS10, Lyu12, MP12] suggest that the time has come to assess the practicality of lattice-based constructions.

The focus of this chapter is therefore to analyze those algorithms and possibly to improve on them, both in term of asymptotic and practical efficiency.

Gaussian Sampling over arbitrary lattices The cost of Klein's algorithm is the same as Babai's algorithm, namely $\mathcal{O}(n^3 \log B)$ (or $\mathcal{O}(n^4 \log^2 B)$ without fast integer arithmetic), where n is the lattice dimension, and B is the maximal norm of the input basis vectors : since B is polynomial in n for trapdoor bases used in lattice cryptography, the usual cost is $\tilde{\mathcal{O}}(n^3)$ (or $\tilde{\mathcal{O}}(n^4)$ without fast integer arithmetic). The main reason behind the cost of Klein's algorithm is the use of long-integer arithmetic : it relies on Gram-Schmidt orthogonalization, which involves rational numbers of bit-length $\mathcal{O}(n \log B)$. A natural way to improve the efficiency is to use floating-point arithmetic (FPA) to replace exact Gram-Schmidt by suitable approximations. Indeed, Klein's algorithm is a variant of Babai's nearest plane algorithm, which itself is simply the size-reduction subroutine used extensively in the LLL algorithm [LLL82] ; and floating-point arithmetic is classically used to speed up LLL (see [Sch88, NS09, MSV09]). But the use of FPA is not straightforward, and it is unclear at first sight how much speed can be gained, if any.

On the other hand, the convolution algorithms of [Pei10, MP12] have two phases : an offline phase (depending on the secret basis only) and an online phase (depending on the target vector). The online phase costs $\tilde{\mathcal{O}}(n^2)$ for q-ary lattices (which are widespread in lattice cryptography), or even $\tilde{\mathcal{O}}(n)$ in the so-called ring setting (*i.e.* special lattices such as NTRU lattices); but the offline phase seems to have the same cost $\tilde{\mathcal{O}}(n^3)$ as Klein's algorithm and involves floating-point arithmetic whose exact cost is not analyzed in [Pei10, MP12]. Both algorithms can use the same offline phase, which will later be referred as Peikert's offline Algorithm. Note that the *offline phase is not a precomputation* : this phase must be *repeated before each sampling*, which is reminiscent of DSA one-time pairs (k, k^{-1}), which can be precomputed as coupons or generated online; but unlike a precomputation it should not be re-used. In some scenarios, this computational cost might be acceptable, but it is clearly valuable to analyze and improve the offline phase.

We develop techniques to improve all three general lattice samplers [GPV08, Pei10, MP12],which provides the first algorithms with quasi-optimal complexity to sample the spherical Gaussian distribution over lattices : they run in quasi-linear time in the size of the input secret basis. More precisely, our optimized variant of Klein's algorithm runs in $\tilde{\mathcal{O}}(n^2)$ and our variant of Peikert's offline algorithm

runs in average time $\tilde{\mathcal{O}}(n)$ in some Ring-Setting (where n is the lattice dimension). In both cases, our improvements do not introduce any loss of quality.

To do so, we study how much lattice sampling can be sped up using FPA. As a starting point, we present FPA variants of Klein's algorithm with statistically close output. Surprisingly, this basic FPA variant has the same asymptotic complexity $\tilde{\mathcal{O}}(n^3)$ as Klein's algorithm ; the reason is that the precision needs to be at least linear in the security parameter. Still, we also present an optimized algorithm with an improved complexity $\tilde{\mathcal{O}}(n^2)$: it is based on a so-called laziness technique which combines high and low precision FPA. This optimized complexity only applies to a special class of bases that include NTRUSign bases [HNHGSW03].

Next, we show that the same optimization can be used to speed up Peikert's offline algorithm, improving the total complexity, to bring its offline complexity down to that of its online complexity for both sampling algorithms of [Pei10, MP12]. More precisely, we apply our laziness technique to reduce the offline complexity to $\tilde{\mathcal{O}}(n^2)$. And for certain Ring-Settings (precisely when the ring is $\mathcal{R} = X^b \pm 1$), we show that the offline phase can also be sped up to average quasi-linear time.

Practical Impact of Laziness The practical challenge we are faced with when dealing with real numbers is to make the precision requirement fit the architecture. Ideally, one would want to use *on-silicon* implementation of floating-point arithmetic, so as to spend only one cycle on each operation. As soon as we exceed such requirements, each multiplication needs at least 5 cycles : 3 word-multiplications and 1 word-addition, and a few cycles for software management of carry and renormalization. Then times starts growing quadratically[1] with the precision requirement.

Our complexity results are stated in an asymptotic manner, but our study gives concrete bounds, and show that for crypto-grade lattices, the standard *double-precision* (53-bits) is largely enough as the "low precision" of our Lazy Algorithm, which means that each operation can fit in one cycle of standard desktop/laptop CPU. Our study also suggests that more basis ad-hoc bounds might lead to *single-precision*, so as to fit GPU or other small architectures such as smart-phones.

We measure the practical speed-up of laziness by comparing the running time of `quad float` and `double float` on an Intel I5 CPU, and we measure a ratio between 10 and 20, depending on implementation[2], rather than the optimistic ratio 5. We thus claim that Laziness technique gives a 10- to 20-fold speed-up, and even more when the security parameter grows.

Roadmap We start in Sect. 6.2, we present our basic FPA variant of Klein's algorithm, which we optimize using laziness in Sect. 6.3. In Sect. 6.4, we apply the same optimization to Peikert's Offline Algorithm, and in Sect. 6.5 we detail how to reach quasi-linear time complexity in the ring setting. Eventually, in Sect. 6.6 we apply our results to cryptographically-relevant cases, and give practical mantissa size requirements. Sections 6.7, 6.8 are devoted to technical lemata and proofs of our main theorems, while section 6.9 provides the concrete versions of our main theorems.

6.2 A Basic Floating-Point Variant of Klein's Algorithm

6.2.1 Notation

For the rest of this chapter, we will work with a fixed basis $\mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_n] \in \mathbb{R}^n$ of a full rank lattice $L \in \mathbb{R}^n$. We use $\mathbf{B}^\star = [\mathbf{b}_1^\star \dots \mathbf{b}_n^\star] \in \mathbb{R}^n$ for the Gram-Schmidt orthogonalization of that basis, and μ will denote the lower-triangular transformation matrix, that is such that $\mathbf{B} = \mu \mathbf{B}^\star$.

For any $\sigma > 0$, we let $\sigma_i = \sigma / \|\mathbf{b}_i^\star\|$ and $\hat{\sigma} = \max_{i=1}^n \sigma_i$. Since the $\mathbf{b}_i^\star$'s are orthogonal, we have $\{s_i(B^\star)\} = \{\|\mathbf{b}_i^\star\|\}$, therefore $\hat{\sigma} = \sigma / (\min_{i=1}^n \|\mathbf{b}_i^\star\|) = \sigma / s_n(B^\star) = \sigma \left\|B^{\star -1}\right\|_s \leq \sigma \left\|B^{-1}\right\|_s \|\mu\|_s \leq \sigma \left\|B^{-1}\right\|_s n\hat{\mu}$ where $\hat{\mu} \geq 1$ upper bounds the coefficients of μ.

6.2.2 Floating-Point Arithmetic

We consider floating-point arithmetic (FPA) with m bits of mantissa, which we denote by $\mathbb{FP}_m$: the precision is $\epsilon = 2^{-m+1}$. A floating-point number $\bar{f} \in \mathbb{FP}_m$ is a triplet $\bar{f} = (s, e, v)$ where $s \in \{0, 1\}$, $e \in \mathbb{Z}$

1. until other algorithms such as Karatsuba becomes faster than naïve multiplication, which happens at precision about 10 times the architecture size : at this point the battle for practicality is already lost.

2. The `gcc` compiler (version $\geq$ 4.6) natively includes `quad float` type, following the IEEE standard (113 bits of precision). We measure ratio 30 for multiplication and 10 for addition, that is a factor 20 averaging over the whole algorithm. The `NTL` library also includes non-standard `quad float`, following the "`double double`" technique, giving 106 bits of precision. We measure a factor 16 for both operations.

and $v \in \mathbb{N}_{2^m-1}$, which represents the real number $R(\bar{f}) = (-1)^s \cdot 2^{e-m} \cdot v \in \mathbb{R}$. Every FPA-operation $\bar{\circ} \in \{\bar{+}, \bar{-}, \bar{\times}, \bar{/}\}$ and its respective arithmetic operation on $\mathbb{R}$, $\circ \in \{+, -, \cdot, /\}$ verify :

$$\forall \bar{f}_1, \bar{f}_2 \in \mathbb{FP}_m, \left|R(\bar{f}_1 \bar{\circ} \bar{f}_2) - (R(\bar{f}_1) \circ R(\bar{f}_2))\right| \leq (R(\bar{f}_1) \circ R(\bar{f}_2))\epsilon \tag{6.1}$$

We require a floating-point implementation of the exponentiation function $\bar{\exp}(\cdot)$ and we assume that it verifies a similar error bound : for any $\bar{f} \in \mathbb{FP}_m$, $\left|R(\bar{\exp}(\bar{f})) - \exp(R(\bar{f}))\right| \leq \epsilon$. Finally, we note that if an integer $x \in \mathbb{Z}$ verifies $|x| \leq 2^m$, it can be converted to a float $\bar{f} \in \mathbb{FP}_m$ with no error, i.e. $R(\bar{f}) = x$. For the rest of the chapter, we omit the function R and consider $\mathbb{FP}_m$ as a subset of $\mathbb{R}$.

6.2.3 Typed Pseudo-code

For this chapter, we will make use of typed pseudo-code to clearly differentiate the various representation of numbers we are using. We also introduce several useful primitives.

Types Variables are typed, and the type is given at each initialization and assignment, as follows : *variable* $\leftarrow$ *value* : *type*. We use a simpler syntax for the definition of local functions : $\{$*variable* $\mapsto$ *value*$\}$. Functional types are denoted by $(t_1 \to t_2)$.

Primitives We use the basic arithmetic operations $\{+, -, \cdot, /\}$, as well as squaring $\square^2$ and exponentiation exp ; the arguments are either integers in $\mathbb{Z}$, or floating-point numbers in $\mathbb{FP}_m$. We extend these notations to vectors and matrices. We also use the following additional primitives :
RandInt$(a, b) : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$: return a random uniform integer in the range $[a, b]$.
RandFloat$_m() :$ void $\to \mathbb{FP}_m$: return a random uniform float in the range $[0, 1)$.
ExtRandFloat$_{m',m}(r) : \mathbb{FP}_{m'} \to \mathbb{FP}_m$: return a random uniform floating-point number in the range $[r, r+2^{-m'})$. For a random $r \leftarrow$ **RandFloat**$_{m'}()$, the output follows the same distribution as **RandFloat**$_m()$.

6.2.4 Description

The goal of Gaussian lattice sampling is to efficiently sample lattice points according to a distribution statistically close to $D_{L,\sigma,\mathbf{c}}$. All lattice samplers known [Kle00, GPV08, Pei10, MP12] have constraints on the parameter σ and the statistical distance, which are related to the so-called smoothing parameter. The sampling parameter σ determines the average distance of the sampled lattice point to the target point : the smaller σ, the better for cryptographic applications. For instance, σ impacts the verification threshold of lattice-based signatures [GPV08] and therefore the security of the scheme ; a lower quality forces to increase lattice parameters. And for a security level of λ bits, we need a statistical distance less than $2^{-\lambda}$.

Klein's sampling algorithm rely on a 1-dimensional sampling subroutine, that given a center $c \in \mathbb{R}$ and a variance σ, outputs an integer according to $D_{\mathbb{Z},\sigma,c}$; for the floating point variant the main issue will be the precision at which the center c has been computed.

Algorithm 13 describes both Klein's algorithm [Kle00] and our basic floating-point variant : given a basis $\mathbf{B}$ of a lattice L, a target $\mathbf{c}$ and a parameter σ, the algorithm outputs a vector with distribution statistically close to $D_{L,\sigma,\mathbf{c}}$. It uses two subroutines : **DecomposeGS**$_m$ (Alg. 14) to compute the coordinates t_i's of the target vector $\mathbf{c}$ with respect to the Gram-Schmidt basis $\mathbf{B}^\star$, and **Sample**$\mathbb{Z}_m$ (Alg. 12) to sample according to the Gaussian distribution over $\mathbb{Z}$. Algorithm 13 comes in two flavors :

- **SampleLattice**$_\infty$ is the exact version, which corresponds to Klein's original algorithm [Kle00]. The $\mu_{i,j}$'s and the t_i's are represented exactly by rational numbers, and all the computations use exact integer arithmetic. Assuming $\sigma \in \mathbb{Q}$, we can only ensure that $\sigma_i \in \sqrt{\mathbb{Q}}$, thus we can represent them exactly by their square. We also assume that this version has access to a perfect primitive (or an oracle) **Sample**$\mathbb{Z}_\infty(\sigma_i, t_i, \tau = \infty)$ that given $t_i, \sigma_i^2 \in \mathbb{Q}$ answers an integer $x : \mathbb{Z}$ exactly according to the distribution $D_{\mathbb{Z},\sigma_i,t_i}$. It does not matter how to sample such a perfect distribution, as the purpose of this perfect algorithm is to be a reference for inexact ones.
- **SampleLattice**$_m$ is our basic floating-point version, using $\mathbb{FP}_m$. The matrices μ and $\mathbf{B}^\star$ and values σ_i may have been pre-computed exactly, but only approximations are stored.

The description of **SampleLattice**$_\infty$ differs from the original description [Kle00, GPV08] only in the way we compute and update the coordinates t_i's. In our version, the final value of t_i before it is used is

Algorithm 12 Sample$\mathbb{Z}_m$: Rejection Sampling for Discrete Gaussian on $\mathbb{Z}$

Input: A center $t : \mathbb{FP}_m$, and a parameter $\sigma : \mathbb{FP}_m$, and a tailcut parameter $\tau : \mathbb{FP}_m$
Output: output $x : \mathbb{Z}$, with distribution statistically close to $D_{\mathbb{Z},t,\sigma}$

1: $h \leftarrow -1/2\sigma^2 : \mathbb{FP}_m$; $x_{\max} \leftarrow \lceil t + \tau\sigma \rceil : \mathbb{Z}$; $x_{\min} \leftarrow \lfloor t - \tau\sigma \rfloor : \mathbb{Z}$
2: $x \leftarrow$ **RandInt**$(x_{\min}, x_{\max}) : \mathbb{Z}$; $p \leftarrow \exp(h \cdot (x-t)^2) : \mathbb{FP}_m$
3: $r \leftarrow$ **RandFloat**$_m() : \mathbb{FP}_m$; **if** $r < p$ **then** return x
4: **Goto** Step 2.

Algorithm 13 SampleLattice$_m$: Gaussian Sampling over a lattice

Input: a (short) lattice basis $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_n) : \mathbb{Z}^{n \times n}$, parameter $\sigma : \mathbb{FP}_m$, A target vector $\mathbf{c} : \mathbb{Z}^{1 \times n}$, and a tailcut parameter $\tau : \mathbb{FP}_m$ **Precomputation :** The GS decomposition $(\mathbf{B}^\star = (\mathbf{b}_1^\star, \ldots, \mathbf{b}_n^\star), (\mu_{i,j}) = (\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_n))$, norms $r_i = \|\mathbf{b}_i^\star\| : \mathbb{FP}_m$ and $\sigma_i = \sigma / r_i : \mathbb{FP}_m$
Output: a vector $\mathbf{v} : \mathbb{Z}^{1 \times n}$ drawn approximately from $D_{L,c,\sigma}$ where $L = \mathcal{L}(\mathbf{B})$

1: $\mathbf{v}, \mathbf{z} \leftarrow \mathbf{0} : \mathbb{Z}^n$; $\mathbf{t} \leftarrow$ **DecomposeGS**$_m(\mathbf{c}, B^\star) : \mathbb{FP}_m$
2: **for** $i = n$ downto 1 **do**
3: $z_i \leftarrow$ **Sample**$\mathbb{Z}_m(\sigma_i, t_i, \tau) : \mathbb{Z}$
4: $\mathbf{v} \leftarrow \mathbf{v} + z_i \cdot \mathbf{b}_i : \mathbb{Z}^n$; $\mathbf{t} \leftarrow \mathbf{t} - z_i \cdot \boldsymbol{\mu}_i : \mathbb{FP}_m^n$
5: **end for**
6: return $\mathbf{v}$

$t_i = \langle \mathbf{c}, \mathbf{b}_i^\star \rangle / r_i^2 - \sum_{j>i}^n z_j \mu_{j,i}$, which matches with the original value :

$$t_i' = \left\langle \mathbf{c} - \sum_{j>i}^{n} z_j \mathbf{b}_j \ , \ \mathbf{b}_i^\star \right\rangle / r_i^2 = \left(\langle \mathbf{c}, \mathbf{b}_i^\star \rangle - \sum_{j>i}^{n} z_j \langle \mathbf{b}_j, \mathbf{b}_i^\star \rangle \right) / r_i^2 = t_i$$

We unroll this computation and update the sum after each value z_i is known. This allows a parallelization up to n processors without the usual $\log n$ factor required for summing up all terms.

Since we use the matrix μ in the main loop, we might want to get rid of $B^\star$ for the **DecomposeGS** algorithm, to save some precomputation and storage, by computing $\mathbf{c}' \leftarrow \mathbf{c} \cdot B^T$ and then solving the triangular system $\mathbf{y}\, \mu^T = \mathbf{c}'$. Solving this system also requires n^2 operations, however when using FPA, it would produce a relative error exponential in the dimension n, because we recursively use previous results.

Our main loop may also be seen as solving a triangular system, where we apply Gaussian rounding at each step. It is worth noting that this additional rounding prevents such relative exponential error, as our proof will show.

Correctness of the exact algorithm SampleLattice$_\infty$ Gentry *et al.* showed in [GPV08] that given as input a lattice basis $\mathbf{B}$ of an n-dimensional lattice L such that $\sigma \geq \|\mathbf{B}^\star\| \cdot \omega(\sqrt{\log n})$, Klein's algorithm [Kle00] outputs lattice points with a distribution statistically close to $D_{L,\sigma,\mathbf{c}}(\mathbf{x})$. For applications, it is more convenient to have a concrete bound on the statistical distance, and to separate this bound from the lattice dimension n. We therefore use the following concrete analysis of Klein's algorithm :

Theorem 6.1 (Concrete version of [GPV08, Th. 4.1]) *Let $n, \lambda \in \mathbb{N}$ be any positive integers, and $\iota = 2^{-\lambda}/(2n)$. For any n-dimensional lattice L generated by a basis $\mathbf{B} \in \mathbb{Z}^{n \times n}$, and for any target vector $\mathbf{c} \in \mathbb{Z}^{1 \times n}$, Alg. 13 is such that the statistical distance $\Delta(D_{L,\sigma,\mathbf{c}}, \mathbf{SampleLattice}_\infty(\mathbf{B}, \sigma, \mathbf{c}))$ is less than $2^{-\lambda}$, under the condition :*

$$\sigma \geq \|\mathbf{B}^\star\| \cdot \eta_\iota(\mathbb{Z}) / \sqrt{2\pi} \quad \textit{where} \ \ \eta_\iota(\mathbb{Z}) \approx \sqrt{(\lambda \ln 2 + \ln n)/\pi} \ .$$

In the next chapter we will give the proof of a more general statement (Theorem 7.7).

Efficiency of SampleLattice$_\infty$ The algorithm **SampleLattice$_\infty$** performs $\mathcal{O}(n^2)$ arithmetic operations on rational numbers of size $\mathcal{O}(n \log B)$, which leads to a complexity of $\tilde{\mathcal{O}}(n^4)$ for cryptographic use. Here, we ignored the calls to the oracle **Sample**$\mathbb{Z}_\infty(\cdot, \cdot, \tau = \infty)$.

Algorithm 14 DecomposeGS$_m$: Decompose a vector c over the GS Basis

Input: A vector $\mathbf{c} : \mathbb{Z}^{1\times n}$, an orthogonal basis $B^\star = (\mathbf{b}_1^\star, \ldots, \mathbf{b}_n^\star) : \mathbb{Q}^{n\times n}$, and $r_i^2 = \|\mathbf{b}_i^\star\|^2 \in \mathbb{FP}_m$
Output: output $\mathbf{t} : \mathbb{Q}^n$ such that $c = t_1\mathbf{b}_1^\star + \cdots + t_n\mathbf{b}_n^\star$
1: $\mathbf{y} \leftarrow \mathbf{c} \cdot B^{\star T}$ $\quad : \mathbb{Z}^{1\times n}$
2: return $(y_1/r_1^2, \ldots, y_n/r_n^2)$

Termination of Sample$\mathbb{Z}_\infty(\cdot,\cdot,\tau<\infty)$ We upper bound the number of trials of Rejection Sampling, ignoring issues related to the transcendental function exp :

Fact 6.2 *If $\sigma \geq 1$ and $\tau \geq 4$, and uniforms $x \leftarrow \mathbb{Z}\cap[x_{min}, x_{max}]$ and $r \leftarrow [0,1)$, we have $\Pr[r < \rho_{\sigma,t}(x)] > 1/(6\tau)$ where $x_{min} = \lceil t - \tau\sigma \rceil$ and $x_{max} = \lfloor t + \tau\sigma \rfloor$.*

Proof: For a random choice of $x \in [x_{\min}, x_{\max}]$, the probability that $|x - t| < \sqrt{2/\pi}\cdot\sigma$ is at least $.6/\tau$. In such a case, we have $p \geq e^{-\pi/4}$. Thus, at each iteration, the probability of termination is greater than $\frac{.6\cdot e^{-\pi/4}}{\tau} > \frac{1}{4\tau}$. □

Thus **Sample$\mathbb{Z}_\infty(\cdot,\cdot,\tau)$** performs less than 4τ trials on average.

6.2.5 Correctness of the FP variant

We give the list of assumptions needed for our correctness results (Theorems 6.3 and 6.7), and which we refer to as conditions $\mathcal{A}$.

Assumption on Gram-Schmidt precomputation. We assume that the Gram-Schmidt values are (possibly approximately) precomputed, and that the computed values $\bar{\mu}_{i,j}$, $\bar{b}_{i,j}^\star$ and $\bar{\sigma}_i$ verify :

$$|\Delta\mu_{i,j}| = |\mu_{i,j} - \bar{\mu}_{i,j}| \leq \hat{\mu}\epsilon, \quad \left|\Delta b_{i,j}^\star\right| = \left|b_{i,j}^\star - \bar{b}_{i,j}^\star\right| \leq \|\mathbf{b}_i^\star\|\,\epsilon,$$
$$|\Delta\sigma_i| = |\sigma_i - \bar{\sigma}_i| \leq \sigma_i\epsilon,$$

where $\hat{\mu}$ denotes the maximal absolute value of the sub-diagonal coefficient of μ. Those condition can be achieved by running the precomputation exactly, then convert the result to floating points of mantissa size m.

Assumption on the target vector. We assume that the components c_i of the input target vector $\mathbf{c}$ satisfy : $|c_i| \leq q$ for a parameter q. This holds in all known cryptographic applications of lattice sampling, for which the lattice is q-ary. But we do not require that the lattice is q-ary.

Assumption on the parameters.

$$\mathcal{A}\begin{cases} \epsilon \leq 0.01, & K^n = (1+\epsilon)^n \leq 1.1, & 1 + nK^n\epsilon \leq 1.01 \\ n\iota \leq 0.01, & \forall i, \sigma_i \geq \eta_\iota(\mathbb{Z}), & \forall i, \sigma_i \geq 1 \\ n \geq 10 & \tau \geq 4 & \end{cases}$$

The assumptions on ϵ are easily achievable for a mantissa size m at least logarithmic in the dimension n. The condition on ι is not restrictive as it needs to be negligible. Similarly, conditions on σ_i's are not restrictive since the security requires all $\sigma_i \geq \frac{1}{\sqrt{2\pi}}\eta_\iota(\mathbb{Z}) > 1$ for security parameters $\lambda \geq 80$.

For the rest of the analysis, we assume that all parameters B, $\mathbf{c}$ and σ are fixed. Our main result states that with enough precision, the outputs of the exact sampler **SampleLattice$_\infty$** and the floating-point sampler **SampleLattice$_m$** are statistically close :

Theorem 6.3 *There exist constants C_λ, C_τ, C_m, such that for any security parameter $\lambda \geq C_\lambda$, and under conditions $\mathcal{A}$, the statistical distance between* **SampleLattice$_m$** *and* **SampleLattice$_\infty$** *is less than $2^{-\lambda}$ on the same input if the following conditions are satisfied :*

$$\tau \geq C_\tau\sqrt{\lambda \log n} \quad m \geq C_m + \lambda + 2\log_2(\left\|B^{-1}\right\|_s) + \log_2\left(\hat{\mu}^2 n^4 (q+\sigma^2)\tau^3\right)$$

Furthermore, under those conditions, the integers manipulated by **SampleLattice$_m$** *can be represented by floating-point numbers without errors.*

The proof is given in Sec. 6.8.1, and the concrete bounds in Sec. 6.9. At the core of the proof is the following fact :

Fact 6.4 (Asymptotic statistical distance of Sample$\mathbb{Z}$) *Let δ_t, δ_σ be any implicit positive decreasing function of m that converge to 0 as $m \to \infty$, and let $\epsilon = 2^{1-m}$. Under the conditions $\sigma, \tau \geq 4$, for any $t \in \mathbb{R}$ and any (implicit functions of m) $\bar{t}, \bar{\sigma} \in \mathbb{FP}_m$ such that $|t - \bar{t}| + |t|\,\epsilon \leq \delta_t$ and $|\sigma - \bar{\sigma}| \leq \delta_\sigma$, the statistical distance $\Delta(D_{\mathbb{Z},\sigma,t}, \mathbf{Sample}\mathbb{Z}_m(\bar{\sigma}, \bar{t}))$ is upper bounded by :*

$$E_{\mathbf{Sample}\mathbb{Z}}(\sigma_i, \tau, \delta_t, \delta_\sigma, \epsilon) \stackrel{def}{=} 3e^{-(\tau+\delta_\tau)^2/2} + \frac{\tau}{\sigma}\mathcal{O}(\delta_t) + \tau^3\sigma^2\mathcal{O}(\epsilon) + \tau^3\sigma\mathcal{O}(\delta_\sigma)$$

where $\delta_\tau = 5\delta_t/(\tau\sigma) + 2.5\delta_\sigma/\sigma + 2\epsilon$

This result is almost tight since $\Delta(D_{\mathbb{Z},\sigma,t}, D_{\mathbb{Z},\sigma,t+\delta}) = \frac{1}{\sigma}\mathcal{O}(\delta)$, up to the extra factor τ that comes from the fact that this algorithm makes τ trials. The rest of the proof proceed by numerical analysis to bound the error made on the t_i before invocation of the **Sample**$\mathbb{Z}_m$ routine. This fact explains the term λ in our correctness Theorem 6.3 : to get $D_{\mathbb{Z},\sigma,t}$ up to a statistical distance of $2^{-\lambda}$ one needs to know the λ first bits of t. Additional terms rise from the error made during the computation; in particular it also depends on the size of the matrix $\mathbf{B}^{\star-1}$ that drives the error made during the **DecomposeGS**$_m$.

6.2.6 Efficiency

We deduce the efficiency of the basic floating-point sampler from Theorem 6.3. We first analyze **Sample**$\mathbb{Z}_m$:

Fact 6.5 *There is a constant C_m such that for any $m \geq C_m$, and any $\tau \geq 1$, **Sample**$\mathbb{Z}_m(\cdot, \cdot, \tau)$ performs less than 4τ trials on the average.*

This can be easily derived from Fact 6.2 using the error bound of Fact 6.21. This ensures that the algorithm **SampleLattice**$_m$ performs $\sim 6n^2$ $\mathbb{FP}_m$-operations as long as $\tau = o(n)$.

Arbitrary bases To minimize the FPA-precision m in Theorem 6.3, we need to evaluate $\log(\left\|\mathbf{B}^{-1}\right\|_s)$: this is always less than $\approx n\log(\mathbf{B})$ by Cramer's rule. This leads to the constraint $m \geq \lambda + n\ell$ where ℓ is logarithmic in n and B, yielding a $\tilde{\mathcal{O}}(n^3)$ bit-complexity as long as $\lambda = \mathcal{O}(n)$, or $\tilde{\mathcal{O}}(n^4)$ without fast integer arithmetic.

The exact algorithm **SampleLattice**$_\infty$ also has complexity $\tilde{\mathcal{O}}(n^3)$. However, the constants are likely to be smaller for the FPA sampler. Indeed, the exact algorithm must handle integers of size $\log(\max_{1\leq i\leq n} \mathrm{Vol}(\mathbf{b}_1, \ldots, \mathbf{b}_i))$, whereas the quantity $\log(\left\|\mathbf{B}^{-1}\right\|_s)$ is typically smaller, though they have similar worst-case asymptotical bounds. And the constants of the FPA sampler can be improved by processing the basis, for instance using LLL reduction.

Furthermore, in cryptographic applications, we may focus on bases B of a particular shape. More precisely, we will consider the following type of basis :

Small-inverse bases A sequence $\mathcal{C} = (\mathcal{C}_n)$ of square matrices generating q_n-ary lattices of dimension n is a class of *small-inverse bases* if there exists a polynomial function f such that for any basis $\mathbf{B} \in \mathcal{C}_n$, $\|\mathbf{B}\|_s \leq f(n)$ and $\left\|\mathbf{B}^{-1}\right\|_s \leq f(n)$.

In particular, the bases used by the NTRUSign signature scheme [HNHGSW03] form a small-inverse class (see [HNHGSW03]). For such bases, we only need $m \geq \lambda + \ell$ for ℓ logarithmic in λ. This still gives a $\tilde{\mathcal{O}}(n^3)$ complexity for cryptographic use (when $\lambda \sim n$), but with much better constants.

6.3 Optimizing the FP Variant of Klein's Algorithm

Overview We now describe our optimized sampler, which is more efficient than the basic sampler, due to a better use of FPA. The analysis of the basic sampler showed that it was sufficient to compute t_i up to $\approx \lambda$ bits below the unity to get an error below $2^{-\lambda}$ on the output distribution. However, a careful analysis of the rejection sampling algorithm (Alg. 12) shows that most of the time, many of those bits are not used : the precision of t_i impacts the precision of $p = \rho_{\sigma,t}(x)$, which is only used to make a comparison with a uniform random real $r \in [0, 1)$. For all $j > 1$, such a comparison is determined by the first j bits, except with probability 2^{-j} (exactly when the j first bits of r and p match) ; and on average only the two first bits are consumed to decide the comparison.

However, we still need to decide properly this comparison even when the first $j \leq \lambda$ bits match, to output a proper distribution. This suggests a new strategy : compute lazily the bits of t_i and p. We first only compute most significant bits and backtrack for additional bits until the comparison can be determined. We choose a simple laziness control, using only two levels of precision (for simplicity, but also for practical efficiency). Informally, we choose $k \leq \lambda$, and compute t_i up to a precision m' that only guarantees the first k bits of p, draw the first k bits of the random real r. If the comparison is decided with those k bits, continue normally. Otherwise (which happens with probability less than 2^{-k}) recompute t_i and p at a precision m to ensure λ correct bits.

6.3.1 Description

Our optimized sampler $\mathbf{LazySampleLattice}_{m',m}$ (Alg 15) works with two floating-point types, $\mathbb{FP}_m$ (high precision) and $\mathbb{FP}_{m'}$ (low precision), where $m > m'$. The algorithm works similarly to the original one, except it now works most of the time at low precision m'. The subroutine for sampling over $\mathbb{Z}$ is replaced by $\mathbf{LazySample}\mathbb{Z}_{m',m}$, which takes the usual arguments at low precision, plus an error bound, and access to high-precision arguments : σ is precomputed thus requiring no special care, however, the access to high precision value of t is given through a function that takes no argument.

This new subroutine $\mathbf{LazySample}\mathbb{Z}_{m',m}$ (Alg. 16) works identically to the original $\mathbf{Sample}\mathbb{Z}_{m'}$ as long as the decisive comparison is trusted, i.e. as long as the difference $|r' - p'|$ is higher than the error bound δ_p. Otherwise, the high precision is triggered, and high-precision inputs are requested through the function F. Then all sample trials are computed with high precision.

Algorithm 15 $\mathbf{LazySampleLattice}_{m',m}$: Lazy Gaussian Sampling over a lattice

Input: Same as **SampleLattice** plus low precision versions of $\mu, \mathbf{B}^\star$ and σ_i's values : $\mu', \mathbf{B}^{\star\prime} : \mathbb{FP}_{m'}^{n \times n}$, $\sigma_i' : \mathbb{FP}_{m'}$, and an error bound δ_p

Output: Same as **SampleLattice**

$\mathbf{v}, \mathbf{z} \leftarrow \mathbf{0} : \mathbb{Z}^n$
$\mathbf{t}' \leftarrow \mathbf{DecomposeGS}_{m'}(\mathbf{c}, B^{\star\prime}) : \mathbb{FP}_{m'}^n$
for $i = n$ downto 1 **do**
  $F_i \leftarrow \{() \mapsto \langle \mathbf{c}, \mathbf{b}_i^\star \rangle - \langle \mathbf{z}, \lceil \mu^T \rfloor_i \rangle\} : (\text{void} \to \mathbb{FP}_m)$
  $z_i \leftarrow \mathbf{LazySample}\mathbb{Z}_{m',m}(\sigma_i', \tau, t_i', \delta_p, \sigma_i, F_i) : \mathbb{Z}$
  $\mathbf{v} \leftarrow \mathbf{v} + z_i \cdot \mathbf{b}_i : \mathbb{Z}^n$; $\mathbf{t}' \leftarrow \mathbf{t}' - z_i \cdot \boldsymbol{\mu}_i' : \mathbb{FP}_{m'}^n$
end for
return $\mathbf{v}$

Algorithm 16 $\mathbf{LazySample}\mathbb{Z}_{m',m}(\sigma', \tau, t', \delta_p : \mathbb{FP}_{m'},\ \sigma : \mathbb{FP}_m, F : (\text{void} \to \mathbb{FP}_m))$

$h' \leftarrow -1/2\sigma'^2 : \mathbb{FP}_{m'}$; $x_{\max} \leftarrow \lceil t' + \tau\sigma' \rceil : \mathbb{Z}$; $x_{\min} \leftarrow \lfloor t' - \tau\sigma' \rfloor : \mathbb{Z}$; highprec $\leftarrow$ false : bool
$x \leftarrow \mathbf{RandInt}(x_{\min}, x_{\max}) : \mathbb{Z}$; $r' \leftarrow \mathbf{RandFloat}_{m'}() : \mathbb{FP}_{m'}$
if not(highprec) **then**
  $p' \leftarrow \exp(h' \cdot (x - t')^2) : \mathbb{FP}_{m'}$
  if $|r' - p'| \leq \delta_p$ **then** $\{t \leftarrow F() : \mathbb{FP}_m$; $h \leftarrow 1/2\sigma^2 : \mathbb{FP}_m$; highprec $\leftarrow$ true $\}$
  else if $r' < p'$ **then** return x
end if
if highprec **then**
  $r \leftarrow \mathbf{ExtRandFloat}_{m',m}(r') : \mathbb{FP}_m$; $p \leftarrow \exp(h \cdot (x - t)^2) : \mathbb{FP}_m$
  if $r < p$ **then** return x
end if
Goto Step 2.

6.3.2 Correctness

We need to determine a proper value for the error bound δ_p in terms of the basis and m' (the size of the low precision), to ensure correctness. For this parameter, the lower the better, since it determines the probability to trigger the re-computation of t at high precision, as detailed in the next section. The behavior of the new subroutine is analyzed by the following :

Lemma 6.6 (Informal version of 6.23) *The behavior of* $\mathbf{LazySample}\mathbb{Z}_{m,m'}$ *given approximate inputs* $\sigma \pm \delta_\sigma$ *and* $t \pm \delta_t$ *and* δ_p*, is similar to* $\mathbf{Sample}\mathbb{Z}_m$ *on input* σ, t *under the condition :*

$$\delta_p \geq \sigma^2 \mathcal{O}(\epsilon') + \sigma \mathcal{O}(\delta_\sigma) + \mathcal{O}(\delta_t)/\sigma \quad \text{where } \epsilon' = 2^{1-m'}$$

From this lemma, we prove the correctness of $\mathbf{LazySampleLattice}_{m',m}$, summarized by the following result.

Theorem 6.7 *There exist constants* $C_\lambda, C_\tau, C_m, C_{m'}, C_{\delta_p}$*, such that for any security parameter* $\lambda \geq C_\lambda$*, and under Conditions* $\mathcal{A}$*, the statistical distance between* $\mathbf{LazySampleLattice}_{m,m'}$ *and* $\mathbf{SampleLattice}_\infty$ *is less than* $2^{-\lambda}$ *on the same input if the following conditions are satisfied :*

$$\begin{aligned}
\tau &\geq C_\tau \sqrt{\lambda \log n} \\
m &\geq C_m + \lambda + 2\log_2(\left\|\mathbf{B}^{-1}\right\|_s) + \log_2\left(\hat{\mu}^2 n^4 q \sigma^2 \tau^3\right) \\
m' &\geq C_{m'} + 2\log_2(\left\|\mathbf{B}^{-1}\right\|_s) + \log_2\left(\hat{\mu}^2 n^4 (Q+\sigma^2) \tau^3\right) \\
\delta_p &\geq 2^{-k} \text{ where } k = m' - \left(C_{\delta_p} + 2\log_2(\left\|\mathbf{B}^{-1}\right\|_s) + \log_2\left(\hat{\mu}^2 n^3 \tau \sigma^2 q\right)\right)
\end{aligned}$$

Furthermore, under those conditions, the integers manipulated by the algorithm can be represented by low-precision floating-point numbers ($\mathbb{FP}_{m'}$*) without errors.*

The proof is given in Sec. 6.8.2, and the concrete bounds in Sec. 6.9.

6.3.3 Efficiency

The error bound δ_p impacts the efficiency of the optimized sampler as follows :

Lemma 6.8 *Under the conditions of Theorem 6.7, each call to* $\mathbf{LazySample}\mathbb{Z}_{m,m'}$ *triggers high precision with probability less than* $8\tau\delta_p$*. On the average, the algorithm* $\mathbf{LazySampleLattice}_{m,m'}$ *performs less than* $\mathcal{O}(n^2\tau\delta_p)$ *high-precision floating-point operations.*

Proof: At each trial performed by $\mathbf{LazySample}\mathbb{Z}_{m,m'}$, the probability to trigger high precision is less than $2\delta_p$: indeed it happens only if the randomness $r' \leftarrow [0,1)$ falls in the interval $[p' - \delta_p, p' + \delta_p]$. It remains to bound the average number of trials performed by $\mathbf{LazySample}\mathbb{Z}_{m,m'}$. The condition of Theorem 6.7 ensures that it behaves similarly to $\mathbf{Sample}\mathbb{Z}_m$. Thus, for a large enough m, Fact 6.5 ensures that the average number of trials is less than 4τ.

Triggering high precision during $\mathbf{LazySample}\mathbb{Z}_{m,m'}$ requires $\mathcal{O}(n)$ high-precision FPA operations. This subroutine is called n times, thus on the average less than $\mathcal{O}(n^2\tau\delta_p)$ high-precision FPA operations. □

This leads to our main result : with Small-Inverse bases, the discrete Gaussian distribution can be sampled in quasi-quadratic time, with an exponentially small statistical distance, and no sacrifice on the quality compared to the analysis of [GPV08].

Theorem 6.9 (Gaussian Sampling in quasi-quadratic time) *Let* $(\mathcal{C}_n)$ *be an Small-Inverse class of size-reduced basis. For any implicit function* λ*, such that* $\lambda \sim n$*, and* σ *polynomial in* n*, there exists implicit functions* m, m', τ, δ_p *of* n *such that, for any basis* $B \in \mathcal{C}_n$ *generating a lattice* L *:*

- $\mathbf{LazySampleLattice}_{m,m'}(\mathbf{B}, \sigma, \mathbf{c}, \tau, \delta_p)$ *runs in expected time* $\tilde{\mathcal{O}}(n^2)$ *without fast integer arithmetic.*
- $\Delta(D_{L,\sigma,\mathbf{c}}, \mathbf{LazySampleLattice}_{m,m'}(\mathbf{B}, \sigma, \mathbf{c}, \tau, \delta_p)) \leq 2^{-\lambda}$ *whenever* σ *verifies* $\sigma \geq \|\mathbf{B}^\star\| \, \eta_\iota(\mathbb{Z})/\sqrt{2\pi}$ *with* $\iota = 2^{-\lambda}/(4n)$*.*

Proof: For an small-inverse class of bases, the conditions of Theorem 6.7 can be satisfied with functions verifying :

$$\tau = \mathcal{O}(\sqrt{n}), m = \mathcal{O}(n), m' = \mathcal{O}(\log n), \delta_p = \mathcal{O}(1/n^{5/2}).$$

Lemma 6.8 states that on the average, less than $\mathcal{O}(n^2\tau\delta_p)$ high-precision operations are performed, which in our case is a $\mathcal{O}(1)$. Without fast integer arithmetic, The total complexity is thus less than $\mathcal{O}(n^2)\mathcal{O}(m'^2) + \mathcal{O}(1)\mathcal{O}(m^2) \leq \tilde{\mathcal{O}}(n^2)$. □

6.4 Optimizing Peikert's Offline Algorithm, General Case

Peikert [Pei10] proposed a different sampling algorithm, based on convolution, and inspired by NTRUSign's perturbation countermeasure [HNHGSW03]. This algorithm offers a different trade-off than Klein's algorithm. At the cost of some quality (see [Pei10] for details). The online phase is essentially a randomized variant of Babai's simple rounding algorithm, and involve no floating points for q-ary lattices, thus, it runs in $\tilde{\mathcal{O}}(n^2)$ time, and even $\tilde{\mathcal{O}}(n)$ in the ring setting.

Recall that Babai's simple rounding algorithm proceed as $\mathbf{z} = \lfloor \mathbf{x} \cdot \mathbf{B}^{-1} \rceil \cdot \mathbf{B}$. Peikert's online phase simply replace then rounding operation by 1-dimensional Gaussian Sampling. In the case of q-ary lattices,

Algorithm 17 Peikert's Online Algorithm

Input: A basis $\mathbf{B} : \mathbb{Z}^{n \times n}$ of a lattice $\mathbf{L}$, its inverse $\mathbf{B}^{-1} : \mathbb{R}^{n \times n}$, a parameter $\sigma : \mathbb{FP}_m$, a target vector $\mathbf{t} \in \mathbb{R}^n$

Output: An integer vector $\mathbf{z} \in \mathbb{Z}^n$ following the Discrete Gaussian Distribution $\mathcal{D}_{\mathbf{L},\sqrt{\mathbf{\Sigma_B}},\mathbf{t}}$ over $\mathbf{L}$ of covariance $\mathbf{\Sigma_B} = \sigma^2 \cdot \mathbf{B}^t \cdot \mathbf{B}$

1: $\mathbf{x} \leftarrow \mathbf{t} \cdot \mathbf{B}^{-1}$
2: **for** $i = 1$ to n **do** $y_i \leftarrow$ **Sample**$\mathbb{Z}_m(\sigma, x_i, \tau)$
3: return $\mathbf{z} \leftarrow \mathbf{y} \cdot \mathbf{B}$

this can be implemented very efficiently using the fact that $\mathbf{B}^{-1} \in \frac{1}{q}\mathbb{Z}$: one can therefore implement the phase using only operations mod q and q^2.

Still, this bare algorithm samples a non-spherical Gaussian, so a perturbation, computed offline, is added to the target points. However, the offline phase does require long-integer arithmetic, and it is not fully analyzed in [Pei10] but seems to be $\tilde{\mathcal{O}}(n^3)$ (even $\tilde{\mathcal{O}}(n^4)$ without fast integer arithmetic) like Klein's algorithm.

Algorithm 18 Peikert's Offline Algorithm

Input: A square-root $\mathbf{L}$ of $\mathbf{\Sigma}' = (s^2 - \eta^2)\mathbf{Id} - \mathbf{\Sigma_B} \in \mathcal{S}_n^+$

Output: An integer vector $\mathbf{z} \in \mathbb{Z}^n$ following the Centered Discrete Gaussian Distribution over $\mathbb{Z}^n$ of covariance $\sigma^2\mathbf{Id} - \mathbf{\Sigma_B}$: $\mathcal{D}_{\mathbb{Z}^n,\sqrt{\sigma^2\mathbf{Id}-\mathbf{\Sigma_B}}}$

1: choose $\mathbf{x} : \mathbb{R}^n$ as a Continuous Gaussian of covariance $\mathbf{Id}$
2: $\mathbf{y} = \mathbf{x} \cdot \mathbf{L}$
3: **for** $i = 1$ to n **do** $z_i \leftarrow$ **Sample**$\mathbb{Z}_m(\eta, y_i, \tau)$
4: return $\mathbf{z}$

Algorithm 19 Peikert's Full Algorithm

Input: A basis $\mathbf{B} : \mathbb{Z}^{n \times n}$ of a lattice $\mathbf{L}$, its inverse $\mathbf{B}^{-1} : \mathbb{R}^{n \times n}$, a parameter $\sigma : \mathbb{FP}_m$, a target vector $\mathbf{t} \in \mathbb{R}^n$

Output: An integer vector $\mathbf{z} \in \mathbb{Z}^n$ following the Discrete Gaussian Distribution $\mathcal{D}_{\mathbf{L},\sigma,\mathbf{t}}$ over $\mathbf{L}$ of covariance $\sigma^2\mathbf{Id}$

1: Choose $\mathbf{p} \leftarrow D_{\mathbb{Z}^n,\sqrt{\sigma^2\mathbf{Id}-\mathbf{\Sigma_B}}}$ (offline phase)
2: Set $\mathbf{t}' \leftarrow \mathbf{t} + \mathbf{p}$
3: Output $\mathbf{z} \leftarrow D_{\mathbb{Z}^n,\sqrt{\mathbf{\Sigma_B}},\mathbf{t}'}$

In the latter work of [MP12], a new kind of trapdoor is build, and is ingeniously designed for algorithmic efficiency, and geometric quality. While this allow an even faster online phase, the same kind of offline generated perturbation is required. We refer to this common perturbation generation as Peikert's Offline Algorithm.

6.4.1 Efficiency of Peikert Offline phase

In both [Pei10, MP12], the online phase samples from a non spherical Gaussian, of variance $\mathbf{\Sigma_B} = \sigma\mathbf{B}^t\mathbf{B} \in \mathcal{S}_n^+$ for some matrix $\mathbf{B}$ and some scalar $\sigma \geq \eta_\epsilon(\mathbb{Z})$. To achieve sphericity, one can perturb the target vector by adding another discrete Gaussian noise, of covariance $\mathbf{\Sigma}$ such that $\mathbf{\Sigma} + \mathbf{\Sigma_B} = s \cdot \mathbf{Id}$ for some scalar s. This noise is generated by the offline phase described in Alg. 18.

To implement this, it is suggested in [Pei10] to compute a square root $\mathbf{L}$ s.t. $\mathbf{L}\mathbf{L}^t = \mathbf{\Sigma}'$ via a Cholesky decomposition. The parameters selected to reach security λ are $\eta = \tau = \eta_\iota(\mathbb{Z}) = \tilde{\mathcal{O}}(\sqrt{\lambda})$. The choice of the floating-point precision is not discussed in [Pei10, MP12], however as quick analysis shows that one should take $m = \lambda + \ell$ where ℓ is logarithmic in n, s and τ. Thus, a naive implementation would have a running-time of $\tilde{\mathcal{O}}(n^2\lambda^2)$, the main cost being a non-structured matrix-vector product : that is n^2 floating-point operations, at precision $\tilde{\mathcal{O}}(\lambda)$.

6.4.2 Applying Laziness to Peikert's Offline Algorithm

Like in Klein's Sampling algorithm, the offline phase of Peikert's algorithm [Pei10] only uses non integer values to compute the input of the $\mathbf{SampleLatticeZ}_m(\eta, \cdot, \tau)$ subroutine. High precision bits of this input are useless except with small probability : one may apply the laziness technique to improve efficiency to $\tilde{\mathcal{O}}(n^2)$, by replacing the subroutine by $\mathbf{LazySample}\mathbb{Z}_{m',m}$. We sketch a proof : details and concrete bounds can be adapted from Sect. 6.3.

The floating-point computation $y_j = \sum_{i=1}^{n} x_i \mathbf{L}_{j,i}$ with m bits of precision produces an error less than $\tilde{\mathcal{O}}(n^2 \|x\|_\infty \|\mathbf{L}\|_\infty \epsilon)$ where $\epsilon = 2^{1-m}$. For $\tau = \tilde{\mathcal{O}}(\sqrt{n})$ we have that $\|x\|_\infty \leq \tau$ with overwhelming probability, and $\|\mathbf{L}\|_\infty \leq \|\mathbf{L}\|_s \leq s$ since $\mathbf{L}^t\mathbf{L} = \mathbf{C}' \leq \sigma^2\mathbf{Id}$. The error propagation is thus polynomial in n, and Lemma 6.6 ensures correction with the following parameters :

$$\tau = \mathcal{O}(\sqrt{n}), m = \mathcal{O}(n), m' = \mathcal{O}(\log n), \delta_p = \mathcal{O}(1/n^{5/2}).$$

Similarly to Lemma 6.8, one easily proves that, on average, less than $\mathcal{O}(n^2\tau\delta_p)$ high-precision operations are performed, which in our case is $\mathcal{O}(1)$. Without fast integer arithmetic, the total complexity is thus less than $\mathcal{O}(n^2)\mathcal{O}(m'^2) + \mathcal{O}(1)\mathcal{O}(m^2) \leq \tilde{\mathcal{O}}(n^2)$. □

6.5 Reaching Quasi-Linear Complexity in the Ring-Setting $\mathcal{R} = \mathbb{Z}_q[X]/(X^b \pm 1)$

Laziness, our second improvement over the Offline when used in [Pei10, MP12] is simply to select another square-root algorithm to preserve matrix structures. Indeed, if B is structured (block-circulant or block-anti-circulant), then $\mathbf{\Sigma} = s\mathbf{Id} - B \cdot B^t$ has the same structure by Fact 6.10; but that structure is broken by the Cholesky decomposition.

Instead, we suggest to use a another square-root algorithm, like the Babylonian Method, or the Denman-Beavers iteration [DB76], since, as it we will show in the next section preserve some structures.

6.5.1 Structured Square-Root for $\mathcal{R} = \mathbb{Z}_q[X]/(X^b \pm 1)$

Specific Properties of the Ring-Setting $\mathcal{R} = \mathbb{Z}_q[X]/(X^b \pm 1)$ We show that in those Ring-Setting it is possible to use a structure-preserving square root algorithm. It includes many popular Ring-Setting for cryptography purposes : $\mathbb{Z}_q[X]/(X^b - 1)$ for the class of NTRU lattices [HNHGSW03], and some cyclotomic lattices $\mathbb{Z}_q[X]/(\Phi_m)$ the m-th cyclotomic ring, when m is a power of two, made popular by the hardness results of [LPR10].

When $P(X) = X^b - 1$ (resp. $P(X) = X^b + 1$) the integer representation $B \in \mathcal{M}^{bk \times bl}(\mathbb{Z})$ of any $\mathcal{R}$-basis is a b-block circulant, (resp. b-block anti-circulant) matrix, *i.e.* a matrix composed with $(b \times b)$-blocks of the form :

$$\begin{bmatrix} a_1 & a_2 & \cdots & a_b \\ a_b & a_1 & \cdots & a_{b-1} \\ \vdots & \ddots & \ddots & \vdots \\ a_2 & \cdots & a_b & a_1 \end{bmatrix}, \quad \text{resp.} \quad \begin{bmatrix} a_1 & a_2 & \cdots & a_b \\ -a_b & a_1 & \cdots & a_{b-1} \\ \vdots & \ddots & \ddots & \vdots \\ -a_2 & \cdots & -a_b & a_1 \end{bmatrix}.$$

The family are noted $\mathcal{C}_b$ (resp. $\mathcal{A}_b$).These two family are of matrices is stable under ring operations (addition, product and inverse, when defined) because of the ring isomorphism with matrices over $\mathcal{R}$. Such isomorphisms also exist for other polynomials P, defining other b-block structures. However, circulant and anti-circulant structures provide a key property for our improvement :

Fact 6.10 *Matrix families $\mathcal{C}_b$ and $\mathcal{A}_b$ are stable under transposition.*

From this, we deduce that $\mathbf{\Sigma} = s\mathbf{Id} - \mathbf{B} \cdot \mathbf{B}^t \in \mathcal{C}_b$ (or $\mathcal{A}_b$) when working in those rings. At this point one would want to find a square root of $\mathbf{\Sigma}$ that is still structured. Interestingly, the solution is to be found in algorithms that were designed to a extract a different notion of square root; namely the Babylonian Method, or the Denman-Beavers iteration [DB76]. Indeed, those algorithms are searching for an $\mathbf{Y}$ such

that $\mathbf{Y} \cdot \mathbf{Y} = \mathbf{X}$, without symmetry requirement on $\mathbf{X}$, and no guarantee of convergence in general. Lemma 6.11 will prove that given an input $\mathbf{X} \in \mathcal{S}_n^+$, they converge (very fast), to some $\mathbf{Y} \in \mathcal{S}_n^+$, ensuring that $\mathbf{Y}^t \cdot \mathbf{Y} = \mathbf{X}$, as we would like. It also proves structure preservation. For simplicity, we focus our proofs on the Babylonian method (a special case of Newton iteration), but the Denman-Beavers is better in practice, both for convergence speed and numerical stability.

Definition The Babylonian Method approximates the limit of the sequence :

$$\mathbf{Y}_0(\mathbf{X}) = \mathbf{Id}; \quad \mathbf{Y}_{k+1}(\mathbf{X}) = (\mathbf{Y}_k(\mathbf{X}) + \mathbf{X} \cdot \mathbf{Y}_k(\mathbf{X})^{-1})/2 \tag{6.2}$$

and if this sequence converges to an invertible limit $\mathbf{Y}(\mathbf{X})$, it must verify $\mathbf{Y}(\mathbf{X}) = \frac{1}{2}(\mathbf{Y}(\mathbf{X}) + \mathbf{X} \cdot \mathbf{Y}(\mathbf{X})^{-1})$, which is equivalent to $\mathbf{Y}(\mathbf{X}) \cdot \mathbf{Y}(\mathbf{X}) = \mathbf{X}$. The Denman-Beavers iteration is similar, using the sequences :

$$\begin{cases} \mathbf{Y}_0(\mathbf{X}) &= \mathbf{X} \\ \mathbf{Z}_0(\mathbf{X}) &= \mathbf{Id} \end{cases} \quad \begin{cases} \mathbf{Y}_{k+1}(\mathbf{X}) &= \left(\mathbf{Y}_k(\mathbf{X}) + \mathbf{Z}_k(\mathbf{X})^{-1}\right)/2 \\ \mathbf{Z}_{k+1}(\mathbf{X}) &= \left(\mathbf{Z}_k(\mathbf{X}) + \mathbf{Y}_k(\mathbf{X})^{-1}\right)/2 \end{cases} \tag{6.3}$$

it verifies the invariant $\mathbf{Y}_k \cdot \mathbf{Z}_k^{-1} = \mathbf{Z}_k^{-1} \cdot \mathbf{Y}_k = \mathbf{X}$, and if it converges, the limit $\mathbf{Y}$ of $\mathbf{Y}_k$ verifies $\mathbf{Y} \cdot \mathbf{Y} = \mathbf{X}$.

Lemma 6.11 *Let $\mathbf{X} \in \mathcal{S}_n^+$ be a definite positive symmetric matrix, then the Babylonian Method, as defined by the sequence $\mathbf{Y}_k(\mathbf{X})$ in (6.2) converges quadratically[3] to some $\mathbf{Y}(\mathbf{X}) \in \mathcal{S}_n^+$. Furthermore, if $\mathbf{X} \in \mathcal{C}_b$ (resp. $\mathcal{A}_b$) then $\mathbf{Y}(\mathbf{X})$ also belongs to $\mathcal{C}_b$ (resp. $\mathcal{A}_b$). Similar results also hold for the Denman-Beavers iteration (6.3).*

Proof: Since $\mathbf{X}$ is a definite positive symmetric matrix, it can be written as $\mathbf{X} = \mathbf{Q}\mathbf{D}\mathbf{Q}^t$ where $\mathbf{D}$ is a diagonal matrix with strictly positive entries and $\mathbf{Q}$ is orthogonal. By induction, one proves that $\forall k \geq 0, \mathbf{Y}_k(\mathbf{X}) = \mathbf{Q} \cdot \mathbf{Y}_k(\mathbf{D}) \cdot \mathbf{Q}^t$, where $\mathbf{Y}_k(\mathbf{D})$ is also diagonal. The i-th diagonal entry of $\mathbf{Y}_k(\mathbf{D})$, noted $y_k^{(i)}$ verifies $y_{k+1}^{(i)} = \frac{1}{2}(y_k^{(i)} + \mathbf{D}_{i,i}/y_k^{(i)})$. This sequence corresponds exactly to the Babylonian Method on the real number $\mathbf{D}_{i,i}$, which is known to converge quadratically[2] to $\sqrt{\mathbf{D}_{i,i}}$. Thus, $\mathbf{Y}_k(\mathbf{D})$ converges quadratically to a diagonal matrix $\mathbf{D}'$ with strictly positive entries, ensuring that $\mathbf{Y}_k(\mathbf{X})$ converges to $\mathbf{Q}\mathbf{D}'\mathbf{Q}^t$ which is definite positive symmetric.

Now, for structure preservation, consider the set $\mathcal{S}_{n,b}$ of b-block circulant matrices in $\mathcal{M}_n(\mathbb{R})$. This set is stable by sum, products and inverse (when it exists). Thus, if $\mathbf{X} \in \mathcal{S}_{n,b}$, by induction we have that $\forall k \geq \mathbf{Y}_k(\mathbf{X}) \in \mathcal{S}_{n,b}$. It remains to note that $\mathcal{S}_{n,b}$ is topologically closed to conclude on the limit.

Proofs are similar for the set $\mathcal{S}'_{n,b}$ of b-block anti-circulant matrices, and when replacing the Babylonian Method by the Denman-Beavers Method. □

Finally, we note that intermediate results of those algorithms are also structured, reducing the complexity of each step by a factor $\tilde{\mathcal{O}}(n)$.

6.5.2 Improved Efficiency

Assuming the square root $\mathbf{L}$ of $\mathbf{\Sigma}$ was pre-computed using one of the structure preserving square-root Algorithm described below, each computation of $\mathbf{y} = \mathbf{x} \cdot \mathbf{L}$ at precision m' can now be done in time $\tilde{\mathcal{O}}(nm'^2)$, but some coordinate may need to be recomputed at precision m. Using a similar analysis than Sect. 6.4.2 with :

$$\tau = \mathcal{O}(\sqrt{n}), m = \mathcal{O}(n), m' = \mathcal{O}(\log n), \delta_p = \mathcal{O}(1/n^{7/2}).$$

we show that the average[4] time spent on the computation of $\mathbf{y} = x \cdot \mathbf{L}$ is indeed $\tilde{\mathcal{O}}(n)$.

Comb Laziness and Structured-Square-Root, we move the complexity bottleneck to the **LazySample**$\mathbb{Z}$ subroutine, which is called n times and requires $\tilde{\mathcal{O}}(\tau) = \tilde{\mathcal{O}}(\sqrt{\lambda})$ trials in average. For $\lambda \sim n$, this leads to an overall average complexity of $\tilde{\mathcal{O}}(n^{1.5})$.

To reach quasi-linear complexity we need a third trick, detailed in the next section (Sec. 6.5.3). There, we improve the rejection sampling algorithm **Sample**$\mathbb{Z}$ so that it only needs a constant number of trials on the average. This is done by sampling from a distribution before rejection that is much closer to the target distribution than the uniform distribution used in **Sample**$\mathbb{Z}$.

Combining the three techniques, we eventually obtain an implementation of Peikert's offline phase which runs in average[4] quasi-linear time. These results also apply to the recent variant of Micciancio and Peikert [MP12].

3. The number of correct bits grows quadratically with the number of iteration k : $|s_k - s_\infty| \leq c\ 2^{-c'k^2}$ for some $c, c' > 0$

4. We explain what we mean by average. As high-precision is triggered independently with small probability over n trials,

6.5.3 Gaussian Sampling over $\mathbb{Z}$ with Constant Trials

In this section we provide a better algorithm to sample from the distribution $D_{\mathbb{Z},\sigma,c}$ tailcutted with parameter τ for a fixed value of σ. We still use rejection sampling, however we replace the uniform distribution over $\{\lfloor c-\tau\sigma\rfloor \dots \lceil c+\tau\sigma\rceil\}$ by a (tailcutted) discrete Gaussian distribution $D_{\mathbb{Z},\sigma,\lfloor c\rceil}$. It verifies two essential properties for our application : first it can also be sampled very efficiently, using $\mathcal{O}(\log(\tau\sigma))$ access to a precomputation table of $\mathcal{O}(\tau\sigma)$ elements. Secondly, it fits better to $D_{\mathbb{Z},\sigma,c}$ than the uniform distribution, thus improving the acceptance rate of the sampled value, leading to a faster sampling algorithm.

Sampling from $D_{\mathbb{Z},\sigma,z}$ for an integer $z \in \mathbb{Z}$. Trivially, the problem can be reduced to sample $x \leftarrow D_{\mathbb{Z},\sigma,0}$ and output $x+z \in \mathbb{Z}$. For fixed values of σ and τ, this is doable very efficiently using a precomputation table : let $T[-\lceil\tau\sigma\rceil \dots \lceil\tau\sigma\rceil]$ be a table containing the value cumulative distribution function of $D_{\mathbb{Z},\sigma,0}$. Then one may sample from $D_{\mathbb{Z},\sigma,0}$ by choosing a uniform random real $r \in [0,1)$ and outputting the only i such that $T[i] \le r < T[i+1]$. Finding such an index i can be done using only $\mathcal{O}(\log(\tau\sigma))$ access to the table T by a binary search. Yet in the next chapter 7 we will develop alternative algorithm for this task without such a large precomputation table.

Acceptance Rate. The rejection sampling technique samples $x \leftarrow \mathcal{D}'$ and accepts this sample with probability $r_\alpha(x) = \alpha P_{y\leftarrow\mathcal{D}}[y=x]/P_{y\leftarrow\mathcal{D}'}[y=x]$. The parameter α must be chosen so that $r_\alpha(x) \le 1$ for any x in the domain of $\mathcal{D}'$. iterating this process until acceptance produces a sample following the distribution $\mathcal{D}$. In our case, we have $\mathcal{D}' = D_{\mathbb{Z},\sigma,\lfloor c\rceil}$ and $\mathcal{D} = D_{\mathbb{Z},\sigma,c}$. We note $c' = c - \lfloor c\rceil$ the fractional part of c. The acceptance rate $r_\alpha(x)$ of each element x is as follows (ignoring smoothing issues) :

$$r_\alpha(x) = \alpha e^{-\frac{(x-c)^2}{\sigma^2}} \Big/ e^{-\frac{(x-\lfloor c\rceil)^2}{\sigma^2}} = \alpha e^{\frac{2xc'}{\sigma^2}} \cdot e^{\frac{c^2-\lfloor c\rceil^2}{\sigma^2}}$$

Using the tailcut distribution, we have $|x| \le \tau\sigma$ and $|c|' \le 1/2$, thus $e^{\frac{2xc'}{\sigma^2}} \in \left[e^{-\tau/\sigma}, e^{\tau/\sigma}\right]$. Choosing $\alpha = e^{-\frac{\tau}{\sigma} - \frac{c^2-\lfloor c\rceil^2}{\sigma^2}}$, we obtain that : $r_\alpha(x) \in \left[e^{-2\tau/\sigma}, 1\right]$.

Application to Peikert's Offline Algorithm. To reach a security level λ, Peikert's offline phase uses samples from the tailcut distribution $D_{\mathbb{Z},\sigma,c}$ for parameters $\sigma = \tau = \tilde{\mathcal{O}}(\sqrt{\lambda})$. For such parameters, the acceptance rate of each x in the tailcut domain verifies $r_\alpha(x) \ge e^{-2}$, thus rejection sampling terminates on the average after less than e^2 trials.

However its not clear in general if this alternative rejection sampling should be used for Klein's sampler : it requires samples from $D_{\mathbb{Z},\sigma_i,c}$ for n different values of σ_i, and those values may be quite large (even exponential) thus preventing us from using the precomputation table T.

6.6 Mantissa Sizes in Practice

For the sake of readability, our theorems were formulated asymptotically, but all our results can also be stated in a concrete manner. For practical implementation, one should rely on an ad-hoc approach to derive better bounds (such as done in [PS08]). However, it seems interesting to have at least an estimate of concrete mantissa sizes. Note that our Lazy variants of those algorithms do not compute more low-precision operations than the original algorithms (plus a small amount of high precision operations), thus our improvements are fully stated by the mantissa size of those operations. Our final concrete results are given in Section 6.9.

Nominal Low Precision The nominal Low Precision is defined as $\mathcal{N} = \log_2(b\delta_p/\epsilon')$ where ϵ' and δ_p refers to the parameters for low precision floats of Lazy Sampling, and b is the block-size in the Ring-Setting (taking $b=1$ when non Ring based acceleration is used). The importance of the Nominal Low precision is the following : for each bit over this value for the low precision mantissa size m', the average ratio of high precision operation over low precision operation is halved. Namely, only a $2^{\mathcal{N}-m'}$ fraction

the running times of the optimized Klein's Sampler and optimized Peikert's Offline Phase are bounded by some function $\tilde{\mathcal{O}}(n^2)$, except with negligible probability. However, when applying laziness in the ring setting, triggering high-precision once in the whole algorithm raises this instance's running time to $\tilde{\mathcal{O}}(n\lambda^2)$: only the average cost is below that bound. And dealing with average running times is less problematic in an offline phase, than in an online phase which is more subject to timing attacks.

will be done at high precision out of the $O(n^2)$ operation of our Klein's Algorithm variant, or $\tilde{\mathcal{O}}(n^2/b)$ of Peikert's Offline Algorithm where b is the block size.

Application to NTRUSign type bases The NTRUSign lattices are the most compact trapdoor-lattices known for cryptographic applications, but no provable-security property is known for such lattices. Still, the closest vector problem for this class of lattices is believed to be hard, but the NTRUSign signature scheme needs to be repaired against information leakage [NR06, DN12b] of the signature algorithm. This make them a good candidate for testing the practicality of Gaussian Sampling.

From the results of Section 6.9, we derive concrete bounds for mantissa size, when applying Algorithms [Kle00, GPV08, Pei10] over NTRUSign basis, described in Table 6.1. While the non Lazy variant has to run all operations at high precision, this table shows that taking standard double-precision float as low precision (53 bits), less than one in a million operation will be done at high precision. Yet, those figures might needs small adjusments because applying the technique of [GPV08] would increase the acceptance threshold.

Table 6.1 – FP Klein's Algorithm on NTRUSign-type basis

Security Parameter : λ	80	112	128	160	192	256
Dimension : n	314	394	446	526	626	698
Smoothing Parameter : $\eta_\epsilon(\mathbb{Z})$	4.4	5.1	5.5	6.1	6.6	7.6
FPA-Lazy Klein's Sampling [Kle00, GPV08]						
Quality : $\|B\|/\sqrt{n}$	61.8	73 .6	73.4	139.9	144.0	165.5
High precision : $m = -\log_2(\epsilon)$	116	150	170	206	242	300
Nominal Low precision : $\mathcal{N} = \log_2(\delta_p/\epsilon')$	26.3	26.9	30.7	32.3	32.9	33.5
FPA-Lazy Peikert's Offline Algorithm [Pei10, MP12]						
Quality : $\|B\|_s/\sqrt{n}$	335	399	429	902	976	1122
High precision : $m = -\log_2(\epsilon)$	107	140	157	191	222	287
Nominal Low precision : $\mathcal{N} = \log_2(b\delta_p/\epsilon')$	25.4	26.3	27.5	28.2	28.9	29.2

All parameters measurements leading to those bounds have been done out of the worst case of 50 bases of each size.

Application to the New Trapdoors of Micciancio and Peikert [MP12] While NTRUSign lattices seem to provide reasonable security in practice, there are other constructions providing trapdoors for random (or pseudo-random) lattices [Ajt99, AP09, MP12]. Such lattices enjoy very strong security notion (such as worst-case to average case connection), but are still far from the efficiency of the NTRUSign heuristic trapdoor generation.

The most efficient construction so far is the one of Micciancio and Peikert [MP12], and it has been crafted to improve the online efficiency of Gaussian Sampling. Using their proposed parameters, we obtain a nominal low precision of less than 30 bits (and less than 38 bits in the Ring-Setting). Independently of other potential improvements, it means that laziness allows to run their offline phase using mostly double-precision.

6.7 Technical Lemmata

6.7.1 Error Propagation of FPA Operations

Fact 6.12 (Error propagation during a product) *Let $m \in \mathbb{Z}$ be a positive integer and $\epsilon = 2^{1-m}$. For any $a, b \in \mathbb{R}$ and $\delta_a, \delta_b \in \mathbb{R}$:*

$$|(a+\delta_a)(b+\delta_b) - ab| \leq |a\delta_b| + |b\delta_a| + |\delta_a\delta_b|$$

Similarly, the floating-point product verifies for all $\bar{a}, \bar{b} \in \mathbb{FP}_m$: if $|a-\bar{a}| \leq \delta_a$ and $\left|b-\bar{b}\right| \leq \delta_b$, where $2\delta_b \leq |b|$, then

$$\left|\left(\bar{a}\bar{\times}\bar{b}\right) - ab\right| \leq |a\delta_b| + |b\delta_a| + |\delta_a\delta_b| + (|a| + |\delta_a|)(|b| + |\delta_b|)\epsilon$$

Fact 6.13 (Error propagation during a division) *Let $m \in \mathbb{Z}$ be a positive integer and $\epsilon = 2^{1-m}$. For any $a, b \in \mathbb{R}$ and $\delta_a, \delta_b \in \mathbb{R}$ where $2|\delta_b| \leq |b|$, we have :*

$$\left|\frac{a+\delta_a}{b+\delta_b} - \frac{a}{b}\right| \leq \frac{2}{|b|}|\delta_a| + \left|\frac{2a}{b}\right||\delta_b|$$

Similarly, the floating-point division verifies for all $\bar{a}, \bar{b} \in \mathbb{FP}_m$ *: if* $|a - \bar{a}| \leq \delta_a$ *and* $\left|b - \bar{b}\right| \leq \delta_b$*, where* $2\delta_b \leq |b|$*, then*

$$\left|\left(\bar{a} \bar{/} \bar{b}\right) - \frac{a}{b}\right| \leq \frac{2}{|b|}|\delta_a| + \left|\frac{2a}{b}\right||\delta_b| + 2\frac{2\,|\delta_a| + |a|\,(1 + 2\,|\delta_b|)}{|b|}\epsilon$$

Lemma 6.14 (Propagation of errors during the computation of a sum, adapted from [PS08]) *Let* $m \in \mathbb{Z}$ *be a positive integer and* $\epsilon = 2^{1-m}$*. Let* $a_i \in \mathbb{R}$*, and* $\bar{a_i}$ *be a floating-point approximation of* a_i *such that for* $i \leq n$ $|a_i - \bar{a}_i| \leq \delta_i$*, with* $\delta_i \geq 0$*. Let* $\bar{u}_j$ *be the intermediate values of the floating-point computation of a sum,* i.e. $\bar{u}_0 = 0, \bar{u}_{j+1} = \bar{u}_j \bar{+} \bar{a}_j$*, and* $u_j = \sum_{i=1}^{j} a_i$*. Then the error on the final result satisfies :*

$$\Delta u_n = |\bar{u}_n - u_n| \leq \epsilon\, n\, K^n S + (1 + \epsilon\, n\, K^n) \sum_{i=1}^{n} \delta_i, \text{ with } S = \sum_{i=1}^{n} |a_i|$$

where $K = 1 + \epsilon$*.*

6.8 Proof of Correctness Theorems 6.3 and 6.7

The proofs of those theorems involve many technical lemmatas, which are proved in the next Section 6.8.3 and 6.8.4.

6.8.1 Proof of Theorem 6.3

Here, we prove Theorem 6.3, restated for reader convenience :

Theorem 6.15 *There exist constants* C_λ, C_τ, C_m*, such that for any security parameter* $\lambda \geq C_\lambda$*, and under conditions* $\mathcal{A}$*, the statistical distance between* **SampleLattice**$_m$ *and* **SampleLattice**$_\infty$ *is less than* $2^{-\lambda}$ *on the same input if the following conditions are satisfied :*

$$\tau \geq C_\tau \sqrt{\lambda \log n} \qquad\qquad m \geq C_m + \lambda + 2\log_2(\left\|\mathbf{B}^{-1}\right\|_s) + \log_2\left(\hat{\mu}^2 n^4 (q + \sigma^2)\tau^3\right)$$

Furthermore, under those conditions, the integers manipulated by **SampleLattice**$_m$ *can be represented by floating-point numbers without errors.*

The proof requires various facts and lemmas, whose proofs are given later in this section.

We set the mantissa size m, and the inputs, B and $\mathbf{c}$ and analyze the statistical distance between **SampleLattice**$_m$ and **SampleLattice**$_\infty$, by bounding the error propagation of each step, until full decomposition.

We let $\mathbf{z}_i$ be the distribution of the sampled values $(z_n \dots z_i)$ during the execution of **SampleLattice**$_\infty$, $\bar{\mathbf{z}}_i$ be its analogue for **SampleLattice**$_m$, and we use the following statistical distance :

$$\Delta\mathbf{z}_i = \Delta(\mathbf{z}_i, \bar{\mathbf{z}}_i) = \frac{1}{2} \sum_{\mathbf{y} \in \mathbb{Z}^{n-i+1}} |\Pr[\mathbf{z}_i = \mathbf{y}] - \Pr[\bar{\mathbf{z}}_i = \mathbf{y}]|$$

The statistical distance between the two algorithms is given by $\Delta\mathbf{z}_1 = \Delta(\mathbf{z}_1, \bar{\mathbf{z}}_1)$ as there is a one-to-one correspondence between the output $\mathbf{v}$ and the sampled $\mathbf{z} = (z_n \dots z_1)$.

Algorithm 20 ConditionalSample$_m$: Conditional Distribution of z_i

Input: same as **SampleLattice** plus an index $i : \mathbb{Z}$, and $\mathbf{y} = (y_n \dots y_{i+1}) : \mathbb{Z}^{n-i}$

1: $\mathbf{t} \leftarrow$ **DecomposeGS**$(\mathbf{c}, \mathbf{B}^\star) : \mathbb{FP}_m^n$
2: **for** $j = n$ downto $i + 1$ **do**
3: $\quad t_i = t_i - y_j \cdot \mu_{j,i} : \mathbb{FP}_m$
4: **end for**
5: $z_i \leftarrow$ **Sample**$\mathbb{Z}(\sigma_i, t_i, \tau)$
6: return z_i

Propagation of statistical distances. We define $z_i^{\mathbf{y}}$ (resp. $\bar{z}_i^{\mathbf{y}}$) as the distribution of z_i of the sampling algorithm $\mathbf{SampleLattice}_\infty$ (resp. in $\mathbf{SampleLattice}_m$), conditioned by $(z_n \dots z_{i+1}) = \mathbf{y}$. Those two distributions can be sampled by the $\mathbf{ConditionalSample}_\infty$ (resp. $\mathbf{ConditionalSample}_m$) version of Algorithm 20, and we define the associated statistical distance : $\Delta z_i^{\mathbf{y}} = \Delta(z_i^{\mathbf{y}}, \bar{z}_i^{\mathbf{y}})$. Similarly, we define $t_i^{\mathbf{y}}$ (resp. $\bar{t}_i^{\mathbf{y}}$) as value of t_i under the same condition $\mathbf{z}_{i+1} = \mathbf{y}$.

Fact 6.16 (Recursive bound on $\Delta\mathbf{z}_i$) *For all $i < n$ we have the following :*

$$\Delta\mathbf{z}_i \le \Delta\mathbf{z}_{i+1} + \sum_{\mathbf{y}\in\mathbb{Z}^{n-i}} \Pr[\mathbf{z}_{i+1} = \mathbf{y}]\, \Delta z_i^{\mathbf{y}}$$

Bounding $\Delta z_i^{\mathbf{y}}$. The statistical distance between the results comes from two sources : the intrinsic imperfectness of $\mathbf{Sample}\mathbb{Z}$ which use FPA operations compared to the exact distribution with similar parameters; and the FPA errors made while computing $\bar{\sigma}_i$ and $\bar{t}_i^{\mathbf{y}}$. More formally : $\Delta z_i^{\mathbf{y}} = \Delta(D_{\mathbb{Z},\sigma_i,t_i^{\mathbf{y}}}, \mathbf{Sample}\mathbb{Z}(\bar{\sigma}_i, \bar{t}_i^{\mathbf{y}}))$. We keep on decomposing the error using the following Fact :

Fact 6.17 (Asymptotic statistical distance of $\mathbf{Sample}\mathbb{Z}$) *Let δ_t, δ_σ be any implicit positive decreasing function of m that converge to 0 as $m \to \infty$, and let $\epsilon = 2^{1-m}$. Under the conditions $\sigma, \tau \ge 4$, for any $t \in \mathbb{R}$ and any (implicit functions of m) $\bar{t}, \bar{\sigma} \in \mathbb{FP}_m$ such that $|t - \bar{t}| + |t|\,\epsilon \le \delta_t$ and $|\sigma - \bar{\sigma}| \le \delta_\sigma$, the statistical distance $\Delta(D_{\mathbb{Z},\sigma,t}, \mathbf{Sample}\mathbb{Z}_m(\bar{\sigma}, \bar{t}))$ is upper bounded by :*

$$E_{\mathbf{Sample}\mathbb{Z}}(\sigma_i, \tau, \delta_t, \delta_\sigma, \epsilon) \stackrel{def}{=} \frac{3E_{\mathbf{tailcut}}(\tau, \delta_\tau)}{2} + \frac{\tau}{\sigma}\mathcal{O}(\delta_t) + \tau^3\sigma^2\mathcal{O}(\epsilon) + \tau^3\sigma\mathcal{O}(\delta_\sigma)$$

where $\delta_\tau = 2\delta_t/(\tau\sigma) + \delta_\sigma/\sigma + 2\epsilon$

A proper bound for $\delta_\sigma \le E_{\sigma_i} = \sigma_i\epsilon$ follows from assumptions $\mathcal{A}$. A bound on δ_t follows from the following Lemma :

Lemma 6.18 (Error during the sampling loop) *Under Conditions $\mathcal{A}$, and if $\|\mathbf{y}\|_1 \le \hat{y}$, the final error $\Delta t_i^{\mathbf{y}}$ made by the floating-point version of algorithm 20 is less than*

$$E_{\mathbf{loop}}(\hat{y}, \epsilon) \stackrel{def}{=} \left(\frac{4.3}{\check{r}}n^2q + 2.4n\hat{\mu}\hat{y} + 3.1\sqrt{n}q\hat{r}\right)\epsilon \qquad \le \left(\frac{n^2q}{\check{r}} + n\hat{\mu}\hat{y} + \sqrt{n}q\hat{r}\right)\mathcal{O}(\epsilon)$$

where $\hat{r} = \max_{i=1}^n(\|\mathbf{b}_i^\star\|)$, $\check{r} = \min_{i=1}^n(\|\mathbf{b}_i^\star\|)$ and $\hat{\mu} \ge |\mu_{i,j}|$ for any i, j.

Note that this bound verifies $E_{\mathbf{loop}}(\hat{y}, \epsilon) \ge |t^{\mathbf{y}}|\,\epsilon$ whenever $\|\mathbf{y}\|_1 \le \hat{y}$, allowing us to take $\delta_t = 2E_{\mathbf{loop}}(\hat{y}, \epsilon)$ when applying Fact 6.17. It remains to eliminate vectors $\mathbf{y}$ that are in the tail of the distribution using the following lemma.

Lemma 6.19 (Tailcut of $\mathbf{z}_i$ distribution) *Under Conditions $\mathcal{A}$, for $\hat{y} = n(q + \tau\sigma)\left\|\mathbf{B}^{-1}\right\|_s$, then we have for all $i < n$:*

$$\Pr[\|\mathbf{z}_i\|_1 \ge \hat{y}] \le 4E_{\mathbf{tailcut}}(\tau, 0).$$

We set $\hat{y} = n(q + \tau\sigma)\left\|\mathbf{B}^{-1}\right\|_s$, and $p_\tau = E_{\mathbf{tailcut}}(\tau, 2E_{\mathbf{loop}}(\hat{y}, \epsilon) + 3\epsilon) \ge E_{\mathbf{tailcut}}(\tau, 0)$. We may sum up those bounds and conclude.

$$\begin{aligned}
\Delta Z_1 &\le 4np_\tau + \sum_{i=1}^n E_{\mathbf{Sample}\mathbb{Z}}(\sigma_i, \tau, E_{\mathbf{loop}}(\hat{y}, \epsilon), E_{\sigma_i}, \epsilon) \\
&\le 4np_\tau + \sum_{i=1}^n \left(\frac{p_\tau}{2} + \frac{\tau}{\sigma_i}\mathcal{O}(E_{\mathbf{loop}}(\hat{y}, \epsilon)) + \tau^3\sigma_i^2\mathcal{O}(\epsilon) + \tau^3\sigma_i\mathcal{O}(\Delta\sigma_i)\right) \\
&\le 4np_\tau + \sum_{i=1}^n \left(\frac{p_\tau}{2} + \frac{\tau}{\sigma_i}\mathcal{O}(E_{\mathbf{loop}}(\hat{y}, \epsilon)) + \tau^3\sigma_i^2\mathcal{O}(\epsilon)\right) \\
E_{\mathbf{loop}}(\hat{y}) &= \left(n^2q\check{r}^{-1} + n^2\left\|\mathbf{B}^{-1}\right\|_s\hat{\mu}(q + \tau\sigma) + \sqrt{n}q\hat{r}\right)\mathcal{O}(\epsilon) \\
&\le n^3\hat{\mu}\left\|\mathbf{B}^{-1}\right\|_s(q + \tau\sigma)\mathcal{O}(\epsilon) \qquad \text{since } \sigma \ge 4\hat{r} \text{ and } \check{r}^{-1} \le n\hat{\mu}\left\|\mathbf{B}^{-1}\right\|_s
\end{aligned}$$

It remains to use the inequalities $\sigma/\check{r} \geq \sigma_i \geq 4$ from $\mathcal{A}$ to conclude :

$$\begin{aligned}\Delta Z_1 &\leq n\mathcal{O}(p_\tau) + \sum_{i=1}^{n}\left[n^3\hat{\mu}\left\|\mathbf{B}^{-1}\right\|_s(q+\tau\sigma) + \tau^3\sigma^2n^2\hat{\mu}^2\left\|\mathbf{B}^{-1}\right\|_s^2\right]\mathcal{O}(\epsilon)\\ &\leq n\mathcal{O}(p_\tau) + \left[n^4\hat{\mu}\left\|\mathbf{B}^{-1}\right\|_s(q+\tau\sigma) + \tau^3\sigma^2n^3\hat{\mu}^2\left\|\mathbf{B}^{-1}\right\|_s^2\right]\mathcal{O}(\epsilon).\end{aligned}$$

Note that $p_\tau = E_{\mathbf{tailcut}}(\tau, 2E_{\mathbf{loop}}(\hat{y},\epsilon)+3\epsilon)$ is less than $E_{\mathbf{tailcut}}(\tau, 0.1) \leq \tau e^{-\mathcal{O}(\tau^2)}$ for a large enough mantissa size m, thus for a certain constant C_τ, the condition $\tau \geq C_\tau\sqrt{\lambda\log n}$ ensures that the first term $(n\mathcal{O}(p_\tau))$ is less than $2^{-\lambda}/2$. Similarly, for a certain constant C_m, the condition $m \geq C_m + \lambda - 2\log_2(\left\|\mathbf{B}^{-1}\right\|_s) + \log_2\left(\hat{\mu}^2n^4(q+\sigma^2)\tau^3\right)$ ensures that the second term is less than $2^{-\lambda}/2$, which concludes the proof. □

6.8.2 Proof of Theorem 6.7

Here, we prove Theorem 6.7, restated for reader convenience :

Theorem 6.20 *There exist constants $C_\lambda, C_\tau, C_m, C_{m'}, C_{\delta_p}$, such that for any security parameter $\lambda \geq C_\lambda$, and under Conditions $\mathcal{A}$, The statistical distance between* $\mathbf{LazySampleLattice}_{m,m'}$ *and* $\mathbf{SampleLattice}_\infty$ *is less than $2^{-\lambda}$ on the same input if the following conditions are satisfied :*

$$\begin{aligned}\tau &\geq C_\tau\sqrt{\lambda\log n}\\ m &\geq C_m + \lambda + 2\log_2(\left\|\mathbf{B}^{-1}\right\|_s) + \log_2\left(\hat{\mu}^2n^4q\sigma^2\tau^3\right)\\ m' &\geq C_{m'} + 2\log_2(\left\|\mathbf{B}^{-1}\right\|_s) + \log_2\left(\hat{\mu}^2n^4(Q+\sigma^2)\tau^3\right)\\ \delta_p &\geq 2^{-k} \text{ where } k = m' - \left(C_{\delta_p} + 2\log_2(\left\|\mathbf{B}^{-1}\right\|_s) + \log_2\left(\hat{\mu}^2n^3\tau\sigma^2q\right)\right)\end{aligned}$$

Furthermore, under those conditions, the integers manipulated by the algorithm can be represented by low-precision floating-point numbers ($\mathbb{FP}_{m'}$) without errors.

Proof: We proceed by proving that the condition on δ_p is sufficient to apply Lemma 6.23. The rest of the proof is similar to the proof of Theorem 6.3 on the correctness of $\mathbf{SampleLattice}_m$.

As in the proof of Theorem 6.3, we take $\hat{y} = n(q+\tau\sigma)\left\|\mathbf{B}^{-1}\right\|_s$ and using Lemma 6.19 and Fact 6.18, we obtain that the error made on t_i during its computation at low precision is less than $E_{\mathbf{loop}}(\hat{y},\epsilon') = n^3\hat{\mu}(q+\tau\sigma)\left\|\mathbf{B}^{-1}\right\|_s\mathcal{O}(\epsilon')$, except with probability less than $4E_{\mathbf{tailcut}}(\tau,0)$. The error on the low precision σ_i is less than $\sigma_i\epsilon'$. We recall that $\sigma/\check{r} \geq \sigma \geq 1$ and that $\check{r} \leq (1+n\hat{\mu})\left\|\mathbf{B}^{-1}\right\|_s$. Thus, the following parameters are valid to apply Lemma 6.23 :

$$\begin{aligned}\delta_t &= 2E_{\mathbf{loop}}(\hat{y},\epsilon') = n^3\hat{\mu}(q+\tau\sigma)\left\|\mathbf{B}^{-1}\right\|_s\mathcal{O}(\epsilon')\\ \delta_{\sigma_i} &= n\sigma\hat{\mu}\left\|\mathbf{B}^{-1}\right\|_s\mathcal{O}(\epsilon')\\ \delta_p &= n^3(\tau\sigma^2+q)\hat{\mu}^2\left\|\mathbf{B}^{-1}\right\|_s^2\mathcal{O}(\epsilon')\end{aligned}$$

The rest of the proof is similar to the proof of Theorem 6.3. □

6.8.3 Errors During Gaussian Sampling over $\mathbb{Z}$

The following fact have been simplified to improve readability : the result is only given asymptotically and is not tight, and a factor $\approx \tau^2$ is lost compared to our optimal proof.

Fact 6.21 (Asymptotical error during the computation of ρ) *Let δ_t, δ_σ be any implicit positive decreasing functions of m that converge to 0 as $m \to \infty$; as well as $\epsilon = 2^{1-m}$. Under the condition $\sigma \geq 4$, for any $x \in \mathbb{Z}$, any $t \in \mathbb{R}$ and any (implicit functions of m) $\bar{t}, \bar{\sigma}$ such that $|t-\bar{t}| \leq \delta_t$ and $|\sigma - \bar{\sigma}| \leq \delta_\sigma$, then we have the following asymptotical bound on the error $\Delta\rho_{\sigma,t}(x) = \left|\rho_{\sigma,t}(x) - \bar{\rho}_{\bar{\sigma},\bar{t}}(x)\right|$ using $\mathbb{FP}_m$ arithmetic :*

$$\Delta\rho_{\sigma,t}(x) \leq \mathcal{O}(\delta_t)/\sigma + (1+(x-t)^2)\mathcal{O}(\epsilon) + ((x-t)^2/\sigma)\mathcal{O}(\delta_\sigma) = E_\rho(\sigma,\delta_t,\delta_\sigma)$$

Proof: We will often use the following elementary fact : for any (implicit) function x of m, we have $\mathcal{O}(x(1+o(1))) = \mathcal{O}(x)$. Note that by definition, $\epsilon, \delta_t, \delta_\sigma$ are $o(1)$. The constraint on σ from $\mathcal{A}$ ensures that $x/\sigma \leq x/4$: thus, writing $\mathcal{O}(x)/\sigma = \mathcal{O}(x)$ is allowed, as the function behind $\mathcal{O}$ can be made independent of σ.

The error during the computation of σ^2, according to Fact 6.12 is less than $\Delta\sigma^2 = \sigma\mathcal{O}(\delta_\sigma) + \sigma^2\mathcal{O}(\epsilon)$. The error on the constant π is $\mathcal{O}(\epsilon)$, and we can apply Fact 6.13 to derive an error bound on h : $\Delta h = \left|h - \bar{h}\right| \leq \mathcal{O}(\epsilon) + \mathcal{O}(\delta_\sigma)/\sigma$.

For sufficiently large m, we have $|x| \leq 2^m$, which ensures that there is no error made during the implicit conversion of x to a floating-point number, i.e. $\Delta[x] = |x - \bar{x}| = 0$.

We derive successively the following bounds :

$$\begin{aligned}
\Delta[x-t] &= |(x-t) - (\bar{x}\bar{-}\bar{t})| \leq \mathcal{O}(\delta_t) + |x-t|\,\mathcal{O}(\epsilon) \\
\Delta[(x-t)^2] &= \left|(x-t)^2 - (\bar{x}\bar{-}\bar{t})^{\bar{\times}2}\right| \leq 2\,|x-t|\,\Delta[x-t] + (|x-t| + \Delta[x-t])^2\epsilon \\
&\leq |x-t|\,\mathcal{O}(\delta_t) + (x-t)^2\mathcal{O}(\epsilon) \\
\Delta[h\cdot(x-t)^2] &= \left|h\cdot(x-t)^2 - \bar{h}\bar{\times}(\bar{x}\bar{-}\bar{t})^{\bar{\times}2}\right| \\
&\leq |h|\,\Delta[(x-t)^2] + (x-t)^2\Delta[h] + (h+\Delta[h])((x-t)^2 + \Delta[(x-t)^2])\epsilon \\
&\leq 1/\sigma^2\left(\mathcal{O}(\delta_t) + (x-t)^2\mathcal{O}(\epsilon)\right) + (x-t)^2\left(\mathcal{O}(\epsilon) + \mathcal{O}(\delta_\sigma)/\sigma\right) + ((x-t)^2/\sigma^2)\mathcal{O}(\epsilon) \\
&\leq \mathcal{O}(\delta_t)/\sigma + (x-t)^2\mathcal{O}(\epsilon) + ((x-t)^2/\sigma)\mathcal{O}(\delta_\sigma)
\end{aligned}$$

We then use the assumption on the floating-point implementation of $\bar{\exp}$, and conclude using the fact that exp is 1-Lipschitzian over $]-\infty, 0]$

$$\begin{aligned}
\Delta\rho_{\sigma,t}(x) = \left|\rho_{\sigma,t}(x) - \bar{\rho}_{\bar{\sigma},\bar{t}}(x)\right| &= \left|\exp(h\cdot(x-t)^2) - \bar{\exp}\left(\bar{h}\bar{\times}(\bar{x}\bar{-}\bar{t})^2\right)\right| \\
&\leq \left|\exp\left(h\cdot(x-t)^2\right) - \exp\left(\bar{h}\bar{\times}(\bar{x}\bar{-}\bar{t})^2\right)\right| + \epsilon \leq \Delta[h\cdot(x-t)^2] + \epsilon.
\end{aligned}$$

□

However, with more technicalities one may prove the following.

Fact 6.22 (Error during ρ computation) *Let $m \in \mathbb{Z}$ be a positive integer and $\epsilon = 2^{1-m}$. Let $\bar{t}, \bar{\sigma} \in \mathbb{FP}_m$ be at distance respectively at most δ_t and δ_σ from $t, \sigma \in \mathbb{R}$. We also assume that the following inequalities hold : $\sigma \geq 4$, $\sigma\delta_\sigma \leq 0.01, \delta_t \leq 0.01, \sigma^2\epsilon \leq 0.01$. We then have the following error bound on $\Delta\rho_{\sigma,t}(x) = \left|\rho_{\sigma,t}(x) - \bar{\rho}_{\bar{\sigma},\bar{t}}(x)\right|$ for any integer x such that $|x| \leq 2^m$:*

$$\Delta\rho_{\sigma,t}(x) \leq 10\sigma^2\epsilon + 4.3\sigma\delta_\sigma + \frac{.7}{\sigma}\delta_t$$

Proof of Fact 6.17 : Error during the Sampling over $\mathbb{Z}$

Proof: Let $x_{\min} = t - \tau\sigma$ and $\bar{x}_{\min} = \bar{t}\bar{-}\tau\bar{\times}\bar{\sigma}$. The error $\Delta x_{\min} = |x_{\min} - \bar{x}_{\min}|$ is less than $\delta_t + \tau\delta_\sigma + 2\tau\sigma\epsilon = \tau\sigma\delta_\tau$. One derive a similar error bound for $x_{\max} = t + \tau\sigma$.

We now define $X = \mathbb{Z} \cap [\bar{x}_{\min}, \bar{x}_{\max}]$ and bounds $p_{\neg X} = \Pr\left[\neg(x \in X) | x \leftarrow D_{\mathbb{Z},\sigma,t}\right]$ using Corollary 6.25 : $p_{\neg X} \leq 3E_{\mathbf{tailcut}}(\tau, \delta_\tau) \leq 3E_{\mathbf{tailcut}}(4, 0.1) \leq 0.01$ as $\delta_\tau \leq 0.1$ for sufficiently large m.

We set $R_X = \rho_{\sigma,t}(X)$, $R'_X = \bar{\rho}_{\bar{\sigma},\bar{t}}(X)$, define $\Delta\rho(x) = \left|\rho_{\sigma,t}(x) - \bar{\rho}_{\bar{\sigma},\bar{t}}(x)\right|$. Note that $|R_X - R'_X| \leq \Delta\rho(X) \stackrel{\text{def}}{=} \sum_{x\in X}\Delta\rho(x)$. We now focus on decomposing the main error term :

$$\begin{aligned}
&2\Delta(D_{\mathbb{Z},\sigma,t}, \mathbf{Sample}\mathbb{Z}_m(\bar{\sigma}, \bar{t}, \tau)) \\
&\leq \sum_{x\in\mathbb{Z}} |D_{\mathbb{Z},\sigma,t}(x) - \Pr\left[\mathbf{Sample}\mathbb{Z}_m(\bar{\sigma}, \bar{t}, \tau) = x\right]| \leq p_{\neg X} + \sum_{x\in X}\left|\frac{\rho_{\sigma,t}(x)}{R_X} - \frac{\bar{\rho}_{\bar{\sigma},\bar{t}}(x)}{R'_X}\right| \\
&\leq p_{\neg X} + \frac{1}{R_X R'_X}\sum_{x\in X}\left|R'_X\rho_{\sigma,t}(x) - R_X\bar{\rho}_{\bar{\sigma},\bar{t}}(x)\right| \\
&\leq p_{\neg X} + \frac{1}{R_X R'_X}\left[\Delta\rho(X)R'_X + |R_X - R'_X|\sum_{x\in X}\rho_{\sigma,t}(x)\right] \\
&\leq p_{\neg X} + \frac{1}{R_X R'_X}\left[\Delta\rho(X)R'_X + \Delta\rho(X)R_X\right] \leq p_{\neg X} + \left(\frac{1}{R'_X} + \frac{1}{R_X}\right)\Delta\rho(X)
\end{aligned}$$

We can bound $\Delta\rho(X)$ using Lemma 6.21 :

$$\begin{aligned}\Delta\rho(X) &\leq \sum_{x\in X} \Delta\rho_{\sigma,t}(x)\\ &\leq \sum_{x\in X} \mathcal{O}(\delta_t)/\sigma + (1+(x-t)^2)\mathcal{O}(\epsilon) + ((x-t)^2/\sigma)\mathcal{O}(\delta_\sigma)\\ &\leq \#X\left(\mathcal{O}(\delta_t)/\sigma + (1+\tau^2\bar{\sigma}^2)\mathcal{O}(\epsilon) + (\tau^2\bar{\sigma}^2/\sigma)\mathcal{O}(\delta_\sigma)\right)\\ &\leq \tau\mathcal{O}(\delta_t) + \tau^3\sigma^3\mathcal{O}(\epsilon) + \tau^3\sigma^2\mathcal{O}(\delta_\sigma)\end{aligned}$$

We now study lower bounds for R_X and R'_X. We have :

$$R_X = \sum_{x\in X} e^{-\frac{(x-t)^2}{\sigma^2}} \geq (1-p_{\neg X})\rho_{\sigma,t}(\mathbb{Z}).$$

For $\iota = 0.01$, Lemma 3.18 shows $\eta_\iota(\mathbb{Z}) \leq 2.6 \leq 4 \leq \sigma$. As $\int_0^1 \rho_{\sigma,t}(\mathbb{Z})dt = \int_{-\infty}^{\infty}\rho_{\sigma,0}(x)dx = \sigma$, the bound from Lemma 3.17 gives $\sigma \in \left[\frac{1-\iota}{1+\iota}, 1\right]\rho_{\sigma,0}(\mathbb{Z})$ and $\rho_{\sigma,t}(\mathbb{Z}) \in \left[\frac{1-\iota}{1+\iota}, 1\right]\rho_{\sigma,0}(\mathbb{Z})$, which let us conclude

$$R_X \geq (1-p_{\neg X})\left(\frac{1-\iota}{1+\iota}\right)^2\sigma \geq 0.9\sigma$$

.

For R'_X we use the fact that $|R_X - R'_X| \leq \Delta\rho(X)$, thus, for large enough m : $R'_X \geq 0.8\sigma$. We can conclude :

$$\Delta(D_{\mathbb{Z},\sigma,t}, \mathbf{Sample}\mathbb{Z}_m(\bar{\sigma},\bar{t},\tau)) \leq \frac{3E_{\mathbf{tailcut}}(\tau,\delta_\tau)}{2} + \frac{\tau}{\sigma}\mathcal{O}(\delta_t) + \tau^3\sigma^2\mathcal{O}(\epsilon) + \tau^3\sigma\mathcal{O}(\delta_\sigma)$$

□

Error during the Lazy Sampling over $\mathbb{Z}$

Lemma 6.23 *Let $m, m' \in \mathbb{Z}$ be positive integers, with $m > m'$ and $\epsilon = 2^{1-m}, \epsilon' = 2^{1-m'}$. Let $t, \sigma \in \mathbb{FP}_m$ and $\tau, t', \sigma', \delta_p \in \mathbb{FP}_{m'}$ such that $|t-t'|+|t|\,\epsilon' \leq \delta_t$ and $|\sigma-\sigma'| \leq \delta_\sigma$, $\sigma \geq 1$. We assume $E_{\mathbf{tailcut}}(\tau,\delta_\tau) \leq 0.001$ with $\delta_\tau = \delta_t/(\tau\sigma) + \delta_\sigma/\sigma + 2\epsilon'$ as well as $|t'| \bar{+} \tau \bar{\times} \sigma' \leq 2^{m'}$. Finally, let $F = () \mapsto t$ be a constant function of type $(void \to \mathbb{FP}_m)$. If the parameter δ_p verifies $\delta_p \geq 4\sigma^2\epsilon' + 1.7\sigma\delta_\sigma + (1.7/\sigma)\delta_t$, we have :*

$$\Delta(D_{\mathbb{Z},\sigma,t}, \mathbf{LazySample}\mathbb{Z}_{m',m}(\sigma',\tau,t',\delta_p,\sigma,F)) \leq \frac{3E_{\mathbf{tailcut}}(\tau,\delta_\tau)}{2} + \frac{\tau}{\sigma}\mathcal{O}(\delta_t) + \tau^3\sigma^2\mathcal{O}(\epsilon) + \tau^3\sigma\mathcal{O}(\delta_\sigma)$$

Proof: Let $x_{\min} = t - \tau\sigma'$ and $x'_{\min} = t' \bar{-} \tau \bar{\times} \bar{\sigma}'$. The error $\Delta x_{\min} = |x_{\min} - x'_{\min}|$ is less than $\delta_t + \tau\delta_\sigma + 2\tau\sigma\epsilon' = \tau\sigma\delta_\tau$. We have a similar error bound for $x_{\max} = t + \tau\sigma$. We now define $X = \mathbb{Z} \cap [x'_{\min}, x'_{\max}]$ and bound $p_{\neg X} = P(\neg(x \in X) | x \leftarrow D_{\mathbb{Z},\sigma,t})$ using Corollary 6.25 : $p_{\neg X} \leq 3E_{\mathbf{tailcut}}(\tau,\delta_\tau)$. We denote by $\bar{\rho}$ (resp. $\bar{\rho}'$) the computation of ρ at precision m (resp. m'). For $x \in X$, we obtain using Fact 6.22 twice :

$$\begin{aligned}\Delta\rho(x) &= \left|\bar{\rho}_{\sigma,t} - \bar{\rho}'_{\sigma',t'}\right|\\ &\leq |\bar{\rho}_{\sigma,t}(x) - \rho_{\sigma,t}(x)| + \left|\rho_{\sigma,t}(x) - \bar{\rho}'_{\sigma',t'}(x)\right|\\ &\leq 1.6\sigma^2\epsilon + 1.6\sigma^2\epsilon' + 1.7\sigma\delta_\sigma + \frac{1.7}{\sigma}\delta_t\\ &\leq 3.2\sigma^2\epsilon' + 1.7\sigma\delta_\sigma + \frac{1.7}{\sigma}\delta_t\end{aligned}$$

We note that $\delta_p \geq \Delta\rho(x) + 2\epsilon'$, which ensures that high precision is triggered whenever the random real $r' \in [0,1]$ falls in the interval $[\rho'_{\sigma',t'}(x) - \Delta\rho(x), \rho'_{\sigma',t'}(x) + \Delta\rho(x)]$. Thus, whatever the value of the boolean variable highprec, and for some chosen $x \in X$, the behavior of the main loop of $\mathbf{LazySample}\mathbb{Z}_{m',m}(\sigma',\tau,t',\delta_p,\sigma,F)$ is similar to the original $\mathbf{Sample}\mathbb{Z}_m(\sigma,t,\tau)$.

The rest of the proof is similar to the one of Fact 6.17. □

6.8.4 Error during the Sampling Loop

Fact 6.24 (Error during the Decomposition over the Gram-Schmidt) *Under conditions $\mathcal{A}$, the error Δt_i^I made by the floating-point version of* **DecomposeGS** *algorithm on the i^{th} component is less than*

$$E_{\mathbf{dec}}(\epsilon) = \left(\frac{3.5}{\check{r}} n^2 q + 3\sqrt{n} q \hat{r}\right)\epsilon \qquad \leq q\left(n^2 \check{r}^{-1} + \sqrt{n}\hat{r}\right)\mathcal{O}(\epsilon)$$

Proof: We bound the error on t_i^I after its decomposition over the Gram-Schmidt basis. First, let us analyze the error on the sum for $y_i = \sum_{j=1}^{n} x_i b_{i,j}^\star$.

The values x_i's are integers computed exactly, bounded by q. This leads to a maximum error of $q\epsilon$ when converted to float, thus the error on the product $x_i b_{i,j}^\star$ is by Fact 6.12 less than $2qr_i\epsilon + q(1+\epsilon)r_i(1+\epsilon)\epsilon$. As $\epsilon \leq 0.01$ we can bound this error by the simpler value $\delta_i = 3.03qr_i\epsilon$. Finally, the sum $S = \sum_{j=1}^{n} \left|x_j b_{i,j}^\star\right|$ is bounded by nqr_i.

We can now apply Lemma 6.14 and deduce :

$$\begin{aligned} |\Delta y_i| &\leq n^2 K^n q r_i \epsilon + (1 + nK^n\epsilon)\sum_{i=1}^{i}\delta_i \\ &\leq 1.1n^2 q r_i \epsilon + 1.01n(3.03 q r_i \epsilon) \qquad \leq 1.5 n^2 q r_i \epsilon \end{aligned}$$

Note that this computation intends to compute $y_i = \langle \mathbf{c}, \mathbf{b}_i^\star \rangle$, thus $|y_i| \leq \|\mathbf{c}\| \times \|\mathbf{b}_i^\star\| \leq \sqrt{n} q r_i$. This bound also implies $t_i^I \leq \sqrt{n} q / \check{r}$. To conclude, we use Fact 6.13 :

$$\begin{aligned} \left|\Delta t_i^I\right| &\leq \frac{2}{r_i^2}\left|\Delta y_i\right| + 2\frac{|y_i|}{r_i^2}\epsilon r_i^2 + 2\frac{2\left|\Delta y_i\right| + |y_i|\,(1+2\epsilon)}{r_i^2}\epsilon \leq \frac{2+4\epsilon}{r_i^2}\left|\Delta y_i\right| + 3\left|y_i\right|\epsilon + \frac{2.04\left|y_i\right|}{r_i^2}\epsilon \\ &\leq \frac{3.1}{r_i} n^2 q\epsilon + 3\sqrt{n} q r_i \epsilon + \frac{3\sqrt{n}q}{r_i}\epsilon \leq \frac{3.5}{\check{r}} n^2 q\epsilon + 3\sqrt{n} q \hat{r}\epsilon = E_{\mathbf{dec}}(\epsilon) \end{aligned}$$

□

Proof of Fact 6.18 We first unroll the computation of t_i^y as : $t_i^y = t_i^I - \sum_{j>i} y_j \mu_{i,j}$, in this summation, the errors on the inputs are bounded by :

- $\delta_0 = E_{\mathbf{dec}}(\epsilon)$ the bound previously proven for the term t_i^I in Fact 6.24.
- $2\left|y_i\right|\hat{\mu}\epsilon + \left|y_i\right|(1+\epsilon)\hat{\mu}(1+\kappa)\epsilon \leq \left|y_i\right| 3.03\hat{\mu}\epsilon = \delta_i$ for the term $\mu_{i,j} y_i$.

Furthermore, the sum $S = \left|t_i^I\right| + \sum_{j>i}\left|y_j \mu_{i,j}\right|$ is less than $\frac{\sqrt{n}q}{r_i} + \hat{\mu}\left\|\mathbf{y}\right\|_1$. We apply Lemma 6.14 once again and deduce :

$$\begin{aligned} |\Delta t_i^y| &\leq \epsilon n K^n \left(\frac{\sqrt{n}q}{r_i} + \hat{\mu}\left\|\mathbf{y}\right\|_1\right) + (1 + \epsilon n K^n)\left(E_{\mathbf{dec}}(\epsilon) + \sum_{j=1}^{i} 3.03\left|y_i\right|\hat{\mu}\epsilon\right) \\ &\leq 1.1n\left(\frac{\sqrt{n}q}{r_i} + \hat{\mu}\left\|\mathbf{y}\right\|_1\right)\epsilon + (1+2n\epsilon)\left(E_{\mathbf{dec}}(\epsilon) + 3.03\hat{\mu}\left\|\mathbf{y}\right\|_1\epsilon\right) \\ &\leq 1.1n\left(\frac{\sqrt{n}q}{r_i} + \hat{\mu}\left\|\mathbf{y}\right\|_1\right)\epsilon + 1.01 E_{\mathbf{dec}}(\epsilon) + 3.1\hat{\mu}\left\|\mathbf{y}\right\|_1\epsilon \\ &\leq \frac{1.1}{\check{r}} n^{3/2} q\epsilon + 2n\hat{\mu}\left\|\mathbf{y}\right\|_1\epsilon + \frac{3.6}{\check{r}} n^2 q\epsilon + 3.1\sqrt{n} q\hat{r}\epsilon + 3.1\hat{\mu}\left\|\mathbf{y}\right\|_1\epsilon \\ &\leq \left(\frac{4.3}{\check{r}} n^2 q + 2.4 n\hat{\mu}\hat{y} + 3.1\sqrt{n} q\hat{r}\right)\epsilon = E_{\mathbf{loop}}(\hat{y}, \epsilon) \end{aligned}$$

□

6.8.5 Other proofs

Corollary 6.25 (Tailcut error, Corollary of [MR04, Lemma 2.10]) *Let L be an n-dimensional lattice, $\iota \leq 1/2$, $\sigma \geq \eta_\iota(L)/\sqrt{2\pi}$, $\tau > 1$ $\delta_\tau \in (0,1)$ and $\mathbf{c} \in \mathbb{R}^n$. For $x \leftarrow D_{L,\sigma,\mathbf{c}}$ we have :*

$$\Pr\left[\|\mathbf{x} - \mathbf{c}\| \geq (1-\delta_\tau)\tau\sigma\right] \leq 3 E_{\mathbf{tailcut}}(\tau, \delta_\tau)^n$$

where $E_{\mathbf{tailcut}}(\tau, \delta_\tau) \stackrel{def}{=} \tau\sqrt{2\pi e}\cdot e^{-(1-\delta_\tau)^2\tau^2/2}$.

Proof of Fact 6.16

$$\begin{aligned}
2\Delta\mathbf{z}_i &= \sum_{\mathbf{x}\in\mathbb{Z}^{n-i}}\sum_{y\in\mathbb{Z}}\Big|\Pr[\mathbf{z}_i=(x_n,\ldots,x_{i+1},y)]-\Pr[\bar{\mathbf{z}}_i=(x_n,\ldots,x_{i+1},y)]\Big| \\
&= \sum_{\mathbf{x}\in\mathbb{Z}^{n-i}}\sum_{y\in\mathbb{Z}}\Big|\Pr[\mathbf{z}_{i+1}=\mathbf{x}]\Pr[z_i^{\mathbf{x}}=y]-\Pr[\bar{\mathbf{z}}_{i+1}=\mathbf{x}]\Pr[\bar{z}_i^{\mathbf{x}}=y]\Big| \\
&= \sum_{\mathbf{x}\in\mathbb{Z}^{n-i}}\sum_{y\in\mathbb{Z}}\Big|\left(\Pr[\mathbf{z}_{i+1}=\mathbf{x}]-\Pr[\bar{\mathbf{z}}_{i+1}=\mathbf{x}]\right)\Pr[\bar{z}_i^{\mathbf{x}}=y]\Big| \\
&\qquad\qquad +\Big|\left(\Pr[z_i^{\mathbf{x}}=y]-\Pr[\bar{z}_i^{\mathbf{x}}=y]\right)\Pr[\mathbf{z}_{i+1}=\mathbf{x}]\Big| \\
&\le \sum_{\mathbf{x}\in\mathbb{Z}^{n-i}}\left[\Big|\Pr[\mathbf{z}_{i+1}=\mathbf{x}]-\Pr[\bar{\mathbf{z}}_{i+1}=\mathbf{x}]\Big|\sum_{y\in\mathbb{Z}}\Pr[\bar{z}_i^{\mathbf{x}}=y]\right] \\
&\quad + \sum_{\mathbf{x}\in\mathbb{Z}^{n-i}}\left[\Pr[\mathbf{z}_{i+1}=\mathbf{x}]\sum_{y\in\mathbb{Z}}\Big|\Pr[z_i^{\mathbf{x}}=y]-\Pr[\bar{z}_i^{\mathbf{x}}=y]\Big|\right]
\end{aligned}$$

Using the fact that $\sum_{y\in\mathbb{Z}}\Pr[\bar{z}_i^{\mathbf{x}}=y]=1$, we conclude that :

$$\Delta\mathbf{z}_i \le \Delta\mathbf{z}_{i+1}+\sum_{\mathbf{x}\in Z^{n-i}}\Pr[\mathbf{z}_{i+1}=\mathbf{x}]\,\Delta z_i^{\mathbf{x}}\square$$

Proof of Lemma 6.19 By the definition of the exact algorithm, the distribution of $\mathbf{x}\leftarrow\mathbf{z}_1$ is such that $\mathbf{v}=\mathbf{x}\cdot B$ approximately follows the distribution $D_{L,\mathbf{c},\sigma}$. More precisely, $\Delta(\mathbf{v},D_{L,\mathbf{c},\sigma})\le 2n\iota\le 0.02$ according to the proof of Theorem 6.1 and assumption $\mathcal{A}$.

We apply Fact 6.25 to deduce :

$$\Pr[\|\mathbf{x}-\mathbf{c}\|\ge\tau\sigma]\le 1.02\cdot 3E_{\mathbf{tailcut}}(\tau,0)^n\le 4E_{\mathbf{tailcut}}(\tau,0)$$

thus, except with probability less than $4E_{\mathbf{tailcut}}(\tau,0)$ we have $\|\mathbf{v}-\mathbf{c}\|\le\sqrt{n}\tau\sigma$. In this case $\|\mathbf{v}\|\le\sqrt{n}\,(q+\tau\sigma)$ and $\|\mathbf{x}\|=\left\|\mathbf{v}\cdot\mathbf{B}^{-1}\right\|\le\|\mathbf{v}\|\left\|\mathbf{B}^{-1}\right\|_s\le\sqrt{n}\,(q+\tau\sigma)\left\|\mathbf{B}^{-1}\right\|_s$.

It remains to observe that $\mathbf{x}'\leftarrow z_i$ is identical to $\mathbf{x}'=(x_i,\ldots x_n)$ with $\mathbf{x}=(x_1,\ldots x_n)\leftarrow z_1$ to conclude :

$$\|\mathbf{x}'\|_1\le\sqrt{n-i+1}\,\|\mathbf{x}\|\le\sqrt{n}\,\|\mathbf{x}\|\le n\left\|\mathbf{B}^{-1}\right\|_s(q+\tau\sigma)$$

except with probability less than $4E_{\mathbf{tailcut}}(\tau,0)$. □

6.9 Concrete Mantissa Size Requirement

6.9.1 Concrete Bounds For FPA-Klein Algorithm and its Lazy Variant

High precision Our concrete version of Theorems 6.3 and 6.7 bounds the statistical distance Δ between $\mathbf{SampleLattice}_m$ and $\mathbf{SampleLattice}_\infty$:

$$\begin{aligned}
\Delta\le&\left(32\,\frac{\sigma^2}{\check{r}^2}+\frac{4.7\left(2.4\,(\sigma\tau_2+q)\hat{\mu}n^2\left\|B^{-1}\right\|_s+4.3\,n^2q/\check{r}+3.1\sqrt{n}q\hat{r}\right)\hat{r}}{\sigma}+2.3\,\tau\right)n\epsilon \\
&+4n\,E_{\mathbf{tailcut}}(\tau_2,0)^n+3/2nE_{\mathbf{tailcut}}(\tau,.01)
\end{aligned}$$

Note that we introduce a second tailcut parameter τ_2 which does not appear in the algorithm, that can be chosen smaller than τ, giving a little concrete improvement of about 4 bits. This might seem marginal for high-precision computations, but this small number of bits may be critical at low precision.

Low Precision Similarly, we derive a concrete bound for the correctness condition of the Lazy Algorithm 6.7 **LazySampleLattice**.

$$\delta_p\ge\left(14.3\,\frac{\sigma^2}{\check{r}^2}+\frac{1.7\left(2.4\,(\sigma\tau_2+q)\hat{\mu}n^2\left\|B^{-1}\right\|_s+4.3\,n^2q/\check{r}+3.1\sqrt{n}q\hat{r}\right)\hat{r}}{\sigma}\right)\epsilon'$$

6.9.2 Concrete Bounds For Peikert's Offline Algorithm and its Lazy Variant

Concrete bounds to corresponding to the proof sketch of Sec. 6.4.2 are as follow :

$$\begin{aligned}\Delta \leq & 4np_\tau + \frac{3}{2}nE_{\mathbf{tailcut}}(\tau, .01) + \left(\frac{9.5n\left\|B\right\|_s\tau}{\eta} + 6.1\eta^2 + 2.3\tau\right)n\epsilon \\ \delta_p \geq & \left(1 + 4\eta^2 + \frac{3.5n\left\|B\right\|_s\tau}{\eta}\right)\epsilon'\end{aligned}$$

Chapter 7

Discrete Gaussian Sampling without Floating-Point Arithmetic

Une inégalité moche ne peut être optimale.
Nicolas Gama [Gam08].

Résumé

La première section de ce chapitre fait partie de la publication *Lattice Signature and Bimodal Gaussian*, co-signé avec A. Durmus, T. Lepoint et V.Lyubashevsky et publiée à Crypto 2013 ; les autres contributions de cette publication font l'objet du prochain chapitre. La seconde section présente des travaux en cours.

Au cours du chapitre précèdent, nous avons étudié la propagation d'erreur au sein des variantes flottantes des algorithmes de tirage aléatoire gaussian, celui de Klein et celui de Peikert. Bien que nous ayons amélioré l'efficacité théorique et pratique de ces algorithmes l'utilisation de nombres flottants reste un problème pour les petites architectures tel que les cartes à puce. Cependant, pour les signatures, il n'est pas nécessaire de recourir à des algorithmes aussi généraux ; en suivant l'approche proposée dans [Lyu12] plutôt que celle de [GPV08] (c'est-à-dire une approche à la Fiat-Shamir plutôt que *hacher-puis-signer*). Pour la mise en oeuvre du schéma de Lyubashevsky [Lyu12], il suffit de tirages aléatoires selon $D_{Z,\sigma}$, c'est-à-dire une Gaussienne discrète unidimensionnelle et centrée ; ainsi que rejet avec une probabilité impliquant la fonction exponentielle. Dans la section 7.1, nous développerons des algorithmes spécialisés pour ces tâches sans recours à des opérations sur des nombres flottants. L'outil principal de ce chapitre est la technique de tirage aléatoire avec rejet (voir lemme 2.7) qui permet dans un premier temps d'approcher une distribution (de façon prévisible) et ensuite de rectifier l'erreur en rejetant certains tirages.

En utilisant la même approche, nous montrons en section 7.2 qu'il est possible d'éviter toute utilisation d'arithmétique flottante pour le tirage selon $D_{L,\sigma,\mathbf{c}}$ lorsque le réseau L est q-aire et que la cible $\mathbf{c}$ est entière ; et ceux en proposant une nouvelle variante de l'algorithme de Klein. En theorie, ces nouveaux resultats offre la mme complexite asymptotique que la version flottante et parresseuse du chapitre 7 ; mais il est fort possible que ces nouvelles techniques s'avère meilleure en pratique, surtout sur architecture restreinte. De surcroît, nous espérons que cette nouvelle technique s'applique aussi à la phase hors-ligne de l'algorithme de Peikert.

Abstract

The first section of this chapter was part of the publication *Lattice Signature and Bimodal Gaussian*, cosigned with A. Durmus, T. Lepoint et V.Lyubashevsky and published at Crypto 2013 ; the other contributions of this publication are given in the next chapter. The second section presents work in progress.

In the previous chapter we have studied the error propagation in the FPA variants of Klein's and Peikert's Gaussian Sampler. While we were able to improve the theoretical and practical efficiency of those algorithms ; the use of FPA makes Gaussian Sampling still problematic on small architectures

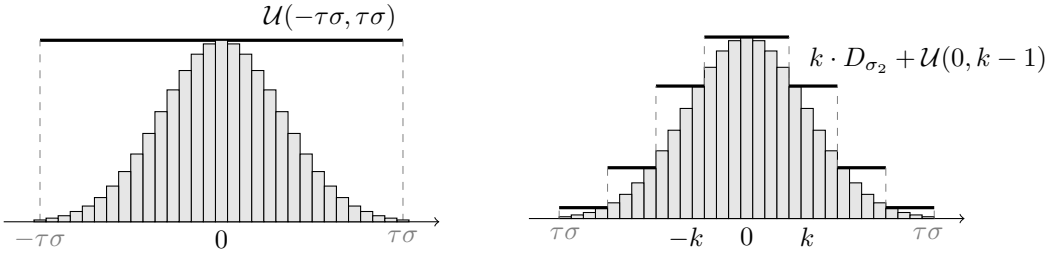
such as smartcards. Yet, for signatures, one may not need such general Gaussian Sampling ; by following the approach of [Lyu12] rather than [GPV08] (in other words, using Fiat-Shamir paradigm rather than *hash-then-sign*). To implement the scheme of Lyubashevsky [Lyu12], one only requires a sampler for $D_{Z,\sigma}$, that is centered one-dimensional discrete Gaussian ; but also to reject according to a probability involving the exponential function. In section 7.1, we will develop specialized algorithm for those task that do not involve Floating-Point operations. The main tool of this Chapter is the Rejection Sampling technique (Lemma 2.7), that allows us to approximate distribution (in a precisely predictable way) and then rectify the error by rejection some of the samples.

Using the same approach, we show in 7.2 that it is possible to avoid the use of Floating Point Arithmetic also for general Gaussian Sampling $D_{L,\sigma,\mathbf{c}}$ of an integer q-ary lattices L and an integer target $\mathbf{c}$; by another variant of Klein's algorithm. In theory, we reach the same asymptotical complexity as the FPA lazy version of chapter 7 ; but it quite plausible that this new techniques would be better in practice, especially on constrained devices. We also hope that this new technique can be adapted to Peikert's offline algorithm.

TABLE 7.1 – Comparison of Gaussian Sampling technique over $\mathbb{Z}$.

	Floating Point exp	Precomputation Storage	Table Look-ups	Entropy Consumption
Naïve Reject.	$.8\tau$	0	0	$.8\tau \log_2 \tau\sigma$
C.D. Table Alg.	0	$\lambda\tau\sigma$	$\log_2 \tau\sigma$	$2.1 + \log_2 \sigma$
Our Algorithm	0	$\lambda \log_2(2.4\tau\sigma^2)$	$1.5 \log_2 \sigma$	$6 + 3\log_2 \sigma$

FIGURE 7.1 – Rejection Sampling



(a) from uniform distribution (repetition rate $\approx$ 10)

(b) from our adapted distribution (repetition rate $\approx$ 1.47)

7.1 Efficient 1-dimensional Gaussian Sampling

Since its introduction, and with the noticeable exception of NTRU, lattice-based cryptosystems operating at a standard security level have remained out of reach of constrained devices by *several orders of magnitude*. A first step towards a practical lattice-based signature scheme was achieved by [GLP12] with an implementation on a low-cost FPGA, by avoiding Gaussians, at the cost of some compactness and security compared to [Lyu12].

Until recently, all known algorithms to sample according to a distribution statistically close to a discrete Gaussian distribution on a lattice [GPV08] require either long-integer arithmetic [DN12a] at some point or large memory storage [Pei10, GD].

Our approach might can be seen as a Ziggurat method adapted to Discrete Gaussian ; with the particularity that we avoid explicit probability computation (that would require floating points or large tables) by using the algebraic propeties of the exp function. Some of our technique are similar to independant work of Karney [Kar13], that focus on simple algorithm to sample exact continuous Gaussians.

Section Outline The main goal of this section is to show how to efficiently sample discrete Gaussian without resorting to large precomputed tables, nor transcendental function evaluations. The first step is being able to sample according to a Bernoulli distribution with bias of the form $\exp(-x/f)$ (and $1/\cosh(x/f)$) without actually computing transcendental functions (Section 7.1.2). The second step is to build an appropriate and efficient distribution (see Figure 7.2(b)) as input of rejection sampling to reduce its rejection rate (Section 7.1.3). Our new algorithm still requires precomputed tables, but of much smaller smaller size ; precisely of size logarithmic in σ rather than linear. Precise comparison is given in Table 7.1.

7.1.1 Discrete Gaussian Sampling : Prior Art

Laziness. Laziness is an algorithmic trick saving both computation and entropy consumption ; for our purpose, it is used in two cases of application. First, as in many compiler, when computing $a \wedge b$ and $a \vee b$, b is not always evaluated depending on the value of a. The second concerns the comparisons $r < c$: the result might be decided only knowing their first different bit ; for a uniform $r \in [0, 1)$, only 2 bits are needed on average. In practice however, one may apply this technique word by word rather than bit by bit.

Sampling with a Constant Bias. Sampling from a distribution statistically close to a Bernoulli variable $\mathcal{B}_c$ for a given bias c is easy : to get a variable $(2^{-\lambda})$-close to $\mathcal{B}_c$, take an approximation of c up

to λ correct bits, then sample a uniform real $r \in [0,1)$ up to λ bits of precision and answer 1 if and only if $r < c$.

General Algorithm. A general algorithm to sample according to a discrete Gaussian distribution $\mathcal{D}_{\sigma,c}$ centered in $c \in \mathbb{R}$ was proposed in [GPV08] and is depicted on Figure 7.2(a) for $c = 0$. It uses rejection sampling from the uniform distribution $\mathcal{U}$ over $[c-\tau\sigma, c+\tau\sigma]$ by outputting a uniform integer x with probability $p(x) = \exp(-(x-c)^2/(2\sigma^2))$. This algorithm requires about $2\tau/\sqrt{2\pi}$ trials in average, and thus $\mathcal{O}(\tau \log_2(\sigma))$ bits of entropy using laziness. The main drawback is the need to compute the exp function at very high-precision. Additionally, an average of $2\tau/\sqrt{2\pi} \approx 10$ trials until acceptance is rather expensive. We address those issues in Sections 7.1.2 and 7.1.3, where we show how to avoid explicit computations of exp and decrease repetition rate to 1.47.

Cumulative Distribution Table (CDT). In [Pei10], Peikert suggested to use a cumulative distribution table to sample more efficiently (with complexity $O(\log_2 \sigma)$) when c is known in advance. One tabulates the approximate cumulative distribution of the desired distribution, *i.e.* the probabilities $p_z = \Pr[x \leq z : x \leftarrow \mathcal{D}_{\sigma,c}]$ for $z \in [c-\tau\sigma, c+\tau\sigma]$, precomputed with λ bits of precision. At sampling time, one generates $y \in [0,1)$ uniformly at random, performs a binary search through the table to locate some $z \in \mathbb{Z}$ such that $y \in [p_{z-1}, p_z)$ and outputs z. This approach consumes $\mathcal{O}(\log_2(\sigma))$ bits of entropy, which is optimal up to a constant factor. The main drawback of this approach resides in the size of the table. Taking our set of parameters (see Section 8.5), the storage requirement is $(\lambda\tau\sigma)$ bits, *i.e.* up to 560kb for parameters of the scheme in the next chapter 8, which is unsuitable for many embedded devices.

Combination with the Knuth-Yao Algorithm. In an extensive study on discrete Gaussian distributions [GD], Galbraith and Dwarakanath suggest to combine the previous method with the Knuth-Yao algorithm. This leads to a significant decrease of the table size by a factor slightly less than 2. Unfortunately, the obtained tables remain prohibitively large.

In the following, we show how to achieve a much smaller precomputation storage (up to 4kb for parameters of chapter 8) at the expense of more input entropy (see Table 7.1).

7.1.2 Efficient Sampling of $\mathcal{B}_{\exp(-x/f)}$ and $B_{1/\cosh(x/f)}$

Requirements The scheme of [Lyu12] requires, to implement the rejection step, samples according $\mathcal{B}_{\exp(-x/f)}$ where x is a bounded integer and f a fixed real. The scheme presented in the next chapter 8 will additionnally require Bernouilli distribution of the form $B_{1/\cosh(x/f)}$. Our sampler for $\mathcal{B}_{\exp(-x/f)}$ will also be useful later to build our efficient Gaussian Sampler.

Main Idea Our solution uses the fact that appropriate combinations of Bernoulli variables can easily produce new Bernoulli variables with combined biases. We make use of that observation to avoid an explicit computation of c and require much less precomputed values. Typically, if one has access to Bernoulli variables $\mathcal{B}_a, \mathcal{B}_b$ three new Bernoulli variables are easily derived from them : $\mathcal{B}_{1-a} = \neg\mathcal{B}_a$, $\mathcal{B}_{ab} = \mathcal{B}_a \wedge \mathcal{B}_b$ and $\mathcal{B}_{a+b-ab} = \mathcal{B}_a \vee \mathcal{B}_b$. We will build a new operator $\oslash$ such that $\mathcal{B}_a \oslash \mathcal{B}_b = \mathcal{B}_{a/(1-(1-a)b)} = \mathcal{B}_a \oslash \mathcal{B}_b$, allowing one to homomorphically introduce fractions in to the Bernoulli algebra.

Efficient Bernoulli Sampling with Exponential Biases. The problem is as follows : for a fixed real f, a positive integer $x \leq 2^\ell$ given as input, sample a random Boolean according to $\mathcal{B}_{\exp(-x/f)}$. Using the simple homomorphic property of the exponential function, our approach, implemented by Algorithm 21, requires only ℓ precomputed entries, and *no* evaluation of transcendental functions.

Lemma 7.1 *For any integer $x > 0$, Algorithm 21 outputs a bit according to $\mathcal{B}_{\exp(-x/f)}$.*

Proof: Denoting the binary decomposition of x by $x = \sum_{i=0}^{\ell-1} x_i 2^i$ with $x_i \in \{0,1\}$, we have

$$\mathcal{B}_{\exp(-x/f)} = \mathcal{B}_{\exp\left(-\sum_i x_i 2^i/f\right)} = \mathcal{B}_{\prod_i \exp(-x_i 2^i/f)} = \bigwedge_{i \text{ s.t. } x_i=1} \mathcal{B}_{\exp(-2^i/f)} \ .$$

□

Algorithm 21 Sampling $\mathcal{B}_{\exp(-x/f)}$ for $x \in [0, 2^\ell)$

Input: $x \in [0, 2^\ell)$ an integer in binary form $x = x_{\ell-1} \cdots x_0$
Precomputation: $c_i = \exp(-2^i/f)$ for $0 \le i \le \ell - 1$
  for $i = \ell - 1$ **to** 0
    if $x_i = 1$ **then**
      sample $A_i \leftarrow \mathcal{B}_{c_i}$
      if $A_i = 0$ **then** return 0
  return 1

Algorithm 22 Sampling $\mathcal{B}_a \oslash \mathcal{B}_b$

  sample $A \leftarrow \mathcal{B}_a$
  if A **then** return 1
  sample $B \leftarrow \mathcal{B}_b$
  if $\neg B$ **then** return 0
  restart

Remark Notice that Algorithm 21 is defined so that the smallest probabilities are checked first, so that the algorithm can terminate faster. Notice that this algorithm is very fast, at worst $2\lceil \log_2(x) \rceil$ bits of entropy, and much less on average for random x.

Efficient Bernoulli Sampling with Inverse Hyperbolic Cosine Biases. During the final rejection step of our signing procedure, one needs to reject with probability $1/\cosh(x/f)$ for a given f. Recall that

$$\frac{1}{\cosh(x/f)} = \frac{2}{\exp(|x|/f) + \exp(-|x|/f)} = \frac{\exp(-|x|/f)}{1/2 + 1/2 \cdot \exp(-2|x|/f)} \,. \tag{7.1}$$

To sample efficiently according to the Bernoulli distribution $\mathcal{B}_{1/\cosh(x/f)}$, we reuse the previous generator for $\mathcal{B}_{\exp(-x/f)}$ with no explicit evaluation of exp or cosh. In order to deal with the fraction in Equation (7.1), we introduce a new operation denoted $\oslash$ and computed according to Algorithm 22.

Lemma 7.2 (Correctness and Efficiency of Algorithm 22) *For any $a, b \in (0, 1)$ we have, $\mathcal{B}_a \oslash \mathcal{B}_b = \mathcal{B}_{a/(1-(1-a)b)}$ and Algorithm 22 terminates after an average of $1/(1-(1-a)b)$ trials.*

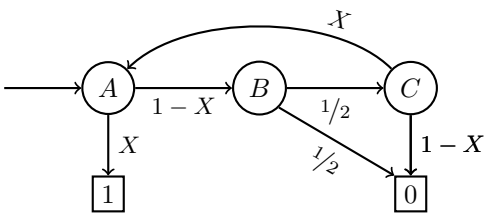
Proof: At each trial, the probability of restarting is $(1-a)b$. Now, the probability that it outputs 1 is easily computed as the sum over each trial :

$$\Pr[\mathcal{B}_a \oslash \mathcal{B}_b = 1] = a \sum_{k=0}^{\infty} (1-a)^k b^k = \frac{a}{1-(1-a)b}$$ □

□

Corollary 7.3 *For any $X \in \mathbb{R}$ we have : $\mathcal{B}_{1/\cosh(X)} = \mathcal{B}_{\exp(-|X|)} \oslash \left(\mathcal{B}_{1/2} \vee \mathcal{B}_{\exp(-|X|)}\right)$ and Algorithm 22 requires less than 1.53 calls to $\mathcal{B}_{\exp(-|X|)}$ on the average.*

Proof: Correctness is a direct application of previous lemma. Set $X = \exp(-|x|/f)$. Algorithm 22 for the computation of the Bernoulli variable $\mathcal{B}_X \oslash \left(\mathcal{B}_{1/2} \vee \mathcal{B}_X\right)$ can be seen as the following Markov Chain :



Let $\mathbf{M}$ denote the restriction of the transition matrix to the states A, B and C (indexed in that order), and let $\mathbf{v} = (1,0,0)^t$ be the initial density vector. The density vector after k steps is $\mathbf{M}^k \cdot \mathbf{v}$, so the average number of steps through each state A, B and C is given by the vector

$$\mathbf{w} = \sum_{k=0}^{\infty} \mathbf{M}^k \cdot \mathbf{v} = (\mathbf{Id}_3 - \mathbf{M})^{-1} \cdot \mathbf{v}$$

where

$$\mathbf{M} = \begin{pmatrix} 0 & 0 & X \\ 1-X & 0 & 0 \\ 0 & \frac{1}{2} & 0 \end{pmatrix} \quad \text{and} \quad (\mathbf{Id}_3 - \mathbf{M})^{-1} \cdot \mathbf{v} = \frac{1}{2 + X(X-1)} \begin{pmatrix} 2 \\ -2X+2 \\ 1-X \end{pmatrix}.$$

Since the calls to $\mathcal{B}_{\exp(-|x|/f)}$ are performed during the states A and C, the average number of calls to this Bernoulli sampling is $C(X) := w_A + w_C = \frac{3-X}{2+X(X-1)}$. One easily derives an upper bound for $C(X)$:

$$C(X) \leq \frac{5+4\sqrt{2}}{7} \approx 1.523,$$

reached when $X = 3 - 2\sqrt{2}$. □

7.1.3 Sampling Centered Discrete Gaussian Variables over $\mathbb{Z}$

Based on Algorithm 21 to sample efficiently from $\mathcal{B}_{\exp(-x/f)}$, it is now possible to obtain a Gaussian distribution via generic rejection sampling algorithm as in [GPV08], trading high-precision evaluation of transcendental functions for a table of $\log_2(\tau^2\sigma^2)$ precomputed values (see Figure 7.2(a)). However, the algorithm still requires $(2\tau/\sqrt{2\pi}) \approx 10$ trials on average to output an x statistically close to the correct distribution. This is due to the significant distance between the uniform distribution and the target distribution.

In what follows, we introduce a new sampling algorithm with an average number of rejections smaller than 1.47. We achieve that result by sampling from a specific distribution denoted D_{k,σ_2}, for which sampling is easy. The distribution D_{k,σ_2} is much closer to the target distribution $D_{k\sigma_2}$ than the uniform distribution (see Figure 7.2(b)), leading to a huge acceleration of rejection sampling.

The Binary Discrete Gaussian Distribution. Let us introduce the binary discrete Gaussian distribution $\mathcal{D}_{\sigma_2}$, which is a discrete Gaussian with specific variance $\sigma_2 = \sqrt{1/(2\ln 2)} \approx 0.849$ and probability density proportional to

$$\rho_{\sigma_2}(x) = e^{-x^2/(2\sigma_2^2)} = 2^{-x^2} \quad \text{for } x \in \mathbb{Z}\,.$$

We will combine $\mathcal{D}_{\sigma_2}$ with the uniform distribution to produce the distribution D_{k,σ_2} (see Figure 7.2(b)). We will only focus on the positive half of $\mathcal{D}_{\sigma_2}$ denoted $\mathcal{D}^+_{\sigma_2} = \{x \leftarrow \mathcal{D}_{\sigma_2} : x \geq 0\}$. Algorithm 23 is designed to sample according to $D^+_{\sigma_2}$ very efficiently using only unbiased random bits.

Lemma 7.4 *Algorithm 23 outputs positive integers according to $D^+_{\sigma_2}$. On average, the algorithm terminates after $2/\rho_{\sigma_2}(\mathbb{Z}^+) < 1.3$ trials, consuming 2.6 bits of entropy overall.*

Proof: We denote $\rho_{\sigma_2}(I) = \sum_{i\in I} 2^{-i^2}$ for $I \subseteq \mathbb{Z}^+$, the probability that the algorithm returns $x \in \mathbb{Z}^+$ is $\rho_{\sigma_2}(x)/\rho_{\sigma_2}(\mathbb{Z}^+)$ where $\rho_{\sigma_2}(\mathbb{Z}^+) = \sum_{i=0}^{\infty} 2^{-i^2} \approx 1.564$. We now observe that the binary expansion of $\rho_{\sigma_2}(\{0,\ldots,j\})$ is of the form

$$\rho_{\sigma_2}(\{0,\ldots,j\}) = \sum_{i=0}^{j} 2^{-i^2} = 1.1001\underbrace{0\ldots0}_{4}1\underbrace{0\ldots0}_{6}1 \quad \ldots \quad \underbrace{0\ldots0}_{2(j-2)}1\underbrace{0\ldots0}_{2(j-1)}1\,.$$

Thus, each trial of Algorithm 23 implicitly chooses a random real $r \in [0,2)$ that will be rejected if $r > \rho_{\sigma_2}(\mathbb{Z}^+)$. It then computes the cumulative table (scaled by $\rho_{\sigma_2}(\mathbb{Z}^+)$) on the fly and reject if necessary. On average, the algorithm completes after $2/\rho_{\sigma_2}(\mathbb{Z}^+) < 1.3$ trials, consuming 2.6 bits of entropy. □

Building the Centered Discrete Gaussian Distribution. Based on our efficient sampling for the distribution $\mathcal{D}^+_{\sigma_2}$, we can now easily build the positive discrete Gaussian distribution with standard deviation $\sigma = k\sigma_2$ for $k \in \mathbb{Z}^+$. Our Algorithm 24 based on the distribution

$$k \cdot \mathcal{D}^+_{\sigma_2} + \mathcal{U}(0, \ldots, k-1),$$

before rejection, and where we reject the result with probability $\exp(-y(y+2kx)/(2\sigma^2))$ where x and y respectively follow the distributions $D^+_{\sigma_2}$ and $\mathcal{U}(0, \ldots, k-1)$.

Theorem 7.5 *For any integer input k, Algorithm 24 outputs positive integers according to D^+_σ for $\sigma = k\sigma_2$. On average, it requires less than 1.47 trials. Consequently, Algorithm 25 output integers according to D_σ, and requires about $1 + \frac{1}{5\sigma}$ trials.*

Remark Entropy consumption for each trials is : 2.6 bits for $x \leftarrow D^+_{\sigma_2}$, $\log_2 k$ bits for $y \leftarrow \mathcal{U}(\{0, \ldots, k-1\})$, and $\approx 1 + \log_2 \sigma$ for rejection bit $b \leftarrow \mathcal{B}_{\exp(-y(y+2kx)/(2\sigma^2))}$ (measured in practice for this particular distribution), for a total of $\approx 4 + 2\log_2 \sigma$.

Proof: Let us start with the fact that any output z is uniquely written as $kx+y$ for $y \in \{0, \ldots, k-1\}$. The input (resp. desired output) distribution weight function g (resp f) is

$$g(z) = g(kx+y) = \frac{\rho_{\sigma_2}(x)}{k\rho_{\sigma_2}(\mathbb{Z}^+)} \quad \text{and} \quad f(z) = f(kx+y) = \frac{\rho_{k\sigma_2}(kx+y)}{\rho_{k\sigma_2}(\mathbb{Z}^+)}.$$

Since we restrict the distribution to non-negative integers, we have $\exp\left(-\frac{y(y+2kx)}{2\sigma^2}\right) \leq 1$ since x and y are both positive. Therefore, the probability to output some integer z is proportional to

$$\rho_{\sigma_2}(x) \exp\left(-\frac{y(y+2kx)}{2\sigma^2}\right) = \exp\left(-\frac{x^2}{2\sigma_2^2} - \frac{2kxy+y^2}{2\sigma^2}\right) = \exp\left(-\frac{(kx+y)^2}{2\sigma^2}\right) = \rho_{k\sigma_2}(z)\ .$$

The repetition rate M is upper-bounded by

$$M = \max \frac{f}{g} \leq \frac{k\rho_{\sigma_2}(\mathbb{Z}^+)}{\rho_{k\sigma_2}(\mathbb{Z}^+)} \leq \frac{k\rho_{\sigma_2}(\mathbb{Z}^+)}{k\sigma_2\sqrt{\pi/2}} \leq 1.47\ .$$

where the first inequality follows from the sum-integral comparison ($\rho_{k\sigma_2}$ is decreasing over $[0, \infty)$)

$$\rho_{k\sigma_2}(\mathbb{Z}^+) \geq \int_{x=0}^{\infty} \rho_{k\sigma_2}(x)dx = k\sigma_2\sqrt{\pi/2}.$$

□

Finally, we apply Algorithm 25 to build the (full) discrete Gaussian distribution $\mathcal{D}_\sigma$ over $\mathbb{Z}$.

7.1.4 Sampling Non-Centered Discrete Gaussian Variables over $\mathbb{Z}$

Quite obviously, the algorithm presented in the previous section to sample from $D_{\mathbb{Z},\sigma}$ can easily be adapted to sample from $D_{\mathbb{Z},\sigma,c}$ for any integer value $c \in \mathbb{Z}$, by simply outputting $c + D_{\mathbb{Z},\sigma}$. In the same manner, the problem of sampling $D_{\mathbb{Z},\sigma,c}$ for a real value $c \in \mathbb{R}$ can be reduced to the case where $c \in [-1/2, 1/2)$. One may think to use a 1-dimensional version of the rejection technique used in [Lyu12] to transform a sampler $D_{\mathbb{Z},\sigma,0}$ to a sample $D_{\mathbb{Z},\sigma,c}$; yet this requires a repetition rate of about $\exp(\tau\,|c|\,/\sigma)$ for a reasonable tailcut parameter (say $\tau \approx 12$). On the other hand, the smoothing condition allows σ as small as 2 in practice (in theory both τ and σ must grow as the square root of the security parameter, and the ratio is the constant $\tau/\sigma = 2\pi$) ; for a shift c as big as $1/2$, this is a repetition rate of $\exp(\pi) \approx 23$.

Interestingly, in dimension 1, it is possible to improve this algorithm by starting from a slightly wider distribution $D_{\mathbb{Z},\sigma'}$ for $\sigma' > \sigma$. Indeed, let $f(x) = \frac{1}{\sigma'\sqrt{2\pi}} \exp\left(-x^2/(2\sigma'^2)\right)$ be $D_{\mathbb{Z},\sigma}$ the weight of $D_{\mathbb{Z},\sigma'}$ at $x \in \mathbb{Z}$ (ignoring the smoothing renormalization factor in $[1-2\iota, 1+2\iota]$ when $\sigma \geq \eta_\iota(\mathbb{Z})$), and similarly let $g_c(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-(x-c)^2/(2\sigma^2)$ be the weight of the target distribution $D_{\mathbb{Z},\sigma,c}$. To apply rejection sampling properly, on must choose the repetion rate M so that, for any $x \in \mathbb{Z}$ and $c \in [-1/2, 1/2)$, we have $(g_c(x)/f(x)) \leq M$. Unrolling the definitions, this gives

Algorithm 23 Sampling $D^+_{\sigma_2}$

Output: An integer $x \in \mathbb{Z}^+$ according to $D^+_{\sigma_2}$
  Generate a bit $b \leftarrow \mathcal{B}_{1/2}$
  if $b = 0$ **then** return 0
  for $i = 1$ **to** ∞ **do**
    draw random bits $b_1 \dots b_k$ for $k = 2i - 1$
    if $b_1 \dots b_{k-1} \neq 0 \dots 0$ **then** restart
    if $b_k = 0$ **then** return i
  end for

Algorithm 24 Sampling $D^+_{k\sigma_2}$ for $k \in \mathbb{Z}$

Input: An integer $k \in \mathbb{Z}$ ($\sigma = k\sigma_2$)
Output: An integer $z \in \mathbb{Z}^+$ according to D^+_{σ}
  sample $x \in \mathbb{Z}$ according to $D^+_{\sigma_2}$
  sample $y \in \mathbb{Z}$ uniformly in $\{0, \dots, k-1\}$
  $z \leftarrow kx + y$
  sample $b \leftarrow \mathcal{B}_{\exp(-y(y+2kx)/(2\sigma^2))}$
  if $\neg b$ **then** restart
  return z

Algorithm 25 Sampling $D_{\mathbb{Z},k\sigma_2}$ for $k \in \mathbb{Z}$

  Generate an integer $z \leftarrow D^+_{k\sigma_2}$
  if $z = 0$ restart with probability $1/2$
  Generate a bit $b \leftarrow \mathcal{B}_{1/2}$ and return $(-1)^b z$

$$M \geq \frac{\sigma'}{\sigma} \exp\left(\frac{x^2}{2\sigma'^2} + \frac{-x^2 + 2cx - c^2}{2\sigma^2}\right).$$

When $\sigma' = \sigma$, the x^2 terms in the exponential cancels out, and if $c \neq 0$, this expression is not bounded for $x \in \mathbb{Z}$, forcing one to resort to a (costly) tailcut argument to bound x. However, it is not possible in [Lyu12] to choose a σ' much larger than σ since the factor $\frac{\sigma'}{\sigma}$ becomes $(\frac{\sigma'}{\sigma})^n$ in dimension n. Yet, for our current purpose, one can allow a larger σ'; the optimal choice seems to be around $\sigma'/\sigma \approx 5/4$, and we will choose this integer ratio to avoid increasing the size of the integer manipulated by the final algorithm.

Algorithm 26 Sampling $D_{\mathbb{Z},\sigma,c}$ for $c \in \frac{1}{d}\mathbb{Z} \cap [-1/2, 1/2)$

Input: $\sigma \geq \eta_\iota(\mathbb{Z})/\sqrt{2\pi}$, a constant $m \in \frac{1}{25d^2}\mathbb{Z}$ such that $e^m \geq (1 + 2\iota)\frac{5}{4}e^{2/(9\sigma^2)}$
Output: An integer x according to the distribution $x \leftarrow D_{\mathbb{Z},\sigma,c}$
  Generate an integer $x \leftarrow D_{\mathbb{Z},\frac{5}{4}\sigma}$
  With probability $\exp\left(\frac{m - \frac{9}{25}x^2 + 2cx - c^2}{2\sigma^2}\right)$ output x
  Otherwise, **Restart**

Note that the algorithm does not need to manipulate reals. The value in the exp function has the form $n/(50d^2\sigma^2)$ for some efficently computable integer n, therefore on can resort to the $\mathcal{B}_{\exp}$ algorithm (Alg. 21).

Lemma 7.6 (Correctness and Efficiency of Algorithm 26) *Algorithm 26 is correct, that is, its output follows the distribution $D_{\mathbb{Z},\sigma,c}$. If $\iota \in (0, 2^{-5})$ and $\sigma \geq 1$, then there is a valid input $m \in \frac{1}{25d^2}\mathbb{Z}$ such that the algorithm terminates after less than 1.6 trials on the average.*

Proof: According to the previous computation, to apply rejection sampling (lemma 2.7) one must accept a sample with probability

$$\frac{(1 + 2\iota)\sigma'}{M\sigma} \exp\left(\frac{x^2}{2\sigma'^2} + \frac{-x^2 + 2cx - c^2}{2\sigma^2}\right).$$

ith $\sigma' = \frac{5}{4}\sigma$, and where M is chosen such that this expression is always ≤ 1. That is, $e^m = M$ must verify

$$M \geq (1 + 2\iota)\frac{5}{4} \exp\left(\frac{-\frac{9}{25}x^2 + 2cx - c^2}{2\sigma^2}\right)$$

for all x. Note that the polynomial $-x^2 + 2cx - c^2$ reaches its maximum at $x = \frac{25}{9}c$, for a value of $\frac{16}{9}c^2 \leq \frac{4}{9}$. Therefore, it is enough that $e^m = M \geq (1 + 2\iota)\frac{5}{4}e^{2/(9\sigma^2)}$. □

7.2 Klein's Algorithm without Floating-Points Arithmetic

7.2.1 Generalized Klein's Algorithm

Our first point is an analysis of generalized Klein's Algorithm, that is essentially the original algorithm [Kle00] but given auxiliary inputs that may not be the GSO decomposition of the basis $\mathbf{B}$. In other words, this is the same algorithm as before, except that we don,t require that the inputs verify $\mathbf{B} = \mu\mathbf{DQ}$ for some orthogonal matrix $\mathbf{Q}$. We show that this algorithm still provides a discrete Gaussian distribution, but it might not be spherical. Yet if we have $\mathbf{B} = \mu\mathbf{DQ}$ for some approximatly orthogonal matrix $\mathbf{Q}$, the distribution will be perfectly Gaussian, and close to spherical. Our goal will be to use this algorithm on an approximation of the GSO, and rectify the sphericity defect later on. In short, this new algorithm **GenSampleLattice** is the same as the original **SampleLattice** except that we remove the requirement that the input should be the GSO of the basis, and add new variables d_i to replace $\|\mathbf{b}_i^\star\|$.

Algorithm 27 GenSampleLattice : Elliptic Klein's Algorithm

Input: a (short) lattice basis $B = (\mathbf{b}_1, \ldots, \mathbf{b}_n) \in \mathbb{R}^{n\times n}$, a lower triangular matrix $\mu \in \mathbb{R}^{n\times n}$ with unit diagonal, a diagonal matrix $\mathbf{D} = \mathrm{diag}(d_1 \ldots d_n)$ parameter $\sigma \in \mathbb{R}$, A target vector $\mathbf{c} \in \mathbb{Z}^{1\times n}$ and $\sigma_i = \sigma/r_i \in \mathbb{R}$

Output: a vector $\mathbf{v}$ drawn approximately from $D_{L,\sqrt{\mathbf{\Sigma}},\mathbf{c}}$ where $L = \mathcal{L}(\mathbf{B})$ and $\mathbf{\Sigma} = \sigma^2 \cdot (\mathbf{D}^{-1}\mu^{-1}\mathbf{B})^t(\mathbf{D}^{-1}\mu^{-1}\mathbf{B})$

1: $\mathbf{v}, \mathbf{z} \leftarrow \mathbf{0} : \mathbb{Z}^n$
2: $\mathbf{w} = \mathbf{c}\mathbf{B}^{-1}\mu$
3: **for** $i = n$ downto 1 **do**
4: $\quad z_i \leftarrow D_{\mathbb{Z},\sigma/d_i,w_i}$
5: $\quad \mathbf{v} \leftarrow \mathbf{v} + z_i \cdot \mathbf{b}_i$
6: $\quad \mathbf{w} \leftarrow \mathbf{w} - z_i \cdot \mu_i$
7: **end for**
8: return $\mathbf{v}$

Remark For certain type of inputs, this algorithm can be run using integers only. In particular, if $\mathbf{B} \in \mathbb{Z}^{n\times n}$ is the basis of a q-ary lattice, then $\mathbf{B}^{-1} \in \frac{1}{q}\mathbb{Z}^{n\times n}$. If additionally, the target is an integer vector ($\mathbf{c} \in \mathbb{Z}^n$), and if the auxiliary input μ have rational entries with a common denominator p, that is $\mu \in \frac{1}{p}\mathbb{Z}^{n\times n}$, then, all along the algorithm, we have

$$\mathbf{z} \in \mathbb{Z}^n, \mathbf{v} \in \mathbb{Z}^n, \mathbf{w} \in \frac{1}{pq}\mathbb{Z}.$$

In particular, all the internal operations of **GenSampleLattice** can be done with integers (simply scaling the representation by pq) ; and additionally the distribution $D_{\mathbb{Z},\sigma/d_i,w_i}$ can be sampled using algorithm 26 using only integers.

Theorem 7.7 (Output distribution of GenSampleLattice) *Let $\mathbf{B} \in \mathbb{R}^{n\times n}$ be a basis of L, $\mu \in \mathbb{R}^{n\times n}$ be a lower triangular matrix μ with zero diagonal, $\mathbf{D} = \mathrm{diag}(d_1 \ldots d_n)$ be a diagonal matrix, and $\mathbf{c} \in \mathbb{R}^n$ be a target vector. If $\sigma > 0$ is such that $\sigma/d_i > \eta_\iota(\mathbb{Z})$ for some $\iota \in (0, 1/2)$, then the output distribution of* **GenSampleLattice**$(\mathbf{B}, \mu, \mathbf{D}, \sigma, \mathbf{c})$ *is within statistical distance at most $2n\iota$ from $D_{L,\sqrt{\mathbf{\Sigma}},\mathbf{c}}$ for $\mathbf{\Sigma} = \sigma^2 \cdot (\mathbf{D}^{-1}\mu^{-1}\mathbf{B})^t(\mathbf{D}^{-1}\mu^{-1}\mathbf{B})$.*

Proof: For any $\tilde{\mathbf{z}} \in \mathbb{Z}^n$, the probability that $\mathbf{z} = \tilde{\mathbf{z}}$ at the end of **GenSampleLattice**$(\mathbf{B}, \mathbf{D}, \mu, \sigma, \mathbf{c})$ is, where $\mathbf{w} = \mathbf{c}\mathbf{B}^{-1}(\mu + \mathbf{Id}_n)$ denotes the initial value of $\mathbf{w}$

$$\begin{aligned}
\Pr(\mathbf{z} = \tilde{\mathbf{z}}) &= \prod_{i=1}^{n} \Pr(z_i = \tilde{z}_i | \forall j > i, z_j = \tilde{z}_j) \\
&= \prod_{i=1}^{n} \frac{\rho_{\sigma/d_i}(\tilde{z}_i - (\mathbf{w} - \sum_{j>i} T_{j,i}\tilde{z}_i))}{\rho_{\sigma/d_i}(\mathbb{Z} - (\mathbf{w} - \sum_{j>i} T_{j,i}\tilde{z}_i))} \\
&= \prod_{i=1}^{n} \frac{\rho_{\sigma/d_i}(\tilde{z}_i - (\mathbf{w} - [\tilde{\mathbf{z}}\mu - \mathbf{Id}_n]_i))}{\rho_{\sigma/d_i}(\mathbb{Z} - (\mathbf{w} - [\tilde{\mathbf{z}}\mu - \mathbf{Id}_n]_i))} \\
&= \prod_{i=1}^{n} \frac{1}{\rho_{\sigma/d_i}(\mathbb{Z} - (\mathbf{w} - [\tilde{\mathbf{z}}\mu - \mathbf{Id}_n]_i))} \cdot \exp\left(-\sum_{i=1}^{n} \frac{d_i^2}{2\sigma^2}(\tilde{z}_i - (\mathbf{w} - [\tilde{\mathbf{z}}\mu - \mathbf{Id}_n]_i)^2\right) \\
&= \prod_{i=1}^{n} -\frac{1}{\rho_{\sigma/d_i}(\mathbb{Z} - (\mathbf{w} - [\tilde{\mathbf{z}}\mu - \mathbf{Id}_n]_i))} \cdot \exp\left(-\frac{1}{2\sigma^2}\left\|\tilde{\mathbf{z}}\mu - \mathbf{w}\right\|^2\right)
\end{aligned}$$

Since there is a bijective correspondence between $\mathbf{z} \in \mathbb{Z}^n$ and $\mathbf{v} = \mathbf{z} \cdot \mathbf{B} \in L$, for any $\tilde{\mathbf{v}} \in L$ we have $\Pr(\mathbf{v} = \tilde{\mathbf{v}}) = \Pr\left(\mathbf{z} = \tilde{\mathbf{v}} \cdot \mathbf{B}^{-1}\right)$, that is

$$\begin{aligned}
\Pr(\mathbf{v} = \tilde{\mathbf{v}}) &= \prod_{i=1}^{n} \frac{1}{\rho_{\sigma/d_i}(\mathbb{Z} - (\mathbf{w} - [\tilde{\mathbf{z}}\mu - \mathbf{Id}_n]_i))} \cdot \exp\left(-\frac{1}{2\sigma^2}\left\|(\tilde{\mathbf{v}} - \mathbf{c})\mathbf{B}^{-1}(\mathbf{Id}_n + \mu)\right\|^2\right) \\
&= \prod_{i=1}^{n} \frac{1}{\rho_{\sigma/d_i}(\mathbb{Z} - (\mathbf{w} - [\tilde{\mathbf{z}}\mu - \mathbf{Id}_n]_i))} \cdot \exp\left(-\frac{1}{2}(\tilde{\mathbf{v}} - \mathbf{c})\mathbf{\Sigma}^{-1}(\tilde{\mathbf{v}} - \mathbf{c})^t\right) \\
&\in \left[1, \left(\frac{1+\iota}{1-\iota}\right)^n\right] \cdot \frac{1}{\prod_{i=1}^{n} \rho_{\sigma/d_i}(\mathbb{Z})} \cdot \rho_{\mathbf{\Sigma}}(\tilde{\mathbf{v}} - \mathbf{c})
\end{aligned}$$

In particular, the output distribution of $\mathbf{v} \in L$ is within statistical distance at most $2n\iota$ of $D_{L,\mathbf{\Sigma},\mathbf{c}}$. □

In the original version of Klein Algorithm, the inputs were the GSO decomposition of $\mathbf{B}$, precisely we had $\mathbf{B} = \mu\mathbf{D}\mathbf{Q}$ for some orthogonal matrix $\mathbf{Q}$. This implies that

$$\mathbf{\Sigma} = \sigma \cdot (\mathbf{D}^{-1}\mu^{-1}\mathbf{B})^t(\mathbf{D}^{-1}\mu^{-1}\mathbf{B}) = \sigma\mathbf{Q}^t\mathbf{Q} = \sigma\mathbf{Id}_n;$$

that is we obtained a spherical Gaussian, independent of the shape of $\mathbf{B}$.

Here, our strategy is to use an approximation of the GSO of $\mathbf{B}$, which would result in an Gaussian not exactly spherical; we intend to correct its sphericality using rejection sampling. Let us first quantify the lack of sphericality in terms of the quality of the approximation.

Lemma 7.8 (Sphericity Default Bound) *Let $\mathbf{B} \in \mathbb{R}^{n\times n}$ be an invertible matrix, and let μ, $\mathbf{D} = \mathrm{diag}(d_1 \dots d_n)$ and $\mathbf{Q}$ orthogonal form its GSO decomposition $\mathbf{B} = \mu\mathbf{D}\mathbf{Q}$. Let $\epsilon > 0$, $\bar{\mu}$ and $\bar{\mathbf{D}}$, with $\bar{\mathbf{D}}$ a diagonal matrix, be $n \times n$ real matrices such that $\|\bar{\mu} - \mu\|_\infty \leq \epsilon$ and $\left\|\bar{\mathbf{D}} - \mathbf{D}\right\|_\infty \leq \epsilon$. Set $\bar{\mathbf{Q}} = \bar{\mathbf{D}}^{-1}\bar{\mu}^{-1}\mathbf{B}$ and $\mathbf{\Sigma}_{\bar{\mathbf{Q}}} = \bar{\mathbf{Q}}^t\bar{\mathbf{Q}}$, then, for any vector $\mathbf{v} \in \mathbb{R}^n$ we have*

$$\left\|\mathbf{v}\bar{\mathbf{Q}}^{-1}\right\|^2 = \mathbf{v}\mathbf{\Sigma}^{-1}\mathbf{v}^t \in \left[(1-\delta)^2, (1+\delta)^2\right] \cdot \|\mathbf{v}\|^2$$

for $\delta = \left(\left\|\mathbf{B}^{-1}\right\|_s \cdot (\max d_i + \epsilon) + \max \frac{1}{d_i}\right) n\epsilon$. In other terms,

$$\frac{1}{1-\delta} \geq s_1(\sqrt{\mathbf{\Sigma}}) \geq s_n(\sqrt{\mathbf{\Sigma}}) \geq \frac{1}{1+\delta}$$

Proof: Set $\mathbf{E}_1 = \mu - \bar{\mu}$ that is a lower triangular matrix such that $\|\mathbf{E}_1\| \leq \epsilon$, and $\mathbf{E}_2 = \mathbf{D} - \bar{\mathbf{D}}$, that is a diagonal matrix such that $\|\mathbf{E}_1\| \leq \epsilon$. Rewrite $\mathbf{v}\mathbf{\Sigma}^{-1}\mathbf{v}$ as $\left\|\mathbf{v} \cdot \mathbf{B}^{-1}\bar{\mu}\bar{\mathbf{D}}\right\|^2$, and notice that

$$\begin{aligned}
\mathbf{v} \cdot \mathbf{B}^{-1}\bar{\mu}\bar{\mathbf{D}} &= \mathbf{v} \cdot \mathbf{B}^{-1}(\mu + \mathbf{E}_1)(\mathbf{D} + \mathbf{E}_2) \\
&= \mathbf{v} \cdot \mathbf{Q}^{-1} + \mathbf{v} \cdot (\mathbf{B}^{-1}\mathbf{E}_1\mathbf{D} + \mathbf{B}^{-1}\mu\mathbf{E}_2 + \mathbf{B}^{-1}\mathbf{E}_1\mathbf{E}_2)
\end{aligned}$$

Note that $\left\|\mathbf{v}\cdot\mathbf{Q}^{-1}\right\| = \|\mathbf{v}\|$ because $\mathbf{Q}$ is orthogonal. Additionally, we have $\|\mathbf{E}_1\|_s \leq n\epsilon$, therefore

$$\left\|\mathbf{v}\cdot\mathbf{B}^{-1}\mathbf{E}_1\mathbf{D}\right\| \leq \left\|\mathbf{B}^{-1}\right\|_s \|\mathbf{E}_1\|_s \|\mathbf{D}\|_s \|\mathbf{v}\| \leq \left\|\mathbf{B}^{-1}\right\|_s \max d_i n\epsilon \cdot \|\mathbf{v}\|.$$

Similarly,

$$\left\|\mathbf{v}\cdot\mathbf{B}^{-1}\mu\mathbf{E}_2\right\| = \left\|\mathbf{v}\cdot\mathbf{Q}^{-1}\mathbf{D}^{-1}\mathbf{E}_2\right\| \leq \max\frac{1}{d_i}\epsilon\cdot\|\mathbf{v}\|.$$

Eventually,

$$\left\|\mathbf{v}\cdot\mathbf{B}^{-1}\mathbf{E}_1\mathbf{E}_2\right\| \leq \left\|\mathbf{B}^{-1}\right\|_s \|\mathbf{E}_1\|_s \|\mathbf{E}_2\|_s \|\mathbf{v}\| \leq \left\|\mathbf{B}^{-1}\right\|_s n\epsilon^2 \|\mathbf{v}\|$$

□

7.2.2 Sphericity Rectification via Rejection

For a matrix any matrix $\mathbf{B}\in\mathbb{R}^{n\times n}$ one easily see that if $\sigma \leq s_1(\mathbf{B}) \leq s_n(\mathbf{B}) \leq \sigma'$, then, for any vector $\mathbf{v}$, we have

$$\rho_\sigma(\mathbf{v}) \geq \rho_B(\mathbf{v}) \geq \rho_{\sigma'}(\mathbf{v})$$

Yet, to apply rejection sampling we need to take account for the renormalization factor $\frac{1}{\rho_{\sqrt{\mathbf{\Sigma}}}(\mathbf{c}+L)}$

Fact 7.9 (Weight of ρ over a Lattice) *For any lattice $L\subset\mathbb{R}^m$, if $\sigma \geq \frac{1}{\sqrt{2\pi}}\eta_\iota(L)$ for some $\iota\in(0,1/2)$, then for any vector $\mathbf{x}\in\mathbb{R}^n$ we have*

$$\rho_\sigma(L+\mathbf{x}) \in \left[\frac{1-\iota}{1+\iota}, \frac{1+\iota}{1-\iota}\right]\cdot\frac{(\sigma\sqrt{2\pi})^n}{\mathrm{Vol}(L)}$$

Proof: Recall from lemma 3.18, that if $\sigma \geq \frac{1}{\sqrt{2\pi}}\eta_\iota(L)$ and $c\in\mathrm{span}(L)$, we have

$$\rho_{\sigma,\mathbf{c}}(L) \in \left[\frac{1-\iota}{1+\iota}, 1\right]\cdot\rho_{\sigma,0}(L).$$

For a full rank lattice $L\subset\mathbb{R}^n$, and for a measurable fundamental domain $\mathcal{F}$, we have

$$\int_{x\in\mathbb{R}^m}\rho_\sigma(x) = \int_{x\in\mathcal{F}}\rho(x+L) \in \left[\frac{1-\iota}{1+\iota}, 1\right]\cdot\mathrm{Vol}(\mathcal{F})\cdot\rho_\sigma(L).$$

Therefore, from the identity $\int_{x\in\mathbb{R}^m}\rho_\sigma(x) = (\sigma\sqrt{2\pi})^n$ we deduce that $\rho_\sigma(L)\in\left[1,\frac{1+\iota}{1-\iota}\right]\frac{(\sigma\sqrt{2\pi})^n}{\mathrm{Vol}(L)}$and more generally, for any $\mathbf{x}$

$$\rho_\sigma(L+\mathbf{x}) \in \left[\frac{1-\iota}{1+\iota}, \frac{1+\iota}{1-\iota}\right]\cdot\frac{(\sigma\sqrt{2\pi})^n}{\mathrm{Vol}(L)}$$

□

Remark If the covariance matrix $\Sigma = \sigma^2\bar{\mathbf{Q}}^t\bar{\mathbf{Q}}$ where $\bar{\mathbf{Q}} = \mathbf{B}\bar{\mathbf{D}}^{-1}\bar{\mu}^{-1}$, as the output of **GenSampleLattice** on inputs $\mathbf{B},\bar{\mu},\bar{\mathbf{D}}$, then the acceptance probability $\left(\rho_{\sqrt{\mathbf{\Sigma}}}(\mathbf{x}-\mathbf{c})/M\cdot\rho_{\sqrt{\sigma}}(\mathbf{x}-\mathbf{c})\right)$ can be rewritten as

$$\exp\left(\frac{-m+\left\|(\mathbf{x}-\mathbf{c})\bar{\mathbf{Q}}^{-1}\right\|^2 - \|(\mathbf{x}-\mathbf{c})\|^2}{2\sigma^2}\right) \text{ for } M = e^m$$

In particular, if $\mathbf{x},\mathbf{c}$ are integer vectors, $\mathbf{B}^{-1}\in\frac{1}{q}\mathbb{Z}$, and $\bar{\mathbf{D}},\bar{\mu}\in\frac{1}{r}\mathbb{Z}$, then the numerator of the previous fraction is in $\frac{1}{q^2r^4}\mathbb{Z}$; in particular, the rejection step can be implemented using the $\mathcal{B}_{exp}$ algorithm (Alg. 21).

Lemma 7.10 (Correctness and Efficiency of RectifySphere) *Under proper input $\mathbf{\Sigma}$, σ, δ, ι and M as defined in* **RectifySphere**, *the output of this algorithm is at statistical distance at most 2ι from $D_{L,\sigma,\mathbf{c}}$, and the algorithm terminates after M trials in average.*

Algorithm 28 RectifySphere : Rectify non-spherical Gaussian

Input: A symmetric semi-definite positive $n \times n$ matrix $\mathbf{\Sigma}$, positive integers σ, δ such that $\sigma^2\mathbf{Id}_n \leq \mathbf{\Sigma} \leq (1+\delta)^2\sigma^2\mathbf{Id}_n$, and such that $\sigma \geq \eta_\iota(L)$; a constant $M \geq (\frac{1+\iota}{1-\iota})^2(1+\delta)^n$
A sampler for $D_{L,\mathbf{\Sigma},\mathbf{c}}$

Output: A sample $\mathbf{x} \in L$ following distribution $D_{L,\sigma,\mathbf{c}}$

1: Sample $\mathbf{x} \leftarrow D_{L,\mathbf{\Sigma},\mathbf{c}}$
2: With probability $\left(\rho_{\sqrt{\mathbf{\Sigma}}}(\mathbf{x}-\mathbf{c})/M \cdot \rho_{\sqrt{\sigma}}(\mathbf{x}-\mathbf{c})\right)$ output $\mathbf{x}$
3: Otherwise, **restart**

Proof: The proof easily follows from Lemma 3.18 (smoothing bound), Fact 7.9 (weight of ρ) and Lemma 2.7 (correctness of Rejection Sampling). □

7.2.3 Spherical Sampling without Floating-Point-Arithmetic

Approximating and rectifying Spherical Gaussian Using the results of the previous section, one can build an sampler for $D_{L,\sigma,\mathbf{c}}$ without FPA, if L is a q-ary lattice, and given a basis $\mathbf{B}$ for $\sigma \approx \|\mathbf{B}^\star\| \eta_\iota(\mathbb{Z})$. To do so, proceed as follows : consider the GSO decomposition $\mathbf{B} = \mu\mathbf{DQ}$. Then choose a precision $\epsilon = 2^{-m-1}$, such that

$$\delta = \left(\left\|\mathbf{B}^{-1}\right\|_s \cdot (\max d_i + \epsilon) + \max \frac{1}{d_i} \right) n\epsilon = \mathcal{O}(1/n).$$

Then, choose an approximation $\bar{\mu} \in \frac{1}{2^m}\mathbb{Z}^{n\times n}$ of $\bar{\mu}$, and an approximation $\bar{\mathbf{D}} \in \frac{1}{2^m}\mathbb{Z}^{n\times n}$ of $(1+\delta)\mathbf{D}$; and define $\bar{\mathbf{Q}} = \bar{\mathbf{D}}^{-1}\bar{\mu}^{-1}\mathbf{B}$. According to Lemma 7.8, this ensure the sphericity defect of $\bar{\mathbf{Q}}$ is bounded

$$\frac{1+\delta}{1-\delta} \geq s_1(\bar{\mathbf{Q}}) \geq s_n(\bar{\mathbf{Q}}) \geq 1.$$

This means that one can run Algorithm **GenSampleLattice** on the inputs $\mathbf{B}, \bar{\mu}, \bar{\mathbf{D}}, \mathbf{c}, \sigma$ using only numbers in $\frac{1}{q^2 2^{4m}}\mathbb{Z}$, and obtain a distribution $D_{L,\Sigma,\mathbf{c}}$ that is almost spherical ; precisely $\Sigma = \sigma^2\bar{\mathbf{Q}}^t\bar{\mathbf{Q}}$ verifies $\sigma^2\mathbf{Id}_n \leq \mathbf{\Sigma} \leq \left(\frac{1+\delta}{1-\delta}\right)^2 \sigma^2\mathbf{Id}_n$. Because we have chosen $\delta = O(1/n)$, one can therefore apply **RectifySphere** with a constant repetition rate $M = (1+\mathcal{O}(1/n))^n = \mathcal{O}(1)$.

This new algorithm provides the quasi-quadratic running time, that is the efficiency asymptotical efficiency as the lazy variant of Klein's algorithm (see Theorem 6.9), for the same class of basis, namely

Theorem 7.11 (Gaussian Sampling in quasi-quadratic time without FPA) *Let $(\mathcal{C}_n)$ be an Small-Inverse class of size-reduced basis of q-ary lattices, for $q = poly(n)$. For any basis $\mathbf{B} \in (\mathcal{C}_n)$,and $\sigma = poly(n)$ such that $\sigma \geq \|\mathbf{B}^\star\| \cdot \mathcal{O}(\sqrt{n})$, there an exist implicit functions m such that the above algorithm runs in quasi-quadratic time $\tilde{\mathcal{O}}(n^2)$ and outputs a distribution exponentially close to $D_{\mathcal{L}(\mathbf{B}),\sigma,\mathbf{c}}$ when the entries of $\mathbf{c}$ are reduced* mod q.

Proof: Using the notation above, one can choose $m = \mathcal{O}(\log n)$ and obtain $\delta = \mathcal{O}(1)$. Because the lattice is assumed to be q-ary, we have $\mathbf{B}^{-1} \in \frac{1}{q}\mathbb{Z}^{n\times n}$, by definition of size reduction, the coefficient of μ are bounded by $1/2$. This implies that $\bar{\mu} \in \frac{1}{2^m}\mathbb{Z}^{n\times n}$ and Σ have polynomially bounded entries as well. Since the algorithm performs $\mathcal{O}(n^2)$ operations in $\frac{1}{q^2 2^{4m}}\mathbb{Z}$, it is enough to prove that the intermediates values of **GenSampleLattice** for those inputs remain polynomial ; this is provided by the following lemma. □

Lemma 7.12 (Bounds on intermediates values of GenSampleLattice) *If the inputs of algorithm* **GenSampleLattice** *are such that the entries of $\mathbf{B} \in \mathbb{Z}^{n\times n}$, $\mathbf{B}^{-1} \in \frac{1}{q}\mathbb{Z}^{n\times n}$, $\mu \in \frac{1}{p}\mathbb{Z}^{n\times n}$, $\mathbf{\Sigma}$ and $\mathbf{c} \in \mathbb{Z}^n$ are polynomially bounded in n, then all the intermediates values are in $\frac{1}{p^2q^2}\mathbb{Z}$ and are also polynomially bounded in n except with negligible probability in n.*

Proof: We note $\mathbf{v}^{(k)}, \mathbf{w}^{(k)}$ and $\mathbf{z}^{(k)}$ the value of $\mathbf{v}, \mathbf{w}$ and $\mathbf{z}$ in **GenSampleLattice** before the for-loop is ran for $i = k$ (recall that the loops goes from $i = n$ down to 1), and it denotes their final value when $k = 0$.

Recall that the initial values are $\mathbf{w}^{(n)} = \mathbf{c} \cdot \mathbf{B}^{-1}\mu$, $\mathbf{z}^{(n)} = \mathbf{v}^{(n)} = \mathbf{0}$. The loop maintains the following equations :

$$\begin{aligned}
\mathbf{v}^{(k)} &= \mathbf{z}^{(k)} \cdot \mathbf{B} \\
\mathbf{w}^{(k)} &= w^{(n)} - \mathbf{z}^{(k)} \cdot \mathbf{B} \\
z_i^{(k)} &= z_i^{(i-1)} \text{ if } k < i \\
z_i^{(k)} &= 0 \text{ otherwize}
\end{aligned}$$

In particular, the entries of $\mathbf{z}^{(k)}$ are always smaller than the entries of $\mathbf{z}^{(0)}$. Now notice that with overwhelming probability, the entries of $\mathbf{v}^{(0)}$ (the output of the algorithm) are bounded by $\mathcal{O}(n \cdot s_1(\sqrt{\mathbf{\Sigma}}))$ by tailcut bound (lemma 3.20). Therefore the entries $\mathbf{z}^{(0)} = \mathbf{v}^{(0)} \cdot \mathbf{B}^{-1}$ are polynomially bounded, and so are those of $\mathbf{z}^{(k)}$ for any k. One concludes using the previous equations. □

7.3 Conclusion

We have seen that there exists very simple and natural algorithms to sample according to discrete Gaussian over $\mathbb{Z}$, and those algorithm should be able to run on small architectures. Those algorithms should prove quite useful to implement the scheme described in the next chapter on embedded devices.

Concerning sampling over arbitrary lattices, the efficiency of an algorithm such requires a more carefull study, to evaluate precisely its performance, that will be the object of future research, openning the way to trapdoor primitives on embedded devices. Additionnally, it is quite belivable that such a strategy of rectifying a sphere, could also adapt to the Peikert's offline phase, once again to avoid the use of FPA, which allow different trafe-offs.

Open problem : Shorter Sampling In both chapters 6 and 7 our aim was to improve the *efficiency* of known Gaussian Sampling Algorithm. Another goal for practical application would be to also improve the *quality*, that is, being able to sample shorter vectors by relaxing the condition $\sigma \geq \eta_\iota\mathbb{Z}$ where ι decrease exponentially with the security parameter $\iota \approx 2^{-\lambda}$.We spotted the potential lack of tightness in the current arguments used for the correction of those algorithm : the notion of statistical distance is not always tightest way to obtain an indistinguishability statement. For indistinguishability, another measurement was devellopped by in the field of Information Theory, called Kullback Leibler divergence. Preliminary studies suggest that one could take a larger $\iota \approx 2^{-\lambda/2}$; and this would end up decreasing $\eta_\iota\mathbb{Z}$ by a factor $\sqrt{2}$; this could also decrease by a factor about 2 the mantissa size requirement for high-precisions operations of the algorithms of chapter 6. All those improvement might come *for free*; in the sense that they would require no modifications in the algorithms themselves, only the security proof would have to be adapted.

A recent work of Brakersi, Langlois, Peikert, Regev and Stehlé [BLP+13] proposed to use a rejection technique to improve Klein's algorithm allowing them to choose ι only polynomially small in the dimension n; yet this algorithm was designed for a security reduction, and the question of the practical efficiency of such algorithm is not studied. One issue is that it needs to compute with high precision a rejection rate that includes the value of $\rho_\sigma(\mathbb{Z} + c) = \sum_{x\in\mathbb{Z}} \exp\left(\frac{(x+c)^2}{2\sigma^2}\right)$ for arbitrary value of $c \in \mathbb{R}$. It would be very interesting to see if the techniques of this chapter could be adapted to their algorithm.

Even beyond, one frustrating point of current algorithms is that, even given the best basis of L, we are limited to $\sigma \geq \lambda_1(L)\eta_\iota(\mathbb{Z})$, while for a random lattice we expect the smoothing parameter of the lattice to be $\eta_\iota(L) \approx \frac{\lambda_1(L)}{\sqrt{n}}\eta_\iota(\mathbb{Z})$. Yet, it might be impossible to sample efficiently for σ that small; this problem is likely to be as hard as an approximation of CVPP_γ for a constant approximation factor γ. Exploring this gap seems an interesting research topic.

Chapter 8

BLISS, An optimized Lattice Signature Scheme

She'll make point five past lightspeed. She may not look like much, but she's got it where it counts, kid. I've made a lot of special modifications myself.
Han Solo

Resumé

Ce chapitre reprend les contributions principales de l'article *Lattice Signature and Bimodal Gaussian*, co-signé avec A. Durmus, T. Lepoint et V.Lyubashevsky et publié à Crypto 2013.

Notre résultat principal est la construction d'un schéma de signatures à base de réseaux, qui représente, en pratique comme en théorie, une amélioration par rapport a tous les schémas prouvablement sûrs à base de réseaux proposés à ce jour. Ce nouveau schéma est obtenu en modifiant l'algorithme de tirage avec rejet qui est au cœur du schéma de signature de Lyubashevsky (Eurocrypt, 2012), et de certaines autres primitives à base de réseaux. Notre nouvel algorithme de rejet s'appuie sur une distribution Gaussienne *bimodale*, et combiné avec d'autres modifications, nous obtenons au final des signatures plus courtes d'un facteur asymptotiquement en racine carré du paramètre de sécurité. Les implémentations de notre schéma pour les paramètres de sécurité de 128, 160 et 192 bits se comparent favorablement aux schémas existants tels que RSA ou ECDSA en terme d'efficacité. De plus, notre schéma offre des signatures et des clés publiques plus compactes que tous les schémas à base de réseau proposés auparavant, y compris ceux offrant des niveaux de sécurité *plus faibles*.

Notre implémentation est Open-Source (license CeCILL), disponible sur `http://bliss.di.ens.fr/`.

Abstract

This chapter contains the main contributions of the article *Lattice Signature and Bimodal Gaussian*, coauthored with A. Durmus, T. Lepoint et V.Lyubashevsky and published at Crypto 2013.

Our main result is a construction of a lattice-based digital signature scheme that represents an improvement, both in theory and in practice, over today's most efficient provably secure lattice schemes. The novel scheme is obtained as a result of a modification of the rejection sampling algorithm that is at the heart of Lyubashevsky's signature scheme (Eurocrypt, 2012) and several other lattice primitives. Our new rejection sampling algorithm which samples from a *bimodal* Gaussian distribution, combined with a modified scheme instantiation, ends up reducing the standard deviation of the resulting signatures by a factor that is asymptotically square root in the security parameter. The implementations of our signature scheme for security levels of 128, 160, and 192 bits compare very favorably to existing schemes such as RSA and ECDSA in terms of efficiency. In addition, the new scheme has shorter signature and public key sizes than all previously proposed lattice signature schemes, even those with *lower* security levels.

Or implementation is Open-Source (under CeCILL license), available at `http://bliss.di.ens.fr/`.

TABLE 8.1 – Benchmarking on a desktop computer (Intel Core i7 at 3.4Ghz, 32GB RAM) with our implementation of BLISS and `openssl` 1.0.1c implementation of RSA and ECDSA

Scheme	Security	Sig. Size	SK Size	PK Size	Sign (ms)	Sign/s	Verif. (ms)	Verify/s
BLISS-0	≤ 60 bits	3.3 kb	1.5 kb	3.3 kb	0.241	4k	0.017	59k
BLISS-I	128 bits	5.6 kb	2 kb	7 kb	0.124	8k	0.030	33k
BLISS-II	128 bits	5 kb	2 kb	7 kb	0.480	2k	0.030	33k
BLISS-III	160 bits	6 kb	3 kb	7 kb	0.203	5k	0.031	32k
BLISS-IV	192 bits	6.5 kb	3 kb	7 kb	0.375	2.5k	0.032	31k
RSA 1024	72-80 bits	1 kb	1 kb	1 kb	0.167	6k	0.004	91k
RSA 2048	103-112 bits	2 kb	2 kb	2 kb	1.180	0.8k	0.038	27k
RSA 4096	≥ 128 bits	4 kb	4 kb	4 kb	8.660	0.1k	0.138	7.5k
ECDSA 160	80 bits	0.32 kb	0.16 kb	0.16 kb	0.058	17k	0.205	5k
ECDSA 256	128 bits	0.5 kb	0.25 kb	0.25 kb	0.106	9.5k	0.384	2.5k
ECDSA 384	192 bits	0.75 kb	0.37 kb	0.37 kb	0.195	5k	0.853	1k

8.1 Introduction

Lattice cryptography is arguably the most promising replacement for standard cryptography after the eventual coming of quantum computers. The most ubiquitous public-key cryptographic primitives, encryption schemes [HPS98,LPR10] and digital signatures [Lyu12,GLP12], already have somewhat practical lattice-based instantiations. In addition, researchers are rapidly discovering new lattice-based primitives, such as fully-homomorphic encryption [Gen09], multi-linear maps [GGH12], and attribute-based encryption [SW12], that had no previous constructions based on classical number-theoretic techniques. Even though the above primitives are quite varied in their functionalities, many of them share the same basic building blocks. Thus an improvement in one of these fundamental building blocks, usually results in the simultaneous improvement throughout lattice cryptography. For example, the recent work on the lattice trapdoor generation algorithm [MP12] resulted in immediate efficiency improvements in lattice-based hash-and-sign signatures, identity-based encryption schemes, group signatures, and functional encryption schemes.

In this work, we propose an improvement of another such building block – the *rejection sampling* procedure that is present in the most efficient constructions of lattice-based digital signatures [Lyu12,GLP12], authentication schemes [Lyu09], blind signatures [Rüc10], and zero-knowledge proofs used in multi-party computation [DPSZ12]. As a concrete application, we show that with our new algorithm, lattice-based digital signatures become completely practical. We construct and implement a family of digital signature schemes, named BLISS (Bimodal Lattice Signature Scheme) for security levels of 128, 160, and 192 bits. On standard 64-bit processors, our proof-of-concept implementations are significant improvements over previous lattice-based signatures and compare very favorably to the `openssl` implementations of RSA and ECDSA [1]signatures schemes (see Table 8.1). Note that throughout the chapter, *kb* refers to kilobits.

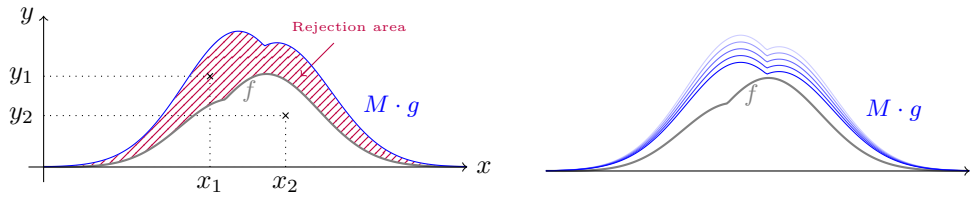
As part of our implementation, we also designed several novel algorithms that could also be of independent interest. Chiefly among them is a new procedure that very efficiently samples from the Gaussian distribution over $\mathbb{Z}^m$ without requiring a very large look-up table. The absence of such an algorithm made researchers avoid using the Gaussian distribution when implementing lattice-based schemes on constrained devices, which resulted in these schemes being less compact than they could have been [GLP12].

8.1.1 Related Work

Rejection Sampling.

Rejection sampling in lattice constructions was first used by Lyubashevsky [Lyu08] to construct a three-round identification scheme. A standard identification scheme is a three round sigma protocol that consists of a commit, challenge, and response stages. The main idea underlying their constructions and security proofs from number theoretic assumptions (e.g. Schnorr and GQ schemes [BP02]) is that the value y committed to in the first stage is used to information-theoretically hide the secret key s in the third stage. This is relatively straight-forward to do in number-theoretic schemes because one can just commit to a random y and then add it to (or multiply it by) some challenge-depending function of s. Since all operations are performed in a finite ring, y being uniformly random hides s. In lattice constructions, however, we need to hide the secret key with a *small* y. The solution is thus to choose y

1. ECDSA on a prime field $\mathbb{F}_p$: ecdsap160, ecdsap256 and ecdsap384 in openssl.



(c) (x_i, y_i) is sampled uniformly in the area under $M \cdot g$, and accepted when $y_i \leq f(x_i)$

(d) M can be reduced when g is better adapted to f

FIGURE 8.1 – Rejection sampling from the distribution of g to get the distribution of f

from a narrow distribution and then perform rejection sampling so that s is not leaked when we add y to it (we describe this idea in much greater detail in Section 8.1.2). The improvements in lattice-based identification schemes (and therefore signature schemes via the Fiat-Shamir transformation) partly came via picking distributions that were more amenable to rejection sampling.

Lattice Signatures.

Early lattice-based signature proposals [GGH97, HPS01, HNHGSW03] did not have security reductions, and they were all subsequently broken because it turned out that every signature leaked a part of the secret key [GS02, NR09, DN12b]. Of the provably-secure signature schemes, [GPV08, Lyu09], [Lyu12, MP12], the most efficient seems to be that of [Lyu12] whose most efficient instantiation has signature and key size on the order of 9kb [GLP12] for an approximately 80-bit security level. [2]

8.1.2 Our Results and Techniques

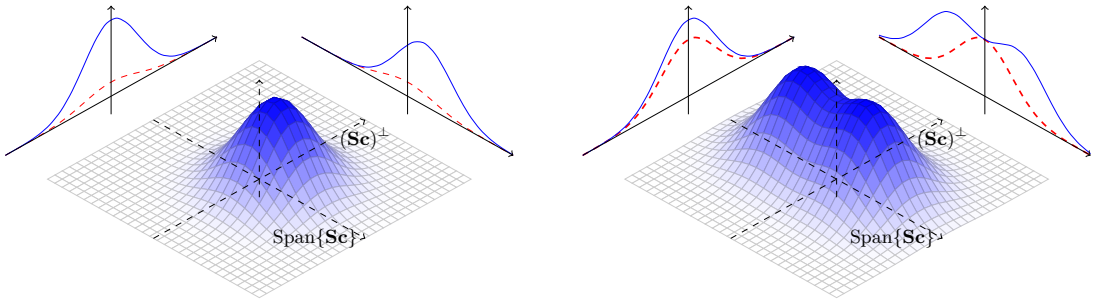
Rejection Sampling and Signature Construction.

To understand the improvement of the rejection sampling procedure in this work, we believe that it is best to first give an overview of rejection sampling and the currently most efficient way in which it is used in [Lyu12]. Rejection sampling is a well-known method introduced by von Neumann [vN51] to sample from an arbitrary target probability distribution f, given the ability to sample according to a different probability distribution g. Conceptually, the method works as follows. A sample x is drawn from g and is accepted with probability $f(x)/(M \cdot g(x))$, where M is some positive real. If it is not accepted, then the process is restarted. It is not hard to prove that if for all x, we have $f(x) \leq M \cdot g(x)$, then the rejection sampling procedure produces exactly the distribution of f. Furthermore, because the expected number of times the procedure will need to be restarted is M, it is crucial to keep M as small as possible by possibly tailoring the function g so that it resembles the target function f as much as possible. In particular, since rejection sampling can be interpreted as sampling a random point (x_i, y_i) in the area under the distribution $M \cdot g$ (see Figure 8.1) and accepting if and only if $y_i \leq f(x_i)$, reducing the area between the two curves will reduce M.

The digital signature from [Lyu12] works as follows (for the sake of this discussion, we will present the simplest version based on SIS) : the secret key is an $m \times n$ matrix $\mathbf{S}$ with small coefficients, and the public key consists of a random $n \times m$ matrix $\mathbf{A}$ whose entries are uniform in $\mathbb{Z}_q$ and $\mathbf{T} = \mathbf{AS} \bmod q$. There is also a cryptographic hash function H, modeled as a random oracle, which outputs elements in $\mathbb{Z}^n$ with small norms. To sign a message digest μ, the signing algorithm first picks a vector $\mathbf{y}$ according to the distribution $D_{\mathbb{Z}^m,\sigma}$, where $D_{\mathbb{Z}^m,\sigma}$ is the discrete Gaussian distribution over $\mathbb{Z}^m$ with standard deviation σ. The signer then computes $\mathbf{c} = H(\mathbf{Ay} \bmod q, \mu)$ and produces a potential signature $(\mathbf{z}, \mathbf{c})$ where $\mathbf{z} = \mathbf{Sc} + \mathbf{y}$. Notice that the distribution of $\mathbf{z}$ depends on the distribution of $\mathbf{Sc}$, and thus on the distribution of $\mathbf{S}$ – in fact, the distribution of $\mathbf{z}$ is exactly $D_{\mathbb{Z}^m,\sigma}$ *shifted* by the vector $\mathbf{Sc}$.

To remove the dependence of the signature on $\mathbf{S}$, rejection sampling is used. The target distribution that we want all the signatures to have is $D_{\mathbb{Z}^m,\sigma}$, whereas we obtain samples from the distribution $D_{\mathbb{Z}^m,\sigma}$ shifted by $\mathbf{Sc}$ (call this distribution $D_{\mathbb{Z}_m,\sigma,\mathbf{Sc}}$). To use rejection sampling, we need to find a positive real M such that for all (or all but a negligible fraction) $\mathbf{x}$ distributed according to $D_{\mathbb{Z}^m,\sigma}$ we have

2. In [GLP12], a 100-bit security level was claimed, but the cryptanalysis we use in this current paper, which combines lattice-reduction attacks with combinatorial meet-in-the-middle techniques [HG07], estimates the actual security level to be around 75-80 bits.



(a) In the original scheme of [Lyu12]. (b) In our scheme.

FIGURE 8.2 – Improvement of Rejection Sampling with Bimodal Gaussian Distributions. In blue is the distribution of $\mathbf{z}$, for fixed $\mathbf{Sc}$ and over the space of all $\mathbf{y}$ in Figure (a) and all $(b, \mathbf{y})$ in Figure (b), before the rejection step and its decomposition as a Cartesian product over $\mathrm{Span}\{\mathbf{Sc}\}$ and $(\mathbf{Sc})^{\perp}$. In dashed red is the target distribution scaled by $1/M$.

$D_{\mathbb{Z}^m,\sigma}(\mathbf{x}) \leq M \cdot D_{\mathbb{Z}_m,\sigma,\mathbf{Sc}}(\mathbf{x})$. A simple calculation (see [Lyu12, Lemma 4.5]) shows that

$$D_{\mathbb{Z}^m,\sigma}(\mathbf{x})/D_{\mathbb{Z}_m,\sigma,\mathbf{Sc}}(\mathbf{x}) = \exp\left(\frac{-2\langle \mathbf{x}, \mathbf{Sc}\rangle + \|\mathbf{Sc}\|^2}{2\sigma^2}\right). \tag{8.1}$$

The value of $\langle \mathbf{x}, \mathbf{Sc}\rangle$ behaves in many ways as a one-dimensional discrete Gaussian, and it can be thus shown that $|\langle \mathbf{x}, \mathbf{Sc}\rangle| < \tau\sigma\|\mathbf{Sc}\|$ with probability $1 - \exp(-\Omega(\tau^2))$. Asymptotically, the value of τ is proportional to the square root of the security parameter. Concretely, if we would like to have, for example, $1 - 2^{-100}$ certainty that $|\langle \mathbf{x}, \mathbf{Sc}\rangle| < \tau\sigma\|\mathbf{Sc}\|$, we would set $\tau = 12$. Thus with probability $1 - \exp(-\Omega(\tau^2))$, we have $\exp\left(\frac{-2\langle \mathbf{x},\mathbf{Sc}\rangle + \|\mathbf{Sc}\|^2}{2\sigma^2}\right) \leq \exp\left(\frac{2\tau\sigma\|\mathbf{Sc}\| + \|\mathbf{Sc}\|^2}{2\sigma^2}\right)$. So if $\sigma = \tau\|\mathbf{Sc}\|$, we will have $D_{\mathbb{Z}^m,\sigma}(\mathbf{x})/D_{\mathbb{Z}_m,\sigma,\mathbf{Sc}}(\mathbf{x}) \leq \exp\left(1 + \frac{1}{2\tau^2}\right)$. Therefore if we set $M = \exp\left(1 + \frac{1}{2\tau^2}\right)$, we will be able to use rejection sampling to output signatures that are distributed according to $D_{\mathbb{Z}^m,\sigma}$ where $\sigma = \tau\|\mathbf{Sc}\|$ and have the expected number of repetitions be $M \approx \exp(1)$. [3]

Prior to explaining our technique to improve the scheme, we need to state how the verification algorithm in [Lyu12] works. Upon receiving the signature $(\mathbf{z}, \mathbf{c})$ of μ, the verifier checks that $\|\mathbf{z}\|$ is "small" (roughly $\sigma\sqrt{m}$) and also that $\mathbf{c} = H(\mathbf{Az} - \mathbf{Tc} \bmod q, \mu)$. It is easy to check that the outputs of the signing procedure satisfy the two requirements. In this work, we show how to remove the factor τ (in fact even more) from the required standard deviation. Above, we described how to perform rejection sampling when we were sampling potential signatures as $\mathbf{z} = \mathbf{Sc} + \mathbf{y}$. Consider now, an alternative procedure, where we first uniformly sample a bit $b \in \{-1, 1\}$ and then choose the potential signature to be $\mathbf{z} = b\mathbf{Sc} + \mathbf{y}$. In particular $\mathbf{z}$ is now sampled from the distribution $\frac{1}{2}D_{\mathbb{Z}^m,\sigma,\mathbf{Sc}} + \frac{1}{2}D_{\mathbb{Z}^m,\sigma,-\mathbf{Sc}}$. If our target distribution is still $D_{\mathbb{Z}^m,\sigma}$, then, as in the above paragraph, we need to have $D_{\mathbb{Z}^m,\sigma}(\mathbf{x}) / \left(\frac{1}{2}D_{\mathbb{Z}^m,\sigma,\mathbf{Sc}}(\mathbf{x}) + \frac{1}{2}D_{\mathbb{Z}^m,\sigma,-\mathbf{Sc}}(\mathbf{x})\right) \leq M$. By using Equation (8.1) and some algebraic manipulations, we obtain that

$$D_{\mathbb{Z}^m,\sigma}(\mathbf{x}) \Big/ \left(\frac{1}{2}D_{\mathbb{Z}^m,\sigma,\mathbf{Sc}}(\mathbf{x}) + \frac{1}{2}D^m_{-\mathbf{Sc},\sigma}(\mathbf{x})\right) = \exp\left(\frac{\|\mathbf{Sc}\|^2}{2\sigma^2}\right) \Big/ \cosh\left(\frac{\langle \mathbf{x}, \mathbf{Sc}\rangle}{\sigma^2}\right) \leq \exp\left(\frac{\|\mathbf{Sc}\|^2}{2\sigma^2}\right), \tag{8.2}$$

where the last inequality follows from the fact that $\cosh(y) \geq 1$ for all y. Thus for rejection sampling to work with $M = \exp(1)$, as in the previous example, we only require that $\sigma = \|\mathbf{Sc}\|/\sqrt{2}$ rather than $\tau\|\mathbf{Sc}\|$.

Our improvement is explained pictorially in Figure 8.2. Part 8.2(a) shows the rejection sampling as done in [Lyu12]. There, the distribution $D_{\mathbb{Z}^m,\sigma}$ (the dashed red line) must be scaled by a somewhat large factor so that all but a negligible fraction of it fits under $D_{\mathbb{Z}^m,\sigma,\mathbf{Sc}}$. In 8.2(b), which represents our improved sampling algorithm, the distribution from which we are sampling is bimodal having its two centers at $\mathbf{Sc}$ and $-\mathbf{Sc}$. As can be seen from the figure, the distribution $D_{\mathbb{Z}^m,\sigma}$ fits much "better" (*i.e.* needs to be scaled by a much smaller factor) underneath the bimodal distribution and therefore there is a much smaller rejection area between the two curves. As a side note, whereas in (a), a negligible fraction of the scaled $D_{\mathbb{Z}^m,\sigma}$ is still above $D_{\mathbb{Z}^m,\sigma,\mathbf{Sc}}$, in (b), all of $D_{\mathbb{Z}^m,\sigma}$ is underneath the bimodal distribution $\frac{1}{2}D_{\mathbb{Z}^m,\sigma,\mathbf{Sc}} + \frac{1}{2}D_{\mathbb{Z}^m,\sigma,-\mathbf{Sc}}$.

3. More precisely $\sigma = \tau \max_{\mathbf{S},\mathbf{c}} \|\mathbf{Sc}\|$, since we do not know in advance what $\mathbf{Sc}$ will be.

While the above sampling procedure has the potential to produce much shorter signatures since the Gaussian "tail-cut" factor τ is never used, it does not give us an improved signature by itself because the verification procedure is no longer guaranteed to work. The verification checks that $\mathbf{c} = H(\mathbf{Az} - \mathbf{Tc} \bmod q, \mu)$ and so will verify correctly if and only if $\mathbf{Ay} = \mathbf{Az} - \mathbf{Tc} = \mathbf{A}(b\mathbf{Sc} + \mathbf{y}) - \mathbf{Tc} = \mathbf{Ay} + b\mathbf{Tc} - \mathbf{Tc}$, which will only happen if $b\mathbf{Tc} = \mathbf{Tc} \bmod q$ for $b \in \{-1, 1\}$. In other words, we will need $\mathbf{Tc} = -\mathbf{Tc} \bmod q$, which will never happen if q is prime unless $\mathbf{T} = \mathbf{0}$.[4] Our solution, therefore, is to work modulo $2q$ and set $\mathbf{T} = q\mathbf{I}$ where $\mathbf{I}$ is the $n \times n$ identity matrix. In this case $\mathbf{Tc} = -\mathbf{Tc} \bmod 2q$, and so the verification procedure will always work.

Changing the modulus from q to $2q$ and forcing the matrix $\mathbf{T}$ to always be $q\mathbf{I}$ creates several potential problems. In particular, it is no longer clear how to do the key generation, and also the outline for the security proof from [Lyu12] no longer holds. But we show that these problems can be overcome. We will now sketch the key generation and the security proof based on the hardness of the SIS problem in which one is given a uniformly random matrix $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$, and is asked to find a short vector $\mathbf{w}$ such that $\mathbf{Bw} = \mathbf{0} \pmod q$. To generate the public and secret keys, we first pick a uniformly random matrix $\mathbf{A}' \in \mathbb{Z}_q^{n \times (m-n)}$ and a random $(m-n) \times n$ matrix $\mathbf{S}'$ consisting of short coefficients. We then compute $\mathbf{A}'' = \mathbf{A}'\mathbf{S}' \bmod q$ and output $\mathbf{A} = [2\mathbf{A}' | 2\mathbf{A}'' + q\mathbf{I}]$ as the public key. The secret key is $\mathbf{S} = [\mathbf{S}'^T | -\mathbf{I}]^T$. Notice that by construction we have $\mathbf{AS} = q\mathbf{I} \pmod{2q}$ and $\mathbf{S}$ consists of small entries. The dimensions m and n are be picked so that the distribution of $[\mathbf{A}' | \mathbf{A}'\mathbf{S}' \bmod q]$ can be shown to be uniformly random in $\mathbb{Z}_q^{n \times m}$ by the leftover hash lemma.

In the security proof, we are given a random matrix $\mathbf{B} = [\mathbf{A}' | \mathbf{A}''] \in \mathbb{Z}_q^{n \times m}$ by the challenger and we will use the adversary who forges in the signature scheme to find a short vector $\mathbf{w}$ such that $\mathbf{Bw} = \mathbf{0} \pmod q$. We create the public key $\mathbf{A} = [2\mathbf{A}' | 2\mathbf{A}'' + q\mathbf{I}]$ and give it to the adversary. Even though we do not know a secret key $\mathbf{S}$ such that $\mathbf{AS} = q\mathbf{I} \pmod{2q}$, we can still create valid signatures for any messages of the adversary's choosing by picking the $(\mathbf{z}, \mathbf{c})$ according to the correct distributions and then programming the random oracle as is done in [Lyu12]. When the adversary forges, we use the forking lemma to create two equations $\mathbf{Az} = q\mathbf{c} \pmod{2q}$ and $\mathbf{Az}' = q\mathbf{c}' \pmod{2q}$. Combining them together, we obtain $\mathbf{A}(\mathbf{z} - \mathbf{z}') = q(\mathbf{c} - \mathbf{c}') \pmod{2q}$. Under some very simple requirements for $\mathbf{z}, \mathbf{z}', \mathbf{c}$, and $\mathbf{c}'$, the previous equation implies that $\mathbf{A}(\mathbf{z} - \mathbf{z}') = \mathbf{0} \pmod q$ and $\mathbf{z} \neq \mathbf{z}'$. This then implies that $2\mathbf{B}(\mathbf{z} - \mathbf{z}') = \mathbf{0} \pmod q$ and since 2 is invertible modulo q, we have found a $\mathbf{w} = (\mathbf{z} - \mathbf{z}')$ such that $\mathbf{Bw} = \mathbf{0} \pmod q$.

The above scheme construction and proof works for SIS and equally well for Ring-SIS when instantiated with polynomials. As in [Lyu12], we can also construct much more efficient schemes based on LWE and Ring-LWE by creating the matrix $\mathbf{A}'' = \mathbf{A}'\mathbf{S}'$ such that $(\mathbf{A}', \mathbf{A}'')$ is not uniformly random, but only computationally. For optimal efficiency, though, we can create the key in yet a different manner related to the way NTRU keys are generated. The formal construction is described in Section 8.4, but here we just give the intuition. We could create two small polynomials $\mathbf{s}_1, \mathbf{s}_2 \in \mathbb{Z}[x]/(x^n + 1)$ and output the public key as $\mathbf{a} = \frac{\mathbf{q} - \mathbf{s}_2}{\mathbf{s}_1} \pmod{2q}$. Notice that this implies that $\mathbf{as}_1 + \mathbf{s}_2 = \mathbf{q} \pmod{2q}$, and so we can think of the public key as $\mathbf{A} = [\mathbf{a}, \mathbf{1}]$ and the secret key as $\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2]^T$. Assuming that it is a hard problem to find small vectors $\mathbf{w}$ such that $\mathbf{Aw} = \mathbf{0} \pmod{2q}$, the signature scheme instantiated in the above manner will be secure. To those readers familiar with the key generation in the NTRU encryption scheme, the above key generation should look very familiar, except that the modulus is $2q$ rather than q. Since we are not sure what happens when the modulus is $2q$, we show in Section 8.4 how to instantiate our scheme so that it is based on NTRU over modulus q. We then explain how for certain instantiations, this is as hard a problem as Ring-SIS (using the results of Stehlé, Steinfeld [SS11]) and how for more efficient instantiations, it is a *weaker* assumption than the ones underlying the classic NTRU encryption scheme and the recent construction of fully-homomorphic encryption [LATV12].

Cryptanalysis.

Previous cryptanalytic efforts mostly involved computing the Hermite factor as in the work of Gama and Nguyen [GN08]. In our work, we do a more careful cryptanalysis by running the simulations in BKZ 2.0 of Chen and Nguyen [CN11] in combination with the combinatorial meet-in-the-middle attack of Howgrave-Graham [HG07], which is applicable to the instances in this chapter.

4. One may think that a possible solution could be to output the bit b as part of the signature, but this is not secure. Depending on the sign of $\langle \mathbf{z}, \mathbf{Sc} \rangle$, one of the two values of b is more likely to be output than the other, and this leaks information about $\mathbf{S}$.

8.1.3 Discussion and Open Problems

This work presents an improved rejection sampling algorithm and utilizes it to construct the currently most practical lattice-based signature scheme. For optimal efficiency, the security of our scheme relies on the hardness of a type of NTRU problem that has recently (re-)appeared in the literature [LATV12] and, we believe, could play a big role in the future of lattice-based cryptography (see Section 8.2 for the precise definition of the problem). There is currently no cryptanalytic work that suggests that NTRU lattices of dimension $2n$, which contain n "unusually short" vectors, behave any differently than other structured lattices that contain only one such unusually short vector.

We ran experiments that suggest indeed that BKZ behaves similarly in the presence of either only one unusually short vector or a basis of n of them (see Section 8.6.3). Yet we have no explanation of this phenomenon ; we are not sure whether this means that these lattices are indeed as hard as the random lattices that have been exhaustively studied [GN08, CN11], or whether it means that they simply have not been carefully studied (even though this NTRU problem has been around since 1998 [HPS98]).

We believe that understanding the behavior of lattice-reduction algorithms on these NTRU lattices would bring some much-needed clarity to the current state of lattice-based cryptography. In particular, it still remains unclear whether the classical NTRU encryption scheme [HPS98] (where the ciphertexts consist of one element) is more efficient than the "2-element" Ring-LWE based cryptosystem of Lyubashevsky, Peikert and Regev [LPR10]. For the same reason, it is also unclear whether there is any reason in practice to use the key-generation parameters suggested in [SS11] to make the NTRU cryptosystem as secure as Ring-LWE. Recently Lopez-Alt et al. proposed a fully-homomorphic encryption scheme (with some additional features) based on NTRU [LATV12]. On the surface, it seems that this scheme could be even more efficient than the implemented [GHS12] Ring-LWE version of the scheme of Brakerski et al. [BGV12] because the NTRU encryptions are potentially shorter and the homomorphic re-encryption operation does not require the "key switching/dimension reduction" step.

The dearth of lattice cryptanalysis papers stands in contrast to the vast number of articles proposing theoretical lattice-based constructions. Our belief is that this lack of cryptanalytic effort is in part due to the fact that most of the papers with scheme proposals give no concrete targets to attack. One of the proposed instantiations in the present work is a "toy example" that we estimate has approximately 60 bits of security. Thus if it turns out that the NTRU lattices are weaker than believed, it is wholly possible that this example could be broken on a personal computer, and we think this would be of great interest to the practical community. In addition, it could be argued that we do not yet know enough about lattice reduction to be able to propose such "fine-grained" security estimates like 160-bit or 192-bit. But one of the main reasons that we make these proposals is to make it "worthwhile" for cryptanalysts to work on these problems. In short, one of our hopes is that this work spurs on the cryptanalysis that is currently much needed in the field.

8.2 Preliminaries

8.2.1 Hardness Assumptions

All the constructions in this chapter are based on the hardness of the *generalized SIS* (Short Integer Solution) problem, which we define below.

Definition 8.1 ($\mathcal{R}$-$\mathrm{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem) *Let $\mathcal{R}$ be some ring and $\mathcal{K}$ be some distribution over $\mathcal{R}_q^{n\times m}$, where $\mathcal{R}_q$ is the quotient ring $\mathcal{R}/(q\mathcal{R})$. Given a random $\mathbf{A} \in \mathcal{R}_q^{n\times m}$ drawn according to the distribution $\mathcal{K}$, find a non-zero $\mathbf{v} \in \mathcal{R}_q^m$ such that $\mathbf{Av} = \mathbf{0}$ and $\|\mathbf{v}\|_2 \leq \beta$.*

If we let $\mathcal{R} = \mathbb{Z}$ and $\mathcal{K}$ be the uniform distribution, then the resulting problem is the classical SIS problem first defined by Ajtai [Ajt96] in his seminal paper showing connections between worst-case lattice problems and the average-case SIS problem. By the pigeonhole principle, if $\beta \geq \sqrt{m}q^{n/m}$ then the SIS instances are guaranteed to have a solution. Using Gaussian techniques, Micciancio and Regev [MR04] improved Ajtai's result to show that, for a large enough q as a function of n and β, the $\mathrm{SIS}_{q,n,m,\beta}$ problem is as hard (on the average) as the $\tilde{\mathcal{O}}(\sqrt{n}\beta)$-SIVP problem for *all* lattices of dimension n.

In 2006, a ring variant of SIS was introduced independently in [PR06] and [LM06]. In [LM06] it was shown that if $\mathcal{R} = \mathbb{Z}[x]/(x^n + 1)$, where n is a power of 2, then the $\mathcal{R}$-$\mathrm{SIS}^{\mathcal{K}}_{q,1,m,\beta}$ problem is as hard as the $\tilde{\mathcal{O}}(\sqrt{n}\beta)$-SVP problem in all lattices that are ideals in $\mathcal{R}$ (where $\mathcal{K}$ is again the uniform distribution over $\mathcal{R}_q^{1\times m}$).

NTRU Lattices In the NTRU cryptosystem [HPS98] over the ring $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n+1)$, the key generation procedure picks two short secret keys $\mathbf{f}, \mathbf{g} \in \mathcal{R}_q$ (according to some distribution) and computes the public key as $\mathbf{a} = \mathbf{g}/\mathbf{f}$.[5] When the norm of $\mathbf{f}, \mathbf{g}$ is large enough, it can be shown that $\mathbf{a}$ is actually uniformly random in $\mathcal{R}_q$ [SS11], but even when the secret keys do not have enough entropy, their quotient still appears to be pseudorandom, although no proof of this fact is known [LATV12]. In the NTRU cryptosystem (or its more secure modification of [SS11] which is based on the Ring-LWE problem), one encrypts a message μ, represented as a polynomial in $\mathcal{R}_q$ with $\{0,1\}$ coefficients, by picking two short vectors $\mathbf{r}, \mathbf{e} \in \mathcal{R}_q$ and outputting $\mathbf{z} = 2(\mathbf{ar} + \mathbf{e}) + \mu$. The security of the scheme relies on the fact that the distribution of $(\mathbf{a}, \mathbf{z})$ is pseudo-random in $\mathcal{R}_q^2$.

One can define an NTRU version of the SIS problem that is at least as hard as breaking the NTRU cryptosystem. In particular, given an NTRU public key $\mathbf{a}$, find two polynomials $\mathbf{v}_1, \mathbf{v}_2 \in \mathcal{R}_q$ such that $\|(\mathbf{v}_1|\mathbf{v}_2)\| \leq \beta$ and $\mathbf{av}_1 + \mathbf{v}_2 = 0$ in $\mathcal{R}_q$. Notice that $(\mathbf{f}, -\mathbf{g})$ is a solution to this problem, but in fact, finding larger solutions can also be useful in breaking the NTRU cryptosystem. In particular, notice that for any solution $(\mathbf{v}_1|\mathbf{v}_2)$, one can compute $\mathbf{zv}_1 = 2(-\mathbf{rv}_2 + \mathbf{ev}_1) + \mu\mathbf{v}_1$. If β is sufficiently small with respect to $\|(\mathbf{r}|\mathbf{e})\|$, then $\mathbf{z} \cdot \mathbf{v}_1 \bmod 2 = \mu\mathbf{v}_1$, and μ can be recovered. Thus, for certain parameters, the NTRU version of the SIS problem is at least as hard as breaking the NTRU cryptosystem. In Section 8.6, we analyze the hardness of this problem using combinations of lattice [CN11] and hybrid attacks [HG07]. As a side-note, we would like to point out that the NTRU encryption scheme remains hard even after 15 years of cryptanalysis. The weakness in the NTRU signature scheme, which uses the same key generation procedure, is due to the fact that signatures slowly leak the secret key([NR09, DN12b], chapter 5), but this is provably (*i.e.* information-theoretically) avoided in our scheme.

A way to state the NTRU SIS problem in terms of the $\mathcal{R}$-$\mathsf{SIS}^{\mathcal{K}}_{q,1,2,\beta}$ problem is to set $\mathcal{R} = \mathbb{Z}[x]/(x^n+1)$ and let $\mathcal{K}$ be the distribution that picks small $\mathbf{f}, \mathbf{g}$ and outputs the public key $\mathbf{A} = (\mathbf{a}, \mathbf{1}) \in \mathcal{R}_q^{1\times 2}$ for $\mathbf{a} = \mathbf{g}/\mathbf{f}$.

8.3 BLISS : A Lattice Signature Scheme using Bimodal Gaussians

In this section, we present our new signature scheme along with the proof of correctness and the security proof based on the $\mathcal{R} - \mathsf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem. We mention that this is the "simple" version of our algorithm, and the specific implementation of it that uses numerous enhancements is presented in Section 8.4. For simplicity, we present our algorithm for $\mathcal{R} = \mathbb{Z}$, but it works exactly the same way for rings $\mathcal{R} = \mathbb{Z}[x]/(x^n+1)$ (see Section 8.4).

8.3.1 New Signature and Verification Algorithms

Key pairs The secret key is a (short) matrix $\mathbf{S} \in \mathbb{Z}_{2q}^{m\times n}$ and the public key is given by the matrix $\mathbf{A} \in \mathbb{Z}_{2q}^{n\times m}$ such that $\mathbf{AS} = q\mathbf{Id}_n \pmod{2q}$. A crucial property, for our new rejection sampling algorithm, satisfied by the key pair, is that $\mathbf{AS} = \mathbf{A}(-\mathbf{S}) = q\mathbf{Id}_n \pmod{2q}$. Obtaining such a key pair is easy and can be done efficiently. In Section 8.7 we explain the key-generation procedure which results in a scheme whose security is based on the classic $\mathrm{SIS}_{q,n,m,\beta}$ problem and in Section 8.4 we present an "NTRU-like" variant of the key generation which yields a more efficient instantiation of the signature scheme.

Random Oracle Domain We model the hash function H as a random oracle that has uniform output in $\mathbb{B}_\kappa^n$, the set of binary vectors of length n and weight κ. An efficient construction of such a random oracle can be found in Section 8.4.4.

The Signature Algorithm The signer, who is given a message digest μ, first samples a vector $\mathbf{y}$ from the m-dimensional discrete Gaussian distribution D_σ^m and then computes $\mathbf{c} \leftarrow H(\mathbf{Ay} \bmod 2q, \mu)$. He then samples a bit b in $\{0,1\}$ and computes the potential output $\mathbf{z} \leftarrow \mathbf{y} + (-1)^b\mathbf{Sc}$. Notice that $\mathbf{z}$ is distributed according to the bimodal discrete Gaussian distribution $\frac{1}{2}D^m_{\mathbf{Sc},\sigma} + \frac{1}{2}D^m_{-\mathbf{Sc},\sigma}$. At this point we perform rejection sampling and output the signature $(\mathbf{z}, \mathbf{c})$ with probability $1\Big/\left(M\exp\left(-\frac{\|\mathbf{Sc}\|^2}{2\sigma^2}\right)\cosh\left(\frac{\langle\mathbf{z},\mathbf{Sc}\rangle}{\sigma^2}\right)\right)$, where M is some fixed positive real that is set large enough to ensure that the preceding probability is always at most 1. We explain how to set the M in accordance with the standard deviation σ in the next

5. In the original NTRU scheme, the ring was $\mathbb{Z}_q[x]/(x^n-1)$, but lately researchers have also used $\mathbb{Z}_q[x]/(x^n+1)$ when n is a power of 2. Indeed, the latter choice seems at least as secure.

Algorithm 29 Signature Algorithm

Input: Message μ, public key $\mathbf{A} \in \mathbb{Z}_{2q}^{n \times m}$, secret key $\mathbf{S} \in \mathbb{Z}_{2q}^{m \times n}$, standard deviation σ
Output: A signature $(\mathbf{z}, \mathbf{c})$ of the message μ

1: $\mathbf{y} \leftarrow D_\sigma^m$
2: $\mathbf{c} \leftarrow H(\mathbf{A}\mathbf{y} \bmod 2q, \mu)$
3: Choose a random bit $b \in \{0, 1\}$
4: $\mathbf{z} \leftarrow \mathbf{y} + (-1)^b \mathbf{S}\mathbf{c}$
5: **Output**$(\mathbf{z}, \mathbf{c})$ with probability $1 \Big/ \left(M \exp\left(-\frac{\|\mathbf{S}\mathbf{c}\|^2}{2\sigma^2}\right) \cosh\left(\frac{\langle \mathbf{z}, \mathbf{S}\mathbf{c}\rangle}{\sigma^2}\right)\right)$ otherwise **restart**

Algorithm 30 Verification Algorithm

Input: Message μ, public Key $\mathbf{A} \in \mathbb{Z}_{2q}^n$, signature $(\mathbf{z}, \mathbf{c})$
Output: Accept or Reject the signature

1: **if** $\|\mathbf{z}\| > B_2$ **then** Reject
2: **if** $\|\mathbf{z}\|_\infty \geq q/4$ **then** Reject
3: Accept iff $\mathbf{c} = H(\mathbf{A}\mathbf{z} + q\mathbf{c} \bmod 2q, \mu)$

section. If the signing algorithm did not output the signature, then it is restarted and is repeated until something is output. The expected number of iterations of the signing algorithm is M.

The Verification Algorithm The verification algorithm will accept $(\mathbf{z}, \mathbf{c})$ as the signature for μ if the following three conditions hold :

1. $\|\mathbf{z}\| \leq B_2$
2. $\|\mathbf{z}\|_\infty < q/4$
3. $\mathbf{c} = H(\mathbf{A}\mathbf{z} + q\mathbf{c} \bmod 2q, \mu)$

The signer outputs signatures of the form $(\mathbf{z}, \mathbf{c})$ where $\mathbf{z}$ is distributed according to D_σ^m, thus the acceptance bound B_2 should be set a little bit higher than $\sqrt{m}\sigma$, which is the expected value around which the output of D_σ^m is tightly concentrated around; denoting $B_2 = \eta\sqrt{m}\sigma$, one can set η so that $\|\mathbf{z}\| \leq B_2$ is verified with probability $1 - 2^{-\lambda}$ [Lyu12, Lemma 4.4] for the security parameter λ (in practice, $\eta \in [1.1, 1.4]$). For technical reasons in the security proof, we also need that $\|\mathbf{z}\|_\infty < q/4$, but this condition is usually verified whenever the first one is and does restrict the manner in which we choose the parameters for the scheme (see Section 8.3.3). Condition 3 will also hold for valid signatures because

$$\mathbf{A}\mathbf{z} + q\mathbf{c} = \mathbf{A}(\mathbf{y} + (-1)^b\mathbf{S}\mathbf{c}) + q\mathbf{c} = \mathbf{A}\mathbf{y} + \big((-1)^b\mathbf{A}\mathbf{S}\big)\mathbf{c} + q\mathbf{c} = \mathbf{A}\mathbf{y} + (q\mathbf{Id}_n)\mathbf{c} + q\mathbf{c} = \mathbf{A}\mathbf{y} \bmod 2q.$$

8.3.2 Rejection Sampling : Correctness and Efficiency

We now explain how to pick the standard deviation σ and positive real M so that the signing algorithm in the preceding section produces vectors $\mathbf{z}$ according to the distribution D_σ^m. Because $\mathbf{y}$ is distributed according to D_σ^m, it is easy to see that in Step 4 of the signing algorithm, $\mathbf{z}$ is distributed according to $g_{\mathbf{S}\mathbf{c}} = \frac{1}{2}D^m_{\mathbf{S}\mathbf{c},\sigma} + \frac{1}{2}D^m_{-\mathbf{S}\mathbf{c},\sigma}$ for fixed $\mathbf{S}\mathbf{c}$ and over the space of all $(b, \mathbf{y})$. Thus for any $\mathbf{z}^* \in \mathbb{R}^m$, we have

$$\begin{aligned}
\Pr[\mathbf{z} = \mathbf{z}^*] &= \frac{1}{2}D^m_{\mathbf{S}\mathbf{c},\sigma}(\mathbf{z}^*) + \frac{1}{2}D^m_{-\mathbf{S}\mathbf{c},\sigma}(\mathbf{z}^*) \\
&= \frac{1}{2\rho_\sigma(\mathbb{Z}^m)} \exp\left(-\frac{\|\mathbf{z}^* - \mathbf{S}\mathbf{c}\|^2}{2\sigma^2}\right) + \frac{1}{2\rho_\sigma(\mathbb{Z}^m)} \exp\left(-\frac{\|\mathbf{z}^* + \mathbf{S}\mathbf{c}\|^2}{2\sigma^2}\right) \\
&= \frac{1}{2\rho_\sigma(\mathbb{Z}^m)} \exp\left(-\frac{\|\mathbf{z}^*\|^2}{2\sigma^2}\right) \exp\left(-\frac{\|\mathbf{S}\mathbf{c}\|^2}{2\sigma^2}\right) \left(e^{-\frac{\langle \mathbf{z}^*, \mathbf{S}\mathbf{c}\rangle}{\sigma^2}} + e^{\frac{\langle \mathbf{z}^*, \mathbf{S}\mathbf{c}\rangle}{\sigma^2}}\right) \\
&= \frac{1}{\rho_\sigma(\mathbb{Z}^m)} \exp\left(-\frac{\|\mathbf{z}^*\|^2}{2\sigma^2}\right) \exp\left(-\frac{\|\mathbf{S}\mathbf{c}\|^2}{2\sigma^2}\right) \cosh\left(\frac{\langle \mathbf{z}^*, \mathbf{S}\mathbf{c}\rangle}{\sigma^2}\right) .
\end{aligned}$$

The desired output distribution is the centered Gaussian distribution $f(\mathbf{z}^*) = \rho_\sigma(\mathbf{z}^*)/\rho_\sigma(\mathbb{Z}^m)$. Thus, by Lemma 2.7, one should accept the sample $\mathbf{z}^*$ with probability :

$$p_{\mathbf{z}*} = \frac{f(\mathbf{z}^*)}{M g_{\mathbf{S}\mathbf{c}}(\mathbf{z}^*)} = 1 \Big/ \left(M \exp\left(-\frac{\|\mathbf{S}\mathbf{c}\|^2}{2\sigma^2}\right) \cosh\left(\frac{\langle \mathbf{z}^*, \mathbf{S}\mathbf{c}\rangle}{\sigma^2}\right)\right),$$

where M is chosen large enough so that $p_{\mathbf{z}*} \leq 1$. Note that $\cosh(x) \geq 1$ for any x, so it suffices that

$$M = e^{\frac{1}{2\alpha^2}} \tag{8.3}$$

where α is such that $\sigma \geq \alpha \cdot \|\mathbf{Sc}\|$.

Bound on $\|\mathbf{Sc}\|$ First, let us precise that since we use column notation in this chapter, $\|\mathbf{S}\|$ denotes the maximal ℓ_2-norm of the column of $\mathbf{S}$.

Notice that if we fix the repetition rate M, then the standard deviation of the signature $\mathbf{z}$, and therefore also its size, only depend on the maximum possible norm of the vector $\mathbf{Sc}$. For this reason, it is important to obtain a bound as tight as possible on this product. Several upper bounds on $\|\mathbf{Sc}\|$ can be used such as $\|\mathbf{Sc}\| \leq \|\mathbf{c}\|_1 \cdot \|\mathbf{S}\| = \kappa \|\mathbf{S}\|$ (as in [Lyu12]) or $\|\mathbf{Sc}\| \leq s_1(\mathbf{S}) \cdot \|\mathbf{c}\| = s_1(\mathbf{S}) \cdot \sqrt{\kappa}$ where $s_1(\mathbf{S})$ is the singular norm of $\mathbf{S}$. Here we introduce a new measure of $\mathbf{S}$, adapted to the form of $\mathbf{c}$, which helps us achieve a tighter bound than all the previous methods. We believe that this norm and the technique for bounding it could be of independent interest.

Definition 8.2 *For any integer κ, we define $N_\kappa \colon \mathbb{R}^{m \times n} \to \mathbb{R}$ as :*

$$N_\kappa(\mathbf{X}) = \max_{\substack{I \subset \{1,\ldots,n\} \\ \#I = \kappa}} \sum_{i \in I} \left(\max_{\substack{J \subset \{1,\ldots,n\} \\ \#J = \kappa}} \sum_{j \in J} T_{i,j} \right) \quad \textit{where } \mathbf{T} = \mathbf{X}^t \cdot \mathbf{X} \in \mathbb{R}^{n \times n} .$$

The following proposition states that $\sqrt{N_\kappa(\mathbf{S})}$ is also an upper bound for $\|\mathbf{Sc}\|$.

proposition 8.1 *Let $\mathbf{S} \in \mathbb{R}^{m \times n}$ be a real matrix. For any $\mathbf{c} \in \mathbb{B}_\kappa^n$, we have $\|\mathbf{Sc}\|^2 \leq N_\kappa(\mathbf{S})$.*

Proof: Set $I = J = \{i \in \{1, \ldots, n\} : \mathbf{c}_i = 1\}$, which implies $\#I = \#J = \kappa$. Rewriting $\|\mathbf{S} \cdot \mathbf{c}\|^2 = \mathbf{c}^t \cdot \mathbf{S}^t \cdot \mathbf{S} \cdot \mathbf{c} = \mathbf{c}^t \cdot \mathbf{T} \cdot \mathbf{c} = \sum_{i \in I} \sum_{j \in J} T_{i,j}$, we can conclude from the definition of N_κ. □

In practice, we will use this upper bound (see Section 8.4) to bound $\|\mathbf{Sc}\|$ and derive the parameters. Some secret keys $\mathbf{S}$ will be rejected according to the value of $N_\kappa(\mathbf{S})$, which is easily computable. In addition to the gain from the use of bimodal Gaussians, this new upper bound lowers the standard deviation σ by a factor $\approx \sqrt{\kappa}/2$ compared to [Lyu12].

8.3.3 Security Proof

Any adversary able to forge against our signature scheme can solve the $\mathcal{R} - \mathsf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem for $\beta = 2B_2$ where $\mathcal{K}$ is the distribution induced by the public-key generation algorithm.

Theorem 8.2 *Suppose there is a polynomial-time algorithm $\mathcal{F}$ which makes at most s queries to the signing oracle and h queries to the random oracle H, and succeeds in forging with non negligible probability δ. Then there exists a polynomial-time algorithm which can solve the $\mathcal{R} - \mathsf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem for $\beta = 2B_2$ with probability $\approx \frac{\delta^2}{2(h+s)}$. Moreover the signing algorithm produces a signature with probability $\approx 1/M$ and the verifying algorithm accepts a signature produced by an honest signer with probability at least $1 - 2^m$.*

The proof of the theorem follows from standard arguments, and is similar to the proof of [Lyu12]. In particular, the fact that the distribution of the signatures in the scheme does not depend on the secret key means that the simulator can "sign" arbitrary messages without having the secret key by programming the random oracle. Then when the adversary forges on a message, the simulator can extract a solution to the SIS problem. The main difference arises with so called type-2 forgeries, for which the arguments actually become simpler and tighter for the current scheme. It is proved in a sequence of two Lemmas. In Lemma 8.3, we show that our signing algorithm can be replaced by Hybrid 2 (Algorithm 32), and the statistical distance between the two outputs will be at most $\epsilon = s(s+h)2^{-n+1}$. Since Hybrid 2 produces an output with probability exactly $1/M$, the signing algorithm produces an output with probability at least $(1-\epsilon)/M$. Then in Lemma 8.4 we show that if a forger can produce a forgery with probability δ when the signing algorithm is replaced by Hybrid 2, then we can use him to recover a vector $\mathbf{v} \neq \mathbf{0}$ such that $\|\mathbf{v}\| \leq \beta = 2B_2$ and $\mathbf{Av} = \mathbf{0} \bmod q$ with probability at least $\delta^2/(2(s+h))$.

Algorithm 31 Hybrid 1

$\mathbf{y} \leftarrow \mathcal{D}_\sigma^m$
$\mathbf{c} \leftarrow \mathbb{B}_\kappa^n$
Choose a random bit b
$\mathbf{z} \leftarrow (-1)^b\mathbf{Sc} + \mathbf{y}$
With probability $1/\big(M\exp(-\frac{\|\mathbf{Sc}\|^2}{2\sigma^2})\cosh(\frac{\langle \mathbf{z},\mathbf{Sc}\rangle}{\sigma^2})\big)$:
    output $(\mathbf{z}, \mathbf{c})$
    program $H(\mathbf{Az} + q\mathbf{c}, \mu) = \mathbf{c}$

Algorithm 32 Hybrid 2

$\mathbf{c} \leftarrow \mathbb{B}_\kappa^n$
$\mathbf{z} \leftarrow \mathcal{D}_\sigma^m$
With probability $\frac{1}{M}$:
    output $(\mathbf{z}, \mathbf{c})$
    program $H(\mathbf{Az} + q\mathbf{c}, \mu) = \mathbf{c}$

Lemma 8.3 *Let $\mathcal{D}$ be a distinguisher which can query the random oracle H and either the actual signing algorithm (Algorithm 29) or Hybrid 2 (Algorithm 32). If she makes h queries to H and s queries to the signing algorithm that she has access to, then for all but a $1 - e^{\Omega(n)}$ fraction of all possible matrices $\mathcal{A}$, her advantage of distinguishing the actual signing algorithm from the one in Hybrid 2 is at most $s(s+h)2^{-n+1}$.*

Proof: First, we show that the distinguisher $\mathcal{D}$ has advantage at most $s(s+h)2^{-n+1}$ of distinguishing between the real signature scheme and Hybrid 1 (Algorithm 31). The only difference between these algorithms is that, in Hybrid 1, the output of the random oracle is chosen at random from $\mathbb{B}_\kappa^n$ and then programmed as the answer to $H(\mathbf{Az}+q\mathbf{c}, \mu) = H(\mathbf{Ay}, \mu)$ without checking whether the value of $(\mathbf{Ay}, \mu)$ was already set. Now, each time Hybrid 1 is called, the probability of generating a $\mathbf{y}$ such that $\mathbf{Ay}$ is equal to one of the previous values that was queried is at most 2^{-n+1}. Indeed, let us notice that at most $s+h$ values of $(\mathbf{Ay}, \mu)$ will ever be set. With probability at least $1 - e^{\Omega(n)}$, the matrix $\mathbf{A}$ can be written in Hermite Normal Form as $\mathbf{A} = [\bar{\mathbf{A}}\|\mathbf{I}]$. Finally, for any $\mathbf{t} \in \mathbb{Z}_{2q}^n$, since $\sigma \geq 3/\sqrt{2\pi}$, we have

$$\Pr[\mathbf{Ay} = \mathbf{t}; \mathbf{y} \leftarrow \mathcal{D}_\sigma^m] = \Pr[\mathbf{y_1} = (\mathbf{t} - \bar{\mathbf{A}}\mathbf{y}_0); \mathbf{y} \leftarrow \mathcal{D}_\sigma^m] \leq \max_{\mathbf{t}' \in \mathbb{Z}_{2q}^n} \Pr[\mathbf{y_1} = \mathbf{t}'; \mathbf{y_1} \leftarrow \mathcal{D}_\sigma^n] \leq 2^{-n} .$$

Thus if Hybrid 1 is accessed s times, and the probability of getting a collision each time is at most $(s+h)2^{-n+1}$, the probability that a collision occurs after s queries is at most $s(s+h)2^{-n+1}$.

We next emphasize that the outputs of Hybrid 1 and Hybrid 2 follows exactly the same distribution. Indeed, the proof of this fact is a direct consequence of Lemma 2.7 : Hybrid 1 exactly plays the role of the algorithm $\mathcal{A}$ and Hybrid 2 corresponds to $\mathcal{F}$, where $M = \exp(1/(2\alpha^2))$ and

$$f(\mathbf{z}) = \exp\big(-\|\mathbf{z}\|^2/(2\sigma^2)\big), \quad g_{\mathbf{c}}(\mathbf{z}) = \exp\big(-\|\mathbf{z}\|^2/(2\sigma^2)\big)\exp\big(-\|\mathbf{Sc}\|^2/(2\sigma^2)\big)\cosh\big(\langle \mathbf{z}, \mathbf{Sc}\rangle/\sigma^2\big) .$$

By Lemma 2.7 the outputs of Hybrid 1 and Hybrid 2 follows exactly the same distribution (since we have $M \cdot g_{\mathbf{c}} \geq f$ for all $\mathbf{v}$). □

Lemma 8.4 *Suppose there exists a polynomial-time algorithm $\mathcal{F}$ which makes at most s queries to the signer in Hybrid 2, h queries to the random oracle H, and succeeds in forging with probability δ. Then there exists an algorithm of the same time-complexity as $\mathcal{F}$ that for a given $\mathbf{B} \leftarrow \mathcal{K}$ finds a non-zero $\mathbf{v} \in \mathbb{Z}_q^m$ such that $\|\mathbf{v}\| \leq 2B_2$ and $\mathbf{Bv} = 0$ with probability at least $\approx \delta^2/(2(s+h))$.*

Proof: Let $\mathbf{B} \leftarrow \mathcal{K}$ the matrix for the generalized SIS instance we want to solve. We transform slightly $\mathbf{B}$ to create a public key $\mathbf{A}$ as in the signature scheme and publish it as the public key $\mathbf{A} \in \mathbb{Z}_{2q}^{n \times m}$; notice that this modification is such that $\mathbf{A} \bmod q = 2\mathbf{B}$ for our key generation procedures. Therefore finding

a vector $\mathbf{v}$ such that $\mathbf{Av} = \mathbf{0} \bmod q$ yields $\mathbf{Bv} = \mathbf{0} \bmod q$ because 2 is invertible modulo q.[6] Denote by $t = s + h$ the bound on the number of times the random oracle H is called or programmed during $\mathcal{F}$'s attack.

First, we pick random coins ϕ and ψ respectively for the forger and the signer. We also pick the values that will correspond to the responses of the random oracle $\mathbf{c}_1, \ldots, \mathbf{c}_t \xleftarrow{\$} \mathbb{B}_\kappa^n$. We now consider a subroutine $\mathcal{A}$ taking as input $(\mathbf{A}, \phi, \psi, \mathbf{c}_1, \ldots, \mathbf{c}_t)$.

The first step of the subroutine is to initialize $\mathcal{F}$ by giving it the public-key $\mathbf{A}$ and the random coins ϕ. Then, it proceeds to run $\mathcal{F}$. Whenever $\mathcal{F}$ wants some message signed, $\mathcal{A}$ runs the signing algorithm of Hybrid 2 using the signer random coins ψ to produce a signature. During signing or when $\mathcal{F}$ will make queries to the random oracle, the random oracle H will have to be programmed, and the response of H will be the first $\mathbf{c}_i$ in the list $(\mathbf{c}_1, \ldots, \mathbf{c}_t)$ that has not been used yet. (Of course, $\mathcal{A}$ keeps a table of all queries to H, so in case the same query is made twice, the previously answered $\mathbf{c}_i$ will be replied.) When $\mathcal{F}$ finishes running and outputs a forgery (with probability δ), our subroutine $\mathcal{A}$ simply outputs $\mathcal{F}$'s output $\{(\mathbf{z}, \mathbf{c}), \mu\}$.

Recall that the output of $\mathcal{A}$ verifies $\|\mathbf{z}\|_\infty < q/4$ and $\|\mathbf{z}\| \leq B_2$ and $\mathbf{c} = H(\mathbf{Az} + q\mathbf{c}, \mu)$. Notice that if the random oracle H was not queried or programmed on some input $\mathbf{w} = \mathbf{Az} + q\mathbf{c}$, then $\mathcal{F}$ has only a $1/|\mathbb{B}_\kappa^n|$ chance of producing a $\mathbf{c}$ such that $\mathbf{c} = H(\mathbf{w}, \mu)$. Thus with probability $1 - 1/|\mathbb{B}_\kappa^n|$, $\mathbf{c}$ must be one of the $\mathbf{c}_i$'s, and so the probability that $\mathcal{F}$ succeeds in a forgery. Thus the probability that $\mathbf{c} = \mathbf{c}_j$ for some j is $\delta - 1/|\mathbb{B}_\kappa^n|$.

Type 1 Forgery Suppose that $\mathbf{c}_j$ was a response to a signing query made by $\mathcal{F}$ on $(\mathbf{w}', \mu') = (\mathbf{Az}' + q\mathbf{c}_j, \mu')$. Then we would have

$$H(\mathbf{Az} + q\mathbf{c}_j, \mu) = H(\mathbf{Az}' + q\mathbf{c}_j, \mu').$$

If $\mu \neq \mu'$ or $\mathbf{Az} + q\mathbf{c}_j \neq \mathbf{Az}' + q\mathbf{c}_j$, it means that $\mathcal{F}$ found a pre-image of $\mathbf{c}_j$. Therefore with overwhelming probability, we have $\mu = \mu'$ and $\mathbf{Az} + q\mathbf{c}_j = \mathbf{Az}' + q\mathbf{c}_j$. This yields $\mathbf{A}(\mathbf{z} - \mathbf{z}') = \mathbf{0} \bmod 2q$. We know that $\mathbf{z} \neq \mathbf{z}'$ (otherwise the signatures are the same). Moreover, since $\|\mathbf{z}\|_\infty, \|\mathbf{z}'\|_\infty \leq q/4$, we have $\mathbf{z} - \mathbf{z}' \neq 0 \bmod q$. Finally, the condition on the ℓ_2-norm of $\mathbf{z}$ and $\mathbf{z}'$ gives $\|\mathbf{z} - \mathbf{z}'\| \leq 2B$.

Type 2 Forgery Assume now that $\mathbf{c}_j$ was a response to a random oracle query made by $\mathcal{F}$. In this case we record this signature $(\mathbf{z}, \mathbf{c}_j)$ on the message μ, and we generate fresh random elements $\mathbf{c'}_j, \ldots, \mathbf{c'}_t \xleftarrow{\$} \mathbb{B}_\kappa^n$. By the General Forking Lemma of Bellare and Neven [BN06], we obtain that the probability that $\mathbf{c'}_j \neq \mathbf{c}_j$ and the forger uses the random oracle response $\mathbf{c'}_j$ (and the query associated to it) in the forgery is at least

$$\left(\delta - \frac{1}{|\mathbb{B}_\kappa^n|}\right) \cdot \left(\frac{\delta - 1/|\mathbb{B}_\kappa^n|}{t} - \frac{1}{|\mathbb{B}_\kappa^n|}\right).$$

Thus, with the above probability, $\mathcal{F}$ outputs a signature $(\mathbf{z}', \mathbf{c}'_\mathbf{j})$ of the message μ and $\mathbf{Az} + q\mathbf{c}_j = \mathbf{Az}' + q\mathbf{c}'_\mathbf{j}$. We finally obtain

$$\mathbf{A}\big(\mathbf{z} - \mathbf{z}'\big) = q\big(\mathbf{c}_j - \mathbf{c'}_j\big) \bmod 2q\,.$$

Since $\mathbf{c}_j - \mathbf{c'}_j \neq \mathbf{0} \bmod 2$, we have $\mathbf{z} - \mathbf{z}' \neq \mathbf{0} \bmod 2q$. Moreover, we have $\|\mathbf{z} - \mathbf{z}'\|_\infty < q/2$: this implies that $\mathbf{v} = \mathbf{z} - \mathbf{z}' \neq \mathbf{0} \bmod q$. Finally, we have

$$\mathbf{Av} = \mathbf{0} \bmod q \qquad \text{and} \qquad \|\mathbf{v}\| \leq 2B_2\,,$$

that is $\mathbf{v}$ is a solution to a $\mathsf{SIS}^\infty_{q,n,m,\beta}$ with $\beta = 2B_2$. □

8.4 Practical Instantiation of BLISS

In this section, we present a practical instantiation of our signature scheme zhich is inspired by the NTRU key-generation. We present optimizations and discuss implementation issues for each step of the signing algorithm (Algorithm 29). The signature scheme was implemented as a proof of concept on a desktop computer. Parameters proposals and timings are provided in Section 8.5.

6. More precisely, for the SIS-based generation detailed in Section 8.7, denoting $\mathbf{B} = (\mathbf{B}_1 | -\mathbf{B}_2) \in \mathbb{Z}_q^{n \times (m-n)} \times \mathbb{Z}_q^{n \times n}$, define $\mathbf{A} = (2\mathbf{B}_1 | q\mathbf{I}_n - 2\mathbf{B}_2)$. The matrix $\mathbf{A}$ follows the same distribution that in the key generation of Section 8.7 because of the use of the leftover hash lemma, and $\mathbf{A} \bmod q = 2\mathbf{B}$. For the "NTRU-like" key generation used to instantiate our scheme in Section 8.4, we get from the NTRU SIS assumption a matrix $\mathbf{B} = (\mathbf{B}_1 | -\mathbf{B}_2) = (\mathbf{a} | -1)$ where $\mathbf{a} = (2\mathbf{g}+1)/\mathbf{f}$ and we define $\mathbf{A} = (2\mathbf{B}_1 | q\mathbf{1} - 2\mathbf{B}_2) = (2\mathbf{a} | q - 2)$ that is a public key with the same distribution that in Section 8.4. Moreover we get $\mathbf{A} \bmod q = 2\mathbf{B} = (2\mathbf{a} | -2)$.

8.4.1 Key-Generation

Given densities δ_1 and δ_2, we generate random polynomials $\mathbf{f}$ and $\mathbf{g}$ with $d_1 = \lceil \delta_1 n \rceil$ coefficients in $\{\pm 1\}$, $d_2 = \lceil \delta_2 n \rceil$ coefficients in $\{\pm 2\}$ and all other coefficients to 0 until $\mathbf{f}$ is invertible.[7] The secret key is given by $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2)^t = (\mathbf{f}, 2\mathbf{g}+1)^t$.

The public key is then computed as follows : set $\mathbf{a}_q = (2\mathbf{g}+1)/\mathbf{f} \in \mathcal{R}_q$ ($\mathbf{a}_q$ is defined as a quotient *modulo* q). Next, define $\mathbf{A} = (2\mathbf{a_q}, q-2) \in \mathcal{R}_{2q}^{1\times 2}$. One easily verifies that :

$$\begin{aligned} \mathbf{AS} &= 2\mathbf{a}_q \cdot \mathbf{f} - 2(2\mathbf{g}+1) = 0 && \bmod q \\ \mathbf{AS} &= q(2\mathbf{g}+1) = q \cdot 1 = 1 && \bmod 2\ , \end{aligned}$$

that is $\mathbf{AS} = q \bmod 2q$. Finally, $(\mathbf{A}, \mathbf{S})$ is a valid key pair for our scheme.

Denote by $\mathcal{K}_{n,\delta_1,\delta_2}$ the distribution that picks small $\mathbf{f}$ and $\mathbf{g}$ as uniform polynomials with exactly d_1 entries in $\{\pm 1\}$ and d_2 entries in $\{\pm 2\}$ and outputs the public key $\mathbf{B} = (\mathbf{a}, 1) \in \mathcal{R}_q^{1\times 2}$ for $\mathbf{a} = (2\mathbf{g}+1)/\mathbf{f}$.

The public key generated above $\mathbf{A}$ taken modulo q follows the distribution $2\mathcal{K}_{n,\delta_1,\delta_2}$; that is, such key-pair generation algorithm gives a scheme based on $\mathcal{R}\text{-SIS}_{q,1,2,\beta}^{\mathcal{K}_{n,\delta_1,\delta_2}}$.

Rejection According to $N_\kappa(\mathbf{S})$ In practice after generating $\mathbf{S}$, we restart when $N_\kappa(\mathbf{S}) \geq C^2 \cdot 5 \cdot (d_1 + 4d_2) \cdot \kappa$ for a fixed constant C. This constant is chosen so that 25% of the keys are accepted, decreasing the overall security by at most 2 bits.

Computation of $N_\kappa(\mathbf{S})$ Recall that

$$N_\kappa(\mathbf{S}) = \max_{\substack{I \subset \{1,\dots,n\} \\ \#I = \kappa}} \sum_{i \in I} \left(\max_{\substack{J \subset \{1,\dots,n\} \\ \#J = \kappa}} \sum_{j \in J} T_{i,j} \right) \quad \text{where } \mathbf{T} = \mathbf{S}^t \cdot \mathbf{S} \in \mathbb{R}^{n \times n}\ .$$

However, in order to obtain $N_\kappa(\mathbf{S})$, it is not required to compute the $2 \cdot \binom{n}{\kappa}$ sums of the definition. Indeed, if suffices to compute $\mathbf{T} = \mathbf{S}^t \cdot \mathbf{S}$, to sort the columns of $\mathbf{T}$, sum the κ larger values in each line, sort the resulting vector and to sum its κ larger components. Notice moreover that working in $\mathbb{Z}_{2q}[x]/(x^n+1)$ implies that $\mathbf{S}$ is composed of rotations (possibly with opposed coefficients) of $\mathbf{s}_i$'s, and this (ideal) structure is thus also present in $\mathbf{T}$. Thus it suffices to compute the vector

$$\mathbf{t} = \left(\langle \mathbf{s}_1, \mathbf{s}_1 \rangle + \langle \mathbf{s}_2, \mathbf{s}_2 \rangle, \langle \mathbf{s}_1, \mathbf{x} \cdot \mathbf{s}_1 \rangle + \langle \mathbf{s}_2, \mathbf{x} \cdot \mathbf{s}_2 \rangle, \dots, \langle \mathbf{s}_1, \mathbf{x}^{n-1} \cdot \mathbf{s}_1 \rangle + \langle \mathbf{s}_2, \mathbf{x}^{n-1} \cdot \mathbf{s}_2 \rangle \right)\ ,$$

and derive $\mathbf{T} = (\mathbf{t}, \mathbf{x} \cdot \mathbf{t}, \dots, \mathbf{x}^{n-1} \cdot \mathbf{t})$.

Theoretical Bound We provide below a (theoretical) asymptotic bound on $N_\kappa(\mathbf{S})$ for completeness. The following proposition easily generalizes to the form of our secret keys (see Corollary 8.6).

Lemma 8.5 *For a fixed density $\delta \in (0,1)$, and $w = \lceil \delta n \rceil$, let $\mathbf{s} \in \mathbb{Z}[x]/(x^n+1)$ be chosen uniformly in $\mathbb{T}_w^n$, and $\mathbf{S} \in \mathbb{Z}^{n\times n}$ denotes its matrix representation. Then, for any $\epsilon > 0$, we have :*

$$N_\kappa(\mathbf{S}) \leq w\kappa + \kappa^2 \mathcal{O}\left(w^{1/2+\epsilon}\right)$$

except with negligible probability.

Proof: The first term $w\kappa$ arises from the diagonal coefficients of $\mathbf{T} = \mathbf{S}^t \cdot \mathbf{S}$, equals to $\|\mathbf{s}\|^2 = w$. It remains to bound the non-diagonal terms of T. For $i \neq j$,

$$Y_{i,j} = \sum_{1 \leq k \leq n} \varepsilon_{i,j,k} \cdot s_{i+k} \cdot s_{j+k}\ ,$$

where $\varepsilon_{i,j,k} \in \{\pm 1\}$ are some fixed coefficients, and the indices are taken modulo n. The key argument is to split this sum in two parts, so that each part contains only independent terms. This is possible when

7. In order to get a better entropy/length ratio, we include a few entries in $\{\pm 2\}$ in the secret key, increasing resistance to the Hybrid attack.

$i - j \neq 0$ and n is a power of 2 : one easily checks that there exists a set $K \subset \mathbb{Z}_n$ such that $K + i$ and $K + j$ form a partition of $\mathbb{Z}_n$. Thus, we rewrite

$$Y_{i,j} = \sigma_{i,j} + \bar{\sigma}_{i,j} \quad \text{where } \sigma_{i,j} = \sum_{k \in K} \varepsilon_{i,j,k} \cdot s_{i+k} \cdot s_{j+k} \text{ and } \bar{\sigma}_{i,j} = \sum_{k \in \mathbb{Z}_n \setminus K} \varepsilon_{i,j,k} \cdot s_{i+k} \cdot s_{j+k} \ .$$

Focusing on the sum $\sigma_{i,j}$ (a similar argument holds for $\bar{\sigma}_{i,j}$), one can restrict the sum to its non-zero terms and notice that the remaining terms are uniformly random in $\{-1, 1\}$ and independent from each other. Finally $\sigma_{i,j}$ is the sum of at most w uniform variables over $\{-1, 1\}$ and therefore $\sigma_{i,j} \leq w^{1/2+\epsilon}$ except with negligible probability. [8] □

Corollary 8.6 *Let $\mathbf{f}, \mathbf{g} \in \mathbb{Z}[x]/(x^n + 1)$ be chosen uniformly in $\mathbb{T}_w^n$, $\mathbf{F}, \mathbf{G} \in \mathbb{Z}^{n \times n}$ be their matrix representations, and set $\mathbf{S}^t = (\mathbf{F} | 2\mathbf{G} + \mathbf{Id}_n) \in \mathbb{Z}^{n \times 2n}$. Then,*

$$N_\kappa(\mathbf{S}) \leq (5w + 1)\kappa + \kappa^2 \mathcal{O}\left(w^{1/2+\epsilon}\right) \ .$$

Proof: This follows easily from the fact that $\mathbf{S}^t \cdot \mathbf{S} = \mathbf{F}^T \cdot \mathbf{F} + 4\mathbf{G}^T \cdot \mathbf{G} + \mathbf{G} + \mathbf{G}^T + \mathbf{Id}_n$, yielding $N_\kappa(\mathbf{S}) \leq N_\kappa(\mathbf{F}) + 4N_\kappa(\mathbf{G}) + 2\kappa^2 + \kappa$. □

8.4.2 Gaussian Sampling

In Line 1 of Algorithm 29, we want to produce $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2)^t$ where $\mathbf{y}_1, \mathbf{y}_2$ are polynomials over $\mathbb{Z}_{2q}[x]/(x^n + 1)$ with coefficients distributed according to a centered discrete Gaussian distribution of standard deviation σ. In Section 7.1, we provide a new technique to perform *efficiently* discrete Gaussian sampling on constrained devices. However on an environment with enough memory, using the cumulative distribution table algorithm is the easiest and fastest solution (see Table 7.1). Namely, we tabulate the approximate cumulative distribution of the desired distribution, *i.e.* the probabilities $p_z = \Pr[x \leq z : x \leftarrow \mathcal{D}_\sigma]$ for $z \in [-\tau\sigma, \tau\sigma]$, precomputed with λ bits of precision. At sampling time, one generates $y \in [0, 1)$ uniformly at random, then performs a binary search through the table to locate some $z \in \mathbb{Z}$ such that $y \in [p_{z-1}, p_z)$ and outputs z. A GCC-profiling of our program reveals that this step takes about 35% of the entire running-time, including the entropy generation using `sha-512`.

8.4.3 Multiplication of Two Polynomials

In Line 2 of Algorithm 29, the element $\mathbf{Ay} = \mathbf{a}_1 \cdot \mathbf{y}_1 + \mathbf{a}_2 \cdot \mathbf{y}_2 \in \mathbb{Z}_{2q}[x]/(x^n + 1)$ is given as input to the random oracle. Since $\mathbf{a}_2 = q - 2$ is a constant, $\mathbf{a}_2 \cdot \mathbf{y}_2$ is straightforward to obtain. It remains to (efficiently) compute the product of $\mathbf{a}_1$ by $\mathbf{y}_1$ over $\mathbb{Z}_{2q}[x]/(x^n + 1)$.

Because of the particular shape of $\mathbf{a}_1$ in the NTRU-like key generation, namely that $\mathbf{a}_1$ is lifted from $\mathbb{Z}_q[x]/(x^n + 1)$ to $\mathbb{Z}_{2q}[x]/(x^n + 1)$ by multiplying its coefficients by 2 (i.e. $\mathbf{a}_1 = 2 \cdot \mathbf{a'}_1$), computing $\mathbf{a_1} \cdot \mathbf{y}_1$ over $\mathbb{Z}_{2q}[x]/(x^n + 1)$ can be done by computing the product $\mathbf{a'}_1 \cdot \mathbf{y}_1$ over $\mathbb{Z}_q[x]/(x^n + 1)$ and then multiplying the coefficients of the result by 2. Now multiplying two polynomials of $\mathbb{Z}_q[x]/(x^n + 1)$ for q prime is made efficient by choosing a modulus q such that $q = 1 \bmod 2n$: there exists then a primitive $2n$-th root ω of unity modulo q. Finally, the multiplication can be done in complexity $\mathcal{O}(n \log n)$ via Number Theoretic Transform (i.e. Fast Fourier Transform over a finite field). Details on these standard techniques can be found for example in [PG12, Ber]. Notice that one does not need to work with vectors of size $2n$ as the component-wise multiplication of the NTT representations of size n of $\mathbf{a'}_1(\omega x)$ and $\mathbf{y}_1(\omega x)$ gives the NTT representation of $[\mathbf{a'}_1 \cdot \mathbf{y}_1](\omega x) \in \mathbb{Z}_q[x]/(x^n + 1)$.

8.4.4 Hashing to $\mathbb{B}_\kappa^n$

We discuss how to build a hash function outputting uniform vectors in $\mathbb{B}_\kappa^n$ from a standard hash function H (used in Line 2 of Algorithm 29). In [Lyu12], it was suggested to use a Hash function with $\kappa + \log_2 \binom{n}{\kappa}$ bits of output (recall that $\#\mathbb{B}_\kappa^n = \binom{n}{\kappa}$, and thus $\#\mathbb{T}_\kappa^n = 2^\kappa \binom{n}{\kappa}$), and then apply a one-to-one map to $\mathbb{T}_\kappa^n$. Such a mapping can be found in [FS96] but its complexity is quadratic in n; this is quite inefficient especially for large parameters. To avoid this costly algorithm, the authors of [GLP12] used an efficient procedure injectively mapping 160-bit strings to $\mathbb{T}_{32}^{512}$; they increased the value κ from 20 to 32 to gain efficiency, yielding a larger signature size.

8. By Hoeffding bound for example, or classical properties of random walks.

Overview We here give an alternative solution that is both efficient and optimal (i.e. κ is minimal for a target entropy) to produce random elements in $\mathbb{B}_\kappa^n$. In a few words, our approach consists of obtaining $\kappa' > \kappa$ values $x_1 \dots x_{\kappa'}$ in $\mathbb{Z}_n$, and setting the coordinates $\mathbf{c}_{x_i}$ of the challenge $\mathbf{c}$ to 1, starting from $i = 1$, and until $\|\mathbf{c}\|_1 = \kappa$. If some coordinate $\mathbf{c}_{x_j}$ is already set to 1 one just ignore this x_j, and if one runs out of values x_j, we would restart the process using a different seed. In the following we describe more precisely this algorithm and show it indeed produces a uniform random function over $\mathbb{B}_\kappa^n$ if H is indeed a uniform random function over $\mathbb{Z}_n^k$.

Detailed Construction and Correctness Let n be a power of 2 and $H_0 \colon \{0,1\}^* \to \mathbb{Z}_n^{\kappa'}$ with $\kappa' > \kappa$ be a random function outputting $\kappa' \log_2 n$ bits (parsed as κ' elements in $\mathbb{Z}_n$). We consider the set $S \subset \mathbb{Z}_n^{\kappa'}$ of vectors that have at least κ different entries. The probability that a uniform element in $\mathbb{Z}_n^{\kappa'}$ lies in S is :

$$A = 1 - \frac{|\mathbb{Z}_n^{\kappa'} \setminus S|}{|\mathbb{Z}_n^{\kappa'}|}.$$

When A is not negligible, one can efficiently build a random function $\widetilde{H} \colon \{0,1\}^* \to S$ as $\widetilde{H}(x) = H(x|i)$, where i is the smallest index such that $H(x|i) \in S$. This is somehow a rejection sampling technique applied to a random function. Finally, in average, one call of $\widetilde{H}$ requires $1/A$ calls to H.

Claim 8.7 *With the notation above,* $|\mathbb{Z}_n^{\kappa'} \setminus S| \leq \binom{n}{\kappa-1}(\kappa-1)^{\kappa'}$.

Proof: Notice that $\mathbb{Z}_n^{\kappa'} \setminus S$ is the set of vectors over $\mathbb{Z}_n$ of length κ' with at most $\kappa - 1$ distinct coordinates. To obtain this set, one may first choose a subset $K \subset \mathbb{Z}_n$ of size $\kappa - 1$ ($\binom{n}{\kappa-1}$ choices), and then chooses the κ' coordinates in K ($(\kappa-1)^{\kappa'}$ choices). Note that vectors with strictly less than $\kappa - 1$ coordinates have been counted several times. More formally :

$$\mathbb{Z}_n^{\kappa'} \setminus S = \bigcup_{\substack{K \subset \mathbb{Z}_n \\ |K| < \kappa-1}} K^{\kappa'} = \bigcup_{\substack{K \subset \mathbb{Z}_n \\ |K| = \kappa-1}} K^{\kappa'} .$$

□

For our parameters, this gives $1/A \leq 1.00001$ using a 512-bit hash function for H (*e.g.* BLISS-IV : $n = 2^9$, $\kappa' = 56$, $\kappa = 39$).

It remains to map the domain S to $\mathbb{B}_\kappa^n$. For $\mathbf{x} \in S$, let I be the set of the κ first distinct coordinates values of $\mathbf{x}$, and set $f(\mathbf{x}) = \sum_{i \in I} \mathbf{e}_i \in \mathbb{B}_\kappa^n$ where $\mathbf{e}_1, \dots, \mathbf{e}_n$ are the canonical vectors of $\mathbb{Z}^n$. Each image $\mathbf{y} \in \mathbb{B}_\kappa^n$ has the same amount of f-preimages in S, therefore $\widehat{H} \colon \{0,1\}^* \to \mathbb{B}_\kappa^n$ defined as $\widehat{H}(x) = f \circ \widetilde{H}(x)$ is also a random function.

8.4.5 Multiplication of S by a Sparse Vector c

In Line 4 of Algorithm 29, one should compute $\mathbf{Sc}$. Let $\mathbf{S}_i$, $i = 1, 2$ denotes the $n \times n$ matrix over $\mathbb{Z}_{2q}$ whose columns vectors are the $x^j \cdot \mathbf{s}_i$'s for $j = 0, \dots n-1$ In particular we have that $\mathbf{s}_i \cdot \mathbf{c} = \mathbf{S}_i \mathbf{c}$.

Now, since $\mathbf{c}$ is a sparse binary vector, one should not use the NTT to compute $\mathbf{s}_i \cdot \mathbf{c}$ for this step (contrary to Section 8.4.3). Indeed, the absolute value of the coefficients of $\mathbf{s}_1$ and $\mathbf{s}_2$ is smaller than 5, yielding $\|\mathbf{s}_i \cdot \mathbf{c}\|_\infty \leq 5\kappa \ll 2q$, $i = 1, 2$. Therefore, computing $(\mathbf{s}_1 \cdot \mathbf{c})$ and $(\mathbf{s}_2 \cdot \mathbf{c})$ can be performed very efficiently by additions over $\mathbb{Z}$ (i.e. without reduction modulo $2q$) of κ pre-stored columns of $\mathbf{S}_i$. Notice moreover that working over $\mathbb{Z}_{2q}[x]/(x^n + 1)$ allows to reduce the memory storage overhead to zero : all the columns of $\mathbf{S}_i$ are rotations (possibly with opposite coefficients) of $\mathbf{s}_i$.

8.4.6 Rejection Sampling according to $1/\exp$ and $1/\cosh$

In Line 5 of Algorithm 29, one should reject with probability $1/(M \exp(-x/f) \cosh(x'/f))$. To avoid floating-point computations of the transcendental functions exp and cosh, we use the techniques described in Section 7.1 to do it efficiently with a very small memory footprint. Notice that the precomputed values can be the same both for Gaussian sampling and this rejection sampling step.

8.4.7 Signature Compression

Recall that the signature is a pair $(\mathbf{z}, \mathbf{c})$ where $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2)^t$ follows the Gaussian distribution D_σ^{2n}.

Working with A in Hermite Normal Form.

As in [GLP12], in order to compress our signature, we need to have $\mathbf{A}$ in Hermite Normal Form. Now, during the key-generation process, we explicitly constructed $\mathbf{A} = (\mathbf{a}_1, q-2)$ such that

$$\mathbf{a}_1 \cdot \mathbf{s}_1 + (q-2)\mathbf{s}_2 = q \bmod 2q \ .$$

Let us define ζ such that $\zeta \cdot (q-2) = 1 \bmod 2q$. Next, instead of calling the random oracle on $(\mathbf{Ay} \bmod 2q, \mu)$, we call it on $\big((\zeta\mathbf{A})\mathbf{y} \bmod 2q, \mu\big)$ because $\zeta\mathbf{A} = (\zeta\mathbf{a}_1, 1)$ is in Hermite Normal Form.

Dropping Low-weight Bits of $\mathbf{z}_2$.

We denote by d the number of bits we would like to drop in $\mathbf{z}_2$. For every integer x in the range $[-q, q)$ and any positive integer d, x can be uniquely written

$$x = \lfloor x \rceil_d \cdot 2^d + [x \bmod 2^d] \ ,$$

where $[x \bmod 2^d] \in [-2^{d-1}, 2^{d-1})$. Thus $\lfloor x \rceil_d$ can be viewed as the "high-order bits" of x and $[x \bmod 2^d]$ as its "low-order bits".

In [GLP12], the signature size is reduced by dropping almost all the information about $\mathbf{z}_2$ in the signature. Such a strategy impacts on security, as it reduces to an easier SIS problem (it may allow an attacker to forge using longer vectors). Let us describe a similar feature for our signature scheme. First, we replace the random oracle H input by

$$\begin{aligned} \big(\lfloor(\zeta\mathbf{A})\mathbf{y}\rceil_d, \mu\big) &= \big(\lfloor\zeta \cdot \mathbf{a}_1 \cdot \mathbf{y}_1 + \mathbf{y}_2 \bmod 2q\rceil_d, \mu\big) \\ &= \big(\lfloor\zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c} + \mathbf{z}_2 \bmod 2q\rceil_d, \mu\big). \end{aligned} \tag{8.4}$$

The idea of [GLP12], transposed to our settings, was to define a vector $\widetilde{\mathbf{z}_2}$ with coefficients in $\{0, \pm 2^d\}$ and a limited number of coefficients $\widetilde{\mathbf{z}_2}[i] = \mathbf{z}_2[i]$ (coming from the need of reduction modulo $2q$ after the addition with small but non negligible probability) such that

$$\lfloor(\zeta\mathbf{A})\mathbf{y}\rceil_d = \lfloor\zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c} + \widetilde{\mathbf{z}_2} \bmod 2q\rceil_d \ .$$

Unfortunately the workaround which consists in storing some coefficients uncompressed, *i.e.* of the form $\widetilde{\mathbf{z}_2}[i] = \mathbf{z}_2[i]$, yields a signature scheme which is not strongly unforgeable. Indeed it is easy to forge a signature by modifying the least-significant bit of one of the uncompressed values, and this does not modify the high-order bits of the sum with very high probability. [9]

Let us describe how to solve this issue for our signature scheme. We want to replace $\mathbf{z}_2$ by a small vector $\mathbf{z}_2^\dagger$ such that

$$\lfloor(\zeta\mathbf{A})\mathbf{y}\rceil_d = \lfloor\zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c} \bmod 2q\rceil_d + \mathbf{z}_2^\dagger \ .$$

Unfortunately without additional modification, the security proof does not go through because of a similar issue as in [GLP12], *i.e.* the coefficients $\mathbf{z}_2[i]$ of $\mathbf{z}_2$ which, added to $\big(\zeta \cdot (\mathbf{a}_1 \cdot \mathbf{z}_1)[i] + \zeta \cdot q \cdot \mathbf{c}[i]\big)$, force us to reduce the result modulo $2q$ in Equation (8.4). Let us define $p = \lfloor 2q/2^d \rfloor$; we have $2q = p \cdot 2^d + \nu$ with a small ν (typically $\nu = 1$ in our parameters). Now we modify the random oracle H input by

$$\big(\lfloor(\zeta\mathbf{A})\mathbf{y}\rceil_d \bmod p, \mu\big) \ ,$$

and define

$$\mathbf{z}_2^\dagger = \big(\lfloor(\zeta\mathbf{A})\mathbf{y}\rceil_d - \lfloor\zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c} \bmod 2q\rceil_d \bmod p\big) \in [0, p)^n \ .$$

The coefficients of $\mathbf{z}_2^\dagger$ are small modulo p. We redefine the signature to be $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$ instead of $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$, and during the verification, we check that

$$H\left(\mathbf{z}_2^\dagger + \lfloor\zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c} \bmod 2q\rceil_d \bmod p, \mu\right) = \mathbf{c} \ ,$$

that $\left\|(\mathbf{z}_1 \| 2^d\mathbf{z}_2^\dagger)\right\| \leq B_2$ and that $\left\|(\mathbf{z}_1 \| 2^d\mathbf{z}_2^\dagger)\right\| \leq B_\infty$.

Finally, we have the following Theorem :

9. As a direct consequence, the scheme of [GLP12] is not strongly unforgeable.

Theorem 8.8 *Let us consider the signature scheme of Section 8.4.8. Assume that* $d \geq 3$, $q \equiv 1 \bmod 2^{d-1}$, *and* $2B_\infty + (2^d + 1) < q/2$. *Suppose there is a polynomial-time algorithm* $\mathcal{F}$ *which succeeds in forging with non negligible probability. Then there exists a polynomial-time algorithm which can solve the* $\mathcal{R} - \mathsf{SIS}^{\mathcal{K}}_{q,n,m,\beta}$ *problem for* $\beta = 2B_2 + (2^d + 1)\sqrt{n}$.

The differences with the proof of Theorem 8.2 are detailed in Section 8.8.

Compressing Most Significant Bits of $\mathbf{z}_1$ and $\mathbf{z}_2^\dagger$.

The simplest representation of the entries of $\mathbf{z}_1$ then requires $\lceil \log_2(8\sigma) \rceil \leq \log_2(16\sigma)$ bits. Yet, the entropy of these entries is actually smaller :

Fact 8.9 *Let* X *be distributed as* D_σ, *that is a centered discrete Gaussian variable. Then the entropy of* X *is upper-bounded by :*

$$\mathcal{H}(X) \leq \frac{1}{\sigma^3} + \log_2(\sqrt{2\pi e}\sigma) \approx \log_2(4.1\sigma) \ .$$

Now, Huffman coding provides (almost) optimal encoding for data when their distribution is exactly known. More precisely :

Theorem 8.10 (Huffman Coding) *For any random variable* X *over a finite support* S, *there exist an injective prefix-free code* $C\colon S \to \{0,1\}^*$ *such that :*

$$\mathcal{H}(X) \leq E\left[|C(X)|\right] < \mathcal{H}(X) + 1 \ .$$

To keep the compression efficient, we choose to only encode the highest bits of all entries; the lower are almost uniform and therefore we do not loose anything by not compressing them. Moreover, if by packing several independent variables $X_1, \ldots, X_k$, we can decrease the overhead to $1/k$.

8.4.8 Final KeyGen, Sign and Verify Algorithms

In this section, we describe the final algorithms to instantiate BLISS with the parameters of Section 8.5. Notice that to obtain the signature size indicated Table 8.1 (page 94), one need to use Huffman Coding to compress the highest bits of $\mathbf{z}_1$ and $\mathbf{z}_2^\dagger$. Let us define $p = \lfloor 2q/2^d \rfloor$ where d is the number of dropped bits.

Algorithm 33 BLISS Key Generation

Output: Key pair $(\mathbf{A}, \mathbf{S})$ such that $\mathbf{AS} = q \bmod 2q$

1: Choose $\mathbf{f}, \mathbf{g}$ as uniform polynomials with exactly d_1 entries in $\{\pm 1\}$ and d_2 entries in $\{\pm 2\}$
2: $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2)^t \leftarrow (\mathbf{f}, 2\mathbf{g} + 1)^t$
3: **if** $N_\kappa(\mathbf{S}) \geq C^2 \cdot 5 \cdot (\lceil \delta_1 n \rceil + 4\lceil \delta_2 n \rceil) \cdot \kappa$ **then**
4: **restart**
5: **end if**
6: $\mathbf{a}_q = (2\mathbf{g} + 1)/\mathbf{f} \bmod q$ (**restart** if $\mathbf{f}$ is not invertible)
7: **Output**$(\mathbf{A}, \mathbf{S})$ where $\mathbf{A} = (2\mathbf{a}_q, q - 2) \bmod 2q$

Algorithm 34 BLISS Signature Algorithm

Input: Message μ, public key $\mathbf{A} = (\mathbf{a}_1, q - 2) \in \mathcal{R}_{2q}^{1\times 2}$, secret key $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2)^t \in \mathcal{R}_{2q}^{2\times 1}$

Output: A signature $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$ of the message μ

1: $\mathbf{y}_1, \mathbf{y}_2 \leftarrow D_{\mathbb{Z}^n,\sigma}$
2: $\mathbf{u} = \zeta \cdot \mathbf{a}_1 \cdot \mathbf{y}_1 + \mathbf{y}_2 \bmod 2q$
3: $\mathbf{c} \leftarrow H(\lfloor \mathbf{u} \rceil_d \bmod p, \mu)$
4: Choose a random bit b
5: $\mathbf{z}_1 \leftarrow \mathbf{y}_1 + (-1)^b \mathbf{s}_1 \mathbf{c}$
6: $\mathbf{z}_2 \leftarrow \mathbf{y}_2 + (-1)^b \mathbf{s}_2 \mathbf{c}$
7: **Continue** with probability $1 \Big/ \left(M \exp\left(-\frac{\|\mathbf{Sc}\|^2}{2\sigma^2} \right) \cosh\left(\frac{\langle \mathbf{z}, \mathbf{Sc} \rangle}{\sigma^2} \right) \right)$ otherwise **restart**
8: $\mathbf{z}_2^\dagger \leftarrow (\lfloor \mathbf{u} \rceil_d - \lfloor \mathbf{u} - \mathbf{z}_2 \rceil_d) \bmod p$
9: **Output** $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$

TABLE 8.2 – Parameters proposals

Name of the scheme	**BLISS-0**	**BLISS-I**	**BLISS-II**	**BLISS-III**	**BLISS-IV**
Security	Toy ($\leq$ 60 bits)	128 bits	128 bits	160 bits	192 bits
Optimized for	Fun	Speed	Size	Security	Security
Ring Dim. n	256	512	512	512	512
Lattice Dim. $m = 2n$	512	1024	1024	1024	1024
Modulus q	7681	12289	12289	12289	12289
Secret key densities δ_1, δ_2	.55 , .15	.3 , 0	.3 , 0	.42 , .03	.45, .06
Gaussian std. dev. σ	100	215	107	250	271
Max Shift/std. dev. ratio α	.5	1	.5	.7	.55
Weight of the challenge κ	12	23	23	30	39
Secret key N_κ-Threshold C	1.5	1.62	1.62	1.75	1.88
Dropped bits d in $\mathbf{z}_2$	5	10	10	9	8
Verif. thresholds B_2, B_∞	2492, 530	12872, 2100	11074, 1563	10206,1760	9901, 1613
Repetition rate	7.4	1.6	7.4	2.8	5.2
Entropy of challenge $\mathbf{c} \in \mathbb{B}_\kappa^n$	66 bits	132 bits	132 bits	161 bits	195 bits
Signature size	**3.3kb**	**5.6kb**	**5kb**	**6kb**	**6.5kb**
Secret key size	1.5kb	2kb	2kb	3kb	3kb
Public key size	3.3kb	7kb	7kb	7kb	7kb
SIS parameter $\beta/\sqrt{q}$ (as in Theorem8.8)	63= 1.0083^m	441= 1.0060^m	409= 1.0059^m	289= 1.0055^m	231= 1.0053^m
Ring-Unique-SVP parameter $\sqrt{\frac{qm}{2\pi e}}/\lambda_1$	14= 1.0051^m	46= 1.0037^m	46= 1.0037^m	30= 1.0033^m	25= 1.0031^m

Algorithm 35 BLISS Verification Algorithm

Input: Message μ, public key $\mathbf{A} = (\mathbf{a}_1, q-2) \in \mathcal{R}_{2q}^{1\times 2}$, signature $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$
Output: Accept or Reject the signature
1: **if** $\|(\mathbf{z}_1 | 2^d \cdot \mathbf{z}_2^\dagger)\|_2 > B_2$ **then** Reject
2: **if** $\|(\mathbf{z}_1 | 2^d \cdot \mathbf{z}_2^\dagger)\|_\infty > B_\infty$ **then** Reject
3: Accept iff $\mathbf{c} = H\big(\lfloor \zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c} \rceil_d + \mathbf{z}_2^\dagger \bmod p, \mu\big)$

8.5 Parameters and Benchmarks

In this section, we first propose parameters sets for the scheme BLISS described in Section 8.4. Next, we compare the benchmarks of our proof-of-concept implementations with the `openssl` running times of RSA and ECDSA.

8.5.1 Parameters Sets

In Table 8.2, we propose several sets of parameters to implement the $\mathcal{R}$-SIS$^{\mathcal{K}}$ variant of our scheme described in Section 8.4. The signature schemes BLISS-I and BLISS-II are respectively optimized for speed and compactness and offer 128 bits of security (i.e. long-term protection [NIS,ECR]). The signature schemes BLISS-III and BLISS-IV offer respectively 160 and 192 bits of security. The two last lines provides typical measurement security against direct lattice attack in term of Hermite factor, but slightly better attacks exists. Therefore, our security claims are derived from an extensive analysis based on BKZ-2.0 simulation [CN11] in interaction with other techniques [MR09, MM11, HG07] detailed in Section 8.6.

One of the objectives of this work was to determine whether the scheme from [Lyu12] could be improved so as it remains sufficiently secure for a dimension $n = 256$. Even though this seems possible when only considering *direct* lattice attacks, it turns out to be slightly out of reach according to the analysis of Section 8.6. Any additional trick might unlock an extremely efficient 80-bit secure signature scheme ; it seems to us a challenging but worthwhile goal. We do however propose a toy variant BLISS-0 in this dimension for which we expect up to 60 bits of security. Yet, we believe it would require a significant effort to break this toy variant ; we leave it as a challenge to motivate advance in lattice cryptanalysis. Notice that choosing a non power-of-two dimension n would have been possible but yields several unwelcome consequences : on efficiency first as NTT becomes at least twice slower and the geometry is worst (our constant C grows), but also on simplicity as one will no longer work as on the

simple quotient by $x^n + 1$. However, it is possible to get about 100 bits of security in dimension $n = 379$ for signatures of size 4kb. In comparison [Lyu12, **Set-IV**] and [GLP12, **Set-I**] have respective signature size of 15kb and 9.5kb, for claimed security of 100 bits. [10]

8.5.2 Timings

In Table 8.1, we provide running times of our proof-of-concept implementation of our signature scheme with the parameters provided above, on a desktop computer. We also provide running times for the `openssl` implementations of RSA and ECDSA. Notice that, despite the lack of optimization on our proof-of-concept implementation, we derived interesting timings. First, our verification time is nearly the same for each of our variants, and is much faster than the RSA and the (even worse) ECDSA verification implementations of `openssl` by a factor 10 to 30. Secondly, excluding RSA which is really slow, the signature algorithm of BLISS-I is as fast as ECDSA-256 (with the same claimed security). We refer to [NIS, ECR] to get the equivalence between the key length of RSA and ECDSA and the expected security in bits.

Besides, we expect our scheme to be much more suitable for embedded devices than both RSA and ECDSA, mainly because our operation are done with a very small modulus (less than 16 bits). By design, the binary representation of q is `11 0000 0000 0001`, that is q has a very small Hamming weight ; this structure might yield interesting hardware optimizations. The main issue for such architectures is the generation of discrete Gaussian, addressed in Section 7.1.

8.6 Security Analysis

In this section, we describe how known attacks apply to our scheme. First, we describe in Section 8.6.1 combinatorial attacks on the secret key, namely brute-force and meet-in-middle attacks.

Then we consider lattice reduction attacks. Typical measurements of lattice problem hardness (the so called Hermite factor, see [CN11]) are given in Table 8.2 (page 109), measuring how hard it is to find vectors of a given norm in a random lattice. We first apply this measure to the hardness of the underlying SIS problem, as if the lattice used was truly random (*c.f.* Section 8.6.2).

Yet, the lattice is not truly random, as by design it contains unusually short vectors. Therefore, on may try to directly recover the secret by lattice reduction : find the secret key $(\mathbf{f}, \mathbf{g})$ as a short vector in the primal lattice $L = \{(\mathbf{x}, \mathbf{y}) \in \mathcal{R}^2 : \mathbf{a}_q\mathbf{x} + \mathbf{y} = 0 \bmod q\}$. Unfortunately, the only study [GN08] of the behavior of lattice algorithms in the presence of unusually short vectors only consider the unique-SVP problem, in which there is only one unusually short vector. In the NTRU-like case, there is a basis of n of them. We provide new experiments showing that the behavior is similar ; that is, it is dictated by the ratio between the actual shortest vector, its expected length in a random lattice and the Hermite factor (*c.f.* Section 8.6.3).

An alternative attack to recover unusually short vectors of a lattice is to use short (but quite larger) dual lattice $\hat{L}$ vectors to detect its presence, and then recover it [MR09, MM11] using search-to-decision reduction ; quantification of this attack is detailed in Section 8.6.4.

Finally, it is possible to combine lattice reduction and combinatorial techniques : Howgrave-Graham designed in [HG07] an attack against NTRU keys combining a meet-in-the-middle strategy with lattice reduction. This attack applies to our scheme, as detailed in Section 8.6.5, but also on the previous related schemes [Lyu12, GLP12]. Notice that there is no mention of this attack in the security analysis of the latter schemes ; therefore in order to compare, we also include security measurements for those schemes.

We base our security projection on the BKZ 2.0 simulation methodology introduced [CN11] that models the behavior of BKZ including the latest improvements [GNR10, HPS11].

Note that we only sketch the attack principles ; the interested readers should refer to the original articles [HNHGSW03, HG07, CN11, MR09, MM11] for more details. We emphasize that the statistical attacks [NR09, DN12b] provably (*i.e.* information-theoretically) do not apply here because of rejection sampling : the output distribution of the signature scheme is independent of the secret key.

8.6.1 Brute-force and Meet-in-the-Middle Key Recovery Attack

The key-recovery problem is as follows : given $\mathbf{a} \in \mathbb{Z}_q[x]/(x^n + 1)$, find small polynomials $\mathbf{f}, \mathbf{g}$ such that $\mathbf{a}(2\mathbf{g} + 1) - \mathbf{f} = \mathbf{0}$ (knowing that such a solution exists). Precisely, we know that both $\mathbf{f}$ and $\mathbf{g}$ have respectively $d_1 = \lceil n\delta_1 \rceil$ entries ± 1 and $d_2 = \lceil n\delta_2 \rceil$ entries ± 2.

10. Our analysis in Section 8.6 casts doubts on the security claims of [GLP12, **Set-I**].

TABLE 8.3 – Hardness of the underlying SIS instance

Scheme	**BLISS-0**	**BLISS-I**	**BLISS-II**	**BLISS-III**	**BLISS-IV**
SIS parameter $\beta/\sqrt{q}$ (as in Theorem 8.8)	$63 = 1.0083^m$	$441 = 1.0060^m$	$409 = 1.0059^m$	$289 = 1.0055^m$	$231 = 1.0053^m$
Block Size Required	125	215	220	245	260
Enum. Cost $\log_2 T$	**53**	**130**	**136**	**168**	**188**

Brute-force Key Recovery The brute-force attack simply consists in picking a random vector $\mathbf{g}$ according to the key-generation distribution, and checking whether $\mathbf{f} = \mathbf{a}(2\mathbf{g}+1)$ is a small polynomial. To measure the complexity of this attack, one simply measures the entropy of $\mathbf{g}$: this entropy yields a lower bound on the time to exhaust all the possible values. The time complexity of this attack is therefore $T = 2^{d_1+d_2}\binom{n}{d_1}\binom{n-d_1}{d_2}$.

For more complex attacks, it may be simpler to model all the entries of the secret key as independent random variables, each of them having entropy :

$$e = \delta_0 \log_2 \delta_0 + \delta_1 \log_2 \frac{\delta_1}{2} + \delta_2 \log_2 \frac{\delta_2}{2}.$$

In this model, the total entropy is $n \cdot e$, which is at most $\log n$ greater than the true entropy.

Meet-in-the-Middle Attack Odlyzko proposed a MiM attack of running time the square root of the latter attack (but with additional memory consumption). It was designed against the NTRU signature scheme, but it also applies here. We refer to [HNHGSW03] for details, and give only a short explanation of a simplified version : exhaust $\mathbf{g}_1$ as the first half bits of $\mathbf{g}$ and store $\mathbf{g}_1$ in several labeled boxes (of an hash table) according to the values of $\mathbf{f}_1 = \mathbf{a}(2\mathbf{g}_1 + 1)$. Then search for the second half $\mathbf{g}_2$ of $\mathbf{g}$ by computing $\mathbf{f}_2 = \mathbf{a}(2\mathbf{g}_2) \bmod q$: the labeling is designed so that to ensure a collision whenever $\mathbf{f}_1 + \mathbf{f}_2$ is ternary.

This attacks runs in time and memory about $2^{n \cdot e/2}$, since the entropy of a half of the vector is $n \cdot e/2$.

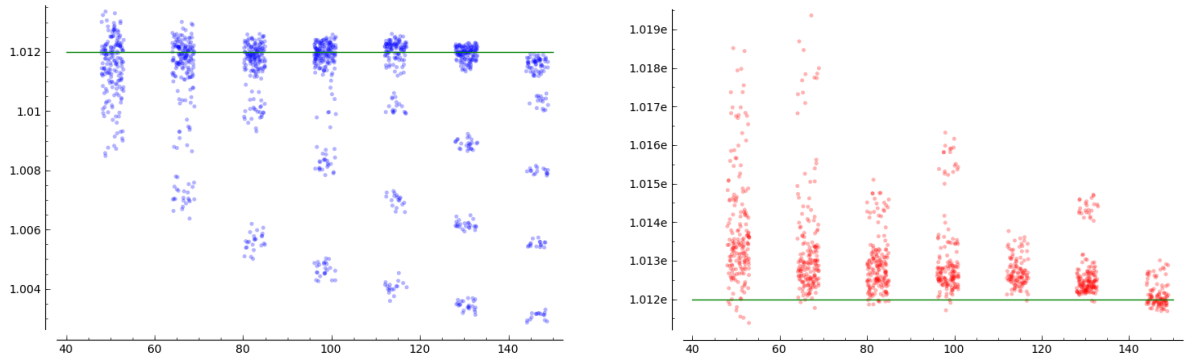
8.6.2 Hardness of the underlying SIS problem

Attack Overview In this section we measure the hardness of forging a signature according to our security proof. We will consider the running time necessary to BKZ algorithm to find a vector of norm $\beta = 2B_2 + (2^d + 1)\sqrt{n}$ in a random q-ary lattice according to latest analysis [CN11]. While the lattice L is not perfectly random because of the presence of unusually short vectors, the next section analyzes how hard it is to detect and find those unusually short vectors.

Remark Note that we have $\beta > q$, yet the q-vectors are not proper solution to the SIS instance since it is required that the short solution is non-null modulo q. This is one of the reason our scheme includes not only constraint on the ℓ_2 but also on the ℓ_∞ norms of signature vectors; this ensures that the reduction provides a vector $\mathbf{v}$ such that $\|\mathbf{v}\|_\infty < q/2$, and thus is non-null modulo q. While we could have chosen larger values for B_∞ and still have a valid security reduction, choosing it as small as possible for correctness can only make the scheme more secure.

Quantification The hardness of this SIS problem is dictated by the ratio $\beta/\sqrt{q}$ and the dimension m, precisely it is necessary to run BKZ with a blocksize providing a Hermite factor $\delta^m < \beta/\sqrt{q}$. The relation between δ the block-size and the running time is interpolated from [CN11].

Margins The cost given in the last line of Table 8.3 is given in number of nodes to visit in the enumeration tree of the enumeration subroutine of BKZ. Each visit requires about 100 CPU cycles, and BKZ needs to perform at least $2n$ such enumerations, adding an additional 10 bits to those numbers. Yet, those numbers does not directly give rise to an attack as they are derived from security reduction; actually forging seems to require finding vectors smaller by a factor 2.



(a) Shortest vector not found

(b) Shortest vector found

FIGURE 8.3 – Results BKZ-20 for $n \in [48, 150]$, $q \in [6000, 25000]$ and binary search on the λ_1-threshold. On horizontal axis is the value of $n + \texttt{random(0,5)}$ and on vertical axis is $\left(\frac{1}{.40}\sqrt{\frac{qm}{2\pi e}}/\lambda_1\right)^{1/2n}$

TABLE 8.4 – Cost of finding the Ring-unique shortest vector via primal lattice reduction

Scheme	**BLISS-0**	**BLISS-I**	**BLISS-II**	**BLISS-III**	**BLISS-IV**
Ring-Unique-SVP parameter $\sqrt{\frac{qm}{2\pi e}}/\lambda_1$	$14 = 1.0051^m$	$46 = 1.0037^m$	$46 = 1.0037^m$	$30 = 1.0033^m$	$25 = 1.0031^m$
Block Size Required	270	> 300	> 300	> 300	> 300
Enum. Cost $\log_2 T$	**200**	$>$ **240**	$>$ **240**	$>$ **240**	$>$ **240**

8.6.3 Primal Lattice Reduction Key Recovery

Attack Overview The attack consists of applying lattice reduction to the primal lattice L hoping that the short vector found will be the secret key. This problem can be seen as a ring variant of the unique-SVP problem.

Quantification The only study of the behavior of BKZ in the presence of short vector is due to Gama and Nguyen [GN08] ; while the theoretical bounds suggest that the shortest vector will be found when λ_2/λ_1 is greater than δ^{2m}, their experiments show that it is in fact sufficient to have a gap of δ^m to actually find the shortest vector. In practice, for BKZ-20, the shortest vector was found when $\lambda_2/\lambda_1 > .48 \cdot 1.01^m$.

We ran similar experiment of BKZ-20 in the case of cyclic lattices, therefore such that $\lambda_1 = \ldots = \lambda_n$. We found that the shortest vector was indeed found when $\sqrt{qm/2\pi e}/\lambda_1$ was greater than $.40 \cdot 1.012^m$ (see Figure 8.3). This is consistent with the results of [GN08] since $\sqrt{qm/2\pi e}$ is the expected length of the shortest vector according to the Gaussian heuristic, and we would also expect $\lambda_2 \approx \sqrt{qm/2\pi e}$ in a random q-ary lattice. Additionally, when this condition was not verified, the resulting shortest vector had length about $\sqrt{q} \cdot 1.012^m$, that is BKZ behaved as if the lattice was truly random ; in other terms, BKZ seems equally efficient at finding unusually short vectors than at detecting their presence. In particular, this justifies the measurements of the previous section.

Therefore it seems reasonable to assume such behavior is similar for larger block sizes, and to measure hardness according to the BKZ 2.0 methodology [CN11].

8.6.4 Dual Lattice Reduction Key Recovery

Attack Overview The attack consists in using short dual lattice vectors as distinguisher for the existence of a very short vector $\mathbf{s}$ in a lattice [MR09] . Then, one may use the distinguisher to completely recover this very short vector using Micciancio and Mol reduction [MM11], inspired by the Goldreich-Levin Theorem [GL89].

TABLE 8.5 – Dual Lattice Reduction Attack Parameters

Scheme	**BLISS-0**	**BLISS-I**	**BLISS-II**	**BLISS-III**	**BLISS-IV**	[Lyu12, **IV**]	[GLP12, **I**]
Best Block Size b	110	220	220	240	245	190	130
Enum. Cost : $\log_2 T$	45	136	136	162	168	103	56
Hermite Factor : δ	1.0088^m	1.0059^m	1.0059^m	1.0056^m	1.0056^m	1.0067^m	1.0081^m
Dist. Advantage : $\log_2 \epsilon$	-5.5	-20	-20	-19	-21	-7	-5
Total Cost : $\log_2(T/\epsilon^2)$	**56**	**177**	**177**	**201**	**211**	**118**	**67**

Quantification For a q-ary lattice L of dimension m, using a vector $\mathbf{v} \in \hat{L}$ (where $\hat{L}$ is the dual lattice) and assuming its direction is random, one is able to distinguish the existence of an unusual short vector $\mathbf{s}$ in the dual with probability $\epsilon = e^{-\pi\tau^2}$, where $\tau = \|\mathbf{v}\| \cdot \|\mathbf{s}\| / (q\sqrt{m})$.

Next, using this distinguisher as an oracle, it is possible to recover one entry of the private key except with small fixed probability, using $1/\epsilon^2$ calls to that oracle. We then iterated over different block-sizes (5 by 5) to minimize the total cost T/ϵ^2, where T is the running time of the enumeration subroutine of BKZ. Our estimations of the cost of the attacks are given in table 8.5

Remark Rather than trying to find the proper secret key $\mathbf{s} = (\mathbf{f}| - 2\mathbf{g} + 1)$ as a short solution to $(2\mathbf{a}_q, 2)^t\mathbf{s} = \mathbf{0} \bmod q$, one would search directly $\mathbf{s}' = (\mathbf{f}|\mathbf{g})$ as a shorter solution to $(\mathbf{a}_q, -1)^t\mathbf{s}' = 1 \bmod q$.

Margins To stay on the safe side, we do not include the additional n^2 factor to the running time of this attack : indeed there is n coordinates to guess, and each BKZ reduction requires at least n enumerations ; one might then be tempted to claim an additional 20 bits of security. Yet it is unclear whether one needs to run the full BKZ reduction to get new short vectors, neither if one can reuse the same short dual vector to guess each coordinate. Even though we do not claim an attack in time 2^{67} on [GLP12], we believe that claiming more than 90 bits of security is a long shot. The difference between our measurement and theirs might be explained by the fact that the authors only considered the case where ϵ was close to 1.

8.6.5 Hybrid MiM-Lattice Key Recovery

Attack Overview The attack from [HG07] uses lattice reduction as a preprocessing step, in order to decrease the search space of combinatorial attacks. Precisely, one first chooses parameters r and R, and applies lattice reduction on the sub-lattice generated by the vectors of the sub-basis $\mathbf{b}_r, \ldots, \mathbf{b}_{R-1}$ (see Figure 8.4), in order to run the MiM attack only over the $2n - R$ last coordinates.

In order to perform the combinatorial attack, one needs to obtain a basis whose last orthogonalized vector is large enough. Precisely, the basis needs to be good enough so that Babai's algorithm properly solves BDD on the error $\mathbf{s}' = (s_1, \ldots, s_R, 0, \ldots, 0)$. A necessary condition is therefore :

$$\langle \mathbf{s}', \mathbf{b}_i^\star \rangle / \|\mathbf{b}_i^\star\|^2 \leq 1/2 \,, \tag{8.5}$$

where the $\mathbf{b}_1^*, \ldots, \mathbf{b}_R^*$ is the Gram-Schmidt orthogonalization of $\mathbf{b}_1, \ldots, \mathbf{b}_R$.

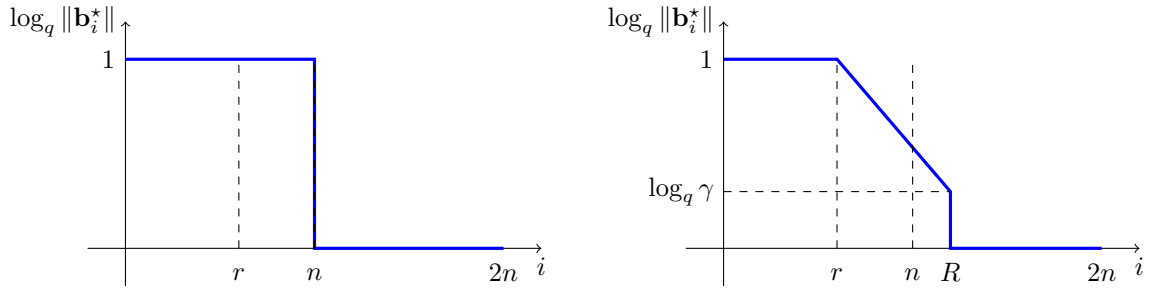
Quantification Once again, we assume that the lattice reduction algorithm provides a basis of random direction. Therefore, we model the quantity $\langle \mathbf{s}', \mathbf{b}_i^\star \rangle / \|\mathbf{b}_i^\star\|^2$ as a Gaussian of standard deviation $\|\mathbf{s}'\| / (\sqrt{R}\,\|\mathbf{b}_i^\star\|)$. Denoting $\gamma = \left\|\mathbf{b}_{R-1}^\star\right\|$, one models by the GSA (geometric series assumption) that $\left\|\mathbf{b}_{R-1-i}^\star\right\| = \gamma \times \delta^{2i}$, where $\delta \leq 1.007$ is the Hermite factor. To verify Equation (8.5) with reasonable probability (say at least 0.01), it is required that $\gamma \geq 2.5\,\|\mathbf{s}'\| / \sqrt{R}$.

We thus determine the security against this attack as follows : to claim λ bits of security, set R so that it takes 2^λ time and memory to exhaust the last $2n - R$ entries of the secret. Recall that e denotes the entropy of a single entry, each step of the Meet in the Middle attack requires $O(n^2)$ operations, and at least $e \cdot R$ bits of storage, therefore we set R such that $R \cdot e = 2\lambda - \log_2(e \cdot R) - \log_2(n^2)$.

Then we determine γ, and run BKZ 2.0 simulation according to [CN11], increasing block-size until $\gamma \geq 2.5\,\|\mathbf{s}'\| / \sqrt{R}$. Finally, deduce the cost of lattice reduction and verify it is greater than 2^λ. Note that r is derived from the behavior of this simulation. Analysis results are described in Table 8.6.

Margins There is a small security margin coming from the fact that we set the parameters so that the attack succeeds with probability 0.01, which would add about 7 bits of security, and again 10 extra bits because BKZ requires at least $2n$ enumeration. More importantly we considered that the attacker has

FIGURE 8.4 – Basis Profile during the Hybrid Attack



(a) Before reduction (b) After reduction

TABLE 8.6 – Hybrid MiM+Lattice Reduction Attack Parameters

Scheme	**BLISS-0**	**BLISS-I**	**BLISS-II**	**BLISS-III**	**BLISS-IV**	[Lyu12, IV]	[GLP12, I]
MiM Search Cost $\log_2 M$	**60**	**128**	**128**	**160**	**192**	**100**	**80**
Entropy per Secret Key Entry	2.11	1.18	1.18	1.60	1.77	1.58	1.58
MiM Search Dimension R	46	194	194	183	201	110	85
Block Size Required	165	245	245	> 300	> 300	220	140
BKZ Enum. Cost $\log_2 T$	**84**	**168**	**168**	$>$ **200**	$>$ **200**	**150**	**60**

2^λ memory available; in practice it is unlikely that an attacker may have as much memory available as number of bit-operations. [11]

8.7 Key Generation for a SIS-Based Scheme

In this section, we explain how to generate the key pair $(\mathbf{A}, \mathbf{S})$ so that

$$\mathbf{AS} = q\mathbf{Id}_n \in \mathbb{Z}_{2q}^{n\times n} ,$$

where the distribution of $\mathbf{A}$ is statistically close to the uniform distribution, in order to obtain a general SIS-based variant of our scheme.

From the Leftover Hash Lemma 2.6, one can deduce the following Lemma for finite linear combinations modulo the prime q :

Lemma 8.11 *Let $m \geq 2$. Set $x_1, \ldots, x_m \leftarrow \mathbb{Z}_q^n$ uniformly and independently, set $s_1, \ldots, s_m \leftarrow (-2^\alpha, 2^\alpha) \cap \mathbb{Z}$, and set $y = \sum_{i=1}^m s_i \cdot x_i \bmod q$. Then $(x_1, \ldots, x_m, y)$ is $1/2\sqrt{q^n/2^{(\alpha+1)\cdot m}}$-uniform over $\mathbb{Z}_q^{n\cdot(m+1)}$.*

Proof: Let us consider the hash function family $\mathcal{H}$ from $(-2^\alpha, 2^\alpha)^m$ to $\mathbb{Z}_q$ in which each member $h \in \mathcal{H}$ is parameterized by the element $(x_1, \ldots, x_m) \in \mathbb{Z}_q^m$. Given $\mathbf{s} \in (-2^\alpha, 2^\alpha)^m$, we define $h(\mathbf{s}) = \sum_{i=1}^m s_i \cdot x_i \in \mathbb{Z}_q$. The hash function family is clearly pairwise independent since q is prime. Therefore by Lemma 2.6, $(h, h(x))$ is $1/2\sqrt{q^n/2^{(\alpha+1)\cdot m}}$-uniform over $\mathbb{Z}_q^{m+1}$. □

SIS-based Scheme.

Define $m' = m + n$. Choose a uniform matrix $\mathbf{A'}_q \in \mathbb{Z}_q^{n\times m}$ and a random small $\mathbf{S'} \in \mathbb{Z}_q^{m\times n}$ with coefficients in $(-2^\alpha, 2^\alpha)$. Define $\mathbf{A}_q = (\mathbf{A'}_q | -\mathbf{A'}_q\mathbf{S'}) \in \mathbb{Z}_q^{n\times m'}$. By Lemma 8.11, the statistical distance between the distribution of $\mathbf{A}_q$ and the uniform distribution over $\mathbb{Z}_q^{n\times m'}$ is at most $n \cdot 1/2\sqrt{q^n/2^{(\alpha+1)\cdot m}}$. Thus, for this statistical distance to be negligible in the security parameter λ, we need

$$m \geq \frac{2(\lambda - 1 + \lceil \log_2(n) \rceil) + n\lceil \log_2(q) \rceil}{\alpha + 1} .$$

11. In 2007, there were no more than 2^{71} bits of storage globally, while all general-purpose computers could execute 2^{87} operations in a year. Storage growth is 23% a year versus 58% for computing power (see `http://news.usc.edu/#!/article/29360/How-Much-Information-Is-There-in-the-World`). There are about 2^{160} atoms on earth.

Set the secret key as $\mathbf{S} = \begin{pmatrix} \mathbf{S}' \\ \mathbf{Id}_n \end{pmatrix} \in \mathbb{Z}_{2q}^{m' \times n}$. One observes that $\mathbf{A}_q \mathbf{S} = \mathbf{0} \bmod q$. It remains to set the public key as $\mathbf{A} = (2\mathbf{A'}_q | q\mathbf{Id}_n - 2\mathbf{A'}_q \mathbf{S'}) \in \mathbb{Z}_{2q}^{n \times m'}$. Then one easily checks that $\mathbf{AS} = q\mathbf{Id}_n$. Also, we have that $\mathbf{A} \bmod q = 2\mathbf{A}_q$ is *uniform* modulo q. Notice that this construction is easily adaptable to the ring settings.

8.8 Security Proof with Dropped Bits

Recall from Section 8.4.7 that a signature is a tuple $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$ with

$$\mathbf{z}_2^\dagger = \lfloor \zeta \cdot \mathbf{a}_1 \cdot \mathbf{y}_1 + \mathbf{y}_2 \rceil_d - \lfloor \zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c} \rceil_d \bmod p \, ,$$

where $p = \lfloor 2q/2^d \rfloor$ and that the random oracle is called on

$$\left(\lfloor \zeta \cdot \mathbf{a}_1 \cdot \mathbf{y}_1 + \mathbf{y}_2 \rceil_d \bmod p, \mu \right) = \left(\mathbf{z}_2^\dagger + \lfloor \zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c} \rceil_d \bmod p, \mu \right) \, .$$

We recall the theorem stating that our scheme is secure when dropping bits.

Theorem 8.12 (Restatement of Theorem 8.8) *Let us consider the signature scheme of Section 8.4.8. Assume that $d \geq 3$, $q \equiv 1 \bmod 2^{d-1}$, and $2B_\infty + (2^d + 1) < q/2$. Suppose there is a polynomial-time algorithm $\mathcal{F}$ which succeeds in forging with non negligible probability. Then there exists a polynomial-time algorithm which can solve the $\mathcal{R}\text{-SIS}^{\mathcal{K}}_{q,n,m,\beta}$ problem for $\beta = 2B_2 + (2^d + 1)\sqrt{n}$.*

The proof of this theorem follows the same blueprint than the proof of Theorem 8.2. Namely, by a straightforward adaptation of Lemma 8.3, one can show that our signing algorithm can be replaced by Hybrid 3 (Algorithm 36). Next, an adaptation of Lemma 8.4 states that if an algorithm can produce a forgery with non-negligible probability when the signing algorithm is replaced by Hybrid 3, then we can use it to recover a vector $\mathbf{v} \neq \mathbf{0} \bmod q$ such that $\|\mathbf{v}\| \leq \beta = 2B_2 + (2^d + 1)\sqrt{n}$ and $\mathbf{Av} = \mathbf{0} \bmod q$.

Algorithm 36 Hybrid 3

1: $\mathbf{c} \leftarrow \mathbb{B}_\kappa^n$
2: $\mathbf{z}_1, \mathbf{z}_2 \leftarrow \mathcal{D}_\sigma^n$
3: With probability $\frac{1}{M}$:
4: $\quad \mathbf{z}_2^\dagger \leftarrow (\lfloor \zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c} + \mathbf{z}_2 \rceil_d - \lfloor \zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c} \rceil_d) \bmod p$
5: $\quad$ output $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$
6: $\quad$ program $H(\lfloor \zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c} + \mathbf{z}_2 \rceil_d \bmod p, \mu) = \mathbf{c}$

Throughout the rest of the section, we focus on the modifications in the proof of Lemma 8.4 to deal with the dropping bits, i.e. we assume that $\mathcal{F}$ succeeds in forging the signature by outputting $(\mathbf{z}_1, \mathbf{z}_2^\dagger, \mathbf{c})$ where $\mathbf{c} = \mathbf{c}_j \in \{\mathbf{c}_1, \ldots, \mathbf{c}_t\}$ was obtained from either a previous signing query, or a previous random oracle query.

We start with the following two facts :

Fact 8.13 *Let $d \geq 2$, q be an integer such that $q \equiv 1 \bmod 2^{d-1}$, and let $p = \lfloor 2q/2^d \rfloor$. Then $p \cdot 2^d = 2q - 2$.*

Fact 8.14 *Let q be an odd integer and define $\zeta \in [0, 2q-1]$ such that $\zeta \cdot (q-2) = 1 \bmod 2q$. Then $\zeta = \frac{q-1}{2}$ if $(q-1)/2$ is odd or $\zeta = \frac{q-1}{2} + q$ if $(q-1)/2$ is even.*

Proof: We have that

$$\frac{q-1}{2} \cdot (q-2) = q \cdot \frac{q-1}{2} - q + 1 = 1 \bmod q \, .$$

Therefore $\zeta = \frac{q-1}{2} \bmod q$ and the fact holds according to the parity of $(q-1)/2$. □

Proof: Assume the challenger has a signature $(\mathbf{z'}_1, \mathbf{z'}_2^\dagger, \mathbf{c'}_j)$ such that

$$\lfloor \zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c}_j \rceil_d + \mathbf{z}_2^\dagger \bmod p = \lfloor \zeta \cdot \mathbf{a}_1 \cdot \mathbf{z'}_1 + \zeta \cdot q \cdot \mathbf{c'}_j \rceil_d + \mathbf{z'}_2^\dagger \bmod p \ .$$

There exists $\mathbf{k} \in \{0, \pm 1\}^n$ such that the following equation holds over $\mathbb{Z}$:

$$\lfloor \zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c}_j \rceil_d - \lfloor \zeta \cdot \mathbf{a}_1 \cdot \mathbf{z'}_1 + \zeta \cdot q \cdot \mathbf{c'}_j \rceil_d + \mathbf{z}_2^\dagger - \mathbf{z'}_2^\dagger = \mathbf{k}p \ .$$

Now we multiply the previous equation by 2^d, and this yields modulo $2q$:

$$\zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c}_j - \mathbf{e} - \zeta \cdot \mathbf{a}_1 \cdot \mathbf{z'}_1 - \zeta \cdot q \cdot \mathbf{c'}_j + \mathbf{e'} + 2^d(\mathbf{z}_2^\dagger - \mathbf{z'}_2^\dagger) = \mathbf{k} \cdot p 2^d \bmod 2q \ ,$$

where $\mathbf{e} = [\zeta \cdot \mathbf{a}_1 \cdot \mathbf{z}_1 + \zeta \cdot q \cdot \mathbf{c}_j \bmod 2q] \bmod 2^d$ and $\mathbf{e'} = [\zeta \cdot \mathbf{a}_1 \cdot \mathbf{z'}_1 + \zeta \cdot q \cdot \mathbf{c'}_j \bmod 2q] \bmod 2^d$. This yields by Fact 8.13 :

$$(\zeta \cdot \mathbf{a}_1) \cdot (\mathbf{z}_1 - \mathbf{z'}_1) + 2^d(\mathbf{z}_2^\dagger - \mathbf{z'}_2^\dagger) + \zeta \cdot q \cdot (\mathbf{c}_j - \mathbf{c'}_j) + (\mathbf{e'} - \mathbf{e}) + 2\mathbf{k} = \mathbf{0} \bmod 2q \ . \tag{8.6}$$

Thus, if we define

$$\mathbf{v} = \big(\mathbf{z}_1 - \mathbf{z'}_1, 2^d(\mathbf{z}_2^\dagger - \mathbf{z'}_2^\dagger) + (\mathbf{e'} - \mathbf{e}) + 2\mathbf{k}\big)^t \in \mathcal{R}^{2\times 1} \ ,$$

we have that

$$(\zeta \cdot \mathbf{a}_1, 1) \cdot \mathbf{v} = \mathbf{0} \bmod q \ ,$$

and thus multiplying by 2 :

$$(\mathbf{a}_1, 2) \cdot \mathbf{v} = \mathbf{0} \bmod q \ .$$

Now, we have that $\|\mathbf{v}\|_2 \leq 2B_2 + (2^d + 1) \cdot \sqrt{n}$ and $\|\mathbf{v}\|_\infty \leq 2B_\infty + (2^d + 1) < q/2$. Indeed

$$\begin{aligned}
\|\mathbf{v}\|_2 &\leq \Big\|\big(\mathbf{z}_1 - \mathbf{z'}_1, 2^d(\mathbf{z}_2^\dagger - \mathbf{z'}_2^\dagger)\big)\Big\|_2 + \big\|\big(\mathbf{0}, (\mathbf{e'} - \mathbf{e} + 2\mathbf{k})\big)\big\|_2 \\
&\leq 2B_2 + \big\|\big(\mathbf{0}, (\mathbf{e'} - \mathbf{e} + 2\mathbf{k})\big)\big\|_\infty \cdot \sqrt{n} \\
&\leq 2B_2 + (\big\|\big(\mathbf{0}, (\mathbf{e'} - \mathbf{e})\big)\big\|_\infty + 2\,\|\mathbf{k}\|_\infty) \cdot \sqrt{n} \\
&\leq 2B_2 + (2^d - 1 + 2) \cdot \sqrt{n} \\
&\leq 2B_2 + (2^d + 1) \cdot \sqrt{n}
\end{aligned}$$

Similarly for the infinite norm, we get

$$\|\mathbf{v}\|_\infty \leq 2B_\infty + (2^d + 1) < q/2.$$

It remains to show that $\mathbf{v} \neq \mathbf{0} \bmod q$ to conclude. By the condition $\|\mathbf{v}\|_\infty < q/2$, it suffices to show that $\mathbf{v} \neq \mathbf{0} \bmod 2q$.

Case #1 :[$\mathbf{z}_1 \neq \mathbf{z}'_1 \bmod 2q$]. Since

$$\mathbf{v} = \big(\mathbf{z}_1 - \mathbf{z'}_1, 2^d(\mathbf{z}_2^\dagger - \mathbf{z'}_2^\dagger) + (\mathbf{e'} - \mathbf{e}) + 2\mathbf{k}\big)^t \ ,$$

we have $\mathbf{v} \neq \mathbf{0} \bmod 2q$. This case includes both type-1 and type-2 forgeries.

Case #2 :[$\mathbf{z}_1 = \mathbf{z}'_1 \bmod 2q$ and $\mathbf{c}_j = \mathbf{c'}_j$]. In that case, we have $\mathbf{e} = \mathbf{e'}$, and for the signatures to be different we have $\mathbf{z}_2^\dagger \neq \mathbf{z'}_2^\dagger$. Therefore

$$\mathbf{v} = \big(\mathbf{0}, 2^d(\mathbf{z}_2^\dagger - \mathbf{z'}_2^\dagger) + 2\mathbf{k}\big)^t \ .$$

Now $\|2\mathbf{k}\|_\infty < 2^d$, then $\mathbf{v} \neq \mathbf{0} \bmod 2q$. This case is only possible for type-1 forgeries.

Case #3 :[$\mathbf{z}_1 = \mathbf{z}'_1 \bmod 2q$, $\mathbf{c}_j \neq \mathbf{c'}_j$ and $\mathbf{z}_2^\dagger = \mathbf{z'}_2^\dagger \bmod 2q$]. In that case, Equation (8.6) yields

$$\mathbf{e'} - \mathbf{e} + 2\mathbf{k} = \zeta \cdot q \cdot (\mathbf{c}_j - \mathbf{c'}_j) \bmod 2q.$$

Now $\mathbf{c}_j - \mathbf{c'}_j \neq \mathbf{0} \bmod 2$, therefore $\mathbf{e'} - \mathbf{e} + 2\mathbf{k} \neq \mathbf{0} \bmod 2q$. Since

$$\mathbf{v} = \big(\mathbf{0}, (\mathbf{e'} - \mathbf{e}) + 2\mathbf{k}\big)^t \ ,$$

we have $\mathbf{v} \neq \mathbf{0} \bmod 2q$. This case is only possible for type-2 forgeries.

Case #4 :[$\mathbf{z}_1 = \mathbf{z}'_1 \bmod 2q$, $\mathbf{c}_j \neq \mathbf{c'}_j$ and $\mathbf{z}_2^\dagger \neq \mathbf{z'}_2^\dagger \bmod 2q$]. In that case

$$\mathbf{v} = \left(\mathbf{0}, 2^d(\mathbf{z}_2^\dagger - \mathbf{z'}_2^\dagger) + (\mathbf{e}' - \mathbf{e}) + 2\mathbf{k}\right)^t .$$

Since $\mathbf{c}_j \neq \mathbf{c'}_j$, there exists i such that $(\mathbf{c}_j)_i \neq (\mathbf{c'}_j)_i$. Without loss of generality, we can assume that $(\mathbf{c'}_j)_i = 1$ and thus $(\mathbf{c}_j)_i = 0$. Therefore,

$$e'_i = \left(x + \zeta \cdot q \bmod 2q\right) \bmod 2^d ,$$

and

$$e_i = x \bmod 2^d ,$$

where $x = \left(\zeta \cdot (\mathbf{a}_1 \cdot \mathbf{z}_1)[i]\right) \bmod 2q$. Now $\zeta \cdot q = q \bmod 2q$ because $\zeta = 1 \bmod 2$ by Fact 8.14. Therefore

$$e'_i = \left(x + q \bmod 2q\right) \bmod 2^d .$$

Now, $\left(x + q \bmod 2q\right) = x \pm q$ over $\mathbb{Z}$. Therefore,

$$\left(e'_i - e_i\right) \bmod 2^d = \left((x \pm q) - x\right) \bmod 2^d$$

is odd. This proves that v_i is odd, and therefore that $\mathbf{v} \neq \mathbf{0} \bmod 2q$. This case is only possible for type-2 forgeries. □

Chapter 9

Conclusion

La cryptographie moderne a cela de fascinant que sa théorie et sa pratique s'opposent, s'émulent et s'inspirent mutuellement à un rythme très élevé. Que cela soit en cryptographie symétrique où asymétrique ; un théorème admis par la communauté est une base solide pour un cryptosystème, mais il laisse souvent ouvert quelques détails essentiels à sa mise en œuvre. Que faire des constantes implicites, des $\tilde{\mathcal{O}}$ et autres ω ? Comment instancier efficacement tel oracle aléatoire ? Comment extrapoler les cryptanalyses connues et évaluer la sécurité pratique d'un certain jeu de paramètres ?

Pour aller jusqu'à la pratique, la cryptographie à base de courbes elliptiques et de couplages a pu séparer la problématique en trois domaines orthogonaux ; un axe s'intéresse à la difficulté du logarithme discret (et problèmes associés) pour différentes constructions de courbes, un autre axe s'intéresse à l'implémentation efficace des opérations sur différentes courbes ; et le dernier axe s'appuie indifféremment sur l'une de ces courbes pour construire des cryptosystèmes et protocoles avancés, dont la sécurité est garantie par la courbe choisie. Si une courbe utilisée s'avère faible, ou si une nouvelle courbe s'avère plus efficace, il est possible de changer la *boîte noire*. Au final, des autorités nationales (NIST, ANSSI) et supranationales (Association des standards IEEE) émettent des recommandations et des normes, faisant le pont entre la théorie et la pratique. La rigidité de ce processus est une garantie contre des choix mal informés, des erreurs de non-spécialistes ; mais c'est une épée de Damoclès. Si jamais une telle courbe standardisée s'avérait beaucoup plus faible que prévue, si une nouvelle attaque rendait accessible avec des moyens raisonnables le déchiffrement et l'usurpation d'identité, c'est tout une infrastructure qui serait paralysée, pendant une durée non négligeable. Mon opinion sur le problème du logarithme discret est qu'il est trop *ad-hoc* : il a été conçu pour la cryptographie, et n'est étudié que par la cryptographie. Cela a été très récemment démontré par Antoine Joux, ayant trouvé un algorithme de logarithme discret pour un large ensemble de courbes avec couplage, et son attaque n'a pas été précédée de beaucoup de signaux de danger.

S'il me semble que la cryptographie à base de réseaux est plus sûre, c'est qu'elle est bien moins ad-hoc. L'algorithmique des réseaux euclidiens a sa propre raison d'être en dehors de la cryptographie car de nombreux problèmes d'optimisation peuvent s'y ramener. C'est la nature des résultats de NP- complétude : aux constantes près, les problèmes de réseaux sont au moins aussi durs que d'autres problèmes étudiés par de nombreuses branches de l'informatique pour résoudre et optimiser les réponses à des questions scientifiques, mais aussi économiques, climatiques... Chaque fois que l'on échoue à trouver un emploi du temps optimal, indirectement, on échoue à trouver le plus court vecteur d'un certain réseau euclidien. Certes, les instances utilisées pour la cryptographie ne sont pas ces instances ultimement difficiles, mais il y a un paramètre continu pour aller des unes aux autres. Enfin, les réductions de pire-cas à cas-moyen nous garantissent qu'il n'y a pas de mauvais choix possible ; pour une dimension fixée, presque tous les réseaux sont aussi durs ; contrairement aux courbes elliptiques ou des choix doivent être fait, parfois mal avisés pour des raisons de sécurités.

Néanmoins, une organisation similaire à celle de la cryptographie basée sur le logarithme discret est encore très loin d'être en place. La première tentative sérieuse de mise en œuvre de la cryptographie à base de réseaux est celle de NTRU : avec la standardisation et NTRUEncrypt, sa résistance a plus de 15 années de cryptanalyse, et son efficacité sans commune mesure avec les autres techniques de chiffrement, la cryptographie à base de réseaux est depuis longtemps un domaine prometteur, et pas seulement d'un point de vue théorique. Cependant, les attaques répétées sur les signatures à base de réseaux (NSS, puis NTRUSign) ont démontré qu'au débuts des années 2000 le domaine n'était pas suffisamment mûr pour sa mise en œuvre. Mais depuis 10 ans, le domaine a connu une poussée phénoménale, surtout dans

ses aspects théoriques. Nous avons maintenant des briques de bases, les problèmes (LWE et SIS), les bons algorithmes (Gaussian Sampling) et les techniques de preuves nous permettant de construire une large variété de primitives et de protocoles offrant une bien meilleure confiance que les outils actuels. Le chiffrement completement homomorphe a démontré la grande souplesse de ce nouvel outil ; il semble grand temps que la pratique rattrape la théorie. C'est l'un des objectifs qui a guidé cette thèse.

L'une de mes conclusions est que l'approche en *boîte noire* se transcrit assez mal pour la mise en œuvre de la cryptographie à base de réseaux. Prenons l'exemple de la mise en œuvre d'un HIBE (chiffrement basé sur l'identité hiérarchique). D'un point de vue formel, les constructions à base de réseau ressemblent à celles à base de couplages ; mais dans le détail il y a une différence essentielle pour la mise en pratique. En couplage, quelle que soit la hauteur h de la hiérarchie, le système complet se base toujours sur le même groupe, même si le nombre d'éléments des clefs et des messages augmente avec h. Pour les réseaux euclidiens, il y a une dépendance – parfois cachée dans les $\tilde{\mathcal{O}}$ – en h des paramètres de réseaux pour assurer la correction du schéma, les erreurs s'accumulant à chaque niveau de hiérarchie. De surcroît, les problèmes sur lesquels on se base ont trois, voire quatre paramètres, qui influent sur les meilleures attaques en pratique et sur l'efficacité du schéma. C'est un exercice délicat d'optimisation, d'autant qu'il y a souvent différentes attaques à considérer. De plus, pour des raisons d'efficacité, le choix de certains paramètres s'avère beaucoup plus contraint et moins continu que ce que les versions théoriques des constructions suggèrent ; c'est ainsi que, de façon surprenante, nos choix d'optimisation pour BLISS proposent la même dimension de réseau n et le même module q pour les trois niveaux de sécurité proposés. Pour optimiser nos paramètres, dans les chapitres 6 et 8 nous nous sommes appuyés sur des outils de calcul formel. Pour ce seul cryptosystème, nous n'avons pas tout automatisé ; mais l'optimisation automatisée est à mon avis la seule approche raisonnable pour choisir les paramètres en pratique de nombreuses constructions à base de réseau. Il est intéressant de noter que du côté de la cryptanalyse, le même cheminement a eu lieu ; les meilleurs algorithmes de réduction de réseaux s'appuient notamment sur une optimisation numérique des paramètres d'élagage de l'arbre de recherche (pruning). Et c'est bien la qualité première du cryptographe que de s'approprier les outils du cryptanalyste.

Ouvrir la *boîte noire* peut aussi permettre de nouvelles optimisations spécifiques à un protocole comme nous l'avons fait pour les signatures (gaussienne bimodale, utilisation de réseaux non uniformes à la NTRU). Pour mieux évaluer la sécurité pratique de ces problèmes potentiellement plus faciles – mais certainement plus efficaces pour la construction de cryptosystèmes – il serait bon de motiver plus de cryptanalyse, quitte à s'attaquer d'abord à des instances extrêmes. La communauté de cryptographie à base de réseaux a jusqu'à récemment été plutôt frileuse à l'idée de donner des paramètres concrets à leurs cryptosystèmes, et encore plus à s'appuyer sur des hypothèses potentiellement plus fortes. L'une des raisons était certainement la peur de discréditer le domaine par des choix trop agressifs de paramètres. La situation change cependant, étonnamment grâce aux nouvelles primitives inaccessibles avant les réseaux. En effet, les premières constructions de chiffrement pleinement homomorphe semblaient tellement lente asymptotiquement, que c'est pour être crédible que des paramètres pratiques ont fini par voir le jour. Mon avis est que la communauté dans son ensemble n'a plus à craindre un discrédit généralisé ; il reste cependant à convaincre que de proposer des hypothèses et des paramètres pratiques n'est pas un risque personnel majeur. J'estime cet effort nécessaire, car la cryptanalyse des réseaux euclidiens n'a certainement pas dit son dernier mot ; et tant que le domaine ne semblera pas avoir atteint son point d'équilibre, toute cette cryptographie nouvelle ne pourra être sereinement mise en œuvre.

Il faut cependant noter un effort récent d'implémentation de certaines constructions (chiffrement homomorphe, fonctions multilinéaires) et il est heureux de voir ces implémentations publiées en *open-source*. Mais au delà des primitives avancées la cryptographie à base de réseau promet aussi des solutions simples et efficaces pour des primitives plus basique telle que le chiffrement et les signatures ; en particulier pour les petites architectures. J'espère voire rapidement apparaître de telle implémentations expérimentales sur microcontrôleur ou carte-à-puce ; cela permettrai de commencer à répondre à d'autre questions essentielles pour la mise en œuvre que la sécurité théorique : la résistance aux attaques par canaux cachés (fuite d'information par consommation de courant ...), et aux attaques par injections de fautes. Les algorithmes étant fondamentalement différents de ceux utilisés auparavant en cryptographie, ces techniques d'attaques et les contremesures seront sans doute bien différentes ; l'aspect parallélisable des calculs pourrait jouer en la faveur de la cryptographie à base de réseaux contre les attaques physiques.

Un autre enseignement de cette thèse, est que les questions pratiques, d'optimisation ou d'implémentation, peuvent motiver de nouveaux problèmes théoriques. Mon exemple serait le chapitre 7, où nous découvrons qu'il est possible d'éviter de recourir à des calculs sur des nombres réels, alors que la définition même des distributions désirées inclut des nombres réels et des fonctions transcendantes. Cela nous oblige à nous appuyer sur des propriétés algébriques et des techniques stochastiques, certes

simples, mais menant à un algorithme efficace et au final assez naturel. On peut imaginer des questions bien plus fascinantes : peut-on éviter le Gaussian Sampling en se basant sur un domaine fondamental (et un algorithme associé) qui cacherait suffisamment bien la base courte utilisée contrairement aux parallélépipèdes de Babai ? Ne pourrait-on pas s'appuyer sur des réseaux d'empilement optimaux ou quasi-optimaux de sphères pour améliorer les constructions à base de réseaux ? Peut-on exploiter algorithmiquement et géométriquement le réseau de Leech Λ_{24}, ou ceux de Barnes-Wall, et leurs symétries pour la cryptographie ? Plus généralement, mon sentiment est que les aspects géométriques des réseaux ne sont pour l'instant vraiment exploités qu'en cryptanalyse, il serait sûrement bénéfique en pratique, et passionnant en théorie de voir des techniques et des résultats de sécurité prouvés s'appuyer sur des notions géométriques.

Bibliographie

[ABB10a] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 553–572, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany. xxv, 35, 59

[ABB10b] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 98–115, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Berlin, Germany. 35

[ABSS93] S. Arora, L. Babai, J. Stern, and Z. Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. In *Proceedings of the 1993 IEEE 34th Annual Foundations of Computer Science*, SFCS '93, pages 724–733, Washington, DC, USA, 1993. IEEE Computer Society. 20

[ACPS09] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 595–618, Santa Barbara, CA, USA, August 16–20, 2009. Springer, Berlin, Germany. 23

[AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *29th ACM STOC*, pages 284–293, El Paso, Texas, USA, May 4–6, 1997. ACM Press. 22

[AFV11] Shweta Agrawal, David Mandell Freeman, and Vinod Vaikuntanathan. Functional encryption for inner product predicates from learning with errors. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 21–40, Seoul, South Korea, December 4–8, 2011. Springer, Berlin, Germany. xxv, 59

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *28th ACM STOC*, pages 99–108, Philadephia, Pennsylvania, USA, May 22–24, 1996. ACM Press. xxiii, 21, 37, 98

[Ajt98] Miklós Ajtai. The shortest vector problem in l2 is np-hard for randomized reductions (extended abstract). In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, STOC '98, pages 10–19, New York, NY, USA, 1998. ACM. 20

[Ajt99] Miklós Ajtai. Generating hard instances of the short basis problem. In *ICALP*, pages 1–9, 1999. xxiii, 32, 71

[Ajt06] Miklós Ajtai. Generating random lattices according to the invariant distribution. Draft of March 2006., 2006. 46

[AKPW13] Joel Alwen, Stephan Krenn, Krzysztof Pietrzak, and Daniel Wichs. Learning with rounding, revisited : New reduction, properties and applications. Cryptology ePrint Archive, Report 2013/098, 2013. `http://eprint.iacr.org/`. 22

[AKS01] Miklós Ajtai, Ravi Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *33rd ACM STOC*, pages 601–610, Crete, Greece, July 6–8, 2001. ACM Press. 20, 25

[Ale03] Michael Alekhnovich. More on average case vs approximation complexity. In *44th FOCS*, pages 298–307, Cambridge, Massachusetts, USA, October 11–14, 2003. IEEE Computer Society Press. 22

[AP09] Joël Alwen and Chris Peikert. Generating shorter bases for hard random lattices. In *In STACS*, pages 75–86, 2009. 33, 71

[Bab86] László Babai. On lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1) :1–13, 1986. 25, 26, 38

[Ban93] W. Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296(1) :625–635, 1993. 17

[Baz01] É. Bazeries. *Les chiffres secrets dévoilés : étude historique sur les chiffres appuyée de documents inédits tirés des différents dépôts d'archives.* Charpentier et Fasquelle, 1901. ix

[Bel11] Steven M. Bellovin. Frank miller : Inventor of the one-time pad. *Cryptologia*, 35(3) :203–222, 2011. xi

[Ber] Daniel J. Bernstein. Fast multiplication and its applications. In Joe Buhler and Peter Stevenhagen, editors, *Algorithmic number theory : lattices, number fields, curves and cryptography*, pages 325–384. Cambridge University Press. 105

[BF03] Dan Boneh and Matthew K. Franklin. Identity based encryption from the Weil pairing. *SIAM Journal on Computing*, 32(3) :586–615, 2003. xv, 7

[BF11] Dan Boneh and David Mandell Freeman. Homomorphic signatures for polynomial functions. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 149–168, Tallinn, Estonia, May 15–19, 2011. Springer, Berlin, Germany. 8

[BFKL94] Avrim Blum, Merrick L. Furst, Michael J. Kearns, and Richard J. Lipton. Cryptographic primitives based on hard learning problems. In Douglas R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 278–291, Santa Barbara, CA, USA, August 22–26, 1994. Springer, Berlin, Germany. 22

[BGJT13] Razvan Barbulescu, Pierrick Gaudry, Antoine Joux, and Emmanuel Thomé. A quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. Cryptology ePrint Archive, Report 2013/400, 2013. `http://eprint.iacr.org/`. xix

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, *ITCS*, pages 309–325. ACM, 2012. 98

[Bli] H. F. Blichfeldt. A new principle in the geometry of numbers, with some applications. In *Transactions of the American Mathematical Society*, volume Vol. 15, No. 3 (Jul., 1914), pages 227–235. American Mathematical Society. 11

[BLP+13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *STOC*, pages 575–584. ACM, 2013. 22, 92

[BN06] Mihir Bellare and Gregory Neven. Multi-signatures in the plain public-key model and a general forking lemma. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 06*, pages 390–399, Alexandria, Virginia, USA, October 30 – November 3, 2006. ACM Press. 103

[Boy10] Xavier Boyen. Lattice mixing and vanishing trapdoors : A framework for fully secure short signatures and more. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010*, volume 6056 of *LNCS*, pages 499–517, Paris, France, May 26–28, 2010. Springer, Berlin, Germany. 35

[Boy13] Xavier Boyen. Attribute-based functional encryption on lattices. In *TCC*, pages 122–142, 2013. xxv, 35, 59

[BP02] Mihir Bellare and Adriana Palacio. GQ and Schnorr identification schemes : Proofs of security against impersonation under active and concurrent attacks. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 162–177, Santa Barbara, CA, USA, August 18–22, 2002. Springer, Berlin, Germany. 94

[Bri88] E. Oran Brigham. *The fast Fourier transform and its applications.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988. 19

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Rafail Ostrovsky, editor, *52nd FOCS*, pages 97–106, Palm Springs, California, USA, October 22–25, 2011. IEEE Computer Society Press. 8

[CDLP13] Kai-Min Chung, Daniel Dadush, Feng-Hao Liu, and Chris Peikert. On the lattice smoothing parameter problem. 2013. 17

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 523–552, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany. xxv, 31, 33, 35, 59

[CN11] Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0 : Better lattice security estimates. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 1–20, Seoul, South Korea, December 4–8, 2011. Springer, Berlin, Germany. 25, 97, 98, 99, 109, 110, 111, 112, 113

[Coc01] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In Bahram Honary, editor, *8th IMA International Conference on Cryptography and Coding*, volume 2260 of *LNCS*, pages 360–363, Cirencester, UK, December 17–19, 2001. Springer, Berlin, Germany. xviii

[Con03] Consortium for Efficient Embedded Security. Efficient embedded security standards #1 : Implementation aspects of NTRUEncrypt and NTRUSign. Version 2.0 available at [IEE03], June 2003. 32, 38, 39, 41

[Coo71] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, STOC '71, pages 151–158, New York, NY, USA, 1971. ACM. xvi, 20

[Cop97] Don Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *Journal of Cryptology*, 10(4) :233–260, 1997. 24

[CRB01] M.A. Collins, J. Rice, and J. Batteer. *Windtalkers*. HarperCollins, 2001. ix

[DB76] Eugene D. Denman and Alex N. Beavers. *The matrix sign function and computations in systems*. American Elsevier, 1976. 68

[DD12] Léo Ducas and Alain Durmus. Ring-LWE in polynomial rings. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 34–51, Darmstadt, Germany, May 21–23, 2012. Springer, Berlin, Germany. xxvi, 23

[DDLL13] Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal gaussians. In Ran Canetti and Juan A. Garay, editors, *CRYPTO (1)*, volume 8042 of *Lecture Notes in Computer Science*, pages 40–56. Springer, 2013. xxvi

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6) :644–654, 1976. xii

[DKRS03] I. Dinur, G. Kindler, R. Raz, and S. Safra. Approximating cvp to within almost-polynomial factors is np-hard. *Combinatorica*, 23(2) :205–243, April 2003. 20

[DMQ13] Nico Döttling and Jörn Müller-Quade. Lossy codes and a new variant of the learning-with-errors problem. In Johansson and Nguyen [JN13], pages 18–34. 22

[DN12a] Léo Ducas and Phong Q. Nguyen. Faster gaussian lattice sampling using lazy floating-point arithmetic. In Wang and Sako [WS12], pages 415–432. xxv, xxvi, 82

[DN12b] Léo Ducas and Phong Q. Nguyen. Learning a zonotope and more : Cryptanalysis of ntrusign countermeasures. In Wang and Sako [WS12], pages 433–450. xxvi, 71, 95, 99, 110

[DPSZ12] Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 643–662, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Berlin, Germany. 94

[Duc10] Léo Ducas. Anonymity from asymmetry : New constructions for anonymous HIBE. In Josef Pieprzyk, editor, *CT-RSA 2010*, volume 5985 of *LNCS*, pages 148–164, San Francisco, CA, USA, March 1–5, 2010. Springer, Berlin, Germany. xxvi

[ECR] ECRYPT II. Ecrypt II yearly report on algorithms and keysizes (2011-2012). Available on `http://www.ecrypt.eu.org/`. 109, 110

[Ell70] J. H. Ellis. The possibility of secure non-scret digital encryption, 1970. xii

[FFS88] U. Feige, A. Fiat, and A. Shamir. Zero-knowledge proofs of identity. *J. Cryptol.*, 1(2) :77–94, August 1988. xiv

[FJK96] A. Frieze, M. Jerrum, and R. Kannan. Learning linear transformations. In *37th Annual Symposium on Foundations of Computer Science (Burlington, VT, 1996)*, pages 359–368. IEEE Comput. Soc. Press, Los Alamitos, CA, 1996. 53

[FM04] Uriel Feige and Daniele Micciancio. The inapproximability of lattice and coding problems with preprocessing. *Journal of Computer and System Sciences*, 69(1) :45–67, 2004. Preliminary version in CCC 2002. 20

[FP83] U. Fincke and Michael Pohst. A procedure for determining algebraic integers of given norm. In J. A. van Hulzen, editor, *Computer Algebra, EUROCAL 83, European Computer Algebra Conference, London, England, March 28-30, 1983, Proceedings*, volume 162 of *Lecture Notes in Computer Science*, pages 194–202. Springer, 1983. 25

[FS87] Amos Fiat and Adi Shamir. How to prove yourself : Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194, Santa Barbara, CA, USA, August 1987. Springer, Berlin, Germany. 30, 31, 34

[FS96] Jean-Bernard Fischer and Jacques Stern. An efficient pseudo-random generator provably as secure as syndrome decoding. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 245–255, Saragossa, Spain, May 12–16, 1996. Springer, Berlin, Germany. 105

[Gam08] Nicolas Gama. *Géométrie des nombres et Cryptanalyse de NTRU*. 2008. 80

[Gau01] Carl Friedrich Gauss. Disquisitiones arithmeticae. 1801. 20

[GD] Steven D. Galbraith and Nagarjun C. Dwarakanath. Efficient sampling from discrete gaussians for lattice-based cryptography on a constrained device. Survey of 2012. Available on `http://www.math.auckland.ac.nz/~sgal018/pubs.html`. 82, 83

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 169–178, Bethesda, Maryland, USA, May 31 – June 2, 2009. ACM Press. xvi, 8, 94

[GGH97] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. In Burton S. Kaliski Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 112–131, Santa Barbara, CA, USA, August 17–21, 1997. Springer, Berlin, Germany. xxiii, xxiv, 31, 32, 37, 38, 95

[GGH12] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. Cryptology ePrint Archive, Report 2012/610, 2012. To appear in Eurocrypt 2013. 94

[GGH13] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Johansson and Nguyen [JN13], pages 1–17. xv, 8

[GHS12] Craig Gentry, Shai Halevi, and Nigel P. Smart. Homomorphic evaluation of the AES circuit. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 850–867, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Berlin, Germany. 8, 98

[GL89] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *21st ACM STOC*, pages 25–32, Seattle, Washington, USA, May 15–17, 1989. ACM Press. 112

[GLP12] Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical lattice-based cryptography : A signature scheme for embedded systems. In Emmanuel Prouff and Patrick Schaumont, editors, *CHES 2012*, volume 7428 of *LNCS*, pages 530–547, Leuven, Belgium, September 9–12, 2012. Springer, Berlin, Germany. xxv, xxvi, 30, 82, 94, 95, 105, 107, 110, 113, 114

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, STOC '82, pages 365–377, New York, NY, USA, 1982. ACM. xiii

[GMSS99] O. Goldreich, D. Micciancio, S. Safra, and J. P. Seifert. Approximating shortest lattice vectors is not harder than approximating closet lattice vectors. *Inf. Process. Lett.*, 71(2) :55–61, July 1999. 20

[GN08] Nicolas Gama and Phong Q. Nguyen. Predicting lattice reduction. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 31–51, Istanbul, Turkey, April 13–17, 2008. Springer, Berlin, Germany. 25, 97, 98, 110, 112

[GNR10] Nicolas Gama, Phong Q. Nguyen, and Oded Regev. Lattice enumeration using extreme pruning. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 257–278, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany. 25, 110

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206, Victoria, British Columbia, Canada, May 17–20, 2008. ACM Press. xxiv, xxv, 21, 22, 31, 33, 34, 35, 37, 41, 57, 59, 61, 62, 66, 71, 80, 81, 82, 83, 85, 95

[GS02] Craig Gentry and Michael Szydlo. Cryptanalysis of the revised NTRU signature scheme. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 299–320, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer, Berlin, Germany. xxiv, 57, 95

[Ham11] Mike Hamburg. Spatial encryption. *IACR Cryptology ePrint Archive*, 2011 :389, 2011. 8

[HG07] Nick Howgrave-Graham. A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 150–169, Santa Barbara, CA, USA, August 19–23, 2007. Springer, Berlin, Germany. 46, 95, 97, 99, 109, 110, 113

[HGP+] J. Hoffstein, N. A. Howgrave Graham, J. Pipher, J. H. Silverman, and W. Whyte. NTRUSIGN : Digital signatures using the NTRU lattice. Full version of [HNHGSW03]. Draft of April 2, 2002, available on NTRU's website. 37, 38, 39, 48

[HGSW03] N Howgrave-Graham, J H Silverman, and W Whyte. A meet-in-the-middle attack on an NTRU private key. 2003. 46, 47, 48

[HHGP+05] Jeff Hoffstein, Nick A. Howgrave-Graham, Jill Pipher, Joseph H. Silverman, and William Whyte. Performances improvements and a baseline parameter generation algorithm for NTRUsign. In *Proc. of Workshop on Mathematical Problems and Techniques in Cryptology*, pages 99–126. CRM, 2005. 39

[HHGPW10] Jeff Hoffstein, Nick Howgrave-Graham, Jill Pipher, and William Whyte. Practical lattice-based cryptography : NTRUEncrypt and NTRUSign. 2010. In [NV10]. 32, 37, 39, 50

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4) :1364–1396, 1999. 4

[HKT11] Thomas Holenstein, Robin Künzler, and Stefano Tessaro. The equivalence of the random oracle model and the ideal cipher model, revisited. In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pages 89–98, San Jose, California, USA, June 6–8, 2011. ACM Press. xii

[HNHGSW03] Jeffrey Hoffstein, Jill Pipher Nick Howgrave-Graham, Joseph H. Silverman, and William Whyte. NTRUSIGN : Digital signatures using the NTRU lattice. In Marc Joye, editor, *CT-RSA 2003*, volume 2612 of *LNCS*, pages 122–140, San Francisco, CA, USA, April 13–17, 2003. Springer, Berlin, Germany. xxiii, xxiv, 31, 32, 33, 37, 60, 64, 67, 68, 95, 110, 111, 125

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU : A ring-based public key cryptosystem. In *Algorithmic Number Theory – Proc. ANTS-III*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer, 1998. xxiii, 32, 37, 38, 94, 98, 99

[HPS01] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NSS : An NTRU lattice-based signature scheme. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 211–228, Innsbruck, Austria, May 6–10, 2001. Springer, Berlin, Germany. xxiv, 95

[HPS11] Guillaume Hanrot, Xavier Pujol, and Damien Stehlé. Analyzing blockwise lattice algorithms using dynamical systems. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 447–464, Santa Barbara, CA, USA, August 14–18, 2011. Springer, Berlin, Germany. 110

[HWH08] Yupu Hu, Baocang Wang, and Wencai He. NTRUSign with a new perturbation. *IEEE Transactions on Information Theory*, 54(7) :3216–3221, 2008. xxiv, 33, 37, 40, 41, 51, 53, 55

[IEE03] IEEE P1363.1. Public-key cryptographic techniques based on hard problems over lattices. See `http://grouper.ieee.org/groups/1363/lattPK/index.html`, June 2003. 33, 37, 38, 123

[JN13] Thomas Johansson and Phong Q. Nguyen, editors. *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*. Springer, 2013. 123, 124, 126

[Jou04] Antoine Joux. A one round protocol for tripartite Diffie-Hellman. *Journal of Cryptology*, 17(4) :263–276, September 2004. xv

[Kah96] David Kahn. *The Codebreakers : The Comprehensive History of Secret Communication from Ancient Times to the Internet*. Scribner, 1996. viii, xi

[Kan87] Ravi Kannan. Minkowski's convex body theorem and integer programming. *Math. Oper. Res.*, 12(3) :415–440, August 1987. 20

[Kar13] Charles F. F. Karney. Sampling exactly from the normal distribution. Technical report, SRI International, March 2013. 82

[Ker83] Auguste Kerckhoffs. *La cryptographie militaire, ou, Des chiffres usités en temps de guerre : avec un nouveau procédé de déchiffrement applicable aux systèmes à double clef*. Librairie militaire de L. Baudoin, 1883. x

[Kle00] Philip N. Klein. Finding the closest lattice vector when it's unusually close. In *Proc. ACM SODA*, pages 937–941, 2000. xxiv, xxv, 33, 37, 41, 57, 59, 61, 62, 71, 88

[Lag73] L. Lagrange. Recherches d'arithmétique. 1773. 20

[LATV12] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In Howard J. Karloff and Toniann Pitassi, editors, *44th ACM STOC*, pages 1219–1234, New York, NY, USA, May 19–22, 2012. ACM Press. 97, 98, 99

[LLL82] Arjen K. Lenstra, Hendrik W. Lenstra Jr., and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Ann.*, 261 :513–534, 1982. xxii, 24, 59

[LM06] Vadim Lyubashevsky and Daniele Micciancio. Generalized compact Knapsacks are collision resistant. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *ICALP 2006, Part II*, volume 4052 of *LNCS*, pages 144–155, Venice, Italy, July 10–14, 2006. Springer, Berlin, Germany. 98

[LM08] Vadim Lyubashevsky and Daniele Micciancio. Asymptotically efficient lattice-based digital signatures. In Ran Canetti, editor, *TCC 2008*, volume 4948 of *LNCS*, pages 37–54, San Francisco, CA, USA, March 19–21, 2008. Springer, Berlin, Germany. xxv

[LP11] Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In Aggelos Kiayias, editor, *CT-RSA 2011*, volume 6558 of *LNCS*, pages 319–339, San Francisco, CA, USA, February 14–18, 2011. Springer, Berlin, Germany. 59

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany. 23, 24, 68, 94, 98

[LPR13] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A toolkit for ring-lwe cryptography. In Johansson and Nguyen [JN13], pages 35–54. 23

[Lyu08] Vadim Lyubashevsky. Lattice-based identification schemes secure under active attacks. In Ronald Cramer, editor, *PKC 2008*, volume 4939 of *LNCS*, pages 162–179, Barcelona, Spain, March 9–12, 2008. Springer, Berlin, Germany. 29, 94

[Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts : Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616, Tokyo, Japan, December 6–10, 2009. Springer, Berlin, Germany. xxv, xxvi, 30, 94, 95

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 738–755, Cambridge, UK, April 15–19, 2012. Springer, Berlin, Germany. xxv, xxvi, 30, 31, 37, 59, 80, 81, 82, 83, 86, 87, 94, 95, 96, 97, 100, 101, 105, 109, 110, 113, 114

[Mer78] Ralph C. Merkle. Secure communications over insecure channels. *Commun. ACM*, 21(4) :294–299, April 1978. xii

[MH78] Ralph C. Merkleand and Martin E. Hellman. Hiding information and signatures in trapdoor knapsacks. *IEEE Transactions On Information Theory*, 24 :525–530, 1978. xix, 33

[Mic01] Daniele Micciancio. Improving lattice-based cryptosystems using the Hermite normal form. In *Proc. of CALC '01*, volume 2146 of *LNCS*. Springer, 2001. 32, 38

[Mil82] F. Miller. *Telegraphic Code to Insure Privacy and Secrecy in the Transmission of Telegrams*. C.M. Cornwell, 1882. xi

[MM11] Daniele Micciancio and Petros Mol. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 465–484, Santa Barbara, CA, USA, August 14–18, 2011. Springer, Berlin, Germany. 109, 110, 112

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices : Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718, Cambridge, UK, April 15–19, 2012. Springer, Berlin, Germany. xxv, 22, 31, 33, 59, 60, 61, 67, 68, 69, 71, 94, 95

[MP13] Daniele Micciancio and Chris Peikert. Hardness of sis and lwe with small parameters. Cryptology ePrint Archive, Report 2013/069, 2013. `http://eprint.iacr.org/`. 22

[MPSW] T. Malkin, C. Peikert, R. A. Servedio, and A. Wan. Learning an overcomplete basis : Analysis of lattice-based signatures with perturbations. 2009 manuscript cited in [Pei10], available as [Wan10, Chapter 6]. 37

[MR04] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on Gaussian measures. In *45th FOCS*, pages 372–381, Rome, Italy, October 17–19, 2004. IEEE Computer Society Press. xxiv, 17, 21, 77, 98

[MR09] Daniele Micciancio and Oded Regev. Lattice-based cryptography. In *Post-quantum cryptography*, pages 147–191. Springer, Berlin, 2009. 59, 109, 110, 112

[MSV09] Ivan Morel, Damien Stehlé, and Gilles Villard. H-LLL : using householder inside LLL. In *Proc. ISSAC '09*, pages 271–278. ACM, 2009. 59

[MV10] Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. In Leonard J. Schulman, editor, *42nd ACM STOC*, pages 351–358, Cambridge, Massachusetts, USA, June 5–8, 2010. ACM Press. 20

[Ngu99] Phong Q. Nguyen. Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from Crypto'97. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 288–304, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Berlin, Germany. 32

[NIS] NIST Special Publication 800-131A. Transitions : Recommendation for transitioning the use of cryptographic algorithms and key lengths. Available on `http://csrc.nist.gov`. 109, 110

[NR06] Phong Q. Nguyen and Oded Regev. Learning a parallelepiped : Cryptanalysis of GGH and NTRU signatures. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 271–288, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Berlin, Germany. xxiv, 31, 32, 33, 37, 39, 40, 41, 42, 43, 46, 48, 50, 51, 52, 71

[NR09] Phong Q. Nguyen and Oded Regev. Learning a parallelepiped : Cryptanalysis of GGH and NTRU signatures. *Journal of Cryptology*, 22(2) :139–160, April 2009. 95, 99, 110

[NS09] Phong Q. Nguyen and Damien Stehlé. An LLL algorithm with quadratic complexity. *SIAM J. Comput.*, 39(3) :874–903, 2009. 59

[NV10] Phong Q. Nguyen and Brigitte Vallée, editors. *The LLL Algorithm : Survey and Applications*. Information Security and Cryptography. Springer, 2010. 20, 21, 24, 125

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 223–238, Prague, Czech Republic, May 2–6, 1999. Springer, Berlin, Germany. xvi, xviii

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem : extended abstract. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 333–342, Bethesda, Maryland, USA, May 31 – June 2, 2009. ACM Press. 22

[Pei10] Chris Peikert. An efficient and parallel gaussian sampler for lattices. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 80–97, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Berlin, Germany. xxv, 33, 57, 59, 60, 61, 67, 68, 71, 82, 83, 127

[PG12] Thomas Pöppelmann and Tim Güneysu. Towards efficient arithmetic for lattice-based cryptography on reconfigurable hardware. In Alejandro Hevia and Gregory Neven, editors, *LATINCRYPT 2012*, volume 7533 of *LNCS*, pages 139–158, Santiago, Chile, October 7–10, 2012. Springer, Berlin, Germany. 105

[PR06] Chris Peikert and Alon Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In Shai Halevi and Tal Rabin, editors, *TCC 2006*, volume 3876 of *LNCS*, pages 145–166, New York, NY, USA, March 4–7, 2006. Springer, Berlin, Germany. 98

[PS96] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 387–398, Saragossa, Spain, May 12–16, 1996. Springer, Berlin, Germany. 30

[PS08] Xavier Pujol and Damien Stehlé. Rigorous and efficient short lattice vectors enumeration. In Josef Pieprzyk, editor, *ASIACRYPT 2008*, volume 5350 of *LNCS*, pages 390–405, Melbourne, Australia, December 7–11, 2008. Springer, Berlin, Germany. 25, 70, 72

[Rab79a] M. O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical report, Cambridge, MA, USA, 1979. xiii

[Rab79b] Michael O. Rabin. Digital signatures and public key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, Massachusetts Institute of Technology, January 1979. 34

[Reg03] Oded Regev. Improved inapproximability of lattice and coding problems with preprocessing. In *IEEE Transactions on Information Theory*, pages 363–370. IEEE, 2003. 20

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93, Baltimore, Maryland, USA, May 22–24, 2005. ACM Press. xxiii, xxiv, 22, 28, 29, 35

[RS10] Markus Rückert and Michael Schneider. Estimating the security of lattice-based cryptosystems. Cryptology ePrint Archive, Report 2010/137, 2010. `http://eprint.iacr.org/`. 59

[RSA78] R.L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21 :120–126, 1978. xiii, xv, xviii

[Rüc10] Markus Rückert. Lattice-based blind signatures. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 413–430, Singapore, December 5–9, 2010. Springer, Berlin, Germany. 94

[Sch88] Claus-Peter Schnorr. A more efficient algorithm for lattice basis reduction. *J. Algorithms*, 9(1) :47–62, 1988. 59

[Sch90] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 239–252, Santa Barbara, CA, USA, August 20–24, 1990. Springer, Berlin, Germany. 29, 30, 31

[SD82] Adi Shamir and Whitfield Diffie. A polynomial-time algorithm for breaking the basic merkle-hellman cryptosystem. In *In Proceedings of the 23rd IEEE Symposium on Foundations of Computer Science*, pages 145–152. IEEE, 1982. xix, 33

[SE93] C. P. Schnorr and M. Euchner. Lattice basis reduction : Improved practical algorithms and solving subset sum problems. In *Math. Programming*, pages 181–191, 1993. 25

[SH95] Claus-Peter Schnorr and Horst Helmut Hörner. Attacking the Chor-Rivest cryptosystem by improved lattice reduction. In Louis C. Guillou and Jean-Jacques Quisquater, editors, *EUROCRYPT'95*, volume 921 of *LNCS*, pages 1–12, Saint-Malo, France, May 21–25, 1995. Springer, Berlin, Germany. 25

[Sha49] C. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal, Vol 28, pp. 656–715*, 1949. xi

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5) :1484–1509, October 1997. xvii

[SS11] Damien Stehlé and Ron Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 27–47, Tallinn, Estonia, May 15–19, 2011. Springer, Berlin, Germany. 32, 97, 98, 99

[SW12] Amit Sahai and Brent Waters. Attribute-based encryption for circuits from multilinear maps. Cryptology ePrint Archive, Report 2012/592, 2012. 94

[vEB81] Peter van Emde Boas. Another np-complete partition problem and the complexity of computing short vectors in a lattice. 1981. xxii, 20

[vN51] John von Neumann. Various techniques used in connection with random digits. *J. Research Nat. Bur. Stand., Appl. Math. Series*, 12 :36–38, 1951. 95

[Wan10] A. Wan. *Learning, cryptography, and the average case.* PhD thesis, Columbia University, 2010. Available at `http://itcs.tsinghua.edu.cn/ atw12/`. 127

[Woo02] John Woo. Windtalkers, 2002. MGM Production. Staring Nicolas Cage. `http://www.imdb.com/title/tt0245562/`. ix

[WS12] Xiaoyun Wang and Kazue Sako, editors. *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, volume 7658 of *Lecture Notes in Computer Science*. Springer, 2012. 123

Résumé

Les réseaux euclidiens font l'objet d'un fort engouement de la part de la communauté de recherche théorique en cryptographie ces dernières années. Ils offrent des fondations peut-être plus solides, et s'avèrent aussi plus souples. Cependant, les efforts d'implémentation efficace de cette cryptographie innovante restent limités : il s'agit essentiellement des cryptosystèmes NTRU introduits à la fin des années 1990. Cette thèse s'inscrit dans cette direction, en se focalisant sur cas des signatures numériques.
Nous présentons d'abord la première attaque pratique sur le schéma de signature NTRUSign lorsque des contremesures sont mises en place, notamment celles proposées par l'entreprise NTRU. Pour cela, nous montrons que l'attaque de Nguyen-Regev est plus robuste que prévue : elle permet d'apprendre des structures plus complexes que des parallélépipèdes, comme les zonotopes et des parallélépipèdes déformés.
Nous nous intéressons ensuite à une autre contremesure : l'échantillonnage Gaussien discret, qui permet de prouver des propriétés de securité, mais qui était jusqu'alors peu efficace. Nous proposons de nouveaux algorithmes adaptés et efficaces pour cette tache, avec et sans virgule flottante.
Nous concluons cette thèse par la conception et l'implémentation d'un nouveau schéma de signature, BLISS, en nous appuyant sur de nombreuses idées du domaine, et en ayant deux objectifs : la sécurité prouvée, et l'efficacité pratique. Nous introduisons l'utilisation de gaussiennes bimodales, qui permet, de façon surprenante, de tirer parti à la fois des progrès sur les signatures sans trappes, et de la génération de trappes à la manière de NTRU. Notre implémentation Open-Source s'avère compétitive avec les normes RSA et ECDSA.

Abstract

Lattices have attracted significant interest in theoretical cryptographic research in the past few years. They offer perhaps stronger foundations, and have also proved very versatile. However, efforts towards efficient implementations of lattice-based cryptosystems have remained limited : they are essentially restricted to NTRU primitives introduced at the end of the 1990s. This thesis goes in this direction, and focuses on digital signatures.
We first present a practical attack on the NTRUSign signature scheme in the presence of countermeasures, such as the one proposed by NTRU. To do this, we show that the Nguyen-Regev attack is more robust than expected : it is able to learn more complex objects than parallelepipeds, such as zonotopes or deformed parallelepipedes.
We then move our attention to an alternative countermeasure that is provably secure, yet not so efficient. We propose new algorithms that are adapted and efficient for this task, with or without usage of floating point.
We conclude this thesis with the design and implementation of a new signature scheme, BLISS, with two objectives in mind : provable security and practical efficiency. We introduce the use of Bimodal Gaussian, which surprisingly allows one to benefit from both trapdoor-less signatures and NTRU-like trapdoor generation. Our implementation is Open-Source, and competes favorably with the RSA and ECDSA standards.