Université Grenoble Alpes

INSTITUT FOURIER

Travail d'étude et de recherche

Algorithme LLL de réduction de réseaux

MAREAU Quentin

Encadré par VITSE Vanessa

Année 2021

Résumé

La notion de réseaux a depuis longtemps laissé place à des problèmes algorithmiques complexes dont le plus connu est celui du "vecteur le plus court" plus communément appelé *SVP* pour "Shortest Vector Problem". Celui-ci consiste à déterminer quel est le vecteur non nul du réseau avec la norme la plus petite et avait été résolu par Lagrange et Euler mais uniquement pour des réseaux de dimension 2. Cependant, le problème pour un réseau et une dimension quelconque est NP-difficile.
C'est pourquoi il est nécessaire d'approcher les solutions de ce problème par ce que l'on appelle une réduction de réseaux. Celle-ci nous permet de trouver à un réseau une base dont les vecteurs sont plus courts et donc plus proches du plus petit vecteur. Il existe de nombreux algorithmes de réduction de réseaux mais l'un des plus célèbres est celui de Arjen Lenstra, Hendrik Lenstra et László Lovász car il tourne en un temps polynomial.
Ainsi dans ce mémoire très inspiré du travail de Steven Galbraith [1], nous étudierons en détails cet algorithme, sa complexité puis son application en cryptographie.

Table des matières

Chapitre 1

Introduction

Afin de débuter ce mémoire, nous allons introduire la notion de réseau qui est le support de travail principal pour tout ce qui va suivre. Cet outil étant au centre de l'algorithme LLL et des exemples qui seront étudiés plus tard tels que le problème du sac-à-dos, il semble nécessaire de bien comprendre de quoi il s'agit et de donner quelques propriétés interessantes à son sujet.

1.1 Réseau : définitions et géneralités

Définition 1.1.1. Soient $m, n \in \mathbb{N}$ tels que $n \leq m$. Soit $\mathscr{B} = (b_1, ..., b_n)$ une famille libre de $\mathbb{R}^m$. On définit alors le réseau R engendré par $\mathscr{B}$ par :

$$R = \left\{ \sum_{i=1}^{n} l_i b_i \, , \, l_i \in \mathbb{Z} \right\}$$

On dit alors que $\mathscr{B}$ est une base du réseau R et que R est un réseau de rang n et de dimension m.
On dit qu'un réseau est euclidien lorsque $\mathbb{R}^m$ est muni d'un produit scalaire $\langle ., . \rangle$.

Remarque. Dans la totalité de ce mémoire, nos réseaux seront euclidiens sans qu'on ait à le préciser.
De plus, il est naturel de dire que $R' \subset R$ est un sous-réseau de R si R' est un réseau.
Enfin, la notion se généralise à tout $\mathbb{R}$-espace vectoriel de dimension finie de la même façon.

Exemple. Deux exemples de réseaux de $\mathbb{R}^2$ sont interessant car assez visuels :
Réseau carré : C'est le réseau engendré par $\left(\begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right)$.
Réseau hexagonal : C'est le réseau engendré par $\left(\begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1/2 \\ \sqrt{3}/2 \end{pmatrix} \right)$.

Leurs noms réspectifs proviennent des formes qu'ils font apparaître :

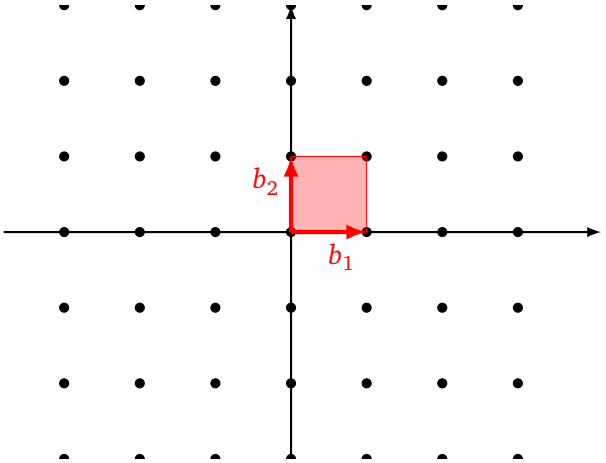

FIGURE 1.1 – Représentation du réseau carré

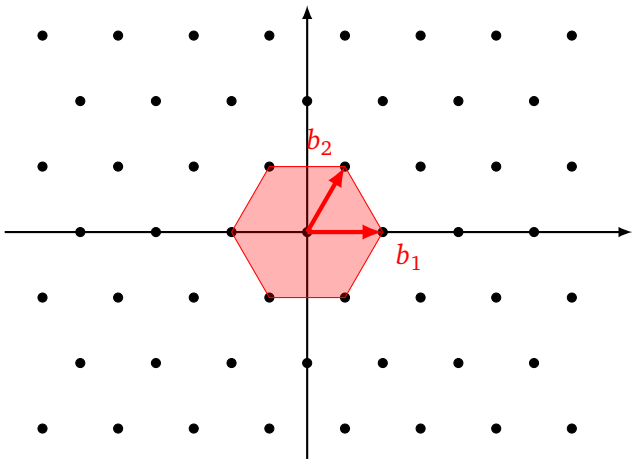

FIGURE 1.2 – Représentation du réseau hexagonal

Il existe différentes définitons de la notion de réseau toutes équivalentes :

Définition 1.1.2. Soit G un sous-groupe de $\mathbb{R}^m$, on dit que G est discret si pour tout $r \geq 0$, l'ensemble $\{g \in G\,,\ \|g\| \leq r\}$ est fini.

Théorème 1.1.1. *On a équivalence des trois assertions suivantes :*

1. *R est un réseau de* $\mathbb{R}^m$.
2. *R est un sous-groupe discret de* $\mathbb{R}^m$.
3. *Pour tout* $x \in R$*, il existe un ouvert V de* $\mathbb{R}^m$ *(pour la topologie usuelle) tel que* $V \cap R = \{x\}$.

Preuve. La preuve complète est traitée en Annexe 1.

Définition 1.1.3. Soit R un réseau de $\mathbb{R}^m$ de base $\mathscr{B} = (b_1, \ldots, b_n)$, on définit le déterminant du réseau par :

$$\det(R) = |\det_{\mathscr{B}_0}(\mathscr{B})| = |\det(P^{\mathscr{B}}_{\mathscr{B}_0})|\ \text{, avec } \mathscr{B}_0 \text{ une base orthonormée de Vect}(b_1, \ldots, b_n).$$

Remarque. De façon pratique, pour les réseaux de rang m, on pourra prendre la base canonique de $\mathbb{R}^m$ pour $\mathscr{B}_0$.

On peut vérifier que cette définition fait bien sens, c'est-à-dire que ce nombre ne dépend que du réseau étudié. En effet si on a deux bases orthonormées $\mathscr{B}_0$ et $\mathscr{B}_0'$ de Vect$(b_1, \ldots, b_n)$, alors on a :

$$|\det(P^{\mathscr{B}}_{\mathscr{B}_0'})| = |\det(P^{\mathscr{B}_0}_{\mathscr{B}_0'} P^{\mathscr{B}}_{\mathscr{B}_0})| = |\det(P^{\mathscr{B}}_{\mathscr{B}_0})||\det(P^{\mathscr{B}_0}_{\mathscr{B}_0'})| = |\det(P^{\mathscr{B}_0}_{\mathscr{B}})|\ \text{, car}\ |\det(P^{\mathscr{B}_0'}_{\mathscr{B}_0})| = 1$$

De plus, si $\mathscr{B} = (b_1, ..., b_n)$ et $\mathscr{B}' = (b_1', ..., b_n')$ sont deux bases de R, on note P la matrice de passage de $\mathscr{B}$ à $\mathscr{B}'$. Pour tout $i \in \{1, ..., n\}$, on a donc $b_i' = \sum_{j=1}^{n} l_j b_j$ avec les $l_j \in \mathbb{Z}$ et donc P est à valeurs entières. De la même façon, P^{-1} sera aussi à valeurs entières et donc $P \in GL_n(\mathbb{Z})$ et $|det(P)| = 1$, ce qui conclut comme fait précédemment.

Définition 1.1.4. Soit R un réseau de rang n de $\mathbb{R}^m$. Si $\mathscr{B} = (b_1, ..., b_n)$ est une base de R, on note $S \in M_n(\mathbb{R})$ la matrice de Gram (aussi appelée matrice de Schäfli) définie par : $S_{(i,j)} = \langle b_i, b_j \rangle$ pour tout $i, j \in \{1, ..., n\}$. En particulier, pour le produit scalaire canonique, $S = B^T B$ où B est la matrice de $\mathscr{B}$ dans la base canonique.

Proposition 1.1.2. *Soit R un réseau de $\mathbb{R}^m$ de rang n. Avec les mêmes notations qu'à la définition précédente, on a :* $\det(R) = \sqrt{\det(S)}$. *En particulier, pour le produit scalaire canonique,* $\det(R) = \sqrt{\det(B^T B)}$.

Preuve. Soit $F = (f_1, ..., f_n)$ un base orthonormée de $\mathrm{Vect}(b_1, ..., b_n)$ pour le produit scalaire $\langle ., . \rangle$ (a priori quelconque). Aussi, la matrice de Gram dans cette base est $S_F = I_n$.
Soit P la matrice de passage de F à $\mathscr{B}$, on a alors que $\det(R) = |\det(P)|$. De plus, si on note S la matrice de Gram dans la base $\mathscr{B}$, on a par changement de base d'une forme bilinéaire : $S = P^T S_F P = P^T P$.
On a donc $\det(S) = \det(P)^2 = \det(R)^2$ et donc $\det(R) = \sqrt{\det(S)}$.

Lemme 1.1.3. *Soient $\mathscr{B} = (b_1, ..., b_n)$ une base d'un réseau R de $\mathbb{R}^m$ et $\mathscr{B}^* = (b_1^*, ..., b_n^*)$ sa base orthogonale associée issue du procédé de Gram-Schmidt. On vérifie :*

1. *Pour tout $i \in \{1, ..., n\}$, $\left\|b_i^*\right\| \leq \|b_i\|$.*
2. $\det(R) = \prod_{i=1}^{n} \left\|b_i^*\right\|$

Preuve. On peut voir le passage de b_i à b_i^* comme : $b_i^* = \pi_i(b_i)$ où π_i est la projection orthogonale sur $(b_1, ..., b_{i-1})^\perp$. Ainsi, on remarque que $\|b_i\| = \|b_i - \pi_i(b_i) + \pi_i(b_i)\|^2 = \left\|b_i - b_i^*\right\|^2 + \left\|b_i^*\right\|^2$ par Pythagore. Ceci conclut alors le point 1.
Le point 2. se montre en choisissant la base orthonormée de Gram-Schmidt pour le calcul du déterminant. Lorsqu'on explicite la matrice de passage P, celle-ci est diagonale supérieure avec les $\left\|b_i^*\right\|$ sur la diagonale. Ainsi on a bien $\det(R) = \det(P) = \prod_{i=1}^{n} \left\|b_i^*\right\|$.

Ce lemme permet aisément de montrer l'inégalité d'Hadamard :

Théorème 1.1.4. *Soit $\mathscr{B} = (b_1, ..., b_n)$ une base d'un réseau R de $\mathbb{R}^m$. On vérifie :*

$$\det(R) \leq \prod_{i=1}^{n} \|b_i\|$$

Définition 1.1.5. Soit R un réseau de $\mathbb{R}^m$ de rang n. On définit les minima successifs de R les réels positifs $\lambda_1 \leq ... \leq \lambda_n$ tels que pour tout $i \in \{1, ..., n\}$, λ_i est le plus petit réel positif tel qu'il existe i vecteurs linéairement indépendants $v_1, ..., v_i$ de R avec $\left\|v_j\right\| \leq \lambda_i$ pour tout $j \in \{1, ..., i\}$.

Remarque. Trouver ces minima n'est pas quelque chose de simple. En effet, ne serait-ce que trouver λ_1 consiste à trouver le vecteur non nul de R avec la norme la plus petite et est un problème difficile.
Par ailleurs, on verra dans la prochaine partie que pour des réseaux de dimension 2, il est possible de

trouver une base du réseau dont les vecteurs ont pour normes les minima successifs. Cependant, pour des dimensions plus grandes, ceci n'est pas toujours possible.

Exemple. On considère dans $\mathbb{R}^5$ le réseau R engendré par les éléments de la base canonique e_1, e_2, e_3, e_4 et $v = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2})$.
Il est facile de voir que tout vecteur de R est de norme supérieure ou égale à 1.

De plus $e_5 = 2v - e_1 - e_2 - e_3 - e_4 \in R$ donc on a la famille $F = (e_1, ..., e_5)$ de R libre et dont tous les vecteurs sont de norme 1. Ceci nous donne alors $\lambda_1 = ... = \lambda_5 = 1$.

Cependant, si une famille libre $B = (u_1, ..., u_5)$ est une base, alors il existe $i \in \{1, ..., 5\}$ tel que $u_i = \sum_{j=1}^{4} \mu_j e_j + \mu_5 v$ et μ_5 impair (sinon on ne pourrait pas construire v par exemple). Or la norme d'un tel vecteur est strictement plus grande que 1, et donc il n'est pas possible de construire une base dont les vecteurs ont pour normes les minima successifs.

Chapitre 2

Réduction de réseaux en dimension 2

Un des buts de la réduction de réseaux est de fournir une base dont les vecteurs sont courts et dont leurs normes sont proches des minima successifs. Or comme nous avons pu le constater plus tôt, il n'est pas toujours possible de construire une base dont les normes des vecteurs sont exactement ces minima. C'est pourquoi nous nous interessons en premier lieu à la dimension 2, qui elle nous permettra toujours d'obtenir une telle base [1].

2.1 Algorithme de Lagrange-Gauss

La réduction en dimension quelconque est un problème contemporain difficile sur plusieurs aspects. Cependant, la réduction de réseaux en dimension 2 avait déjà été étudiée et un algorithme avait été proposé par Lagrange puis par Gauss. Ce qui donnera donc lieu au premier algorithme que nous décrirons : l'algorithme de Lagrange-Gauss [2]. Comme dit précédemment, la réduction proposée ici a pour intérêt qu'elle donne des vecteurs ayant exactement pour normes les minima successifs d'un réseau R.
Le but de cette partie est de comprendre cette réduction pour essayer ensuite de voir comment la généraliser à n'importe quelle dimension. Ainsi, on va d'abord introduire le concept de base réduite au sens de Lagrange et Gauss puis donner des critères de réduction.

Définition 2.1.1. Soit $\mathscr{B} = (b_1, b_2)$ une base de $\mathbb{R}^2$. On dit que cette base est Lagrange-Gauss réduite si pour tout $q \in \mathbb{Z}$, $\|b_1\| \leq \|b_2\| \leq \|b_2 + q b_1\|$

Théorème 2.1.1. *Soient λ_1 et λ_2 les minima successifs d'un réseau R de rang 2. Si $\mathscr{B} = (b_1, b_2)$ est une base Lagrange-Gauss réduite de R, alors $\|b_1\| = \lambda_1$ et $\|b_2\| = \lambda_2$.*

1. Plus précisement, on est capable d'obtenir de telles bases jusqu'en dimension 4.
2. L'algorithme est surtout connu sous le nom d'*algorithme de Gauss*.

Preuve. Premièrement, par définition de notre base $\mathscr{B} = (b_1, b_2)$ Lagrange-Gauss réduite, on a :

$$\forall q \in \mathbb{Z}, \|b_1\| \leq \|b_2\| \leq \|b_2 + q b_1\|$$

Soient $l_1, l_2 \in \mathbb{Z}$ et $v = l_1 b_1 + l_2 b_2 \in R \backslash \{0\}$. Si $l_2 = 0$ alors $\|v\| = |l_1| \|b_1\| \geq \|b_1\|$ car $l_1 \neq 0$.
Si $l_2 \neq 0$, par division euclidienne de l_1 par l_2 dans $\mathbb{Z}$, il existe $q, r \in \mathbb{Z}$ tels que $l_1 = q l_2 + r$ et $0 \leq r < |l_2|$.
Ainsi, on a $v = r b_1 + l_2(b_2 + q b_1)$ qui donne par inégalité triangulaire :

$$\|l_2(b_2 + q b_1)\| = \|v - r b_1\| \leq \|v\| + \|r b_1\|$$

d'où

$$\|v\| \geq |l_2| \|b_2 + q b_1\| - r \|b_1\| = (|l_2| - r) \|b_2 + q b_1\| + r(\|b_2 + q b_1\| - \|b_1\|)$$

En remarquant enfin que $|l_2| - r \geq 1$ et que $\|b_2 + q b_1\| - \|b_1\| \geq 0$ par propriété de la base $\mathscr{B}$, on a :

$$\|v\| \geq \|b_2 + q b_1\| \geq \|b_2\| \geq \|b_1\|$$

Or on rappelle que λ_1 est la norme d'un plus court vecteur de R, ce qui correspond donc bien à $\|b_1\|$ d'après ce qui précède.
Ensuite, on a que $\lambda_2 \leq \|b_2\|$ par existence de la base $\mathscr{B}$. D'autre part, si $\mathscr{F} = (v_1, v_2)$ est une famille libre de R, il est clair qu'un des deux vecteurs n'est pas colinéaire à b_1, supposons ici sans perte de généralités qu'il s'agit de v_1. Ainsi $\|v_1\| \geq \|b_2\|$ et donc on a nécessairement que $\lambda_2 \geq \|b_2\|$ et donc $\lambda_2 = \|b_2\|$

Notations. Dans cette partie, si $b_1, b_2 \in \mathbb{R}^n$, on note $B_i = \|b_i\|^2 = \langle b_i, b_i \rangle$ pour $i \in \{1, 2\}$.

Lemme 2.1.2. *Soit $\mathscr{B} = (b_1, b_2)$ une base de $\mathbb{R}^2$ et $\mu = \dfrac{\langle b_1, b_2 \rangle}{B_1}$, on vérifie :*

$$\forall q \in \mathbb{Z}, \|b_2 - \lfloor \mu \rceil b_1\| \leq \|b_2 - q b_1\| \tag{2.1}$$

Preuve. Premièrement, on remarque que $\langle b_2 - \mu b_1, b_1 \rangle = \langle b_2, b_1 \rangle - \dfrac{\langle b_1, b_2 \rangle}{B_1} \langle b_1, b_1 \rangle = 0$. Ainsi on peut appliquer le théorème de Pythagore qui nous donne :

$$\forall q \in \mathbb{Z}, \|b_2 - q b_1\|^2 = \|b_2 - \mu b_1 + (\mu - q) b_1\|^2 = \|b_2 - \mu b_1\|^2 + (\mu - q)^2 B_1$$

On remarque alors que le premier terme est indépendant de q. D'autre part, $(\mu - q)^2$ atteint son minimum en $q = \lfloor \mu \rceil$ par définition de $\lfloor \mu \rceil$ et donc $\|b_2 - q b_1\|$ aussi. Ainsi $\|b_2 - \lfloor \mu \rceil b_1\| \leq \|b_2 - q b_1\|$ pour tout $q \in \mathbb{Z}$.

Cette étude préliminaire terminée, on peut enfin donner une algorithme ayant pour but, à la manière du procédé de Gram-Schmidt, de constuire une nouvelle base Lagrange-Gauss réduite.

Entrées : Une base (b_1, b_2) de R

Sorties : Une base (b_1, b_2) de R Lagrange-Gauss réduite

$B_1 \leftarrow \|b_1\|^2$

$\mu \leftarrow \langle b_1, b_2\rangle / B_1$

$b_2 \leftarrow b_2 - \lfloor \mu \rceil b_1$

$B_2 \leftarrow \|b_2\|^2$

tant que $B_2 < B_1$ **faire**

  $b_1 \leftrightarrow b_2$

  $B_1 \leftarrow B_2$

  $\mu \leftarrow \langle b_1, b_2\rangle / B_1$

  $b_2 \leftarrow b_2 - \lfloor \mu \rceil b_1$

  $B_2 \leftarrow \|b_2\|^2$

fin

Renvoyer (b_1, b_2)

Algorithme 1 : Algorithme de Lagrange-Gauss

On va étudier cet algorithme plus en détail afin de montrer qu'il termine et qu'il renvoie bien une base de R Lagrange-Gauss réduite.

Terminaison : Si avant de rentrer dans le **while**, on vérifie que $B_1 \leq B_2$, alors le programme termine directement. Sinon, on rentre dans la boucle **while** et donc $B_1 \leftarrow B_2 < B_1$ donc B_1 décroît strictement à chaque itération. Or les valeurs possibles pour B_1 sont discrètes, de fait il n'existe qu'un nombre fini de valeurs possibles pour B_1 plus petites que la valeur initiale. Ainsi B_1 ne peut décroître qu'un nombre fini de fois. Nécessairement, on aura bien une itération pour laquelle $b_2 \leftarrow b_2 - \lfloor \mu \rceil b_1$ donne $B_2 \geq B_1$. De fait le programme termine.

Correction : Premièrement, on va vérifier que l'opération $b_2 \leftarrow b_2 - \lfloor \mu \rceil b_1$ nous permet toujours d'avoir une base. On notera b_2' le vecteur issu de cette opération, on veut alors vérifier que la famille $\mathscr{B}' = (b_1, b_2')$ est toujours une base.

Si $\mathscr{B}'$ est non libre, il existe $\alpha \in \mathbb{Q}$ tel que $b_2' = \alpha b_1 \iff b_2 = (\alpha + \lfloor \mu \rceil) b_1$ donc $\mathscr{B}$ n'est pas libre, ce qui est absurde. Donc $\mathscr{B}'$ est libre.

De plus, pour tout $x \in R$, il existe $(l_1, l_2) \in \mathbb{Z}^2$ tel que $x = l_1 b_1 + l_2 b_2 = (l_1 + m l_2) b_1 + l_2 b_2'$ donc $\mathscr{B}'$ est génératrice et donc une base.

Enfin d'après le lemme (2.1), on a vu que pour tout $q \in \mathbb{Z}$, $\left\|b_2'\right\| \leq \|b_2 - qb_1\|$. Or d'après la terminaison de l'algorithme, on a que pour une certaine itération, on a $B_1 \leq B_2$ et donc, si on reprend les mêmes notations que celle introduite plus tôt pour cette itération, $\|b_1\| \leq \left\|b_2'\right\| \leq \|b_2 - qb_1\|$ pour tout $q \in \mathbb{Z}$ donc $\mathscr{B}'$ est Lagrange-Gauss réduite pour cette dernière itération.

Exemple. Voici deux exemples de l'application de l'algorithme de Lagrange-Gauss :

On peut voir un premier exemple sans détail de l'application de l'algorithme à un réseau dont la base est $(b_1, b_2) = ((1,3),(3,7))$.
Ainsi l'algorithme renvoit la base $(b_1', b_2') = ((1,1),(-1,1))$ dont les vecteurs sont, comme représenté ci-contre, bien plus courts.

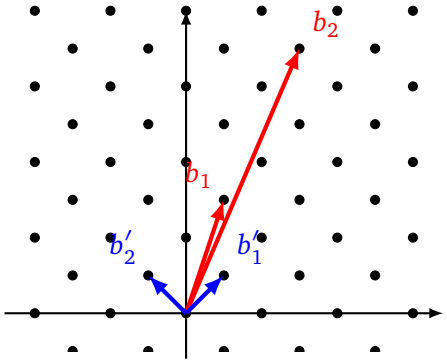

FIGURE 2.1 – Repésentation d'une réduction de réseau

Comme second exemple, on considère maintenant la famille libre $((199,1398),(118,829))$. On va alors décrire quelques étapes de l'algorithme :

- Avant la boucle **while**, $b_1 = (199,1398)$ ne change pas et $\mu = 0.59...$ d'où $\lfloor \mu \rceil = 1$. Donc b_2 devient $b_2' = b_2 - b_1 = (-81,-569)$.
  Ainsi, on a $B_1 = 1994005$ et $B_2 = 330322$ et on rentre donc dans la boucle.
- b_1 et b_2 sont intervertis, B_1 devient B_2 et on calcule $\mu = -2.45...$ d'où $\lfloor \mu \rceil = -2$. Donc b_2 devient $b_2' = b_2 + 2b_1 = (37,260)$.
  Ainsi, on a $B_1 = 330322$ et $B_2 = 68969$ donc la boucle continue.
- On a 4 étapes similaires dans notre boucle qui donnent au final : $b_1 = (-1,-4)$, $b_2 = (-2-1)$.
  On a donc $B_1 = 17$ et $B_2 = 5$: la boucle continue.
- On rentre dans la dernière itération : b_1 et b_2 sont intervertis, B_1 devient alors B_2 et on a donc $\mu = 1.2$ d'où $\lfloor \mu \rceil = 1$. On a alors b_2 qui devient $b_2' = b_2 - b_1 = (1,-3)$
  Enfin, on a $B_1 = 5$ et $B_2 = 10$: le programme termine et renvoie la base : $((-2,-1),(1,-3))$.

2.2 Étude de la complexité

Que ce soit pour cet algorithme ou pour l'algorithme LLL qui sera présenté dans la suite, on restreindra l'étude de la complexité à des réseaux inclus dans $\mathbb{Z}^m$. En effet, ce cas précis est plus simple et nous suffit pour les applications que nous en ferons.

Lemme 2.2.1. *Soient b_1, b_2 deux vecteurs indépendants de $\mathbb{Z}^2$ et $X \in \mathbb{Z}$ tel que $\|b_i\|^2 \leq X$ pour $i \in \{1,2\}$. Alors l'algorithme termine en au plus $O(\log X)$ itérations.*

Preuve. Pour une itération de l'algorithme, on note b_1 et b_2 les vecteurs après la permutation et b_1' et b_2' ceux qui suivent la permutation de l'itération suivante. Ainsi, si on note $\mu = \dfrac{\langle b_1, b_2\rangle}{B_1}$, on a que $b_1' = b_2 - \lfloor\mu\rceil b_1$ et $b_2' = b_1$. On définit aussi $\tilde{b_1}, \tilde{b_2}$ les vecteurs finaux de la seconde itération considérée et $\mu' = \dfrac{\langle b_1', b_2'\rangle}{B_1'}$.

On peut alors montrer que pour toutes les itérations autres que les deux dernières, $\left\|b_1'\right\|^2 \leq \dfrac{\|b_1\|^2}{3}$.

Supposons que $\left\|b_1'\right\|^2 > \dfrac{\|b_1\|^2}{3}$. On rappelle que $\lfloor\mu\rceil = \mu + \varepsilon$ avec $|\varepsilon| \leq \dfrac{1}{2}$ aussi :

$$|\langle b_1', b_2'\rangle| = |\langle b_2 - (\mu+\varepsilon)b_1, b_1\rangle| = |\varepsilon|\,\|b_1\|^2 < \frac{3}{2}\left\|b_1'\right\|^2$$

Donc $|\mu'| < \dfrac{3}{2}$ donc $\lfloor\mu'\rceil \in \{-1, 0, 1\}$. On va alors distinguer les cas :

Si $\lfloor\mu'\rceil = 0$, alors $\tilde{b_1} = b_1'$ et $\tilde{b_2} = b_2'$. Comme nous sommes rentrés dans cette itération, alors on a nécessairement que $\left\|b_1'\right\|^2 \leq \left\|b_2'\right\|^2$. Autrement dit, on vérifie que $\tilde{B_1} \leq \tilde{B_2}$: le programme termine.

Si $\lfloor\mu'\rceil = 1$, alors $\tilde{b_1} = b_1'$ et $\tilde{b_2} = b_2' - b_1'$ donc $\tilde{B_1} = \|b_2 - \lfloor\mu\rceil b_1\|^2 \leq \|b_2 - (\lfloor\mu\rceil + 1)b_1\|^2 = \tilde{B_2}$ par le lemme (2.1) donc le programme termine.

Si $\lfloor\mu'\rceil = -1$, alors $\tilde{b_1} = b_1'$ et $\tilde{b_2} = b_2' + b_1'$ donc $\tilde{B_1} = \|b_2 - \lfloor\mu\rceil b_1\|^2 \leq \|b_2 - (\lfloor\mu\rceil - 1)b_1\|^2 = \tilde{B_2}$ par le lemme (2.1) donc le programme termine.

Donc d'après ce qui précède, on a bien au plus $\log_3 X + 2 = O(\log X)$ itérations.

Lemme 2.2.2. *Soient $a, b, X \in \mathbb{Z}$ tels que $|a| \leq X$ et $|b| \leq X$, alors la complexité du calcul de $\lfloor\dfrac{a}{b}\rceil$ est en $O((\log X)^2)$.*

Preuve. On effectue la division euclidienne de a par b, aussi il existe une unique couple $(q, r) \in \mathbb{Z}^2$ tel que $a = bq + r$ et $0 \leq r < b$.

Donc on a que $\dfrac{a}{b} = q + \varepsilon$ avec $\varepsilon = \dfrac{r}{b} \in [0, 1[$. Donc $\lfloor\dfrac{a}{b}\rceil \in \{q, q+1\}$ et est déterminé en fonction de si $\varepsilon < \dfrac{1}{2}$ ou $\varepsilon \geq \dfrac{1}{2}$.

On a donc que la complexité du calcul de $\lfloor\dfrac{a}{b}\rceil$ est la même que celle de la division euclidienne de a par b. Or d'après la majoration de l'hypothèse, on a bien que cette complexité est en $O((\log X)^2)$.

Théorème 2.2.3. *Soient b_1, b_2 deux vecteurs indépendants de $\mathbb{Z}^2$ et $X \in \mathbb{Z}$ tel que $\|b_i\|^2 \leq X$ pour $i \in \{1,2\}$. Alors la complexité de l'algorithme de Lagrange-Gauss est en $O((\log X)^3)$.*

Preuve. D'après le lemme (2.2.1), on sait déjà que l'algorithme termine en $O(\log X)$ itérations.

Dans la boucle, à part pour le calcul de $\lfloor\mu\rceil$, toutes les affectations ont des complexités en $O(\log X)$ car les coordonnées des vecteurs sont bornées par $\sqrt{X}$. Pour ce qui est du calcul de $\lfloor\mu\rceil$, il est en complexité de $O((\log X)^2)$ par le lemme précédent et la majoration qui précède. Ceci permet alors de conclure.

Chapitre 3

Algorithme LLL

On a vu dans la partie précédente qu'il était possible de touver une base dont les vecteurs étaient de normes égales aux minima successifs en dimension 2. Premièrement, on rappelle qu'une telle base n'existe pas nécessairement en dimension supérieure, mais en plus, il est en pratique difficile de calculer ces minima. On va alors introduire une notion de base réduite dont les critères seront facilement vérifiable et qui en donnera une approximation.
Pour ceci, on s'appuira sur le travail de Arjen Lenstra, Hendrik Lenstra et László Lovász, qui donnent leurs noms au fameux algorithme LLL, pour caractériser et contruire une telle base réduite.

3.1 Base LLL réduite : définition et propriétés

Dans toute cette partie, si on a $\mathscr{B} = (b_1, ..., b_n)$ une famille libre de $\mathbb{R}^m$, on notera $\mathscr{B}^* = (b_1^*, ..., b_n^*)$ la famille issue du procédé d'orthogonalisation de Gram-Schmidt. De plus, pour tout $1 \leq j < i \leq n$, on note $\mu_{i,j} = \dfrac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}$
De plus, on continue de noter λ_i les minima successifs d'un réseau.

On va d'abord commencer par énoncer un lemme concernant la base $\mathscr{B}^*$ qui nous sera très utile pour ce qui suit :

Lemme 3.1.1. *Avec $\mathscr{B}$ et $\mathscr{B}^*$ définis comme précédemment, on a :*

1. *Pour tout $1 \leq i \leq n$, $\left\|b_i^*\right\| \leq \|b_i\|$.*
2. *Pour tout $1 \leq i \leq n$, $\langle b_i, b_i^* \rangle = \langle b_i^*, b_i^* \rangle$.*
3. *Pour tout $1 < k \leq n$, et $1 \leq j < k$, si on note $b_k' = b_k - \lfloor \mu_{k,j} \rceil b_j$ et $\mu_{k,j}' = \dfrac{\langle b_k', b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}$, alors $|\mu_{k,j}'| \leq \dfrac{1}{2}$.*

Preuve. Le point 1. a déjà été vu plus tôt. Le point 2. découle de la bilinéarité du produit scalaire et du choix des $\mu_{i,j}$:

$$\begin{aligned}\langle b_j^*, b_j^*\rangle &= \Big\langle b_j - \sum_{i=1}^{j-1} \frac{\langle b_j, b_i^*\rangle}{\langle b_i^*, b_i^*\rangle} b_i^*, b_j^*\Big\rangle \\ &= \langle b_j, b_j^*\rangle - \sum_{i=1}^{j-1} \frac{\langle b_j, b_i^*\rangle}{\langle b_i^*, b_i^*\rangle}\langle b_i^*, b_j^*\rangle \\ &= \langle b_j, b_j^*\rangle\end{aligned}$$

Pour le point 3., on écrit $\lfloor \mu_{k,j} \rceil = \mu_{k,j} + \varepsilon$ avec $|\varepsilon| \leq \frac{1}{2}$. Aussi, on a :

$$\begin{aligned}\mu'_{k,j} &= \frac{\langle b'_k, b_j^*\rangle}{\langle b_j^*, b_j^*\rangle} \\ &= \frac{\langle b_k - \lfloor \mu_{k,j} \rceil b_j, b_j^*\rangle}{\langle b_j^*, b_j^*\rangle} \\ &= \frac{\langle b_k, b_j^*\rangle}{\langle b_j^*, b_j^*\rangle} - (\mu_{k,j} + \varepsilon)\frac{\langle b_j, b_j^*\rangle}{\langle b_j^*, b_j^*\rangle} \\ &= \varepsilon\end{aligned}$$

Donc $|\mu'_{k,j}| \leq \frac{1}{2}$, ce qui conclut.

Définition 3.1.1. Soient $\mathscr{B}$ et $\mathscr{B}^*$ définies comme précédemment, pour tout $1 \leq i \leq n$, on note $B_i = \left\|b_i^*\right\|^2$. Soit $\frac{1}{4} < \delta < 1$, on dit que la base $\mathscr{B}$ est LLL réduite pour la valeur δ si elle vérifie :

1. Pour tout $1 \leq j < i \leq n$, $|\mu_{i,j}| \leq \frac{1}{2}$
2. Pour tout $2 \leq i \leq n$, $B_i \geq (\delta - \mu_{i,i-1}^2)B_{i-1}$

Remarque. Le point 2 est appelé "condition de Lovász". De plus, en général, la valeur utilisée pour δ est $\delta = \frac{3}{4}$.
On peut interpréter le point 1 comme le fait que la base de L est aussi "orthogonale que possible". En effet, travailler sur un réseau limite les valeurs possibles des $\mu_{i,j}$ et donc on cherche des vecteurs tels que les $\mu_{i,j}$ soient les plus proches de zéro possible.

Maintenant que l'on a défini ce qu'était une base réduite, on va étudier tout l'intérêt d'une telle construction (notons que pour l'instant rien n'affirme qu'une telle base existe). L'essentiel des propriétes de notre base réduite (pour $\delta = \frac{3}{4}$) sera contenu dans le théorème (3.1.4). Pour montrer ce théorème, on va utiliser 2 lemmes :

Lemme 3.1.2. *Soient $\mathscr{B} = (b_1, ..., b_n)$ une base LLL réduite d'un réseau R pour $\delta = \frac{3}{4}$ et $\mathscr{B}^*$ définie comme précédemment. Alors :*

1. *Pour tout $1 \leq j \leq i \leq n$, on a : $B_j \leq 2^{i-j} B_i$*
2. *Pour tout $1 \leq i \leq n$, on a : $B_i \leq \|b_i\|^2 \leq (\frac{1}{2} + 2^{i-2}) B_i$*
3. *Pour tout $1 \leq j \leq i \leq n$, on a : $\left\|b_j\right\| \leq 2^{\frac{i-1}{2}} \left\|b_i^*\right\|$*

Preuve.

1. La condition de Lovász nous donne que pour tout $1 < i \leq n$, $B_i \geq \frac{1}{2} B_{i-1}$. Ainsi on a le résultat par une réccurence immédiate.
2. On rappelle que pour tout $1 \leq i \leq n$, $b_i = b_i^* + \sum_{j=1}^{i-1} \mu_{i,j} b_j^*$. Ainsi, on a :

$$\begin{aligned} \|b_i\|^2 &= \langle b_i, b_i \rangle \\ &= \left\langle b_i^* + \sum_{j=1}^{i-1} \mu_{i,j} b_j^*, b_i^* + \sum_{j=1}^{i-1} \mu_{i,j} b_j^* \right\rangle \\ &= B_i + \sum_{j=1} i - 1 \mu_{i,j}^2 B_j \end{aligned}$$

   Donc par le point 1. et en rappelant que $\mu_{i,j}^2 \leq \frac{1}{4}$ pour tout i, j, on a que $\|b_i\|^2 \leq B_i + \frac{1}{4} \sum_{j=1}^{i-1} 2^{i-j} B_i$.

   On conclut donc grâce à une somme d'une suite géométrique : $\|b_i\|^2 \leq B_i(1 + \frac{1}{4}(2^i - 2)) \leq B_i(\frac{1}{2} + 2^{i-2})$
3. Par le point précédent, on a en particulier que pour tout $1 \leq j \leq i \leq n$, $\left\|b_j\right\|^2 \leq 2^{j-1} B_j$. On reprend alors le point 1. :

$$\left\|b_j\right\|^2 \leq 2^{j-1} B_j \leq 2^{j-1} 2^{i-j} B_i = 2^{i-1} B_i$$

   Ceci donne bien : $\left\|b_j\right\| \leq 2^{\frac{i-1}{2}} \left\|b_i^*\right\|$

Lemme 3.1.3. *Soient $\mathscr{B}$ et $\mathscr{B}^*$ définies comme précédemment. Pour tout $x = \sum_{k=1}^{n} x_k b_k \in R \backslash \{0\}$, on peut noter k_0 le plus grand $k \in \{1, \ldots, n\}$ tel que $x_k \neq 0$. Alors on a que : $\|x\| \geq \left\|b_{k_0}^*\right\|$*

En particulier, on vérifie que :

$$\lambda_1 \geq \min_{1 \leq i \leq n} \left\|b_i^*\right\|$$

Preuve. Soit $x = \sum_{k=1}^{n} x_k b_k \in R$, on note k_0 tel dans l'énoncé. Autrement dit, $x = \sum_{k=1}^{k_0} x_k b_k \in R$. On rappelle que par définition, $b_{k_0}^*$ est orthogonal à $(b_1, \ldots, b_{k_0 - 1})$, ainsi par le point 2. du lemme (3.1.1), on a :

$$|\langle x, b_{k_0}^* \rangle| = |x_{k_0} \langle b_{k_0}, b_{k_0}^* \rangle| = |x_{k_0} \langle b_{k_0}^*, b_{k_0}^* \rangle| = |x_{k_0}| \left\|b_{k_0}^*\right\|^2 \geq \left\|b_{k_0}^*\right\|^2$$

D'autre part, par Cauchy-Schwarz, on a que $|\langle x, b_{k_0}^* \rangle| \leq \|x\| \left\|b_{k_0}^*\right\|$, on a donc le résultat.

Théorème 3.1.4. *Soient $\mathscr{B} = (b_1, ..., b_n)$ une base LLL réduite d'un réseau R pour $\delta = \frac{3}{4}$ et $\mathscr{B}^*$ définie comme précédemment. Alors :*

1. $\|b_1\| \leq 2^{\frac{n-1}{2}} \lambda_1$
2. *Pour tout* $1 \leq j \leq i \leq n$, *on a :* $\left\|b_j\right\| \leq 2^{\frac{n-1}{2}} \lambda_i$
3. *Pour tout* $1 \leq i \leq n$, *on a :* $2^{\frac{1-i}{2}} \lambda_i \leq \|b_i\| \leq 2^{\frac{n-1}{2}} \lambda_i$
4. $\det(R) \leq \prod_{i=1}^{n} \|b_i\| \leq 2^{\frac{n(n-1)}{4}} \det(R)$
5. $\|b_1\| \leq 2^{\frac{n-1}{4}} \det(R)^{1/n}$

Preuve. 1. Pour tout $1 \leq i \leq n$, on a par le point 1. du lemme (3.1.2) que $\left\|b_i^*\right\| \geq 2^{\frac{i-1}{2}} \|b_1\|$. Donc grâce au lemme (3.1.3), on a :

$$\lambda_1 \geq \min_{1 \leq i \leq n} \left\|b_i^*\right\| \geq \min_{1 \leq i \leq n} 2^{\frac{1-i}{2}} \|b_1\| \geq 2^{\frac{1-n}{2}} \left\|b_1^*\right\|$$

En remarquant que $b_1^* = b_1$, on a le résultat.

2. Soient $i \in \{1, \ldots, n\}$ et $(w_1, \ldots, w_i) \in R$ une famille libre telle que $\max_{1 \leq j \leq i} \left\|w_j\right\| = \lambda_i$.
Pour tout $j \in \{1, \ldots, i\}$, on pose $k(j)$ l'entier k_0 du lemme (3.1.3). Ainsi, $\left\|w_j\right\| \geq \left\|b_{k(j)}^*\right\|$.
On suppose sans perte de généralités que $k(1) \leq \cdots \leq k(i)$. De plus on a nécessairement que $j \leq k(j)$. En effet dans le cas contraire, alors on aurait que $(w_1, \ldots, w_j) \in \text{Vect}(b_1, \ldots, b_{j-1})$ et cette famille ne serait donc pas libre. Ceci donne donc :

$$\left\|b_j\right\| \leq 2^{\frac{k(j)-1}{2}} \left\|b_{k(j)}^*\right\| \leq 2^{\frac{n-1}{2}} \left\|w_j\right\| \leq 2^{\frac{n-1}{2}} \lambda_i$$

3. D'après ce qui précède, on a déjà que $\|b_i\| \leq 2^{\frac{n-1}{2}} \lambda_i$.
Comme $\mathscr{B}$ est une base de R, alors pour tout $i \in \{1, \ldots, n\}$, la famille $(b_1, \ldots, b_i)$ est libre. Ainsi par définition, on a : $\lambda_i \leq \max_{1 \leq j \leq i} \left\|b_j\right\|$.
De plus d'après le lemme (3.1.2), on a que pour tout $j \in \{1, \ldots, i\}$, $2^{\frac{1-i}{2}} \left\|b_j\right\| \leq \left\|b_i^*\right\| \leq \|b_i\|$. Ceci conclut.
4. On a vu en introduction que $\prod_{i=1}^{n} \|b_i\| \geq \det(R) = \prod_{i=1}^{n} \left\|b_i^*\right\|$. On a donc par le lemme (3.1.2) :
$$\det(R) \leq \prod_{i=1}^{n} \|b_i\| \leq \prod_{i=1}^{n} 2^{\frac{i-1}{2}} \left\|b_i^*\right\| = 2^{\frac{n(n-1)}{4}} \det(R)$$
5. De la même façon, comme pour tout $i \in \{1, \ldots, n\}$, on vérifie que $\|b_1\| \leq 2^{\frac{i-1}{2}} \left\|b_i^*\right\|$, alors on a :
$$\|b_1\|^n \leq 2^{\frac{n(n-1)}{4}} \det(R)$$
Le résultat s'ensuit.

3.2 Étude de l'algorithme

Comme pour le précédent algorithme, on va maintenant voir par quel moyen il est possible de trouver pour tout réseau R une base LLL réduite. Ainsi l'algorithme LLL a pour but, à partir d'une base quelconque, de construire une nouvelle base qui sera réduite, ce qui permet d'avoir d'une part, l'existence pour tout réseau R d'une telle base, mais surtout une construction explicite nécessaire à son utilisation pratique.
Le principe de l'algorithme est de constuire de nouveaux vecteurs qui permettent vérifier que $|\mu_{i,j}| \leq \frac{1}{2}$. Ensuite, à la manière de l'algorithme de Lagrange-Gauss, on va intervertir les vecteurs un à un et répeter les opérations qui précèdent de façon à réduire petit à petit chaque vecteur jusqu'à ce que la base vérifie la condition de Lovàsz.

Entrées : Une base $\mathscr{B} = (b_1, ..., b_n)$ de R

Sorties : Une base LLL réduite $(b_1, ..., b_n)$ de R

Évaluer base $(b_1^*, ..., b_n^*)$ de Gram-Schmidt et les $\mu_{i,j}$ pour $1 \leq j < i \leq n$
Évaluer $B_i = \langle b_i^*, b_i^* \rangle = \left\| b_i^* \right\|^2$ pour $1 \leq i \leq n$
$k \leftarrow 2$
tant que $k \leq n$ **faire**
    pour *j allant de* $k-1$ *à* 1 **faire**
        $b_k \leftarrow b_k - \lfloor \mu_{k,j} \rceil b_j$
        Réévaluer les $\mu_{k,i}$ pour $1 \leq i \leq j$
    fin
    Si $B_k \geq (\delta - \mu_{k,k-1}^2) B_{k-1}$ **Alors**
        $k = k + 1$
    Fin
    Sinon
        $b_k \longleftrightarrow b_{k-1}$
        Réévaluer $b_k, b_{k-1}, B_k, B_{k-1}$, les $\mu_{k,j}$ pour $1 \leq j < k$ et les $\mu_{i,k}, \mu_{i,k-1}$ pour $k < i \leq n$
        $k \leftarrow \max(2, k-1)$
    Fin
fin
Renvoyer $(b_1, ..., b_n)$

Algorithme 2 : Algorithme LLL

Afin de bien saisir ce qu'il se passe, on va dérouler l'algorithme à la main sur un exemple (pour $\delta = \frac{3}{4}$)

et on prouvera ensuite qu'il correspond bien à nos attentes.

Exemple. On considère la base $\mathscr{B} = (b_1, b_2, b_3) = \left(\begin{pmatrix}-1\\5\\0\end{pmatrix}, \begin{pmatrix}2\\5\\0\end{pmatrix}, \begin{pmatrix}8\\6\\16\end{pmatrix}\right)$. Les vecteurs de Gram-Schmidt associés sont $b_1^* = b_1, b_2^* = \begin{pmatrix}75/26\\15/26\\0\end{pmatrix}, b_3^* = \begin{pmatrix}0\\0\\16\end{pmatrix}$

Initialement , $k = 2$, on a $\lfloor \mu_{2,1} \rceil = \lfloor \frac{23}{26} \rceil = 1$. Ainsi, b_2 devient $\begin{pmatrix}2\\5\\0\end{pmatrix} - \begin{pmatrix}-1\\5\\0\end{pmatrix} = \begin{pmatrix}3\\0\\0\end{pmatrix}$, on actualise les valeurs associées : $\mu_{2,1} = \frac{-3}{26}$ d'où $\mu_{2,1}^2 = \frac{9}{26^2}$, par ailleurs, la base (b_1^*, b_2^*, b_3^*) ne change pas.

Donc $(\frac{3}{4} - 0,013314)B_1 \geq 19 \geq 9 \geq \frac{75^2 + 15^2}{26^2} = B_2$. Ainsi, on échange b_1 et b_2, donc à ce stade, notre base est : $(b_1, b_2, b_3) = \left(\begin{pmatrix}3\\0\\0\end{pmatrix}, \begin{pmatrix}-1\\5\\0\end{pmatrix}, \begin{pmatrix}8\\6\\16\end{pmatrix}\right)$ et sa base orthogonale associée est $(b_1^*, b_2^*, b_3^*) = \left(\begin{pmatrix}3\\0\\0\end{pmatrix}, \begin{pmatrix}0\\5\\0\end{pmatrix}, \begin{pmatrix}0\\0\\16\end{pmatrix}\right)$.

On a toujours $k = 2$ et $\lfloor \mu_{2,1} \rceil = \lfloor \frac{1}{3} \rceil = 0$ ainsi b_2 est inchangé donc (b_1, b_2, b_3) et (b_1^*, b_2^*, b_3^*) ne changent pas. De plus, $(\frac{3}{4} - \frac{1}{9})B_1 \leq 6 \leq 25 = B_2$ donc on passe à l'étape suivante.

On passe maintenant à $k = 3$, on a $\lfloor \mu_{3,2} \rceil = \lfloor \frac{30}{26} \rceil = 1$ d'où b_3 devient $\begin{pmatrix}8\\6\\16\end{pmatrix} - \begin{pmatrix}-1\\5\\0\end{pmatrix} = \begin{pmatrix}9\\1\\16\end{pmatrix}$. Ensuite, on a $\lfloor \mu_{3,1} \rceil = \lfloor \frac{24}{9} \rceil = 3$ et b_3 devient $\begin{pmatrix}9\\1\\16\end{pmatrix} - \begin{pmatrix}8\\0\\0\end{pmatrix} = \begin{pmatrix}0\\1\\16\end{pmatrix}$. Par ailleurs, durant ces deux dernières étapes, la base (b_1^*, b_2^*, b_3^*) ne change pas.

Enfin, $(\frac{3}{4} - \mu_{3,2}^2)B_2 = (\frac{3}{4} - \frac{25}{26^2})26 \leq 6 \leq 256 = B_3$ donc l'algorithme termine et renvoit la base $\mathscr{B} = \left(\begin{pmatrix}3\\0\\0\end{pmatrix}, \begin{pmatrix}-1\\5\\0\end{pmatrix}, \begin{pmatrix}0\\1\\16\end{pmatrix}\right)$ de base orthognale associée $\mathscr{B}^* = \left(\begin{pmatrix}3\\0\\0\end{pmatrix}, \begin{pmatrix}0\\5\\0\end{pmatrix}, \begin{pmatrix}0\\0\\16\end{pmatrix}\right)$. Il est clair que la base vérifie la condition de Lovász, on peut tout de même vérifier la condition sur les $\mu_{i,j}$ qui est n'est pas évidente a priori :

$\mu_{2,1} = \frac{-3}{25}$, $\mu_{3,2} = \frac{5}{256}$ et $\mu_{3,1} = 0$, ainsi leurs valeurs absolues sont toutes inférieures à $\frac{1}{2}$.

Donc la base $\mathscr{B}$ renvoyée est bien LLL réduite.

Remarque. On note que la base orthogonale ne change que lorsque des vecteurs sont intervertis, ce qui explique pourquoi on ne réévalue que les $\mu_{k,i}$ dans la boucle **for**. Pour le justifier, on va montrer que les seules valeurs qui varient tout au long de l'algorithme sont celles explicitiement réévaluées.

On commence par étudier la boucle **for** :

Lemme 3.2.1. *Soient $\mathscr{B}$ une base de R et $k \in \{2, ..., n\}$. Pour tout $1 \leq j < k$, si on note $b_k' = b_k - \lfloor \mu_{k,j} \rceil b_j$, alors $b_k'^* = b_k^*$.*

Preuve. Comme dit en introduction, le passage de b_k à b_k^* peut être vu tel : $b_k^* = \pi_k(b_k)$ où π_k est la projection orthogonale sur $\text{Vect}(b_1, ..., b_{k-1})^\perp$.

Ainsi, pour tout $q \in \mathbb{Z}$, on a $b_k'^* = \pi_k(b_k - qb_j) = \pi_k(b_k) - q\pi_k(b_j) = b_k^* + 0$, ce qui conclut.

On va maintenant justifier comment les variables changent durant le reste de l'algorithme, ce qui nous sera très utile pour la preuve de la terminaison :

Lemme 3.2.2. *Lorsque b_k et b_{k-1} sont intervertis, alors on a :*

1. *Pour tout $1 \leq i < k-1$ et $k < i < n$, les b_i^* sont inchangés.*
2. *b_{k-1}^* devient $b_k^* + \mu_{k,k-1} b_{k-1}^*$ et B_{k-1} devient $B'_{k-1} = B_k + \mu_{k,k-1}^2 B_{k-1}$*
3. *b_k^* devient $\frac{B_k}{B'_{k-1}} b_{k-1}^* - (\mu_{k,k-1} \frac{B_{k-1}}{B'_{k-1}}) b_k^*$ et B_k devient $B'_k = \frac{B_{k-1} B_k}{B'_{k-1}}$*

Preuve. On note b_i' les nouveaux vecteurs après intervertion de de b_k et b_{k-1}, c'est à dire que la plupart des b_i' sont inchangés et $b'_k = b_{k-1}$ et $b'_{k-1} = b_k$, $b_i'^*$, $\mu'_{i,j}$ les nouvelles valeurs du procédé de Gram-Schmidt associés et $B'_i = \left\| b_i'^* \right\|^2$

Il est clair que $b_i'^* = b_i^*$ pour $1 \leq i < k-1$ et de la même façon, $\mu'_{i,j} = \mu_{i,j}$ pour $1 \leq j < i < k-1$.

Encore une fois, pour tout $i > k$, on écrit $b_i^* = \pi_i(b_i)$ comme dans le lemme (3.2.1). Ceci suffit alors pour voir que $b_i = b_i^*$.

De plus, pour tout $1 \leq j < k-1$, on a $\mu'_{k-1,j} = \mu_{k,j}$ et $\mu'_{k,j} = \mu_{k-1,j}$ par définition.

On va utiliser ceci pour montrer le point 2.

$$\begin{aligned} b_{k-1}'^* = b'_{k-1} - \sum_{j=1}^{k-2} \mu'_{k-1,j} b_j'^* &= b_k - \sum_{j=1}^{k-2} \mu_{k,j} b_j^* \\ &= b_k^* + \sum_{j=1}^{k-1} \mu_{k,j} b_j^* - \sum_{j=1}^{k-2} \mu_{k,j} b_j^* \\ &= b_k^* + \mu_{k,k-1} b_{k-1}^* \end{aligned}$$

De plus, comme b_k^* et b_{k-1}^* sont orthogonaux, on a que $B'_{k-1} = B_k + \mu_{k,k-1}^2 B_{k-1}$ par Pythagore. Le point 3 se démontre de la même façon :

$$\begin{aligned} b_k'^* = b'_k - \sum_{j=1}^{k-1} \mu'_{k,j} b_j'^* = b_{k-1} - \sum_{j=1}^{k-2} \mu_{k-1,j} b_j^* - \frac{\langle b_{k-1}, b_k^* + \mu_{k,k-1} b_{k-1}^* \rangle}{B'_{k-1}} (b_k^* + \mu_{k,k-1} b_{k-1}^*) \\ = b_{k-1}^* - \mu_{k,k-1} \frac{\langle b_{k-1}, b_{k-1}^* \rangle}{B'_{k-1}} (b_k^* + \mu_{k,k-1} b_{k-1}^*) \\ = (1 - \mu_{k,k-1}^2 \frac{B_{k-1}}{B'_{k-1}}) b_{k-1}^* - (\mu_{k,k-1} \frac{B_{k-1}}{B'_{k-1}}) b_k^* \end{aligned}$$

On conclut pour $b_k'^*$ par le point précédent : $1 - \mu_{k,k-1}^2 \frac{B_{k-1}}{B'_{k-1}} = \frac{B'_{k-1} - \mu_{k,k-1}^2 B_{k-1}}{B'_{k-1}} = \frac{B_k}{B'_{k-1}}$.

Ainsi, on a donc par Pythagore que $B'_k = \dfrac{B_{k-1}B_k(B_k + \mu^2_{k,k-1}B_{k-1})}{B'^2_{k-1}} = \dfrac{B_{k-1}B_k}{B'_{k-1}}$ par le point 2.

On va alors utiliser ce lemme pour montrer la termnaison :

Terminaison : Pour montrer la terminaison, on va estimer le nombre d'itérations dans le pire des cas de la boucle while. Si on note N le nombre d'itérations, on a que $N = n + 2r$ avec r le nombre d'intervertions $b_k \leftrightarrow b_{k-1}$. De fait, majorer N revient à majorer m. Pour $1 < i \leq n$, on note $d_i = \prod_{j=1}^{i} B_j$.

On va alors étudier $D = \prod_{i=1}^{n} d_i$. D reste constante pendant la boucle **for** vu que les $(b^*_1, ..., b^*_n)$ ne varie pas. On va alors étudier comment D varie lorsque on échange deux vecteurs, et ce grâce au lemme précédent. Soit $k \in \{2, ..., n\}$, on va montrer qu'à force d'intervertir b_k et b_{k-1}, on aura $B_k \geq (\delta - \mu^2_{k,k-1})B_{k-1}$ et donc que k prend la valeur $k+1$.
On suppose alors que l'on échange b_{k-1} et b_k, c'est à dire que $B_k < (\delta - \mu^2_{k,k-1})B_{k-1}$. On note B'_{k-1} et B'_k les nouvelles valeurs de B_{k-1} et B_k ; d'après le lemme (3.2.2), $B'_{k-1} = B_k + \mu^2_{k,k-1}B_{k-1}$ et $B'_k = \dfrac{B_{k-1}B_k}{B'_{k-1}}$. On note de plus d'_i les nouvelles valeurs de d_i.

Il est clair que pour tout $1 \leq i < k-1$, $d'_i = d_i$ vu que B_{k-1} et B_k n'interviennent pas. De plus, on a que $B'_{k-1}B'_k = B_kB_{k-1}$, donc pour tout $k \leq i \leq n$, $d'_i = d_i$ car le produit fait apparaître B_{k-1} et B_k. Enfin, $B'_{k-1} = B_k + \mu^2_{k,k-1}B_{k-1} < (\delta - \mu^2_{k,k-1})B_{k-1} + \mu^2_{k,k-1}B_{k-1}$ par hypothèse.
On a donc $d'_{k-1} < \delta d_{k-1} < d_{k-1}$ et donc D décroît strictement lorsque deux vecteurs sont échangés. Or, le nombre de valeurs possibles de D étant fini, on a bien que le programme termine.

Correction : Il est clair que la condition de Lovàsz est nécessairement vérifiée si le programme termine, il reste donc à vérifier que les $\mu_{j,i}$ sont assez petits. On va commencer par montrer que pour tout $k \in \{2, ..., n\}$, passer de l'étape k à $k+1$ permet de vérifer que $|\mu_{k,j}| \leq \dfrac{1}{2}$ pour tout $1 \leq j \leq k$.
Si on est dans l'itération où k passe à $k+1$, alors les $\mu_{j,i}$ ne sont modifiés que dans la boucle **for**, il suffit donc d'étudier comment les $\mu_{j,i}$ varient dans cette boucle. Soit $j \in \{1, ..., k-1\}$, on note $b'_k = b_k - \lfloor \mu_{k,j} \rceil b_j$ la nouvelle valeur de b_k pour cette itération de la boucle **for** et $\mu'_{k,i}$ les coefficient de Gram-Schmidt associés pour $i \in \{1, ..., k-1\}$. D'après le lemme (3.1.1), $|\mu'_{k,j}| \leq \dfrac{1}{2}$, de plus si on a $j < i \leq k-1$, alors on a :

$$\mu'_{k,i} = \frac{\langle b'_k, b^*_i \rangle}{B_i} = \frac{\langle b_k, b^*_i \rangle}{B_i} - \lfloor \mu_{k,j} \rceil \frac{\langle b_j, b^*_i \rangle}{B_i} = \mu_{k,i}$$

En effet, comme $i > j$, alors $\langle b_j, b^*_i \rangle = 0$ pour les mêmes raisons que $b'^*_k = b^*_k$ *(cf lemme (3.2.1))*.
La boucle **for** étant parcourue par j qui décroit, on a bien que lorsque celle-ci termine, $|\mu_{k,j}| \leq \dfrac{1}{2}$ pour tout

$j \in \{1, ..., k-1\}$.

Enfin, pour conclure, il suffit de remarquer que si b_k et b_{k-1} sont échangés, les seules valeurs $\mu_{i,j}$ modifiées sont les $\mu_{k,j}$ pour $1 \leq j < k$ et les $\mu_{i,k}, \mu_{i,k-1}$ pour $k < i \leq n$. En effet, vu que $k \leftarrow k-1$, ces valeurs seront toutes modifiées correctement dans des itérations ultérieures par ce qui précède.

3.3 Complexité de l'algorithme LLL

L'une des forces de l'algorithme LLL vient justement de sa complexité. En effet, celui-ci permet de résoudre des problèmes difficiles comme par exemple la factorisation de polynômes dans $\mathbb{Q}[X]$ en polynômes irréductibles (voir [2]) et ceci en un temps polynomial.

Dans cette partie, comme nous l'avions fait pour l'algorithme de Lagrange-Gauss, on ne s'attardera qu'à des réseaux inclus dans $\mathbb{Z}^m$.

Lemme 3.3.1. *Soient R un réseau de base $\mathscr{B} = (b_1, \ldots, b_n) \in \mathbb{Z}^m$ et $X \in \mathbb{Z}$ tel que pour tout $i \in \{1, \ldots, n\}$, $\|b_i\|^2 \leq X$. Alors pour tout $\frac{1}{4} < \delta < 1$, l'algorithme LLL termine en $O(n^2 \log X)$ itérations.*

Preuve. On reprend la preuve de la terminsaison de l'algorithme. D'après ce que l'on disait et les hypothèses, on a que $D \leq X^{\frac{n(n+1)}{2}}$. De plus à chaque itération ou l'on augmente pas k, la nouvelle valeur de D, notée D', vérifie : $D' < \delta D$. Donc comme $D \in \mathbb{Z}$ [1], on a au plus $O(\log_\delta(X^{\frac{n(n+1)}{2}}) = O(n^2 \log X)$ itérations.

Maintenant que l'on dispose du nombre d'itérations, comme les opérations se font sur $\mathbb{Q}$ durant la totalité de l'algorithme, il nous faut connaître la taille des entiers concernés pour donner la complexité. En effet, si $r = \frac{p}{q} \in \mathbb{Q}$ vérifie que $|r| \leq X \in \mathbb{Z}$, on n'a a priori aucune borne sur la taille de p et q.

Pour donner une telle borne, on va d'abord devoir démontrer le lemme suivant :

Lemme 3.3.2. *Soient $\mathscr{B}$ et $\mathscr{B}^*$ définies comme précédemment. On rappelle que pour tout $i \in \{1, \ldots, n\}$, $d_i = \prod_{j=1}^{i} B_j$ (avec $d_0 = 1$). De plus, on note $B_{(i)}$ la matrice dont les colonnes sont les b_j pour $1 \leq j \leq i$. On vérifie alors :*

1. *Pour tout $i \in \{1, \ldots, n\}$, on a : $d_i = \det(B_{(i)}^T B_{(i)})$*
2. *Pour tout $i \in \{1, \ldots, n\}$, on a : $d_{i-1} b_i^* \in R$.*
3. *Pour tout $1 \leq j < i \leq n$, on a : $d_j \mu_{i,j} \in \mathbb{Z}$.*

Preuve. Premièrement, on note simplement $R_{(i)}$ le réseau de base $(b_1, \ldots, b_i)$ d'où $d_i = \prod_{k=1}^{i} \|b_k^*\|^2 = \det(R_{(i)})^2$ par le lemme (1.1.3). Aussi par la proposition (1.1.2), on a que $\det(R_{(i)})^2 = \det(B_{(i)}^T B_{(i)})$ ce qui donne bien : $d_i = \det(B_{(i)}^T B_{(i)})$.

1. Les d_i sont tous entiers car le réseau ici considéré est inclus dans $\mathbb{Z}^m$

Deuxièmement, on a que pour tout $i \in \{1,\dots,n\}$, $b_i^* \in \text{Vect}(b_1,\dots,b_i)$ et $\langle b_i^*, b_i^*\rangle = \langle b_i, b_i^*\rangle$ donc il existe des coefficients[2] $a_{i,1},\dots,a_{i,i-1} \in \mathbb{R}$ tels que $b_i^* = b_i - \sum_{j=1}^{i-1} a_{i,j} b_j$.

Ceci donne alors que pour tout $i \geq 2$ et $k \in \{1,\dots,i-1\}$, $\langle b_k, b_i\rangle = \sum_{j=1}^{i-1} a_{i,j}\langle b_k, b_j\rangle$. Si on note respectivement $\underline{a_{i-1}}$ et $\underline{s_{i-1}}$ les vecteurs colonne de taille $i-1$ et de coordonnées $(a_{i,j})$ et $(\langle b_i, b_k\rangle)$, on a :

$$\underline{s_{i-1}} = M_{(i-1)}\underline{a_{i-1}} \quad \text{avec } M_{(i-1)} = B_{(i-1)}^T B_{(i-1)}$$

Comme $\det(M_{(i-1)}) = d_{i-1} > 0$, alors on peut noter $M_{(i-1)}^{-1} = \frac{1}{d_{i-1}}\text{Com}(M_{(i-1)})^T$. Comme les coefficients de $M_{(i-1)}$ sont dans $\mathbb{Z}$, ceux de sa comatrice aussi. Ainsi, comme $d_{i-1}\underline{a_{i-1}} = \text{Com}(M_{(i-1)})\underline{s_{i-1}}$, on a que les $d_{i-1}a_{i,j} \in \mathbb{Z}$ donc $d_{i-1}b_i^* \in R$.

Enfin pour tout $1 \leq j < i \leq n$, on remarque que $d_j = d_{j-1}B_j$ donc :

$$d_j\mu_{i,j} = d_{j-1}B_j\frac{\langle b_i, b_j^*\rangle}{B_j} = \langle b_i, d_{j-1}b_j^*\rangle \in \mathbb{Z}$$

Lemme 3.3.3. *Soient R un réseau de base $\mathscr{B} = (b_1,\dots,b_n) \in \mathbb{Z}^m$ et $X \in \mathbb{Z}$ tel que pour tout $i \in \{1,\dots,n\}$, $\|b_i\|^2 \leq X$. Alors les opérations de l'algorithme LLL s'effectue sur des entiers de taille $O(n\log X)$.*

Preuve. On rappelle que l'on avait pu montrer dans la terminaison de l'algorithme LLL que d'une itération à une autre, les B_i ne pouvaient que diminuer. Ainsi on a bien d'après le point 1. du lemme (3.1.1) que pour tout $i \in \{1,\dots,n\}$, $\left\|b_i^*\right\| \leq \sqrt{X}$. Aussi, d'après le lemme précédent, comme $d_{i-1}b_i^* \in \mathbb{Z}^m$ avec $d_{i-1} \in \mathbb{Z}_{\geq 1}$, on a que les coefficients de b_i^* s'écrivent $\frac{a_{i,j}}{d_{i-1}} \in \mathbb{Q}$ et vérifient $|\frac{a_{i,j}}{d_{i-1}}| \leq \sqrt{X}$ avec les $a_{i,j} \in \mathbb{Z}$. Or comme $d_{i-1} \leq X^{i-1}$ alors les $|a_{i,j}|$ sont majorés par X^i. Ces entiers là sont donc bien de taille $O(n\log X)$.

On va maintenant majorer la norme des b_i au cours de l'algorithme. Pour ceci, il suffit de s'intéresser à la boucle **for** car dans la seconde partie de l'algorithme, les b_i ne sont potentiellement que permutés et donc les normes ne varient pas.

Lorsque l'algorithme commence, les normes des b_i sont déjà majorées par X. De plus pour tout $i \in \{1,\dots,n\}$, b_i n'est modifié que si la variable k de l'algorithme atteint la valeur i. Aussi dans ce contexte, on a montré dans la correction que pour tout $1 \leq j < i$, on a $|\mu_{i,j}| \leq \frac{1}{2}$. On a donc par Pythagore :

$$\|b_i\|^2 = \left\|b_i^* + \sum_{j=1}^{i-1}\mu_{i,j}b_j^*\right\|^2 = \left\|b_i^*\right\|^2 + \sum_{j=1}^{i-1}\mu_{i,j}^2\left\|b_j^*\right\|^2$$

On combine alors la majoration sur les $|\mu_{i,j}|$ et celle des $\left\|b_j^*\right\|$ d'où $\|b_i\|^2 \leq nX$. Ainsi, comme tout au

2. Il faut faire attention au fait que les $a_{i,j}$ ici ne sont pas les $\mu_{i,j}$ habituels

long de l'algorithme, les coefficients des b_i sont entiers, alors ceux-ci sont bien majorés par nX et donc de taille $O(n \log X)$.

Enfin, comme pour tout $1 \leq j < i \leq n$, on vérifie $\|b_i\|^2 \leq nX$, $d_{j-1} \leq X^{j-1}$ et $d_j \in \mathbb{Z}_{\geq 1}$, alors on a :

$$\mu_{i,j}^2 = \frac{|\langle b_i, b_j^* \rangle|^2}{B_j^2} \leq \frac{\|b_i\|^2}{B_j} = \frac{d_{j-1}}{d_j} \|b_i\|^2 \leq X^{j-1} \|b_i\|^2 \leq nX^j$$

Donc par le même procédé que pour les b_i^*, comme $d_i \mu_{i,j} \in \mathbb{Z}$, alors $\mu_{i,j} = \dfrac{\alpha_{i,j}}{d_j} \in \mathbb{Q}$ avec $\alpha_{i,j} \in \mathbb{Z}$ et $|\alpha_{i,j}|^2 \leq nX^{3j}$. S'ensuit le résultat.

Théorème 3.3.4. *Soient R un réseau de base $\mathscr{B} = (b_1, \ldots, b_n) \in \mathbb{Z}^m$ et $X \in \mathbb{Z}$ tel que pour tout $i \in \{1, \ldots, n\}$, $\|b_i\|^2 \leq X$. Alors pour tout $\dfrac{1}{4} < \delta < 1$, la complexité de l'algorithme LLL est en $O(n^5 m (\log X)^3)$.*

Preuve. On rappelle sans preuve que la complexité de l'algorithme de Gram-Schmidt permettant d'évaluer les b_i^* est en complexité $O(n^4 m (\log X)^2)$ (voir théorème (17.3.4) de [1]) donc le calcul des b_i^* est, comme on le verra, négligeable devant le reste de l'algorithme. D'après le lemme (3.3.1), le programme termine en $O(n^2 \log X)$. De plus à chaque boucle, on a au plus n opérations sur des vecteurs de taille m ce qui donne $O(n^3 m \log X)$ opérations. Enfin les entiers des opérations étant de taille $O(n \log X)$ et l'opérations la plus coûteuse étant le calcul de la partie entière des $\mu_{k,j}$ qui est en $O(n^2 (\log X)^2)$ d'après le lemme (2.2.2), ceci donne une complexité en $O(n^5 m (\log X)^3)$.

Remarque. On peut réduire cette complexité à $O(mn^{4+\varepsilon} (\log X)^{2+\varepsilon})$ pour $\varepsilon > 0$ en utilisant des algorithmes de multiplication rapide.

Chapitre 4

Applications de l'algorithme LLL

4.1 Problème du sac-à-dos et cryptosystème de Merkle-Hellman

La réduction de réseaux a de nombreux intérêts pratiques et notamment en cryptographie. Dans cette partie, on va voir comment le fait de pouvoir trouver un vecteur de petite norme permet de facilement résoudre certains problème du sac-à-dos et donc de casser le cryptosystème de Merkle-Hellman qui est basé dessus.

Définition 4.1.1. Soient $n \in \mathbb{N}^*$, $a_1, ..., a_n$ des entiers et $S \geq 0$ tels qu'il existe $c_1, ..., c_n \in \{0, 1\}$ vérifiant :

$$\sum_{i=1}^{n} c_i a_i = S$$

Le problème du sac-à-dos consiste à trouver les c_i étant donnés S et les a_i.

Ce problème est en géneral très difficile (plus précisément, il a été montré NP-complet dans le cas général). Cependant, il existe un situation pour laquelle trouver une solution devient beaucoup plus simple.

Proposition 4.1.1. *Si la suite $a_1, ..., a_n$ du problème du sac à dos est supercroissante, c'est-à-dire que pour tout $1 \leq k < n$, $a_{k+1} > \sum_{i=1}^{k} a_i$, alors le problème se résout en au plus n étapes.*

Preuve. On suppose la suite $a_1, ..., a_n$ supercroissante et soit $S \geq 0$ tel dans le problème du sac-à-dos. On note $n_0 \in \{1, ..., n\}$ l'entier tel que $a_{n_0} \leq S < a_{n_0+1}$ et $n_0 = n$ si $S \geq a_n$. Supposons que $c_{n_0} = 0$, alors même si tous les $c_i = 1$ pour $1 \leq i < n_0$, on aura $\sum_{i=1}^{n_0-1} a_i < a_{n_0} \leq S$ donc nécessairement $c_{n_0} = 1$. On fait la même chose pour $S - a_{n_0}$ et ainsi de suite. On a donc bien que l'on peut trouver les c_i en au plus n étapes.

Le fait de disposer d'un contexte pour lequel ce problème est facile à résoudre est plus qu'important car c'est sur ce seul fait que repose le cryptosystème de Merkle-Hellman.

En effet, l'idée derrière ce processus de chiffrement est de partir d'un sac-à-dos "facile" qui reste privé afin d'en construire un qui ne le sera pas et qui sera rendu publique.

Protocole cryptographique de Merkle-Hellman

Alice et Bob souhaitent communiquer sans que leurs messages ne puissent être lus par un individu auxiliaire qui aurait accès aux données qu'ils s'échangent.

1. Alice choisit/construit une suite supercroissante $s_1, ..., s_n$, un entier $N > \sum_{i=1}^{n} s_i$ et un entier b premier avec N.
   Pour tout $i \in \{1, ..., n\}$, elle calcule $a_i = b.s_i \, mod(N)$.
   La clef publique est alors la suite $(a_1, ..., a_n)$ et la clef privée $(s_1, ..., s_n, b, N)$.
2. Si $m = (m_1, ..., m_n) \in \{0,1\}^n$ est le message que veut transmettre Bob à Alice, alors celui-ci récupère la clef publique d'Alice et lui envoie $C = \sum_{i=1}^{n} m_i a_i$.
3. Alice calcule alors $C' = C.b^{-1} \, mod(N)$ [1] et résout le problème du sac-à-dos pour $(s_1, ..., s_n)$ et $C' \geq 0$, ce qui lui renvoit la suite $(m_1, ..., m_n)$.

Proposition 4.1.2. *Ce protocole permet bien de transmettre des messages. De plus un individu auxiliaire doit résoudre un sac-à-dos quelconque afin de retrouver le message m.*

Preuve. Premièrement, $C' = C.b^{-1} \, mod(N) = \sum_{i=1}^{n} m_i a_i b^{-1} \, mod(N) = \sum_{i=1}^{n} m_i s_i \, mod(N)$.
Comme $N > \sum_{i=1}^{n} s_i$, on a que $C' = \sum_{i=1}^{n} m_i s_i$ et donc résoudre ce sac à dos permet de retrouver m.
Quitte à changer le nombre b choisi, la suite $(a_1, ..., a_n)$ n'est a priori plus supercroissante (il suffit alors de le vérifier lors de la géneration de la clef privée). L'exemple concret qui suit permettra de s'en rendre compte.

Exemple. On considère la suite supercroissante : $s = (1, 5, 9, 17, 38, 101, 179)$, $N = 381$ et $b = 203$.
Ainsi s devient $a = (203, 253, 303, 22, 94, 310, 142)$ qui n'est plus supercroissante (même triée).

Bob désire transmettre le message $m = (1, 0, 0, 1, 0, 1, 0)$, ce qui donne $C = 203 + 22 + 310 = 535$.
Donc comme $122 = 203^{-1} \, mod(381)$, on a $C' = 122.535 = 119 \, mod(381)$.
Or $119 = 101 + 17 + 1$, ce qui donne la suite $(1, 0, 0, 1, 0, 1, 0)$ qui est bien le message attendu.

1. b^{-1} existe car b et N sont premiers entre eux. Il se calcule facilement à l'aide de l'algorithme d'Euclide étendu

4.2 Attaque du protocole de Merkle-Hellman

Comme on vient de le voir, pour attaquer le cryptosystème de Merkle-Hellman, il suffit d'être capable de résoudre un problème du sac à dos quelconque. Bien implémenté, on verra que le problème du sac-à-dos associé a des chances d'être résolu à l'aide d'une réduction de réseaux, ce qui explique alors l'importance de l'algorithme LLL.

Premièrement, pour un sac-à-dos donné par la suite $(a_1, ..., a_n)$ et $S \geq 0$, on considère le réseau de $\mathbb{Z}^{n+1}$ engendré par les vecteurs colonne de la matrice suivante :

$$\mathrm{B} = \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & \ddots & & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \\ a_1 & a_2 & \cdots & a_n & -S \end{pmatrix}$$

Ainsi, si on note la base associée $\mathscr{B} = (b_1, ..., b_n, b_{n+1})$, alors pour la suite $(c_1, ..., c_n)$ solution du problème du sac-à-dos, on a :

$$v = \sum_{i=1}^{n} c_i b_i + b_{n+1} = \begin{pmatrix} c_1 \\ \vdots \\ c_n \\ 0 \end{pmatrix}$$

On a alors $\|v\| \leq \sqrt{n}$. Cependant, on peut multiplier notre problème par $K \geq \sqrt{n+1}$ (c'est-à-dire $\sum_{i=1}^{n} c_i(Ka_i) = KS$) et modifier B en conséquence. Ainsi pour tout $l_1, ..., l_n \in \mathbb{Z}$,

$$w = \sum_{i=1}^{n} l_i(Ka_i) - KS = K(\sum_{i=1}^{n} l_i a_i - S)$$

Donc si les l_i ne sont pas solution du sac-à-dos, alors $|\sum_{i=1}^{n} l_i a_i - S| \geq 1$ donc $\|w\| \geq K \geq \sqrt{n+1}$.

Ceci explique alors pourquoi l'algorithme LLL semble adapté à la résolution du problème du sac-à-dos car il nous donne un base de vecteurs courts. Cependant, il n'est pas certains que la base renvoyée contienne systématiquement le vecteur solution.

Remarque. En réalité, l'algorithme LLL fonctionne beaucoup mieux qu'il ne devrait et renvoit dans la plupart des cas le vecteur attendu.

Exemple. Reprenons l'exemple vu précedemment.

Un individu auxiliaire souhaitant avoir accès au message m doit résoudre le problème du sac-à-dos pour la suite $(203, 253, 303, 22, 94, 310, 142)$ et $C = 535$. On écrit la matrice base du réseau associé :

$$\mathrm{B} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 203 & 253 & 303 & 22 & 94 & 310 & 142 & -535 \end{pmatrix}$$

On applique alors l'algorithme LLL à cette base, ce qui nous renvoit la base associée à la matrice :

$$\mathrm{B} = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 1 & -1 \\ 0 & -1 & 1 & -2 & -1 & -1 & 1 & -1 \\ 0 & -1 & -1 & 1 & 0 & -1 & 0 & 1 \\ 1 & 1 & -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & -1 & 2 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & -1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 2 \\ 0 & 1 & 0 & 0 & -2 & -1 & 0 & 0 \end{pmatrix}$$

On remarque alors que la première colonne, qui est le vecteur le plus court de la base, est le seul à correspondre à un vecteur qui pourrait être notre message m. On lit alors le message sur les 7 premières coordonnées : $m = (1, 0, 0, 1, 0, 1, 0)$

Ainsi on a bien retrouvé le message sans pour autant avoir eu besoin de la clef privée.

Annexe 1 : Réseaux et sous-groupes discrets

Dans cette annexe, on va montrer le théorème sur les sous-groupes discrets de $\mathbb{R}^m$ énoncé en page 2. Cette démonstration est très inspirée de celle donnée par Daniel PERRIN dans [3].

Preuve de (2) $\iff$ (3)

On débute la preuve de ce théorème par cette équivalence qui est, en somme, assez visuelle. Pour ce faire on va d'abord démontrer un petit lemme qui nous sera utile ensuite.

Lemme. *Soit G un sous-groupe de $\mathbb{R}^m$ vérifiant* (3)*, alors G est un fermé.*

Preuve. G vérifie (3) donc en particulier il existe $\varepsilon > 0$ tel que $G \cap B(0,\varepsilon) = \{0\}$ car $0 \in G$.
Soit $(x_n)_{n\geq 0}$ une suite de G convergente vers $x \in \mathbb{R}^m$. La suite $(x_{n+1} - x_n)_{n\geq 0}$ est donc dans G et converge vers 0. Ainsi, par l'hypothèse, d'après ce que l'on a dit plus tôt, cette suite est nulle à partir d'un certains rang N. Ceci nous donne alors que pour tout $n \geq N$, la suite $(x_n)_{n\geq 0}$ est constante et vaut $x_n = x \in G$. Donc G est bien fermé.

Ceci nous permet alors de montrer l'équivalence :

Supposons G discret, et soit $x_0 \in G$. Si $x_0 = 0$, la définition d'un sous-groupe discret convient. Sinon, on pose $R = 2\|x_0\|$, alors l'ensemble $H_R = \{x \in G\,,\ \|g\| \leq R\}$ est fini ; en particulier $r = \min\limits_{x \in H_R \setminus \{x_0\}} \|x - x_0\|$ est défini. On pose alors $r' = \min(r, \|x_0\|)/2$, ainsi $G \cap B(x_0, r') = \{x_0\}$.
Pour la réciproque, on suppose qu'il existe $r \geq 0$ tel que $H_r = \{x \in G\,,\ \|g\| \leq r\}$ est infini. On peut donc construire une suite $(x_n)_{n\geq 0}$ de $H_r = G \cap \overline{B(0,r)}$ dont les termes sont tous distincts. Or H_r est un compact dans $\mathbb{R}^m$ car fermé borné donc quitte à considérer une sous-suite, on peut supposer qu'il existe $x \in H_r$ tel que $x_n \to x$. Or, ceci contredit l'hypothèse sur G car pour tout $\varepsilon > 0$, $B(x,\varepsilon) \cap G$ est infini, ce qui conclut.

Preuve de $(1) \Rightarrow (2)$

Soit $(b_1, ..., b_n)$ la base de notre réseau R, on peut la compléter en une base $(b_1, ..., b_m)$ de $\mathbb{R}^m$. On définit alors l'homéomorphisme $\phi : \mathbb{R}^m \to \mathbb{R}^m$ tel que pour tout $x = \sum_{i=1}^{m} x_i b_i \in \mathbb{R}^m a$, $\phi(x) = (x_1, ..., x_m)$. On remarque alors que $\phi(R) \subset \mathbb{Z}^m$ donc si $\mathbb{Z}^m$ est discret dans $\mathbb{R}^m$, alors R aussi. Ainsi il est facile de voir que pour la norme 1, toute boule centrée en 0 admet un nombre fini de points dans $\mathbb{Z}^m$. Or ici toutes les normes sont équivalente donc ce résultat reste vrai pour la norme 2 que l'on utilisait jusqu'à présent.

Preuve de $(2) \Rightarrow (1)$

Cette implication est la plus difficile à montrer et nécessite de montrer deux lemmes préliminaires :

Lemme. *Soient G un sous groupe de $\mathbb{R}^m$, $a \in G\backslash\{0\}$ et $D = \mathrm{Vect}_{\mathbb{R}}(a)$. Soient H un supplémentaire de D et $p : \mathbb{R}^m \to H$ la projection parallèlement à D.*
Alors pour tout $y \in p(G)$, il existe $x \in G$ tel que $p(x) = y$ et $x = y + \lambda a$ avec $0 \leq \lambda < 1$.

Preuve. Soit $y \in p(G)$, il existe $x_0 \in G$ tel que $p(x_0) = y$. Par définition des supplémentaires, on a donc que $x_0 = y + \mu a$. On pose alors $x = x_0 - \lfloor \mu \rfloor a = y + \lambda a \in G$ avec $0 \leq \lambda < 0$, on vérifie toujours bien $p(x) = y$.

On utilise ce lemme pour en montrer un autre qui servira directement pour la preuve :

Lemme. *Avec les mêmes notations, on vérifie deux points :*

1. *Si G est discret, alors $p(G)$ est un sous-groupe discret de H.*
2. *Si $G \cap D = \mathrm{Vect}_{\mathbb{Z}}(a)$, si le groupe $p(G)$ admet une $\mathbb{Z}$-base $(e_1, ..., e_n)$ et si on a $(b_1, ..., b_n)$ une famille de G telle que pour tout i, $p(b_i) = e_i$, alors $(a, b_1, ..., b_n)$ est une $\mathbb{Z}$-base de G.*

Preuve. Pour le point 1, on considère $\Gamma_r = p(G) \cap B(0, r)$ pour $r \geq 0$. Soit $y \in \Gamma_r$, il existe donc $x = y + \lambda a$ tel que $p(x) = y$ et $0 \leq \lambda < 1$ d'où $\|x\| \leq \|a\| + \|y\| \leq \|a\| + r$. On pose alors $R = r + \|a\|$ et $H_R = G \cap B(0, R)$. Ainsi, si on récapitule, pour tout $y \in \Gamma_r$, il existe $x \in H_R$ tel que $p(x) = y$. Or H_R est fini par hypothèse donc Γ_r aussi : $p(G)$ est discret.

Pour le point 2, on va montrer que la famille $(a, b_1, ..., b_n)$ est libre et que $G = \mathrm{Vect}_{\mathbb{Z}}(a, b_1, ..., b_n)$.
Premièrement, soient $\lambda_1, ..., \lambda_{n+1} \in \mathbb{R}$ tels que $\lambda_1 a + \sum_{i=1}^{n} \lambda_{i+1} b_i = 0$. On applique p, ainsi : $\sum_{i=1}^{n} \lambda_{i+1} e_i = 0$ d'où $\lambda_2 = ... = \lambda_{n+1} = 0$ par indépendance des e_i et donc comme a est non nul, $\lambda_1 = 0$ aussi : la famille est libre.
Deuxièmement, soit $x \in G$, il existe $\lambda_2, ..., \lambda_{n+1} \in \mathbb{Z}$ tels que $p(x) = \sum_{i=1}^{n} \lambda_{i+1} e_i$ d'où $p(x - \sum_{i=1}^{n} \lambda_{i+1} b_i) = 0$.
Ainsi, le vecteur $y = x - \sum_{i=1}^{n} \lambda_{i+1} b_i$ vérifie que $y \in G$ et $y \in Ker(p) = D$, donc $y \in \mathrm{Vect}_{\mathbb{Z}}(a)$, ce qui conclut.

Maintenant que l'on a montré ce lemme fondamental, on va l'utiliser pour prouver que (2) $\Rightarrow$ (1) par récurrence sur m.

Cas $m = 1$: On connait les sous-groupes additifs de $\mathbb{R}$, ainsi comme G est discret, il n'est pas dense donc de la forme $a\mathbb{Z}$.

Hérédité : Soit $m \geq 1$ tel que tout sous-groupe discret de $\mathbb{R}^m$ est un réseau. Soit G un sous-groupe discret de $\mathbb{R}^{m+1}$ et soit $x \in G\backslash\{0\}$. On pose alors $D = \mathrm{Vect}_{\mathbb{R}}(x)$ et on considère p la projection sur un supplémentaire H de D parallèlement à D.

D'après le cas $m = 1$, $D \cap G = a\mathbb{Z}$ pour un certains $a \in G$. De plus, par le 1. du lemme précédent, $p(G)$ est un sous-groupe discret de H et par hypothèse de récurrence, comme H est isomorphe à $\mathbb{R}^m$, il existe une $\mathbb{Z}$-base de $p(G)$. Donc par le point 2. du théorème précédent, on construit une $\mathbb{Z}$-base de G, ce qui conclut.

Bibliographie

[1] Steven D. Galbraith. *Mathematics of Public Key Cryptography*, chapter 16, 17 & 19. 2012.

[2] A. Lenstra, H. Lenstra, and L. Lovàsz. Factoring Polynomials with Rational Coefficients. `http://www.math.leidenuniv.nl/~hwl/PUBLICATIONS/1982f/art.pdf`.

[3] Daniel Perrin. Réseaux et applications. `https://www.imo.universite-paris-saclay.fr/~perrin/Sevres/reseaux.pdf`.

[4] Antonio Vera. Analyses de l'algorithme de Gauss. Applications à l'analyse de l'algorithme LLL. `https://tel.archives-ouvertes.fr/tel-01073359/document`.

[5] Brigitte Vallée. La réduction des réseaux. Autour de l'algorithme de Lenstra, Lenstra, Lovász. `http://www.numdam.org/article/ITA_1989__23_3_345_0.pdf`.