ECO | Exact Computing

Réduction de réseaux
Adaptations d'idées provenant du cas polynomial au cas entier.

HAI001I : Stage académique

Lucas Noirot
(lucas.noirot@etu.umontpellier.fr)

Encadrant
Romain Lebreton
(romain.lebreton@lirmm.fr)

17 février - 27 juin 2025

Remerciements

Je tiens en premier lieu à remercier **Romain**, mon encadrant, pour son accompagnement tout au long de ce stage. Toujours patient et bienveillant, même lorsque je disais que $a\mathbb{Z} + b\mathbb{Z} = (a + b)\mathbb{Z}$. Il a su créer un environnement stimulant et agréable. Ses explications, sa disponibilité et sa bonne humeur ont largement contribué à ma progression scientifique et à mon bien-être personnel. Je lui suis sincèrement reconnaissant pour son soutien tant humain que rigoureux.

Un immense merci également à **Eleonora**, qui avait généreusement accepté d'être mon encadrante par intérim, au cas où Romain aurait été indisponible. J'ai pu mesurer toute l'ampleur du travail colossal effectué par les chercheurs au quotidien, un véritable engagement et une source d'inspiration.

Je souhaite remercier **Katharina** pour m'avoir offert l'opportunité de présenter mes travaux au Lattice Club. Si l'exercice était stressant, la bienveillance de l'équipe a transformé ce moment en une expérience enrichissante et mémorable.

Un grand merci aussi à M. **Goncalves** et M. **Giroudeau** pour leurs retours. Parfois piquants mais instructifs, leurs commentaires sont comme la moutarde : on apprend vite à les apprécier et à en saisir toute la valeur pédagogique.

Merci à Mme **Fourcadier** et à toute l'équipe administrative en charge de la gestion des stages. Leur réactivité et leur efficacité rendent le parcours administratif agréable.

Je tiens à exprimer toute ma gratitude à mes collègues de bureau, **Quentin**, **Geoffroy** et **Logan**. Quentin, avec qui j'ai partagé tant de discussions enrichissantes m'a immédiatement inclus dans l'équipe. Geoffroy, dont la lucidité remarquable sur le monde académique a toujours donné lieu à des échanges stimulants. Logan, quant à lui, véritable incarnation du « chill guy originel », a apporté au bureau une sérénité et une bonne humeur constantes.

Un grand merci également à **Federico**, véritable rayon de soleil du laboratoire, dont les anecdotes et l'amour communicatif de l'Italie ont toujours illuminé mes journées. Merci aussi à **Laz**, pour sa gentillesse permanente.

Merci à **Andrei** pour son enthousiasme intellectuel hors-norme et ses réflexions captivantes sur la complexité de Kolmogorov, appliquée jusque dans le monde des fourmis. Ces moments d'échange furent toujours aussi étonnants qu'amusants. Merci aussi à **Gabrielle** qui, malgré nos rares échanges, m'a offert l'occasion unique de comprendre pleinement un accent anglais !

Merci à **Matteo** et **Paul**, fidèles compagnons des repas du mardi midi.

Un clin d'œil particulier aux personnes du foodtruck, surtout par ces fortes chaleurs.

Je ne saurais conclure ces remerciements sans mentionner les chats du laboratoire, maîtres incontestés des lieux, ainsi que l'incontournable machine à café. Fidèle alliée de mes neurones.

Enfin, ma reconnaissance la plus profonde va à **Julie**, ma compagne. Son écoute et sa patience ont été essentiels, sa présence a été mon plus précieux soutien.

Toutes ces interactions, grandes ou petites, ont contribué significativement à mon équilibre personnel pendant cette période intense. Souvent, ce sont ces gestes simples qui ont l'impact le plus fort sur le bien-être quotidien. Merci infiniment à toutes et tous.

Table des matières

Liste des Algorithmes

Table des figures

Notations, complexité et acronymes

Cette section fait office de lexique et recense l'ensemble des notations et acronymes utilisées tout au long de ce mémoire.

Notations

$\mathbb{Z}$	L'ensemble des entiers relatifs.
$\mathbb{Z}_n$	L'ensemble des entiers modulo n.
$\mathbb{Q}$	L'ensemble des rationnels.
$\mathbb{R}$	L'ensemble des réels.
$\mathbb{K}$	Un corps quelconque.
$\mathbb{F}_p$	Un corps fini de caractéristique p.
$\mathbb{F}_p[x]$	L'anneau des polynômes univariés à coefficients dans le corps fini $\mathbb{F}_p$.
$\mathbb{F}_p^{\leq d}[x]$	Les polynômes à coefficients dans $\mathbb{F}_p$ de degré inférieur ou égal d.
$\mathbb{F}_p[x]^{m\times n}$	Les matrices $m \times n$ à coefficients dans $\mathbb{F}_p[x]$.
$\mathscr{L}$	Un réseau, désigné par une lettre majuscule calligraphiée.
$\mathscr{L}(B)$	Le réseau engendré par la matrice B.
B^t	La transposée de la matrice B.
B^*	La base de Gram-Schmidt associée à B.
$\mathbf{b}, \mathbf{g}, \ldots$	Les vecteurs de $\mathbb{R}^n$, notés en gras.

Complexité

Multiplication dans $\mathbb{F}_p^{\leq d}[x]$	$\mathrm{M}(d) = \mathcal{O}(d \times \log d \times \log\log d)$.
Multiplication dans $\mathbb{F}_p^{n\times n}$	$\mathrm{MM}(n) = \mathcal{O}(n^\omega)$.
Multiplication dans $\mathbb{F}_p^{\leq d}[x]^{m\times n}$	$\mathrm{MM}(n,d)$[1] $= \mathcal{O}(\mathrm{MM}(n)\,\mathrm{M}(d)) = \tilde{\mathcal{O}}(n^{\omega 2}d)$.

Symboles

■	Le symbole marquant la fin d'une démonstration.
$\leftarrow, :=$	Les symboles d'affectations d'un algorithme.

Acronymes

LLL	Lenstra–Lenstra–Lovász
BKZ	Block Korkine-Zolotarev
SVP	Shortest Vector Problem
SIVP	Shortest Independant Vector Problem
CVP	Closest Vector Problem

1. $\mathrm{MM}(n,d)$ peut également être obtenu via une approche d'évaluation-interpolation sur une suite géométrique, ce qui permet d'améliorer certaines bornes de complexité.

2. ω est l'exposant optimal de la multiplication matricielle.

Guide de Lecture

DANS ce mémoire, chaque définition est suivie d'un exemple concret illustrant la notion en question, ainsi que d'un contre-exemple visant à en exposer les subtilités et exceptions éventuelles. Cette approche permet de mieux comprendre les conditions et les limitations associées à chaque concept. L'objectif est de clarifier les différences entre les situations avec lesquelles une définition est applicable et celles où elle ne l'est pas, afin de renforcer la compréhension approfondie des théorèmes et constructions présentés.

Ce mémoire s'adresse à un lecteur ayant une certaine familiarité avec l'algèbre linéaire et la théorie des algorithmes, mais qui n'est pas nécessairement spécialiste des réseaux. Les chapitres peuvent être lus dans l'ordre, mais selon l'objectif ou le temps disponible du lecteur, voici un aperçu de leur contenu et de leur rôle dans le fil directeur du travail :

- **Introduction** Présente les réseaux euclidiens, la notion de réduction de réseau et la problématique du stage : peut-on adapter des techniques issues du cas polynomial au cas entier ? Ce chapitre donne également un aperçu de la motivation cryptographique.
- **Chapitre 1. Réseaux euclidiens :** Définitions formelles, exemples, propriétés fondamentales, ainsi que les problèmes algorithmiques classiques comme SVP et CVP. Un passage essentiel pour qui n'est pas encore familier avec ces objets.
- **Chapitre 2. Réduction de réseaux euclidiens :** Introduction détaillée à l'algorithme LLL, avec explication de son fonctionnement et un exemple complet. Ce chapitre est central pour comprendre l'approche de réduction dans le cas entier.
- **Chapitre 3. Correction, terminaison et complexité de LLL :** Contient les preuves rigoureuses de correction et d'efficacité de LLL. Le lecteur peu intéressé par les détails techniques pourra survoler ce chapitre en ne retenant que les idées-clés.
- **Chapitre 4. Réseaux polynomiaux :** Pose le cadre algébrique du cas polynomial, et introduit des outils spécifiques comme le degré de ligne ou la notion de décalage. Utile pour comprendre la suite, même sans entrer dans tous les détails.
- **Chapitre 5. Réduction de réseaux polynomiaux :** Présente les algorithmes de réduction exacts en temps polynomial dans le cadre polynomial : WeakPopovForm, M-Basis, PM-Basis. Ces méthodes inspireront les adaptations proposées ensuite.
- **Chapitre 6. Adaptation au cas euclidien :** Partie centrale du stage. On y explore comment transposer les méthodes du cas polynomial vers les réseaux entiers : algorithmes d'approximation, techniques de décalage, et analyse de complexité. Ce chapitre est le cœur exploratoire et prospectif du mémoire.
- **Annexes :** Fournissent les rappels nécessaires sur les groupes, anneaux, modules et notions d'algèbre linéaire. Elles permettent de clarifier certains fondements utilisés dans les chapitres principaux.

Le lecteur pressé pourra se limiter à la lecture des chapitres 1, 2, 4 et 6 pour avoir une vision d'ensemble du problème traité et des méthodes proposées.

Déroulé du stage et motivations

Ce stage de Master 2 s'est déroulé au sein de l'équipe ECO (Exact COmputing) du LIRMM, à l'Université de Montpellier. Il s'inscrit dans le cadre d'un stage académique en informatique théorique, axé sur l'étude de la réduction de réseaux euclidiens, un sujet en calcul formel et en cryptographie. L'ensemble du code développé durant le stage est disponible à l'adresse suivante : `[lien vers dépôt/code]`.

Objectifs et déroulé du stage

L'objectif principal du stage était d'essayer d'apater des techniques de réductions de réseaux euclidiens avec des techniques issues du monde polynomiale, en s'appuyant à la fois sur les fondements théoriques et sur des expérimentations.

- **Phase 1 : Compréhension des réseaux euclidiens et de l'algorithme LLL**
  Le début du stage a été consacré à l'étude approfondie de l'algorithme de Lenstra–Lenstra–Lovász (LLL), de sa preuve de correction ainsi que de la bibliographie associée sur les réseaux euclidiens. Implémentation de LLL en SageMath. Cette étape a également permis de préparer la rédaction du rapport intermédiaire.
- **Phase 2 : Compréhension des réseaux polynomiaux et algorithmes de réduction**
  Dans un second temps, le travail s'est tourné vers le cas des réseaux polynomiaux : lecture de la littérature, compréhension des algorithmes (BASIS, PM-BASIS), et mise en perspective avec le cas euclidien. Implémentation dans SageMath.
- **Phase 3 : Rapport intermédiaire et retour encadrant**
  La remise du rapport intermédiaire a permis de faire le point sur les avancées, de mieux cadrer le positionnement du stage, et de structurer l'état de l'art.
- **Phase 4 : Réflexions d'adaptation au cas entier**
  Durant la seconde moitié du stage, des pistes d'adaptation d'algorithmes issus du cas polynomial ont été explorées pour les réseaux entiers.

Participation à la vie du laboratoire

Durant le stage, j'ai également eu l'occasion de m'impliquer dans l'environnement de recherche local :

- Présentation d'un exposé de 1h30 (en anglais) sur l'algorithme LLL et sa preuve lors du *Lattice Club*, un séminaire interne dédié aux réseaux.
- Participation hebdomadaire aux séances du Lattice Club, animées notamment par Katarina et d'autres intervenants, où différents aspects de la cryptographie à base de réseaux ont été abordés.
- Présence régulière aux séminaires du laboratoire organisés les mardis.

Introduction

Un réseau euclidien peut être intuitivement vu comme un ensemble discret et régulier de points dans l'espace $\mathbb{R}^n$, formant un sous-groupe discret additif. À titre d'exemple, dans le plan $\mathbb{R}^2$, un réseau correspond aux intersections d'un quadrillage régulier. Malgré leur ressemblance avec les espaces vectoriels classiques, les réseaux euclidiens possèdent des propriétés spécifiques et complexes, rendant invalides de nombreux résultats habituellement vérifiés dans les $\mathbb{K}$-espaces vectoriels. Cette complexité fait des réseaux euclidiens un objet d'étude particulièrement riche à la frontière de plusieurs domaines de mathématiques et d'informatiques, en particulier le domaine de la cryptographie.

Un des problèmes fondamentaux liés aux réseaux euclidiens est la réduction de réseaux, qui consiste à déterminer une bonne base de ce dernier, c'est-à-dire une base qui facilite la résolution efficace de divers problèmes algorithmiques liés aux réseaux euclidiens, comme celui de trouver le vecteur le plus court d'un réseau, ou trouver le vecteur le plus proche d'une cible donnée. Ces deux problèmes sont connus pour être NP-complets et constituent précisément la difficulté à la base des cryptosystèmes reposant sur les réseaux euclidiens. Tandis que ces questions apparaissent simples et intuitives en basse dimension, elles deviennent rapidement très complexes et coûteuses à résoudre en grande dimension. Un exemple emblématique de l'ambivalence algorithmique des réseaux euclidiens est l'algorithme LLL (Lenstra-Lenstra-Lovász), que nous étudierons, initialement célèbre pour avoir permis la cryptanalyse de systèmes basés sur le problème du sac à dos, mais qui joue aujourd'hui paradoxalement un rôle clé dans la conception de nouveaux schémas cryptographiques robustes.

Pour mieux comprendre les difficultés intrinsèques de la réduction de réseaux euclidiens, il est pertinent d'étudier les réseaux polynomiaux, une classe analogue aux réseaux euclidiens. Alors que les réseaux euclidiens posent des problèmes NP-difficiles, les réseaux polynomiaux peuvent être réduits exactement et efficacement en temps polynomial. Cette différence fondamentale ouvre une piste prometteuse qui définit le thème de ce stage :

> *Problématique du stage*
> Serait-il possible d'adapter certaines méthodes exactes de réduction, initialement développées pour les réseaux polynomiaux, au cas des réseaux euclidiens ?

L'objectif de ce stage est d'explorer, d'analyser et de comparer les deux approches distinctes mais étroitement reliées en s'appuyant notamment sur l'algorithme LLL dans le cas euclidien, tout en cherchant à le réinterpréter à travers des idées issues du contexte polynomial. Nous tenterons en particulier d'identifier clairement les similarités fondamentales ainsi que les différences essentielles qui se dégagent de cette comparaison. À partir de ces observations, une partie centrale du travail consistera à évaluer précisément l'efficacité potentielle de ces adaptations méthodologiques, à identifier les obstacles théoriques ou pratiques susceptibles de freiner leur transposition, et à proposer des pistes d'améliorations et d'explorations futures.

La pertinence de cette démarche s'inscrit dans un contexte où les systèmes de sécurité actuels, reposant sur la difficulté de la factorisation des grands nombres premiers et du calcul de logarithmes discrets, risquent d'être mis à mal par les progrès en informatique quantique. À l'opposé, les réseaux euclidiens apparaissent comme une alternative prometteuse en cryptographie post-quantique, plusieurs de leurs problèmes fondamentaux semblant résister efficacement aux attaques quantiques. Ainsi, approfondir la compréhension et améliorer les techniques de réduction de réseaux euclidiens représente un enjeu crucial pour le développement futur de la cryptographie face aux défis quantiques.

CHAPITRE **1**

Réseaux euclidiens

L'ÉTUDE des réseaux euclidiens puise ses racines dans les travaux mathématiques du XVIII[e] siècle, notamment ceux de Leonhard Euler sur l'organisation géométrique des points dans l'espace. C'est cependant en 1891 qu'Hermann Minkowski établit véritablement les fondements modernes avec l'introduction de la théorie géométrique des nombres, reliant explicitement les réseaux à divers problèmes d'optimisation et de minimisation. Ses résultats joueront ultérieurement un rôle déterminant dans le développement de la cryptographie moderne.

Néanmoins, c'est au cours du XX[e] siècle, et plus particulièrement à partir des années 1990, que les réseaux euclidiens connaissent une véritable intégration dans la cryptographie.

Les travaux de chercheurs tels que Ajtai, Dwork ou Regev marquent alors un tournant décisif, en démontrant que les difficultés algorithmiques intrinsèques aux réseaux peuvent constituer une base solide pour la conception de nouveaux systèmes cryptographiques résistants aux attaques conventionnelles et quantiques. Ce chapitre vise précisément à introduire de manière approfondie les concepts fondamentaux liés aux réseaux euclidiens.

1.1 *Définitions et exemples*

Nous considérons un espace euclidien, c'est-à-dire un espace vectoriel réel de dimension finie muni d'un produit scalaire, noté $\langle \mathbf{f}, \mathbf{g} \rangle$. Dans ce chapitre, nous utiliserons le produit scalaire usuel défini par $\langle \mathbf{f}, \mathbf{g} \rangle := \mathbf{f} \cdot \mathbf{g}^t = \sum_{i=1}^{n} \mathbf{f}_i \mathbf{g}_i$, lequel induit la norme-2 donnée par $\|\mathbf{f}\|_2 = \sqrt{\sum_{i=1}^{n} \mathbf{f}_i^2}$. Pour alléger les notations, nous omettrons l'indice 2. Ce chapitre se contentera d'exposer les résultats classiques, sans chercher à les redémontrer. Rappelons qu'au sein de $\mathbb{R}^n$, toutes les normes sont équivalentes, ce qui signifie qu'aucune ne change la nature intrinsèque des problèmes que nous aborderons. Nous nous concentrerons néanmoins sur la norme-2 pour sa simplicité géométrique, car elle offre une mesure intuitive des longueurs et des angles, point essentiel pour étudier la structure des réseaux euclidiens. Afin de simplifier l'écriture, nous désignerons par $\mathcal{B} = (\mathbf{b}_i)_{1 \leq i \leq n}$ une base de $\mathbb{R}^n$ et par $\mathcal{B}^* = (\mathbf{b}_i^*)_{1 \leq i \leq n}$ sa base orthogonalisée par le procédé de Gram-Schmidt. Les matrices B et B^* (sans calligraphie) seront composées (en ligne) des vecteurs de la famille $(\mathbf{b}_i)_{1 \leq i \leq n}$ et $(\mathbf{b}_i^*)_{1 \leq i \leq n}$ respectivement.

On commence par introduire la notion de réseau euclidien. Pour un rappel sur les groupes ou les modules le lecteur est invité à lire les annexes correspondantes. Le point de vue de module nous sera utile dans la suite de ce manuscrit.

Définition 1.1. Soit $n \in \mathbb{N}^*$, $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^n$. Les définitions suivantes sont équivalentes. [1]

- Un **réseau euclidien** $\mathscr{L}$ est un sous-groupe discret additif de $\mathbb{R}^n$.
  - — Sous-groupe additif : $\mathbf{0} \in \mathscr{L}$, et pour tout $\mathbf{x}, \mathbf{y} \in \mathscr{L}$, $\mathbf{x} + \mathbf{y}, -\mathbf{x} \in \mathscr{L}$
  - — Discret : $\forall \mathbf{x} \in \mathscr{L}, \exists \varepsilon > 0$ tel que $\mathcal{B}(\mathbf{x}, \varepsilon) \cap \mathscr{L} = \{\mathbf{x}\}$ (où $\mathcal{B}(\mathbf{x}, \varepsilon)$ désigne la boule ouverte de rayon ε centrée en $\mathbf{x}$).

1. Plusieurs définitions équivalentes d'un réseau euclidien coexistent dans la littérature, selon que l'on se place ou non dans un espace euclidien $\mathbb{R}^n$, ou dans un espace $\mathbb{R}^n$ équipé explicitement d'une forme quadratique définie positive. Dans tous les cas, l'idée générale reste la même : un réseau euclidien est un sous-ensemble discret de $\mathbb{R}^n$ formé par toutes les combinaisons linéaires entières d'un ensemble de vecteurs générateurs.

- Un **réseau euclidien** $\mathscr{L}$ est un $\mathbb{Z}$-module libre de type fini de $\mathbb{R}^n$.

Définition 1.2. Un **sous-réseau** $\mathscr{L}'$ de $\mathscr{L}$ est un sous groupe de $\mathscr{L}$, on notera $\mathscr{L}' \subseteq \mathscr{L}$.

Exemple. Les entiers de Gauss, définit par $\mathbb{Z}[i] := \mathbb{Z} \oplus i\mathbb{Z}$ forment un réseau de rang 2 dans $\mathbb{C}$, c'est même un anneau.

Exemple. Un exemple plus exotique, $\mathbb{Z} \oplus \sqrt{2} \cdot \mathbb{Z}$ est un réseau de rang 2 dans $\mathbb{R}$.

Contre exemple. $\mathbb{Q}$ n'est pas un réseau euclidien , car $\mathbb{Q}$ est dense dans $\mathbb{R}$, ce qui brise la discrétude, bien que ce soit un sous-groupe de $\mathbb{R}$.

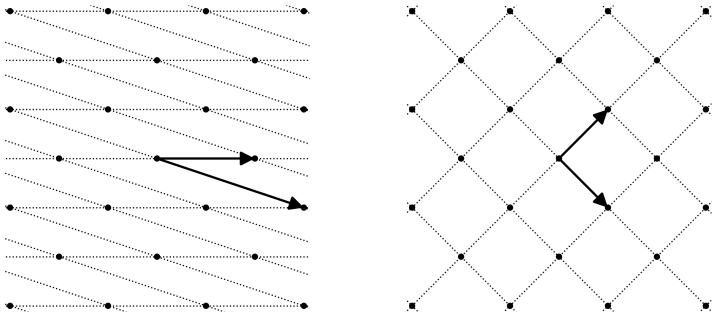
En particulier on peut montrer qu'il existe une famille $\mathbb{Z}$-libre maximale $(\mathbf{b}_i)_{1\leq i\leq m}$ dans $\mathscr{L}$ telle que

$$\mathscr{L} = \bigoplus_{1\leq i\leq m} \mathbb{Z}\mathbf{b}_i := \{a_1\mathbf{b}_1 + \cdots a_m\mathbf{b}_m : a_i \in \mathbb{Z}\}$$

Cette famille est appelée **base** de $\mathscr{L}$, si on note B la matrice de la famille $(\mathbf{b}_i)_{1\leq i\leq m}$ on notera $\mathscr{L}(B)$ le réseau de base B, donc **engendré** par la famille $(\mathbf{b}_i)_{1\leq i\leq m}$. L'entier m est commun à toutes les bases de $\mathscr{L}$ et on l'appelle **rang** de $\mathscr{L}$. Lorsque $n = m$, on dit que le réseau est de **rang plein**.

Exemple. On a la suite d'inclusions $2\mathbb{Z} \subset \mathbb{Z} \subset \frac{1}{2}\mathbb{Z}$.[2] Bien que $\mathrm{rang}(2\mathbb{Z}) = \mathrm{rang}(\mathbb{Z}) = \mathrm{rang}(\frac{1}{2}\mathbb{Z})$, ces ensembles sont distincts. Cela montre que, contrairement aux espaces vectoriels, avoir une relation d'inclusion et avoir le même rang ne suffit pas à garantir l'égalité des réseaux.

Existe-t-il une notion de **bonne** base ? Nous verrons qu'une base idéale est celle qui est en un sens la plus orthogonale possible. Il n'existe pas toujours de base strictement orthogonale, ce qui justifie la notion de quasi-orthogonalité. Nous allons rajouter des façons de mesurer la qualité d'une base dans le chapitre suivant et introduire la notion de réduction, qui consistera à trouver une telle base.



(a) $B = \begin{pmatrix} 3 & -1 \\ 2 & 0 \end{pmatrix}$. (b) $B' = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$

FIGURE 1.1 – Deux bases pour le même réseau de $\mathbb{R}^2$.

Définition 1.3. La **taille**[3] d'un réseau $\mathscr{L}(B)$ est $|\det(B)|$ et est noté $|\mathscr{L}|$. La taille d'un réseau est indépendante de la base choisie.

Remarque. On peut définir de façon équivalente $|\mathscr{L}| := \sqrt{\det(\mathrm{Gram}(B))}$ où $\mathrm{Gram}(B) = B^t \cdot B$. On peut voir $|\mathscr{L}|$ comme le **volume du domaine fondamental** de $\left\{ \sum_{i=0}^{n} \lambda_i \mathbf{b}_i \mid 0 \leq \lambda_i < 1 \right\}$.

2. On rappelle que pour $a \in \mathbb{R}$, on a $a\mathbb{Z} = \{an \in \mathbb{R} | n \in \mathbb{Z}\}$.

3. **taille** et **volume** sont synonymes, mais l'usage le plus courant dans la littérature anglaise est "déterminant du réseau" ou "volume".

Théorème 1.1 (Inégalité d'Hadamard)**.** Soit $B \in M_n(\mathbb{K})$ et $\mathscr{L}(B)$ un réseau. Alors

$$|\mathscr{L}| = |\det(B)| \leq \prod_{i=1}^{n} \|\mathbf{b}_i\|$$

La borne est atteinte si, et seulement si $(\mathbf{b}_i)_{1\leq i\leq n}$ est une famille orthogonale.

Toutes les bases d'un réseau euclidien diffèrent d'une transformation de déterminant ± 1. L'ensemble de ces transformations est connu sous le nom de groupe unimodulaire. Un ensemble de points non alignés dans un réseau ne constitue pas une base si son déterminant est différent de $\pm|\mathscr{L}|$.

Proposition 1.1. Soit $\mathscr{L}$ et $\mathscr{L}'$ deux réseaux de rang n de base B et B'. Alors $\mathscr{L} = \mathscr{L}'$ si et seulement si il existe $U \in \mathrm{GL}_n(\mathbb{Z})$ tel que $B' = BU$. Où $\mathrm{GL}_n(\mathbb{Z}) = \{M \in \mathbb{Z}^{n\times n} | \det(M) = \pm 1\}$.

Remarque. On a l'action de groupe

$$\begin{aligned} \mathrm{GL}_n(\mathbb{Z}) \times \mathrm{GL}_n(\mathbb{R}) &\longrightarrow \mathrm{GL}_n(\mathbb{R}) \\ (U, \mathbf{B}) &\longmapsto \mathbf{B}U \end{aligned}$$

Un réseau est exactement une orbite de cette action. Nous verrons dans le chapitre suivant que la réduction de réseaux consiste à trouver un bon représentant pour chaque orbite.

Nous présentons maintenant deux **invariants** fondamentaux d'un réseau :

- La **taille du vecteur minimal** du réseau, notée $\lambda_1(\mathscr{L})$.
- Le **volume du réseau**, aussi appelé la **taille du réseau** souvent désigné par $|\mathscr{L}|$.

Proposition 1.2. Soit $\mathscr{L}$, $\mathscr{L}'$ deux réseaux de $\mathbb{R}^n$ tel que $\mathscr{L}' \subseteq \mathscr{L}$ alors $\frac{|\mathscr{L}'|}{|\mathscr{L}|} \in \mathbb{N}$.

Remarque. Ce résultat est une conséquence directe du théorème de Lagrange en théorie des groupes.

Définition 1.4. On appelle **minimum d'un réseau**[4] $\mathscr{L}$ la quantité

$$\lambda_1(\mathscr{L}) = \min_{\substack{v\in\mathscr{L} \\ v\neq 0}} \|v\|$$

Plus généralement, pour $k \in \{1, \ldots, n\}$, on pose $\lambda_k(\mathscr{L})$ le plus petit réel r tel qu'il existe k vecteurs $\mathbb{R}$-linéairement indépendants dans $\mathscr{L}$ de norme au plus r.

Remarque. $\lambda_1(\mathscr{L})$ correspond à la distance minimale entre deux points quelconques de $\mathscr{L}$.

Exemple. Soit $\mathscr{L}$ le réseau de la figure 1.1. On a $|\mathscr{L}| = 2$ et $\lambda_1(\mathscr{L}) = \lambda_2(\mathscr{L}) = \sqrt{2}$.

Théorème 1.2 (Premier théorème de Minkowski)**.** Pour tout $n \in \mathbb{N}^*$, il existe une constante $C_n > 0$ telle que pour tout réseau $\mathscr{L}$ de $\mathbb{R}^n$, on a :

$$\lambda_1(\mathscr{L}) \leq C_n |\mathscr{L}|^{1/n}$$

On peut prendre cette constante égale à $C_n = (2/\sqrt{\pi})\Gamma(n/2+1)^{1/n}$.[5] On appelle **constante de**

4. on peut aussi le définir comme $\lambda_1(\mathscr{L}) = \min\{r > 0 : |\mathcal{B}(r) \cap \mathscr{L}| > 1\} \in \mathbb{R}_+$

5. La fonction Γ, appelée fonction gamma, généralise la notion de factorielle aux nombres réels (et complexes). Elle est définie, pour tout $z \in \mathbb{C}$, par la formule suivante :

$$\Gamma(z) = \int_0^{+\infty} t^{z-1} e^{-t}\, dt.$$

Hermite-Minkowski le carré de la constante optimale possible pour cette inégalité, noté γ_n, en particulier, on a $\gamma_n \leq C_n$. En développant, on a

$$\gamma_n = \sup_{\dim(\mathscr{L})=n} \gamma(\mathscr{L}), \quad \text{où } \gamma(\mathscr{L}) := \frac{\lambda_1(\mathscr{L})^2}{\det(\mathscr{L})^{2/n}}$$

Proposition 1.3.

$$\gamma_n = 4\left(\frac{\Delta_n}{V_n}\right)^{2/n} \quad \forall n \in \mathbb{N}^*$$

où $V_n = \frac{\pi^{n/2}}{\Gamma(\frac{n}{2}+1)}$ et Δ_n est la densité d'un empilement compact[6] de densité maximum d'hypersphères.

On ne connaît pas la valeur exacte de γ_n pour tout n. Seuls les valeurs dans le tableau suivant sont connues de manières exactes.[7]

n	1	2	3	4	5	6	7	8	24
γ_n^n	1	$\frac{4}{3}$	2	4	8	$\frac{64}{3}$	64	256	4^{24}

FIGURE 1.2 – Valeurs connues de γ_n

On sait que $(\gamma_n)_n$ est une suite d'ordre de croissance linéaire mais on ne sait pas si elle est croissante.

Exemple. Le **réseau d'Eisenstein**, ou **réseau en nid d'abeille**, est un réseau de $\mathbb{R}^2$ de rang 2, engendré par la base $\begin{pmatrix} 1 & \frac{1+\sqrt{3}}{2} \\ 1 & \frac{1-\sqrt{3}}{2} \end{pmatrix}$. On a $|\mathscr{L}| = \sqrt{3}, \lambda_1 = \sqrt{2}, \gamma(\mathscr{L}) = \frac{2}{\sqrt{3}}$. En traçant des sphères de centre les points du réseaux et de rayon $\sqrt{2}$, on obtient l'empilement compact le plus dense en dimension 2.

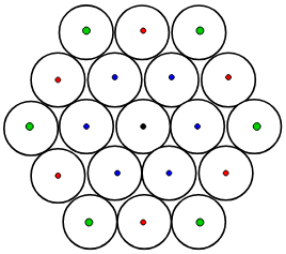


FIGURE 1.3 – Empilement hexagonal dans $\mathbb{R}^2$

Proposition 1.4. Soit $\mathscr{L} \subset \mathbb{R}^n$ un réseau de base $(\mathbf{b}_i)_{1\leq i\leq n}$ et sa base de Gram-Schmidt associée $(\mathbf{b}_i^*)_{1\leq i\leq n}$. Alors pour tout $\mathbf{v} \in \mathscr{L}\backslash\{\mathbf{0}\}$, on a

$$\lambda_1(\mathscr{L}) \geq \|\mathbf{v}\| \geq \min_{1\leq i\leq n} \|\mathbf{b}_i^*\|$$

Tous ces résultats et inégalités vont nous donner des critères de mesure de la qualité de réduction d'une base dans le chapitre sur la réduction de réseaux euclidiens.

6. Un empilement compact d'une collection d'objets est un agencement de ces objets de telle sorte qu'ils occupent le moins d'espace possible.

7. **En dimension** $n = 1$, n'importe quel réseau de dimension 1 atteint cette borne. **En dimension** $n = 2$, le réseau optimal est celui des entiers d'Eisenstein, également appelé réseau en nid d'abeille. **En dimension** $n = 3$, le réseau optimal est présenté dans (HALES et al. 2015), mais sa démonstration nécessite environ 130 pages de minimisation de fonctions analytiques. Des avancées majeures ont été obtenues pour les **dimensions** $n = 8$ et $n = 24$ grâce à la mathématicienne ukrainienne Maryna Viazovska, qui a démontré l'optimalité du réseau E_8 (VIAZOVSKA 2016) et, en collaboration, celle du réseau de Leech (COHN et al. 2016).

1.2 *Réseaux définis par relations plutôt que par générateurs*

Définition 1.5. Soit $\mathscr{L} \subset \mathbb{R}^n$ un réseau de base $(\mathbf{b}_i)_{1\leq i\leq n}$. Le **dual** de $\mathscr{L}$ est défini par

$$\mathscr{L}^\vee := \{\mathbf{x} \in \mathbb{R}^n \mid \forall \mathbf{y} \in \mathscr{L}, \langle \mathbf{x}, \mathbf{y}\rangle \in \mathbb{Z}\}.$$

Proposition 1.5. On a $\mathscr{L} = \mathscr{L}(\mathbf{b}_1, \ldots, \mathbf{b}_m)$ si et seulement si $\mathscr{L}^\vee = \mathscr{L}(\mathbf{b}_1^\vee, \ldots, \mathbf{b}_n^\vee)$, où $\mathbf{b}_i^\vee$ vérifie $\langle \mathbf{b}_i^\vee, \mathbf{b}_i\rangle = \delta_{i,j}$.[8]

Proposition 1.6. $\mathscr{L}^\vee$ est un réseau de base :

- $(B^t)^{-1}$ lorsque le réseau est de rang plein.
- $B(B^tB)^{-1}$ lorsque le réseau n'est pas de rang plein.

Proposition 1.7. De la proposition précédente découle les propriétés suivantes :

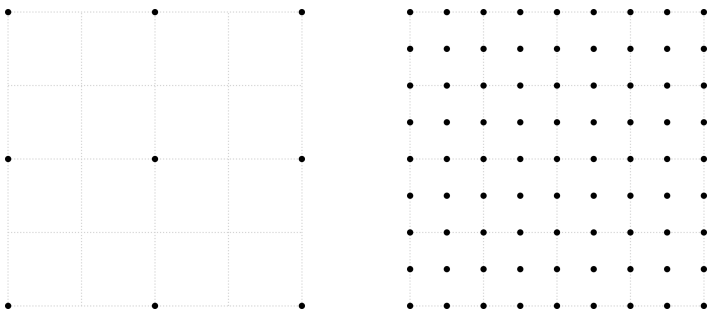
- $\mathrm{rang}(\mathscr{L}) = \mathrm{rang}(\mathscr{L}^\vee)$.
- $|\mathscr{L}^\vee| = |\mathscr{L}|^{-1}$.
- $(\mathscr{L}^\vee)^\vee = \mathscr{L}$

Définition 1.6. On dit que $\mathscr{L}$ est **auto-dual** si $\mathscr{L} = \mathscr{L}^\vee$.

Proposition 1.8. On a les propriétés suivantes :

- $(a\mathscr{L})^\vee = \frac{1}{a}\mathscr{L}^\vee$ pour tout $a \in \mathbb{R}^*$.
- $(\mathbb{Z}u)^\vee = \frac{1}{\|u\|}\mathbb{Z}u$ pour tout $u \in \mathbb{R}^m\backslash\{\mathbf{0}\}$.

Exemple. Le réseau dual de $\mathbb{Z}^n$ est $\mathbb{Z}^n$ et est donc auto-dual.



(a) $2\mathbb{Z}^2$ (b) $\frac{1}{2}\mathbb{Z}^2$

FIGURE 1.4 – Un réseau et son dual.

Proposition 1.9. Soit $\mathscr{L}_1, \mathscr{L}_2$ des réseaux, alors

$$(\mathscr{L}_1 \oplus \mathscr{L}_2)^\vee = \mathscr{L}_1^\vee \oplus \mathscr{L}_2^\vee$$

Lemme 1.1. Soit $\mathscr{L}$ un réseau de dimension n. On a

- $\lambda_1(\mathscr{L}) \cdot \lambda_1(\mathscr{L}^\vee) \leq n$,
- $\lambda_1(\mathscr{L}) \cdot \lambda_n(\mathscr{L}^\vee) \geq 1$.

8. $\delta_{i,j}$ est le symbole de Kronecker, il vaut 1 si $i = j$ et 0 sinon.

(BANASZCZYK 1993) a démontré une relation encore plus forte entre les minima d'un réseau et ceux de son dual, connue sous le nom de théorème de transfert.

Théorème 1.3 (Théorème de transfert)**.** Soit $\mathscr{L}$ un réseau de dimension n. On a

$$1 \leq \lambda_1(\mathscr{L}) \cdot \lambda_n(\mathscr{L}^\vee) \leq n.$$

Définition 1.7. On définit le réseau euclidien $A_n \subset \mathbb{R}^{n+1}$ par :

$$A_n = \left\{ (x_1, \ldots, x_{n+1}) \in \mathbb{Z}^{n+1} \;\middle|\; \sum_{i=1}^{n+1} x_i = 0 \right\}$$

Exemple. On a :

$$\begin{aligned} &A_0 = \{0\} \subset \mathbb{R}, \\ &A_1 = \{(x, -x) \in \mathbb{Z}^2\}, \text{ donc } A_1 \text{ est de rang } 1 \\ &A_2 = \{(x, y, z) \in \mathbb{Z}^3 \mid x + y + z = 0\}, \\ &\quad = \langle a_1, a_2 \rangle \quad \text{où } a_1 = (1, -1, 0), \quad a_2 = (0, 1, -1), \\ &\quad \text{donc } A_2 \text{ est de rang } 2. \end{aligned}$$

Proposition 1.10. Pour tout $n \in \mathbb{N}$, A_n a pour matrice génératrice :

$$B_n := \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ -1 & 1 & 0 & \cdots & 0 \\ 0 & -1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 & 1 \end{pmatrix} \in M_n(\mathbb{Z})$$

Proposition 1.11. A_n est un réseau euclidien de rang n, pour tout $n \in \mathbb{N}$.

Il existe une classe particulière de réseaux qui joue un rôle important en cryptographie.

Définition 1.8 (Réseau q-aire)**.** Un réseau $\mathscr{L}$ est un réseau q-aire si

$$q\mathbb{Z}^n \subseteq \mathscr{L} \subseteq \mathbb{Z}^n.$$

Étant donnée une matrice $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ pour certains entiers $n, m, q \in \mathbb{N}$, on peut définir deux réseaux :

$$\mathscr{L}_q(\mathbf{A}) = \{\mathbf{y} \in \mathbb{Z}^m : \mathbf{y} = \mathbf{A}\mathbf{s} \bmod q \text{ pour un certain } \mathbf{s} \in \mathbb{Z}^n\}$$

$$\mathscr{L}_q^\perp(\mathbf{A}^T) = \left\{\mathbf{y} \in \mathbb{Z}^m : \mathbf{A}^T\mathbf{y} = 0 \bmod q\right\}$$

Les deux réseaux sont de dimension m. Le premier est engendré par les lignes de $\mathbf{A}$ et a pour déterminant q^{m-n}, tandis que le second contient tous les vecteurs orthogonaux aux lignes de $\mathbf{A}$ et a pour déterminant q^n.
De plus, ils sont liés par la dualité des réseaux, c'est-à-dire :

$$\mathscr{L}_q^\perp(\mathbf{A}^T) = q \cdot \mathscr{L}_q(\mathbf{A})^\vee \quad \text{et} \quad \mathscr{L}_q(\mathbf{A}) = q \cdot \mathscr{L}_q^\perp(\mathbf{A}^T)^\vee.$$

1.3 *Quelques problèmes algorithmiques liés aux réseaux euclidiens*

Cette section s'inspire largement des travaux de Boudgoust (2023), auxquels le lecteur intéressé pourra se référer pour un traitement plus approfondi. On définira deux problèmes algorithmiques important sur les réseaux euclidiens. Il y en a beaucoup plus, (Stephens-Davidowitz 2015) donne un aperçu des réductions entre ces problèmes.

1.3.1 *Des problèmes faciles*

Certains problèmes liés aux réseaux euclidiens sont relativement simples à résoudre, notamment la vérification de l'appartenance d'un vecteur à un réseau donné, ou encore la décision de l'égalité de deux bases de réseaux. Le lecteur intéressé pourra s'exercer en tentant de résoudre ces problèmes.

> ***Adhésion***
> Étant donné une base B d'un réseau $\mathscr{L}$, et $\mathbf{v} \in \mathbb{R}^n$, décider si $\mathbf{v} \in \mathscr{L}$.

> ***Équivalence***
> Étant donné deux bases B et B', décider si $\mathscr{L}(B) = \mathscr{L}(B')$,

1.3.2 *Le problème du vecteur le plus court*

Considérons le problème suivant, paramétré par la dimension n du réseau :

> ***Shortest Vector Problem (SVP), NP-complet**, (Ajtai 1996).*
> Étant donné une base B d'un réseau $\mathscr{L}$, trouver un vecteur $\mathbf{v} \neq \mathbf{0}$ tel que $\|\mathbf{v}\|_2 = \lambda_1(\mathscr{L})$.

On ne connaît que des algorithmes demandant au moins un nombre exponentiel d'opérations pour résoudre ce problème, même en utilisant des algorithmes quantiques. Les algorithmes de type **énumération**[9] et les algorithmes de type **crible**[10] se démarquent pour ce problème. Le calcul d'un plus court vecteur dans un réseau euclidien de $\mathbb{R}^n$ est en général un problème difficile qui sert de fondation à de nombreuses primitives cryptographiques. On s'intéresse souvent à la version approximative :

> ***SIVP**$_\gamma$, où $\gamma > 0$*
> Étant donné une base B du réseau $\mathscr{L}$, trouver des vecteur $\mathbf{v}_1, \dots \mathbf{v}_n \neq \mathbf{0}$ linéairement indépendants tel que $\|\mathbf{v}_i\|_2 \leq \gamma \cdot \lambda_i(\mathscr{L})$ pour tout i.

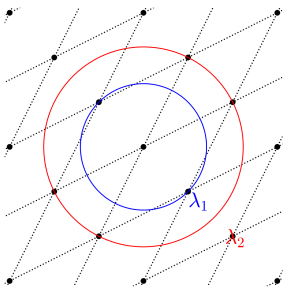


Figure 1.5 – Une instance de SIVP.

9. ils énumèrent tous les vecteurs du réseau qui sont dans une certaine boule bien choisie, en pratique ils sont utilisés jusqu'aux dimensions $n \approx 80$. On peut leur ajouter des optimisations et des heuristiques.

10. on génère deux listes d'éléments du réseau, puis on construit la liste de toutes les différences entre les éléments des deux listes. On espère obtenir des vecteurs plus court. On recommence le procédé. Le temps d'exécution est en $2^{\mathcal{O}(n)}$.

L'état des connaissances actuelles est le suivant :

- Pour $\gamma = \mathcal{O}(1)$, le problème est prouvé **NP-complet**, (AJTAI 1996).
- Pour $\gamma = \text{poly}(n)$, il existe des algorithmes en **temps exponentiel**.
- Pour $\gamma = 2^{\mathcal{O}(n)}$, l'algorithme **LLL** (LENSTRA 1982) permet de le résoudre en **temps polynomial**.

GapSVP$_\gamma$*, où* $\gamma > 0$
Étant donné une base B du réseau $\mathscr{L}$, et $r \in \mathbb{R}_+^*$. Décider si $\lambda_1(\mathscr{L}) \leq r$ (instance positive) ou $\lambda_1(\mathscr{L}) > \gamma \cdot r$ (instance négative).

Théorème 1.4 (BANASZCZYK 1993,STEPHENS-DAVIDOWITZ 2015)**.** .
SVP_γ n'est pas plus simple que GapSVP_γ.

Conjecture 1.1. Il n'existe aucun algorithme classique ou quantique en temps polynomial qui approxime les problèmes de réseaux SVP_γ, GapSVP_γ ou à un facteur polynomial près γ (pour tous les réseaux d'entrée possibles).

1.3.3 *Le problème du vecteur le plus proche*

Un autre problème important concerne la recherche de vecteurs proches d'une cible dans un réseau.

Closest Vector Problem (CVP), NP-complet*, (AJTAI 1996).*
Étant donnés $t \in \mathbb{R}^n$, un réseau $\mathscr{L}(B)$, trouver $\mathbf{v} \in \mathscr{L}$ tel que

$$\|t - \mathbf{v}\|_2 = d(t, \mathscr{L}) := \min_{v \in \mathscr{L}} \{\|t - \mathbf{v}\|_2\}.$$

Le problème CVP est en général difficile pour un réseau arbitraire. Cependant, pour certaines familles spécifiques de réseaux, comme $\mathbb{Z}^n$, des algorithmes en temps polynomial sont connus. La qualité de la base choisie joue un rôle crucial dans la résolution du problème. De même, on peut considérer une version approximative :

CVP$_\gamma$*,* $\gamma > 0$
Étant donnés $t \in \mathbb{R}^n$, un réseau $\mathscr{L}(B)$, trouver $\mathbf{v} \in \mathscr{L}$ tel que $\|t - \mathbf{v}\|_2 \leq \gamma \cdot d(t, \mathscr{L})$.

GapCVP$_\gamma$*,* $\gamma > 0$
Étant donné $r \in \mathbb{R}_+^*$, $t \in \mathbb{R}^n$, un réseau $\mathscr{L}(B)$. Décider si il existe $\mathbf{v} \in \mathscr{L}$ tel que $\|t - \mathbf{v}\|_2 \leq r$ (instance positive) ou $\|t - \mathbf{v}\|_2 > \gamma \cdot r$ (instance négative)

Théorème 1.5 (GOLDREICH et al. 1999)**.** .
GapSVP_γ se réduit à GapCVP_γ en temps polynomial.

Théorème 1.6. Il existe un algorithme qui résout $\mathbf{CVP}_{\exp(n)}$ en temps polynomial via l'algorithme **LLL**.

L'efficacité des algorithmes dépend grandement de la qualité de la base du réseau euclidien choisie. Le chapitre sur la réduction abordera des techniques pour améliorer la base, via l'algorithme **LLL**. [11]

11. En dimension fixée, résoudre **exactement** le problème **SVP** pour la norme $\|\cdot\|_\infty$ fournit en fait une γ-approximation (avec γ dépendant de la dimension) pour le problème **SVP** dans la norme $\|\cdot\|_2$. Il existe notamment une constante C telle que $\|v\|_2 \leq C\|v\|_\infty$ pour tout $v \in \mathbb{R}^n$. Ainsi, un vecteur minimisant $\|v\|_\infty$ donne un vecteur $\sqrt{n}$-proche du vecteur réellement le plus court en norme euclidienne.

CHAPITRE **2**

Réduction de réseaux polynomiaux

DANS ce chapitre, nous nous concentrerons sur un algorithme de réduction de réseaux euclidiens s'exécutant en temps polynomial : l'algorithme LLL, du nom de ses auteurs A. Lenstra, H. Lenstra et L. Lovász. L'algorithme LLL possède de nombreuses applications, notamment en cryptanalyse de schémas basés sur le problème du sac à dos, en factorisation efficace de polynômes, ou encore dans le calcul rapide de décompositions en forme normale d'Hermite (HNF). Le lecteur pourra consulter (HAVAS, MAJEWSKI et MATTHEWS 1998) pour un aperçu plus complet de ses usages.
Nous ne traiterons pas d'autres algorithmes de réductions de réseaux euclidiens comme BKZ, afin de rester dans un cadre plus élémentaire. L'algorithme LLL repose sur une idée simple mais puissante : il produit une approximation entière de la décomposition de Gram-Schmidt et réorganise les vecteurs de la base pour en améliorer la structure. Un rappel sur le procédé d'orthogonalisation de Gram-Schmidt est dans l'annexe dédiée. Celle-ci ne sera pas rappelée ici afin de préserver la concision du texte. Au cours de mon stage j'ai réalisé une présentation de l'algorithme LLL, ainsi qu'une idée rapide de sa preuve, que vous pourrez trouver ici.

2.1 *La base réduite*

Dans les problèmes liés aux réseaux euclidiens, une base orthogonale représenterait une base idéale.

> ***Problème***
> La base B^* (de $\mathbb{R}^n$) n'est généralement pas une base du réseau $\mathscr{L}(B)$.

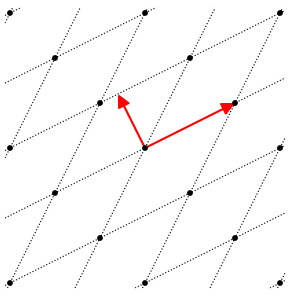


FIGURE 2.1 – Base de Gram-Schmidt qui n'est pas une base de $\mathscr{L}$.

Nous cherchons une base de $\mathscr{L}$ qui *approxime* la base de Gram–Schmidt aussi fidèlement que possible.

Notation. On définit l'entier le plus proche de $x \in \mathbb{R}$ par $\lceil x \rfloor = \lfloor x + 1/2 \rfloor$.

Proposition 2.1. On a $|x - \lceil x \rfloor| \leq \frac{1}{2}$ pour tout $x \in \mathbb{R}$.

Dans la suite de cette explication, le lecteur est invité à porter une attention particulière aux * qui dénotent un vecteur de la base de Gram-Schmidt. L'opération d'orthogonaliation de Gram-Schmidt nous donne :

$$\mathbf{b}_2^* \coloneqq \mathbf{b}_2 - \mu_{2,1}\mathbf{b}_1^* \notin \mathscr{L} \tag{2.1}$$

Mais $\mathbf{b}_1^* = \mathbf{b}_1 \in \mathscr{L}$, en prenant $k \in \mathbb{Z}$, on a $\mathbf{b}_2 - k\mathbf{b}_1 \in \mathscr{L}$, on pourrait donc réaliser l'opération

$$\mathbf{b}_2 \coloneqq \mathbf{b}_2 - k\mathbf{b}_1 \in \mathscr{L}$$

On peut donc choisir $k \in \mathbb{Z}$ qui minimise $\langle \mathbf{b}_2, \mathbf{b}_1 \rangle$, ce qui est réalisé par $k \coloneqq \lceil \mu_{i,j} \rfloor$. On en déduit donc l'opération

$$\mathbf{b}_2 \coloneqq \mathbf{b}_2 - \lceil \mu_{2,1} \rfloor \, \mathbf{b}_1 \in \mathscr{L} \tag{2.2}$$

On peut donc étendre ce procédé par récurrence pour construire la nouvelle famille $(\mathbf{b}_i)_{1 \leq i \leq n}$.
En refaisant les calculs on peut montrer que la base de Gram-Schmidt associée est inchangée, mais que les nouveaux coefficients $|\mu_{i,j}| < \frac{1}{2}$ d'après la proposition 2.1.

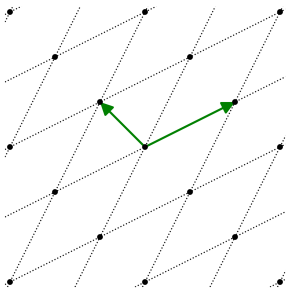


FIGURE 2.2 – Base de $\mathscr{L}$ proche de la base Gram-Schmidt .

Définition 2.1. Soit $(\mathbf{b}_i)_{1 \leq i \leq n}$ une base d'un réseau et U la matrice triangulaire supérieure telle que $B = UB^*$. est dite **propre** [1] si

$$\max_{1 \leq i < j \leq n} |\mu_{i,j}| \leq \frac{1}{2}. \tag{2.3}$$

Contre exemple. Bien que la proprification impose une certaine contrainte sur les coefficients de projection, elle ne garantit pas à elle seule que les vecteurs de la base soient presque orthogonaux.

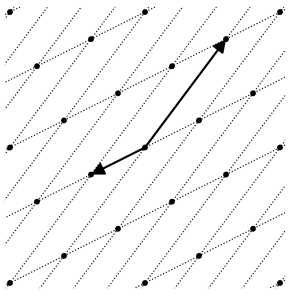


FIGURE 2.3 – Base propre mais peu orthogonale

Idéalement, nous souhaiterions trouver une base $(\mathbf{b}_i)_{1 \leq i \leq n}$ du réseau $\mathscr{L}$ telle que :

$$\|\mathbf{b}_1\| = \lambda_1(\mathscr{L}), \quad \|\mathbf{b}_2\| = \lambda_2(\mathscr{L}), \quad \ldots, \quad \|\mathbf{b}_n\| = \lambda_n(\mathscr{L})$$

Ceci implique $\|\mathbf{b}_1\| \leq \cdots \leq \|\mathbf{b}_n\|$, mais cela est trop difficile de trouver une telle base car cela reviendrait à résoudre SIVP.

1. Une base propre est aussi connue sous le nom de base size-réduite dans la littérature.

Définition 2.2. Une base $(\mathbf{b}_i)_{1\leq i\leq m}$ satisfait la **condition de Lovász**[2] si :

$$\|\mathbf{b}_i^*\|^2 \leq 2\|\mathbf{b}_{i+1}^*\|^2 \quad \text{pour tout } 1 \leq i < n$$

Remarque. On peut interpréter cette condition comme une forme de quasi-croissance des normes $\|\mathbf{b}_i^*\|$: elle n'exige pas que celles-ci soient strictement croissantes, mais impose que toute éventuelle décroissance soit contrôlée, autrement dit, qu'elles ne décroissent pas trop rapidement.

Dès lors, il est naturel de se demander pourquoi ne pas échanger les vecteurs lorsque cette condition de Lovász n'est pas satisfaite.

Exemple. Si l'on applique cette idée à l'exemple précédent, alors après permutation des vecteurs concernés et une nouvelle phase de réduction, on obtient à nouveau une base améliorée :

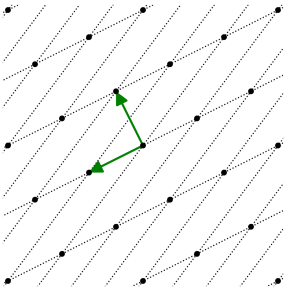


FIGURE 2.4 – Nouvelle base size-réduite

Définition 2.3. Une base $\mathcal{B}$ d'un réseau $\mathscr{L}(\mathcal{B})$ est dite **LLL-réduite** si

- $\mathcal{B}$ est propre.
- $\mathcal{B}$ satisfait la condition de Lovàsz.

Remarque. Chaque vecteur de la base réduite a une norme au moins égale à la moitié de celle du précédent, garantissant ainsi une décroissance modérée.

Théorème 2.1. Soit $\mathcal{B}$ une base réduite du réseau $\mathscr{L} \subseteq \mathbb{R}^n$ et soit $v \in \mathscr{L} \setminus \{0\}$. Alors

$$\|\mathbf{b}_1\| \leq 2^{(n-1)/2} \cdot \|\mathbf{v}\|$$

En particulier, ce résultat s'applique à un vecteur $\mathbf{v} \in \mathscr{L}$ de plus petite norme non nulle, c'est-à-dire un vecteur atteignant $\lambda_1(\mathscr{L})$. On en déduit donc :

$$\|\mathbf{b}_1\| \leq 2^{(n-1)/2} \cdot \lambda_1(\mathscr{L}),$$

ce qui montre que LLL fournit en temps polynomial un vecteur de norme à un facteur $2^{(n-1)/2}$ près du plus court vecteur du réseau. Autrement dit, l'algorithme LLL résout approximativement le problème du plus court vecteur (SVP) avec un facteur d'approximation $\gamma = 2^{(n-1)/2}$. Par extension, en renvoyant les vecteurs de la base réduite, LLL permet également de résoudre le problème SIVP (*Shortest Independent Vectors Problem*) avec le même facteur d'approximation.

2.2 *Fonctionnement et exemple*

Nous présentons à présent l'algorithme de Lenstra–Lenstra–Lovász (LLL), dans sa forme classique. Originellement LLL est apparu dans l'article de 1982 et servait à factoriser des polynômes à coefficients rationnels.

2. Une condition plus générale : $(\delta - \mu_{i+1,i}^2)\,\|\mathbf{b}_i^*\|^2 \leq \|\mathbf{b}_{i+1}^*\|^2$ pour $1 \leq i \leq n$, où $\delta \in \left]\frac{1}{4}, 1\right]$

Algorithme 1 : *LLL*

Entrée : Une base $B = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$
Sortie : Une base réduite $G = (\mathbf{g}_1, \ldots, \mathbf{g}_n)$ de B

```
1  Pour i = 1 à n faire
2      g_i ← b_i
3  (B*, U) ← Gram-Schmidt (B)
4  Tant que i ≤ n faire
5      Pour j = i − 1, i − 2, ..., 1 faire
6          g_i ← g_i − ⌈μ_{i,j}⌋ g_j
7          Mettre à jour B*, U
8      Si i > 1 et ‖g*_{i−1}‖² > 2‖g*_i‖² alors
9          Échanger g_{i−1} et g_i
10         Mettre à jour B*, U
11         i ← i − 1
12     Sinon
13         i ← i + 1
14 Retourner G = (g_1, ..., g_n)
```

L'algorithme débute par le calcul de la base orthogonalisée de Gram–Schmidt (**ligne 3**), qui sert de support aux opérations de réduction. Le principe général repose sur l'application répétée de deux types d'étapes : des *réductions de taille* (**lignes 5 à 7**) visant à raccourcir les vecteurs sans sortir du réseau, et des *permutations* de vecteurs (**ligne 8**) effectuées lorsque la condition de Lovász, qui contrôle la décroissance des normes, n'est pas satisfaite.

L'ensemble de ces opérations est imbriqué dans une boucle `while` qui se répète tant qu'une condition de progression n'est pas remplie. Nous verrons que cette condition de Lovász garantit non seulement une amélioration à chaque étape, mais également la terminaison de l'algorithme en un nombre fini d'itérations.

Exemple. Soit

$$B = \begin{pmatrix} 1 & 1 & 1 \\ -1 & 0 & 2 \\ 3 & 5 & 6 \end{pmatrix} \in GL_3(\mathbb{Z}).$$

On a $|\mathscr{L}(B)| = 9, \quad A = \max_{1 \le i \le 3} \|\mathbf{b}_i\| = 70$

On va essayer d'estimer un plus court vecteur L'algorithme *LLL* commence par calculer la décomposition de Gram-Schmidt de B, pour plus de détails sur le calcul de cette décomposition, ce calcul est effectué dans l'annexe Rappels d'algèbres linéaire.

On obtient la décomposition

$$U = \begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ \frac{14}{3} & \frac{13}{14} & 1 \end{pmatrix}, \quad B^* = \begin{pmatrix} 1 & 1 & 1 \\ -\frac{4}{3} & -\frac{1}{3} & \frac{5}{3} \\ -\frac{3}{7} & \frac{9}{14} & -\frac{3}{14} \end{pmatrix}.$$

Voici un tableau récapitulant les principales étapes de l'algorithme. Le tableau est volontairement détaillé et fourni, le lecteur pourra revenir sur cet exemple pour comprendre ce que sont d_1, d_2, D ou la signification de $\mathrm{Gram}(G)$.

	G	U	G^*	d_1, d_2 D	$\begin{pmatrix}\|\mathbf{g}_1^*\|^2\\ \|\mathbf{g}_2^*\|^2\\ \|\mathbf{g}_3^*\|^2\end{pmatrix}$	$\mathrm{Gram}(G)$
	$\begin{pmatrix}1 & 1 & 1\\ -1 & 0 & 2\\ 3 & 5 & 6\end{pmatrix}$	$\begin{pmatrix}1 & 0 & 0\\ \frac{1}{3} & 1 & 0\\ \frac{14}{3} & \frac{13}{14} & 1\end{pmatrix}$	$\begin{pmatrix}1 & 1 & 1\\ -\frac{4}{3} & -\frac{1}{3} & \frac{5}{3}\\ -\frac{3}{7} & \frac{9}{14} & -\frac{3}{14}\end{pmatrix}$	$3, 14$ 42	$\begin{pmatrix}3\\ \frac{14}{3}\\ \frac{9}{14}\end{pmatrix}$	$\begin{pmatrix}3 & 1 & 14\\ 1 & 1 & 9\\ 14 & 9 & 70\end{pmatrix}$
proprification $\mathbf{g}_3$	$\begin{pmatrix}1 & 1 & 1\\ -1 & 0 & 2\\ 0 & 1 & 0\end{pmatrix}$	$\begin{pmatrix}1 & 0 & 0\\ \frac{1}{3} & 1 & 0\\ \frac{1}{3} & \frac{-1}{14} & 1\end{pmatrix}$	$\begin{pmatrix}1 & 1 & 1\\ -\frac{4}{3} & -\frac{1}{3} & \frac{5}{3}\\ -\frac{3}{7} & \frac{9}{14} & -\frac{3}{14}\end{pmatrix}$	$3, 14$ 42	$\begin{pmatrix}3\\ \frac{14}{3}\\ \frac{9}{14}\end{pmatrix}$	$\begin{pmatrix}3 & 1 & 1\\ 1 & 5 & 0\\ 1 & 0 & 1\end{pmatrix}$
Lovasz $\mathbf{g}_2 \leftrightarrow \mathbf{g}_3$	$\begin{pmatrix}1 & 1 & 1\\ 0 & 1 & 0\\ -1 & 0 & 2\end{pmatrix}$	$\begin{pmatrix}1 & 0 & 0\\ \frac{1}{3} & 1 & 0\\ \frac{1}{3} & \frac{-1}{2} & 1\end{pmatrix}$	$\begin{pmatrix}1 & 1 & 1\\ -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3}\\ -\frac{3}{2} & 0 & \frac{3}{2}\end{pmatrix}$	$3, 2$ 6	$\begin{pmatrix}3\\ \frac{2}{3}\\ \frac{9}{2}\end{pmatrix}$	$\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ \frac{1}{3} & \frac{-1}{2} & 1\end{pmatrix}$
Lovasz $\mathbf{g}_1 \leftrightarrow \mathbf{g}_2$	$\begin{pmatrix}0 & 1 & 0\\ 1 & 1 & 1\\ -1 & 0 & 2\end{pmatrix}$	$\begin{pmatrix}1 & 0 & 0\\ 1 & 1 & 0\\ \frac{1}{3} & \frac{-1}{2} & 1\end{pmatrix}$	$\begin{pmatrix}0 & 1 & 0\\ 1 & 0 & 1\\ -\frac{3}{2} & 0 & \frac{3}{2}\end{pmatrix}$	$1, 2$ 2	$\begin{pmatrix}1\\ 2\\ \frac{9}{2}\end{pmatrix}$	$\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ \frac{1}{3} & \frac{-1}{2} & 1\end{pmatrix}$
proprification $\mathbf{g}_2$	$\begin{pmatrix}0 & 1 & 0\\ 1 & 0 & 1\\ -1 & 0 & 2\end{pmatrix}$	$\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ \frac{1}{3} & \frac{-1}{2} & 1\end{pmatrix}$	$\begin{pmatrix}0 & 1 & 0\\ 1 & 0 & 1\\ -\frac{3}{2} & 0 & \frac{3}{2}\end{pmatrix}$	$1, 2$ 2	$\begin{pmatrix}1\\ 2\\ \frac{9}{2}\end{pmatrix}$	$\begin{pmatrix}1 & 0 & 0\\ 0 & 2 & 1\\ 0 & 1 & 5\end{pmatrix}$

On obtient la base LLL réduite :

$$\mathbf{g}_{reduced} = \begin{pmatrix}0 & 1 & 0\\ 1 & 0 & 1\\ -1 & 0 & 2\end{pmatrix} \in M_3(\mathbb{Z})$$

Remarque.

- La valeur d_3 ne nous intéresse pas car il s'agit d'un invariant,
- $Gram(G)$ se rapproche petit a petit d'une matrice diagonale.

CHAPITRE **3**

Correction, terminaison et complexité de LLL

Le lecteur pourra choisir de passer ce chapitre en première lecture. Les preuves et les résultats secondaires (lemmes et démonstrations annexes) sont regroupés en annexe afin d'alléger la lecture. Dans tous les cas, il suffira de garder à l'esprit que la terminaison de l'algorithme LLL repose sur la décroissance d'une quantité notée D, dont l'étude sera essentielle dans la suite. Le lecteur pourra également, s'il le souhaite, consulter directement les résultats relatifs à la complexité algorithmique de LLL.

3.1 *Correction*

Théorème 3.1 (Correction)**.** L'algorithme *LLL* calcule une base réduite de $\mathscr{L}$.

Algorithme 2 : *Proprification de* $\mathbf{g}_i$

Pour $j = i-1, i-2, \ldots, 1$ **faire**
  Proprification partielle de $\mathbf{g}_i$

Lemme 3.1. *Proprification de* $\mathbf{g}_i$ ne change pas G^* et à la fin on a

$$|\mu_{i,l}| \leq \frac{1}{2} \text{ pour } 1 \leq l < i.$$

On étudie maintenant l'étape de réduction, l'idée consiste à réorganiser les vecteurs afin de garantir une progression quantifiable, qui assurera la terminaison de LLL.

Algorithme 3 : *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$

Si $i > 1$ ***et*** $\left\|\mathbf{g}_{i-1}^*\right\|^2 > 2\left\|\mathbf{g}_i^*\right\|^2$ **alors**
  Échanger $\mathbf{g}_{i-1}$ et $\mathbf{g}_i$
  Mettre à jour (B^*, U)
  $i \leftarrow i - 1$
Sinon
  $i \leftarrow i + 1$

Lemme 3.2. Supposons que $\mathbf{g}_{i-1}$ et $\mathbf{g}_i$ sont échangés à l'étape *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$.
On note h_k les vecteurs après échange et h_k^* leur base orthogonale de Gram-Schmidt.
Alors

1. $\mathbf{h}_k^* = \mathbf{g}_k^*$ pour tout $k \in \{1, \ldots, n\} \setminus \{i-1, i\}$,
2. $\|\mathbf{h}_{i-1}^*\|^2 < \frac{3}{4}\|\mathbf{g}_{i-1}^*\|^2$,
3. $\|\mathbf{h}_i^*\| \leq \|\mathbf{g}_{i-1}^*\|$.

Algorithme 4 : *LLL*

Tant que $i \leq n$ **faire**
  Proprification de $\mathbf{g}_i$
  Réduction de $\mathbf{g}_{i-1}, \mathbf{g}_i$

Lemme 3.3. Au début de chaque itération de la boucle à l'étape *LLL*, les invariants suivants sont vérifiés :

$$|\mu_{k,l}| \leq \frac{1}{2} \quad \text{pour } 1 \leq l < k < i, \qquad \|\mathbf{g}_{k-1}^*\|^2 \leq 2\|\mathbf{g}_k^*\|^2 \quad \text{pour } 1 < k < i.$$

3.2 *Terminaison et complexité*

Théorème 3.2 (Terminaison et complexité)**.** On pose $A = \max_{1 \leq i \leq n} \|\mathbf{g}_i\|$. L'algorithme *LLL* termine et utilise $\mathcal{O}(n^4 \log A)$ opérations arithmétiques sur des entiers.

La difficulté est de montrer que la boucle Tant que ne va pas s'exécuter indéfiniment.

Lemme 3.4.

1. Orthogonalisation de Gram-Schmidt nécessite $\mathcal{O}(n^3)$ opérations dans $\mathbb{Z}$.
2. *Proprification de* $\mathbf{g}_i$ nécessite $\mathcal{O}(n^2)$ opérations dans $\mathbb{Z}$.
3. *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$ nécessite $\mathcal{O}(n)$ opérations dans $\mathbb{Z}$.

Il reste à borner le nombre d'itérations de la boucle Tant que à l'étape *LLL*.
Pour tout $1 \leq k \leq n$, on pose

$$\mathbf{g}_k = \begin{pmatrix} \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_k \end{pmatrix} \in \mathbb{Z}^{k \times n}, \quad d_0 = 1, \quad d_k = \det(\mathbf{g}_k \cdot \mathbf{g}_k^T) \in \mathbb{Z}.$$

Lemme 3.5. Pour tout $1 \leq k \leq n$, on a :

$$d_k = \prod_{1 \leq l \leq k} \|\mathbf{g}_l^*\|^2 > 0.$$

Lemme 3.6.

1. *Proprification de* $\mathbf{g}_i$ ne change pas d_k pour tout $1 \leq k \leq n$.
2. Si $\mathbf{g}_{i-1}$ et $\mathbf{g}_i$ sont échangés à l'étape *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$, et si d_k^* désigne la nouvelle valeur de d_k, alors :
$$d_k^* = d_k \quad \text{pour tout } k \neq i-1, \quad \text{et} \quad d_{i-1}^* \leq \frac{3}{4} d_{i-1}.$$

Preuve.

1. D'après le lemme 2.2 *Proprification de* $\mathbf{g}_i$ ne modifie pas $\mathbf{g}_k^*$ et donc ne modifie pas d_k.
2. Pour $k \neq i-1$, une exécution de *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$ multiplie $\mathbf{g}_k$ par une matrice de permutation, donc $d_k^* = d_k$
   De plus, on a
$$d_{i-1} \overset{(2.6)}{=} \prod_{1 \leq l \leq i-1} \|\mathbf{g}_l^*\|^2 \overset{(2.3)}{\leq} \frac{3}{4} \prod_{1 \leq l \leq i-1} \|\mathbf{h}_l^*\|^2 \overset{(2.6)}{=} \frac{3}{4} d_{i-1}^*$$

■

On pose

$$D = \prod_{1 \leq k < n} d_k, \quad A = \max_{1 \leq i \leq n} \|\mathbf{g}_i\|$$

On désigne D_0 désigne la valeur de D au début de l'algorithme, on a $1 \leq D \in \mathbb{Z}$ et

$$\begin{aligned} D_0 &= \|\mathbf{g}_1^*\|^{2(n-1)} \|\mathbf{g}_2^*\|^{2(n-2)} \cdots \|\mathbf{g}_{n-1}^*\|^2 \\ &\leq \|\mathbf{g}_1\|^{2(n-1)} \|\mathbf{g}_2\|^{2(n-2)} \cdots \|\mathbf{g}_{n-1}\|^2 \\ &\leq A^{n(n-1)} \end{aligned}$$

Puisque g_i^* est une projection de g_i pour tout i.

Lemme 3.7.

1. *Proprification de* $\mathbf{g}_i$ ne modifie pas D.
2. D diminue d'au moins un facteur $3/4$ si un échange a lieu dans *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$

Preuve.

1. D'après le lemme 2.7 *Proprification de* $\mathbf{g}_i$ ne modifie pas d_k et donc ne modifie pas D.
2. Si $\mathbf{g}_{i-1}$ et $\mathbf{g}_i$ sont échangés lors de l'exécution de *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$, en notant D^* la nouvelle valeur de D, alors d'après le lemme 2.7

$$d_k^* = d_k, \quad d_{i-1}^* \leq \frac{3}{4} d_{i-1} \text{ donc } D^* \leq \frac{3}{4} D.$$

■

À tout moment de l'algorithme, soit $e \in \mathbb{N}$ le nombre d'échanges effectués jusqu'à présent, et e^* le nombre de fois où la branche alternative (le *else*) dans *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$ a été prise.

Lemme 3.8. On a

$$e \leq \log_{4/3} D_0 \in \mathcal{O}(n^2 \log A)$$

Preuve. Soit D_e la valeur de D après e échanges.
On doit avoir

$$1 \ \leq \ D_e \ \leq \ \left(\frac{3}{4}\right)^e D_0 \leq \ \left(\frac{3}{4}\right)^e A^{n(n-1)}.$$

En appliquant $log_{3/4}$, aux extrémités de l'inégalité.

$$0 = \log_{3/4}(1) \ \geq \ e + \log_{3/4}(A^{n(n-1)}) = e + n(n-1)\frac{\log A}{\log(3/4)}.$$

On en déduit que $e \ \leq n(n-1)\frac{\log A}{-\log(3/4)}$ et donc $e \in \mathcal{O}(n^2 \log A)$

■

Preuve de la terminaison et la complexité.
Comme i est décrémenté de 1 lors d'un échange et incrémenté de 1 sinon l'entier $i + e - e^*$ est constant tout au long de *LLL*.
Initialement $i + e - e^* = 2$ et à la fin de *LLL* on a $n + 1 + e - e^* = 2$. On en déduit donc que $e + e^* = 2e + n - 1 \in \mathcal{O}(n^2 \log A)$. et donc d'après le lemme 2.5 le coût total de *LLL* est $\mathcal{O}(n^2 \times n^2 \log A)$ opérations dans $\mathbb{Z}$. Ce qui acheve la preuve. ■

État de l'art de la réduction de réseaux euclidiens

Voici une rétrospective structurée des avancées majeures dans la réduction de réseaux euclidiens.

1982 LLL (Lenstra, Lenstra, Lovász) (Lenstra 1982) introduisent le premier algorithme de réduction polynomial, basé sur une combinaison de *size reduction* et d'une condition de Lovász. Il garantit :

$$\|\mathbf{b}_1\| \leq (4/3)^{(n-1)/2} \cdot \lambda_1(\mathscr{L}), \quad \text{avec complexité binaire } \mathcal{O}(n^5\beta^2)$$

1991 BKZ (Schnorr et Euchner 1994) introduit une approche par blocs. L'algorithme applique un solveur SVP de petite dimension β à des sous-blocs de la base :

$$\|\mathbf{b}_1\| \leq \gamma_\beta^{(n-1)/(\beta-1)} \cdot \lambda_1(\mathscr{L})$$

Le coût est exponentiel en β mais reste efficace pour $\beta \leq 40$ en pratique.

2009 L2 (Nguyen et Stehlé 2009) améliore LLL sur le plan de la stabilité numérique, sans gain théorique majeur sur la qualité de la base. Complexité similaire à LLL, mais plus efficace pour des entrées en flottants.

2011 $\tilde{L}_1$ (Novocin, Stehlé et Villard 2011) propose une version rapide de LLL inspirée du GCD rapide de Knuth–Schönhage. Il introduit une stratégie récursive appelée *Lift-Reduction* et atteint une complexité quasi-linéaire :

$$\mathcal{O}\left(d^{5+\varepsilon}\beta + d^{\omega+1+\varepsilon}\beta^{1+\varepsilon}\right)$$

avec une qualité comparable à LLL : $\|\mathbf{b}_1\| \leq 2^{\alpha n} \cdot |\mathscr{L}|^{1/n}$.

2011 Terminating BKZ (Hanrot, Pujol et Stehlé 2011) propose une modélisation dynamique affine de BKZ. Ils montrent que même interrompu prématurément, BKZ garantit :

$$\|\mathbf{b}_1\| \leq 2^{\frac{\gamma_\beta(n-1)}{2(\beta-1)}+\frac{3}{2}} \cdot |\mathscr{L}|^{1/n}$$

après seulement $\mathcal{O}(n^3/\beta^2 \cdot \log \|B\|)$ appels à un solveur SVP.

2019 KEF (Kirchner, Espitau et Fouque 2021)propose un algorithme heuristique récursif exploitant la *Geometric Series Assumption* (GSA) pour guider la réduction. Il utilise des techniques de FFT et obtient une complexité heuristique :

$$\widetilde{\mathcal{O}}(n^\omega \cdot \log \kappa(B))$$

avec une qualité empirique équivalente à BKZ en grande dimension ($n > 2000$).

2023 Iterated Compression (Ryan et Heninger 2023) présente un algorithme récursif fondé sur des opérations de compression stables, une métrique de *drop*, et des profils dynamiques :

$$\|\mathbf{b}_1\| \leq 2^{\alpha n} \cdot |\mathscr{L}|^{1/n}, \quad \|\mathbf{b}_n^*\| \geq 2^{-\alpha n} \cdot |\mathscr{L}|^{1/n}$$

avec complexité heuristique :

$$\mathcal{O}(n^\omega(C+n)^{1+\varepsilon}), \quad C = \log(\|B\| \cdot \|B^{-1}\|)$$

CHAPITRE **4**

Réseaux polynomiaux

LORSQUE les coefficients des matrices appartiennent à un corps $\mathbb{K}$, les opérations classiques telles que la multiplication, l'inversion, le calcul du déterminant ou la résolution de systèmes linéaires possèdent des complexités comparables. En revanche, lorsqu'on considère des matrices à coefficients dans l'anneau $\mathbb{K}[x]$, des différences apparaissent : si le calcul du déterminant conserve la même complexité que celle du produit matriciel, l'inversion est plus coûteuse. Cette complexité découle de la structure de l'anneau $\mathbb{K}[x]$: bien qu'il s'agisse d'un anneau principal (à la différence de $\mathbb{K}[x, y]$), il ne s'agit pas d'un corps. Ainsi, certaines opérations, comme l'inversion, ne sont plus systématiquement réalisables. Notamment, dans $\mathbb{K}[x]$, seuls les polynômes constants non nuls sont inversibles. Cette restriction impose de repenser et redéfinir rigoureusement plusieurs notions de l'algèbre linéaire. Les matrices à coefficients dans $\mathbb{K}[x]$ sont essentielles dans de nombreuses applications. [1]. Ce chapitre vise à explorer les réseaux polynomiaux et leurs propriétés spécifiques. Ce mémoire avait pour ambition initiale de motiver rigoureusement l'introduction des bases réduites dans le cadre des matrices polynomiales. Toutefois, afin de ne pas aborder un sujet trop éloigné des objectifs du stage, et par souci de concision, nous ne détaillerons pas ici les aspects liés aux mesures de complexité. Ce que j'avais rédigé initialement se retrouve en annexe. Notons simplement qu'il est essentiel d'analyser finement le comportement des matrices polynomiales vis-à-vis des opérations algébriques usuelles, en particulier la multiplication. Cela justifie l'introduction de la notion de degré de ligne, qui jouera un rôle central dans le chapitre suivant.

4.1 *Définitions et exemples*

On commence par introduire la notion de réseau polynomial. Des rappels détaillés sur les anneaux et sur les modules sont dans l'annexe correspondante.

Définition 4.1. Un **réseau polynomial** $\mathscr{L}$ est un $\mathbb{K}[x]$-module libre de type fini.

On peut montrer qu'il existe une famille $\mathbb{K}[x]$-libre maximale $(\mathbf{b}_i)_{1\leq i\leq m}$ dans $\mathscr{L}$ telle que

$$\mathscr{L} = \bigoplus_{1\leq i\leq m} \mathbb{K}[x]\mathbf{b}_i := \{a_1\mathbf{b}_1 + \cdots a_m\mathbf{b}_m : a_i \in \mathbb{K}[x]\}$$

Cette famille est appelée **base** de $\mathscr{L}$, si on note $B \in \mathbb{K}[x]^{m\times n}$ la matrice de la famille $(\mathbf{b}_i)_{1\leq i\leq m}$ on notera $\mathscr{L}(B)$ le réseau de base B, donc **engendré** par la famille $(\mathbf{b}_i)_{1\leq i\leq m}$. L'entier m est commun à toutes les bases de $\mathscr{L}$ et on l'appelle **rang** de $\mathscr{L}$. Lorsque $n = m$, on dit que le réseau est de **rang plein**. Un élément de $\mathbb{K}[x]^{m\times n}$ est appelé matrice polynomiale.

Exemple.

$$B = \begin{pmatrix} 3x+4 & x^9 \\ 5 & x^2+1 \end{pmatrix} \in \mathbb{R}[x]^{2\times 2}$$

est une matrice polynomiale qui représente le réseau $\mathscr{L}(B)$.

1. Par exemple dans l'interpolation bivariée, une étape centrale du décodage des codes de Reed-Solomon

Proposition 4.1. Soient P et Q deux bases de lignes d'un même $\mathbb{F}[x]$-module libre . Alors, il existe une matrice unimodulaire U telle que

$$P \;=\; U\,Q.$$

On observe une analogie structurelle entre les matrices et les modules : de même que les matrices à coefficients dans $\mathbb{K}$ sont naturellement liées aux $\mathbb{K}$-espaces vectoriels, les matrices à coefficients dans $\mathbb{K}[x]$ interviennent dans l'étude des $\mathbb{K}[x]$-modules libres.

4.2 *Notion de degré en ligne*

Notation. On note $[d]$ un polynôme de degré d. Par exemple x^2+1 sera noté $[2]$, x^9 sera noté $[9]$.

Exemple. La multiplication ne se passe pas forcément bien. On voit que cela donne des bornes non pertinentes et il serait utile de rajoute des critères de mesure du degré. Considérons les matrices de degrés :

$$\begin{pmatrix}[100] & [1]\\ [100] & [1]\end{pmatrix}\begin{pmatrix}[1] & [1]\\ [1] & [1]\end{pmatrix} = \begin{pmatrix}[101] & [101]\\ [101] & [101]\end{pmatrix}$$

Définition 4.2.

- Pour $\mathbf{m} = (m_1, \cdots, m_n) \in \mathbb{K}[x]^{1\times n}$, on définit son **degré en ligne** par :
$$\mathrm{rdeg}(\mathbf{m}) = \max_{1\le i\le n} \deg(m_i) \in \mathbb{Z}$$
- Pour $M = \begin{pmatrix}\cdots \mathbf{M_1} \cdots\\ \vdots \\ \cdots \mathbf{M_n} \cdots\end{pmatrix}, \mathbf{M_i} \in \mathbb{K}[x]^{1\times n}\ \ \forall 1 \le i \le n$, on définit son **degré en ligne** par :
$$\mathrm{rdeg}(M) = (\mathrm{rdeg}(\mathbf{M_i}))_{1\le i\le n} \in \mathbb{Z}^n$$

Exemple. Soit $M = \begin{pmatrix}3x+4 & x^9\\ 5 & x^2+1\end{pmatrix} \in \mathbb{F}_2[x]$. Alors $\mathrm{rdeg}(M) = \begin{pmatrix}\mathrm{rdeg}(\ 3x+4\ ,\ x^9\ )\\ \mathrm{rdeg}(\ 5\ ,\ x^2+1\ )\end{pmatrix} = \begin{pmatrix}9\\ 2\end{pmatrix}$

Cette définition du degré de ligne présente une limite : si $c = bA$, on a bien en général $\mathrm{rdeg}(c) \le \mathrm{rdeg}(b) + \mathrm{rdeg}(A)$, mais cette majoration est souvent trop lâche pour nos besoins. Ce qui nous intéresse est de pouvoir caractériser plus finement le degré de c, voire d'obtenir une égalité. Cela motive l'introduction de la définition de degré décalé.

Définition 4.3. Soit $\vec{\mathbf{s}} = (s_1, \cdots, s_n) \in \mathbb{Z}^n$. On appelle $\vec{\mathbf{s}}$ le **vecteur de décalage**.

- Pour $\mathbf{m} = (m_1, \cdots, m_n) \in \mathbb{K}[x]^{1\times n}$, on définit son **degré en ligne $\vec{\mathbf{s}}$-décalé** par :
$$\mathrm{rdeg}_{\vec{\mathbf{s}}}(\mathbf{m}) = \max_{1\le i\le n} (\deg(m_i) + s_i)$$
- Pour $M = \begin{pmatrix}\cdots \mathbf{M_1} \cdots\\ \vdots \\ \cdots \mathbf{M_n} \cdots\end{pmatrix}, \mathbf{M_i} \in \mathbb{K}[x]^{1\times n}\ \ \forall 1 \le i \le n$, on définit son **degré en ligne décalé** par :
$$\mathrm{rdeg}_{\vec{\mathbf{s}}}(M) = \big(\mathrm{rdeg}_{\vec{\mathbf{s}}}(\mathbf{M_i})\big)_{1\le i\le m} \in \mathbb{Z}^m$$

Notation. Soit $\vec{\mathbf{s}} = (s_1, \cdots, s_n) \in \mathbb{Z}^n$. On note $x^{\vec{\mathbf{s}}}$ la matrice diagonale $\begin{pmatrix}x^{s_1} & & \\ & \ddots & \\ & & x^{s_n}\end{pmatrix} \in M_n(\mathbb{K}[x])$.

Proposition 4.2. Soit $A \in \mathbb{K}[x]^{m\times n}$, et $\vec{\mathbf{s}} \in \mathbb{Z}^n$. Alors $\mathrm{rdeg}_{\vec{\mathbf{s}}}(A) = \mathrm{rdeg}(Ax^{\vec{\mathbf{s}}})$.

Exemple. Soit $M = \begin{pmatrix} 3x+4 & x^9 \\ 5 & x^2+1 \end{pmatrix} \in \mathbb{R}_2[x]$ et $\vec{\mathbf{s}} = (8,0)$

Alors

$$\mathrm{rdeg}_{\vec{\mathbf{s}}}(M) = \mathrm{rdeg}(M \cdot x^{\vec{\mathbf{s}}}) = \mathrm{rdeg} \begin{pmatrix} 3x^9+4x^8 & x^9 \\ 5x^8 & x^2+1 \end{pmatrix} = (9,8)$$

Proposition 4.3. Soit $A \in \mathbb{K}[x]^{m\times n}$ et $\vec{\mathbf{s}} \in \mathbb{Z}^n$. Alors, on a les propriétés suivantes :

- $\mathrm{rdeg}_{\vec{\mathbf{s}}}(A) = \vec{\mathbf{v}}$ si et seulement si $\mathrm{rdeg}(x^{-\vec{\mathbf{v}}}Ax^{\vec{\mathbf{s}}}) = 0$.
- $\mathrm{rdeg}_{\vec{\mathbf{s}}}(A) \leq \vec{\mathbf{v}}$ si et seulement si $\mathrm{rdeg}(x^{-\vec{\mathbf{v}}}Ax^{\vec{\mathbf{s}}}) \leq 0$.

Exemple. Soit

$$F = \begin{pmatrix} 1 & 0 & 1 \\ x & 1 & x+1 \\ 1 & x^3+x^2 & x \end{pmatrix}, \quad \vec{\mathbf{u}} = (1,0,0,1).$$

Alors

$$\vec{\mathbf{v}} = \mathrm{rdeg}_{\vec{\mathbf{u}}}(F) = (1,2,3,4) \quad \text{et} \quad x^{-\vec{\mathbf{v}}}Ax^{\vec{\mathbf{s}}} = \begin{pmatrix} 1 & 0 & x^{-1} \\ 1 & x^{-2} & x^{-2}+x^{-1} \\ x^{-2} & x^{-1}+1 & x^{-2} \end{pmatrix}.$$

On va définir un ordre sur les degrés de ligne, bien que non total.

Définition 4.4. Soit $m \in \mathbb{N}^*$ et soient $\mathbf{u} = (u_1, \ldots, u_m)$, $\mathbf{v} = (v_1, \ldots, v_m) \in \mathbb{Z}^m$ deux vecteurs de degrés de ligne, triés par valeur croissante. On définit une relation d'ordre partiel, notée $\leq_{ob}$ (ordre obtenu par composantes), par :

$$\mathbf{u} \leq_{ob} \mathbf{v} \quad \text{si et seulement si} \quad u_i \leq v_i \quad \text{pour tout } i \in \{1, \ldots, m\}.$$

Proposition 4.4. Soit $A \in \mathbb{K}[x]^{m\times n}$, $\vec{\mathbf{b}} \in \mathbb{K}[x]^{1\times m}$ et $\vec{\mathbf{c}} = \vec{\mathbf{b}}A$. Soit $\vec{\mathbf{v}} = \mathrm{rdeg}_{\vec{\mathbf{u}}}(A)$ et $w = \mathrm{rdeg}_{\vec{\mathbf{v}}}(b)$. Alors

$$\mathrm{rdeg}_{\vec{\mathbf{u}}}(c) \leq_{ob} w.$$

Les notions de degrés introduites précédemment, ainsi que les techniques qui en découlent, permettent d'accélérer certains algorithmes dans des situations spécifiques. Cependant, elles présentent des limites structurelles importantes : notamment, les degrés de lignes et de colonnes ne possèdent pas de bonnes propriétés vis-à-vis de la multiplication matricielle. De plus, plusieurs problèmes restent ouverts quant à la possibilité d'obtenir des algorithmes plus efficaces pour le calcul du déterminant ou de l'inverse de matrices polynomiales. Ces obstacles mettent en évidence l'intérêt crucial de la réduction des matrices polynomiales, outil indispensable pour contrôler la croissance des degrés et améliorer ainsi les performances des algorithmes associés.

La réduction des réseaux polynomiaux est une étape essentielle dans plusieurs applications algorithmiques, en particulier dans le décodage efficace des codes de Reed-Solomon généralisés. Cette opération vise à transformer une base quelconque d'un réseau polynomial en une base simplifiée. Dans cette section, nous détaillerons les principaux concepts, outils et algorithmes permettant de réaliser cette réduction en temps polynomial. On notera $\mathbb{K}$ un corps quelconque.

4.3 *Généralités et notion de base d'ordre*

Soit $F \in \mathbb{F}[x]^{m\times n}$, et un **degré de précision** $\sigma \in \mathbb{N}$. On définit

$$(F, \sigma) := \{v \in \mathbb{F}[x]^{1\times m} \mid vF = 0 \mod x^\sigma\}$$

Proposition 4.5. (F, σ) est un réseau polynomial de dimension m.

On va s'intérésser à étudier les bases de ce réseau et définir une notion de base réduite.

Définition 4.5. Une (F, σ)-**base d'ordre** est une base[2] de (F, σ) de degré minimale.

Quelle est la définition du degré ? Que signifie minimale dans ce contexte ?

Définition 4.6. Soit $F \in \mathbb{F}[x]^{m\times n}$. On dit que F est **réduite par ligne** si, pour tout $U \in \mathbb{F}[x]^{m\times m}$ unimodulaire, on a :

$$\operatorname{rdeg}(F) \leq_{ob} \operatorname{rdeg}(UF).$$

Pour parler d'un minimum, il faut un ordre total. On voit enfin la nouvelle définition de base réduite.

Définition 4.7. Une **base d'ordre** est une base de (F, σ) qui est réduite par ligne.

On peut définir une notion de base d'ordre en incluant la notion de décalage des degrés, on munit (F, σ) d'un vecteur de décalage s, qu'on notera (F, σ, s), on peut alors dire qu'une base d'ordre de cet ensemble est une base d'ordre de (Fx^s, σ), on peut l'interpréter intuitivement comme une base réduite pour un décalage, qui sera utile dans le chapitre 6.
L'existence d'une base réduite est garantie, mais elle n'est pas nécessairement unique. Pour assurer l'unicité, une condition supplémentaire est requise : la forme de Popov.

Preuve naïve (incorrecte). Considérons le minimum de tous les $\operatorname{rdeg}(PU)$ triés, pour toutes les matrices unimodulaires $U \in \mathbb{F}[x]^{m\times m}$.
Toute base PU ayant un degré minimal est une base d'ordre. Attention : l'ordre $\leq_{ob}$ n'est pas un ordre total. En effet, il est possible d'avoir deux bases dont les degrés en ligne sont respectivement $(1, 2, 3)$ et $(1, 1, 4)$. On ne peut pas encore garantir l'existence d'un minimum ! ■

Définition 4.8. Soit $A \in \mathbb{F}[x]^{m\times n}$ et soit $v = \operatorname{rdeg}_u(A)$. On définit la **matrice des coefficients dominants**[3] de A, notée $\operatorname{lcoeff}(A) \in \mathbb{F}^{m\times n}$, comme étant la matrice obtenue en extrayant la partie constante de $x^{-v}A$, c'est-à-dire :

$$\operatorname{lcoeff}(A) = \lim_{x\to\infty} x^{-v}A.$$

Exemple. Soit

$$F = \begin{pmatrix} 1 & 0 & 1 \\ x & 1 & 1+x \\ 1 & x^2+x^3 & x \end{pmatrix}, \quad \vec{v} := \operatorname{rdeg}(F) = (0, 1, 3)$$

Alors

$$x^{-\vec{v}} \cdot F = \begin{pmatrix} 1 & 0 & 1 \\ 1 & x^{-1} & x^{-1}+1 \\ x^{-3} & x^{-1}+1 & x^{-2} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} + \mathcal{O}_{x\to\infty}(x^{-1})$$

Proposition 4.6 (Transitivité, revisitée)**.** Soient $c := b \cdot A$, $v = \operatorname{rdeg}_u(A)$ et $w = \operatorname{rdeg}_v(b)$.
Si $\operatorname{lcoeff}(A)$ est **injective à gauche**, alors $\operatorname{rdeg}_u(c) = w$.

Remarque. On a donc l'égalité par rapport à la proposition 4.4.

Proposition 4.7. Soit A une matrice. Si $\operatorname{lcoeff}(A)$ est injective à gauche, alors A est réduite par ligne.

2. au sens des $\mathbb{F}[x]$-modules
3. La matrice des coefficients dominants est accessible dans `SageMath` via la méthode `leading_matrix()`.

Maintenant que l'on connaît la définition d'une base réduite et un critère pour la définir comment la calculer efficacement ?

Définition 4.9. On définit le **pivot** d'une ligne comme l'élément non nul de degré maximal le plus à droite dans cette ligne.

Définition 4.10. Soit $W \in \mathbb{F}[x]^{m\times n}$. La matrice W est dite en **forme Popov faible** si chaque ligne de W possède un pivot, et si les indices de colonnes de ces pivots sont distincts deux à deux.

Exemple. Soit $W_1, W_2 \in M_2(\mathbb{Z}_7[x])$ tel que

$$W_1 = \begin{pmatrix} 3x+4 & x^9 \\ 5 & x^2+1 \end{pmatrix},\ W_2 = \begin{pmatrix} 2x^7+5x^5+3x+4 & x^5 \\ 5 & x^2+1 \end{pmatrix}$$

On note les pivots en rouge. W_1 n'est pas en forme de Popov faible car les pivots sont sur la même colonne, et W_2 est en forme Popov faible , car les pivots, ont des indices de colonnes distincts.

Pour transformer une matrice en forme Popov faible, on peut utiliser l'algorithme proposé dans (Mulders et Storjohann 2003), qui fournit une méthode systématique pour y parvenir en appliquant des transformations unimodulaires par lignes.

Définition 4.11. On appelle **transformation simple** de la ligne k sur la ligne l l'opération consistant à soustraire cx^e fois la ligne k à la ligne l, où $c \in \mathbb{F}$ et $e \in \mathbb{N}$. On dit qu'elle est du **premier type** si les pivots de la ligne k et de la ligne l ont les mêmes indices et du **deuxième type** sinon.

[4]

Algorithme 5 : *WeakPopovForm (Mulders et Storjohann 2003)*

Entrée : $\mathcal{M} \in \mathbb{F}[x]^{n\times m}$
Sortie : $\mathcal{N}$ en forme de Popov faible, obtenue par des transformations simples de premier type appliquées à $\mathcal{M}$

$A \leftarrow$ copie$(\mathcal{M})$
Tant que *A n'est pas en forme de Popov faible* **faire**
  Appliquer une transformation simple du premier type à A
$\mathcal{N} \leftarrow$ copie(A)
Retourner $\mathcal{N}$

Théorème 4.1 (Correction et complexité)**.**

L'algorithme WeakPopovForm est correct. Sa complexité est bornée par $\mathcal{O}(nmrd^2)$ opérations dans le corps de base, où r désigne le rang de la matrice M, et d une borne supérieure sur le degré des coefficients de $\mathcal{M}$.

Exemple.

$$\begin{pmatrix} 3x+4 & x^9 \\ 5 & x^2+1 \end{pmatrix} \xrightarrow{(1)} \begin{pmatrix} 2x^7+3x+4 & 6x^7 \\ 5 & x^2+1 \end{pmatrix} \xrightarrow{(2)} \begin{pmatrix} 2x^7+5x^5+3x+4 & x^5 \\ 5 & x^2+1 \end{pmatrix}$$

1. Ajouter $6x^7$ fois la deuxième ligne à la première ligne.
2. Ajouter x^5 fois la deuxième ligne à la première ligne.

4. L'algorithme suivant est disponible dans `SageMath` via la méthode `weak_popov_form()`.

La matrice finale est en forme Popov faible, les pivots ont des indices de colonnes distincts.

Théorème 4.2. Toute matrice en *forme Popov faible* est réduite par ligne.

4.4 *Algorithmes de calculs de base d'ordre*

Cette section va montrer qu'il existe des algorithmes plus rapide que (MULDERS et STORJOHANN 2003) pour calculer des bases d'ordres.

4.4.1 *Cas initial quand $\sigma = 1$*

Si $\sigma = 1$, alors

$$\begin{aligned}(F, \sigma) &= \{v \in \mathbb{F}[x]^{1\times m} \mid vF = 0 \mod x^1\} \\ &= \{v \in \mathbb{F}^{1\times m} \mid vF = 0\}\end{aligned}$$

Soit $F \in \mathbb{F}^{m\times n}$, nous cherchons une base de $(F, 1)$.
On remarque que si

$$\begin{pmatrix} S \\ K \end{pmatrix} F = \begin{pmatrix} R \\ 0 \end{pmatrix} \text{ avec } R \text{ de rang maximal}$$

alors

$$\begin{pmatrix} xS \\ K \end{pmatrix} F = \begin{pmatrix} xR \\ 0 \end{pmatrix} = 0 \mod x$$

ce qui implique que $\begin{pmatrix} xS \\ K \end{pmatrix}$ est une base de $(F, 1)$ Il existe des candidats naturels pour S et K : K le noyau de F et S le supplémentaire de K. On pourra choisir le noyau K utilisant les lignes de F de plus petit degré, une façon de les calculer consiste alors à obtenir la forme échelonnée par lignes de F, avec une bonne permutation de façon à choisir les lignes de plus petit degrés, ceci aura son importance par la suite.
Il faut donc calculer une décomposition :

$$\underbrace{\begin{pmatrix} L_r & 0 \\ G & I_{m-r} \end{pmatrix}}_{L} \cdot \begin{bmatrix} e_{\tau(1)} \\ \vdots \\ e_{\tau(n)} \end{bmatrix} \cdot F = \underbrace{\begin{pmatrix} E' \\ 0 \end{pmatrix}}_{E}$$

où E est échelonnée en ligne, L est triangulaire inférieure, $r$est le rang de E et τ est une permutation.

Algorithme 6 : *Basis*

Entrée : $F \in (\mathbb{F}[x]_{\leq 0})^{m\times n}$, un vecteur de décalage s.
Sortie : Une $(F, 1, s)$-base d'ordre et son degré de ligne s-décalé.

1 On suppose que s est croissant.
2 Calculer une forme ligne échelonnée $F = \tau \cdot L \cdot E$ avec :
3 **Retourner** $\begin{pmatrix} xL_r & 0 \\ G & I_{m-r} \end{pmatrix}, \quad \tau^{-1}s + [1_r, 0_{n-r}]$

L'algorithme Basis calcul correctement une base d'ordre de $(F, 1)$ d'après la construction ci-dessus, l'hypothèse que s est croissant est primordiale pour obtenir une base d'ordre. On ne cherchera pas à redémontrer ce résultat.

4.4.2 *Algorithmes pour le cas général*

On va devoir découper le problème pour trouver une base d'ordre pour $\sigma > 1$.

Théorème 4.3.

Soit P_1, P_2 des bases d'ordre de (F, σ_1) et (F, σ_2) respectivement, $M \in \mathbb{F}[x]^{m\times n}$ tel que $P_1 F = x^{\sigma_1} M$. Alors : $P_2 P_1$ est une base d'ordre de $(F, \sigma_1 + \sigma_2)$.

On peut découper par pas de 1, on se doute qu'il y aura plus efficace juste après ces lignes, on présente un algorithme itératif **quadratique**.

$$(F,1) \to (F,2) \to (F,3) \to \cdots (F, \sigma-1) \to (F,\sigma)$$

Algorithme 7 : *M-Basis*

Entrée : $F \in (\mathbb{F}[x]_{<\sigma})^{m\times n}$, un vecteur de décalage $\vec{s}$, $\sigma \in \mathbb{N}$
Sortie : Une $(F, \sigma, \vec{s})$-base d'ordre et son degré de ligne $\vec{s}$

1 $P_0 \leftarrow$ *Basis*$(F \bmod x)$
2 **Pour** k *de* 1 *à* $\sigma - 1$ **faire**
3 $\quad F' \leftarrow x^{-k} P_{k-1} F$ `// Décalage du problème`
4 $\quad M_k \leftarrow$ *Basis*$(F' \bmod x)$
5 $\quad P_k \leftarrow M_k P_{k-1}$
6 **Retourner** $P_{\sigma-1}$

Théorème 4.4. La complexité de l'algorithme *Basis* est $\mathcal{O}(m^\omega \sigma^2)$ opérations dans le corps de base.

Remarque.

— On peut réaliser la ligne 4 en $\mathcal{O}(m^\omega \sigma)$, la ligne 5 est en $\mathcal{O}(m^\omega)$ car les coefficients sont entiers.
— Il faut voir la ligne 3 comme un décalage.
— On se sert du théorème 5.3 pour prouver la correction de l'algorithme.

On présente maintenant un algorithme sur le principe diviser-pour-régner **quasi-linéaire** qui découpe la précision en 2.

$$(F,1) \to (F,2) \to (F,4) \to \cdots \to \left(F, \frac{\sigma}{2}\right) \to (F,\sigma)$$

Algorithme 8 : *PM-Basis*

Entrée : $F \in (\mathbb{F}[x]_{<\sigma})^{m\times n}$, un vecteur de décalage $\vec{s}$, $\sigma \in \mathbb{N}$
Sortie : Une $(F, \sigma, \vec{s})$-base d'ordre et son degré de ligne $\vec{s}$

1 **Si** $\sigma = 1$ **alors**
2 $\quad$ **Retourner** Basis$(F \bmod x)$
3 **Sinon**
4 $\quad P_{\text{low}} \leftarrow$ PM-Basis$(F, \lfloor \sigma/2 \rfloor)$ `// Premier sous-problème`
5 $\quad$ Soit F' tel que $P_{\text{low}} F = x^k F'$ `// Décalage du problème`
6 $\quad P_{\text{high}} \leftarrow$ PM-Basis$(F', \lfloor \sigma/2 \rfloor)$ `// Deuxième sous-problème`
7 $\quad$ **Retourner** $P_{high} \cdot P_{low}$ `// Résoudre le problème original`

Théorème 4.5. La complexité de l'algorithme *PM-Basis* est $\mathcal{O}(\mathrm{MM}(m, \sigma)\log(\sigma))$ opérations dans le corps de base.

Remarque.

— Il faut voir la ligne 5 comme un décalage.
— La complexité peut se calculer en écrivant l'arbre binaire associé.
— On se sert du théorème 5.3 pour prouver la correction de l'algorithme.

Les algorithmes que nous avons vus sont des algorithmes pour calculer des bases d'ordre et non réduire des matrices par lignes.

CHAPITRE **5**

Adaptation de la réduction de réseaux polynomiaux au cas des réseaux euclidiens

Ce chapitre constitue le cœur de mon stage. Il vise à explorer l'adaptation des techniques exactes de réduction de réseaux polynomiaux, au cadre des réseaux euclidiens. Il s'agit d'esquisser des pistes, de formuler des questions pertinentes et de suggérer des idées nouvelles. L'ensemble de ce travail a été réalisé en collaboration avec mon encadrant, Romain Lebreton.

5.1 *Fonction potentielle et bornes*

5.1.1 *Les cas polynomial*

Dans l'algorithme *(Mulders et Storjohann 2003)*, chaque opération entraîne soit une diminution du degré total par lignes, soit un déplacement du pivot d'une ligne vers la gauche.
(Nielsen 2013) définit une fonction de *valeur* pour les vecteurs :

$$\begin{array}{rccc} \psi : & \mathbb{F}[x]^{\ell+1} & \longrightarrow & \mathbb{N}_0 \\ & \mathbf{v} & \longmapsto & (\ell+1)\cdot \operatorname{rdeg}\mathbf{v} + \operatorname{LP}(\mathbf{v}) \end{array}$$

où $\operatorname{LP}(\mathbf{v}$ désigne l'indice du pivot de $\mathbf{v}$.

Lemme 5.1. Soit $\mathbf{v}'_j$ le vecteur qui remplace $\mathbf{v}_j$ lors d'une réduction de ligne dans *(Mulders et Storjohann 2003)*. Alors :

$$\psi(\mathbf{v}'_j) < \psi(\mathbf{v}_j).$$

Ainsi, l'algorithme *(Mulders et Storjohann 2003)* repose sur une fonction potentielle qui décroît strictement à chaque opération. Cette fonction atteint nécessairement une valeur minimale, correspondant à une situation où aucune opération supplémentaire n'est possible, c'est-à-dire lorsque la matrice est réduite en ligne. Cette même idée se traduit dans l'algorithme.

5.1.2 *Le cas entier*

Le lecteur pourra utilement se référer à la démonstration de la terminaison de l'algorithme LLL. Dans cette preuve, une quantité D a été introduite, qui décroît d'un facteur $\frac{3}{4}$ à chaque échange de vecteurs. Cette décroissance strictement contrôlée constitue l'invariant principal garantissant la terminaison de l'algorithme.

Définition 5.1 (Rappel)**.** On définit $D \coloneqq \prod_{1\le k<n} d_k = \prod_{1\le k<n} \|\mathbf{b}_k^*\|^{2(n-k)}$

En pratique, la valeur de D peut devenir très grande, il est donc plus pertinent de regarder l'ordre de grandeur en considérant son logarithme $\log(D)$, c'est-à-dire son nombre de bits, puisque c'est comme ça qu'il est considérer dans la démonstration.

Proposition 5.1. $\log(D) = \sum_{k=1}^{n-1} 2(n-k)\log(\|\mathbf{b}_k^*\|)$

On impose dans la preuve de terminaison que $D \geq 1$. Toutefois, $D = 1$ si et seulement si le réseau est isomorphe à $\mathbb{Z}^n$. Un raisonnement similaire montre que très peu de réseaux satisfont $D = 2$. En pratique, la valeur de D reste largement supérieure à 1, et l'algorithme retourne une base LLL-réduite avec $D \gg 1$. Ainsi, bien que le critère de terminaison repose sur une décroissance stricte de D, cela ne reflète pas toujours le fait que l'algorithme a effectivement atteint un état suffisant de réduction.

On peut essayer d'améliorer la borne inférieure, on se place dans l'hypothèse que la base est LLL-réduite et donc satisfait la condition de Lovàsz.

$$\begin{aligned} D &= \prod_{1\leq k<n} \|\mathbf{b}_k^*\|^{2(n-k)} \\ &= \|\mathbf{b}_1^*\|^{2(n-1)} \times \|\mathbf{b}_2^*\|^{2(n-2)} \times \cdots \times \|b_n^*\|^2 \\ &\geq \|\mathbf{b}_1^*\|^{2(n-1)} \times \left(\frac{\|\mathbf{b}_1^*\|}{2}\right)^{2(n-2)} \times \cdots \times \left(\frac{\|\mathbf{b}_1^*\|}{2^{n-2}}\right)^2 \\ &\geq \left(\frac{\|b_1^*\|}{2^{\frac{4}{3}(n-2)}}\right)^{(n-1)n} \end{aligned}$$

Après la remise de ce rapport, mon travail consistera à explorer la portée pratique de cette borne. Contrairement au cas polynomial, le comportement de D dans le cas entier reste difficile à cerner, et l'on ne sait pas précisément jusqu'où cette quantité peut décroître.

De manière similaire au raisonnement précédent, une borne supérieure sur D est également disponible. Dans (Gathen et Gerhard 2003), on trouve :

$$D \leq \left(\max_{1\leq i\leq n} \|\mathbf{g}_i\|\right)^{n(n-1)}.$$

Une question naturelle est alors de savoir si cette borne peut être améliorée. On peut voir que l'exposant $n(n-1)$ peut devenir $(n-1)n$. Plutôt que d'utiliser $\max_{1\leq i\leq n} \|\mathbf{g}_i\|$, on pourrait utiliser un invariant lié au réseau comme $|\mathscr{L}|$.

5.2 *Vers une réduction LLL adaptée aux réseaux d'approximation*

On va donner une définition équivalente du réseau d'approximation (F, σ) mais adaptée pour les réseaux euclidiens.

Définition 5.2. Soit $F \in M_n(\mathbb{Z})$, un degré de précision $\sigma \in \mathbb{N}$, et $p \in \mathbb{N}$. On définit

$$F_{p^\sigma} \coloneqq \{v \in \mathbb{Z}^n | vF = 0 \mod p^\sigma\}$$

Contrairement au cas polynomial, la situation est ici fondamentalement différente. Nous verrons par la suite s'il est nécessaire d'imposer des restrictions sur p et σ, et le cas échéant, lesquelles.

Proposition 5.2. F_{p^σ} est un réseau euclidien de dimension n.

Remarque. F_{p^σ} est un réseau p^σ-aire.

Comment peut-on calculer une base de F_{p^σ} ? Est-il possible d'en extraire une base LLL-réduite, et ce, de manière efficace ? Comme dans le cas polynomial traité pour le réseau (F, σ), il est d'abord

nécessaire de calculer une décomposition [1] de la forme

$$\underbrace{\begin{bmatrix} e_{\tau(1)} \\ \vdots \\ e_{\tau(n)} \end{bmatrix}}_{P} \cdot \underbrace{\begin{bmatrix} L_r & 0 \\ G & I_{m-r} \end{bmatrix}}_{L} \cdot F = \underbrace{\begin{bmatrix} E' \\ 0 \end{bmatrix}}_{E} \tag{5.1}$$

où $r \coloneqq \mathrm{rang}(F)$, P est une matrice de permutation, L est une matrice triangulaire inférieure et E est échelonnée en ligne.

Algorithme 9 : *PLE*(A)

Entrée : $A \in \mathbb{K}^{n \times m}$
Sortie : Matrices P, L, E telles que 6.1 est satisfaite.

$n \leftarrow \text{nrows}(A), \quad m \leftarrow \text{ncols}(A)$
$P \leftarrow I_n, \quad L \leftarrow I_n, \quad E \leftarrow A$
pour $i \leftarrow 0$ **à** $m-1$ **faire**
  $(\text{pivot}, i_{\text{pivot}}) \leftarrow \textsc{Pivot}(E_{*,i}, \{i, \dots, n-1\})$
  si *pivot* $=$ `None` **alors**
    continuer
  si $i_{pivot} \neq i$ **alors**
    Échanger les lignes i et i_{pivot} dans P et E
    pour $k \leftarrow 0$ **à** $i-1$ **faire**
      Échanger $L[i,k]$ et $L[i_{\text{pivot}}, k]$
  pour $j \leftarrow i+1$ **à** $n-1$ **faire**
    $s \leftarrow E[j,i]/\text{pivot}$
    si $s \neq 0$ **alors**
      $E \leftarrow E$ avec ligne j **moins** s fois ligne i
      $L \leftarrow L$ avec ligne j **moins** s fois ligne i
Retourner (P, L, E)

Remarque. Comme c'est vrai pour toute permutations, il serait intéressant de choisir certaines permutations particulières, notamment qui ordonne les vecteurs par norme croissante.

Remarque. J'ai mis trop d'investissement dans cet algorithme, j'en ai fais plusieurs versions, allant d'une version naïve où je calculais tous les pivots pour ordonner les lignes de façon à avoir les mineurs principaux inversibles, à une stratégie plus subtile d'échange quand nécessaire, grâce aux conseils avisés de mon encadrant. Cet algorithme fonctionne sur $\mathbb{Z}_p$ pour p premier, car les pivots sont inversibles, je réfléchissais à une stratégie pour rendre cet algorithme fonctionnel sur $\mathbb{Z}_n$.

Théorème 5.1. L'algorithme PLE calcule correctement une décomposition qui satisfait 6.1.

Contre exemple (Limite de l'algorithme PLE). L'algorithme PLE ne s'applique pas dans tous les cas.
Soit

$$A = \begin{bmatrix} 3 & 4 \\ 4 & 3 \end{bmatrix} \in M_2(\mathbb{Z}/12\mathbb{Z}).$$

1. Il s'agit d'une généralisation de la décomposition $PLF = U$, où U était une matrice triangulaire supérieure. Ici, on relâche cette contrainte en ne demandant que U soit échelonnée par lignes.

Dans cet anneau, les coefficients 3 et 4 ne sont pas inversibles, ce qui empêche de procéder aux opérations de pivot nécessaires.

Ce contre-exemple montre que la validité de l'algorithme repose sur une hypothèse cruciale : *les pivots doivent être inversibles*.

Exemple.

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ -\frac{3}{4} & 1 & 0 & 0 \\ -\frac{1}{2} & 0 & 1 & 0 \\ 0 & -\frac{1}{3} & 0 & 1 \end{pmatrix}}_{L} \times \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{P} \underbrace{\begin{pmatrix} 4 & 2 & 4 & 2 \\ 2 & 1 & 2 & 1 \\ 3 & 3 & 3 & 3 \\ 1 & 1 & 1 & 1 \end{pmatrix}}_{F} = \underbrace{\begin{pmatrix} 4 & 2 & 4 & 2 \\ 0 & \frac{3}{2} & 0 & \frac{3}{2} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}}_{E}$$

On en déduit donc un algorithme pour calculer une base.
De 6.1 on déduit que

$$\begin{bmatrix} e_{\tau(1)} \\ \vdots \\ e_{\tau(n)} \end{bmatrix} \cdot \begin{bmatrix} L_r \cdot p^\sigma & 0 \\ G & I_{m-r} \end{bmatrix} \cdot F = \begin{bmatrix} E' \cdot p^\sigma \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \mod p^\sigma$$

Théorème 5.2. Les matrices

$$\begin{bmatrix} L_r \cdot p^\sigma & 0 \\ G & I_{m-r} \end{bmatrix} \cdot \begin{bmatrix} e_{\tau(1)} \\ \vdots \\ e_{\tau(n)} \end{bmatrix}, \quad \begin{bmatrix} I_r \cdot p^\sigma & 0 \\ G & I_{m-r} \end{bmatrix} \cdot \begin{bmatrix} e_{\tau(1)} \\ \vdots \\ e_{\tau(n)} \end{bmatrix}$$

sont deux bases de F_{p^σ}

Remarque. Il vaut mieux privilégier la seconde base car les r premières lignes sont orthonormées, ce qui nous permet de déduire intuitivement que la base sera de meilleure qualité.

Algorithme 10 : *Basis*$(F, p, \texttt{mode})$

Entrée : Une matrice $F \in \mathbb{K}[x]^{m \times n}$, un scalaire $p \in \mathbb{K}$, et une chaîne `mode` égale à `v1` ou `v2`
Sortie : Une base transformée selon le mode choisi

$G \leftarrow \text{copie}(F)$
$(P, L, E) \leftarrow \text{PLE}(G)$
$r \leftarrow \text{rang}(F)$
si *mode* $=$ *v1* **alors**
  pour $i \leftarrow 0$ **à** $r-1$ **faire**
    Multiplier la ligne i de L par p
si *mode* $=$ *v2* **alors**
  pour $i \leftarrow 0$ **à** $r-1$ **faire**
    Remplacer la ligne i de L par $p \cdot e_i$
      // e_i : i-ème vecteur de la base canonique
Retourner $L \cdot P$

Exemple. En partant de la matrice $F = \begin{pmatrix} 4 & 2 & 4 & 2 \\ 2 & 1 & 2 & 1 \\ 3 & 3 & 3 & 3 \\ 1 & 1 & 1 & 1 \end{pmatrix}$

On peut calculer une base de F_{5^4}

$$\begin{pmatrix} 625 & 0 & 0 & 0 \\ -\frac{1}{2} & 1 & 0 & 0 \\ 0 & 0 & 625 & 0 \\ 0 & 0 & -\frac{1}{3} & 1 \end{pmatrix}$$

On peut donc maintenant définir un algorithme sur un principe diviser pour régner.

Algorithme 11 : *LLL-DAC-PADIQUE*(F, p, σ)

Entrée : $F \in \mathbb{K}^{m \times n}$, un entier premier p, un entier $\sigma \geq 1$
Sortie : Une base "LLL-réduite ?" en précision p^σ

si $\sigma = 1$ **alors**
  Retourner *LLL*(*APPROXIMANTBASIS*(F, p))
$\tau \leftarrow \left\lfloor \frac{\sigma + 1}{2} \right\rfloor$
$V_1 \leftarrow$ LLL-DAC-PADIQUE(F, p, τ) `// Appel récursif sur demi-précision`
$F_{\text{low}} \leftarrow \frac{V_1 \cdot F}{p^\tau}$ `// Mise à jour du problème`
$V_2 \leftarrow$ LLL-DAC-PADIQUE$(F_{\text{low}}, p, \sigma - \tau)$ `// Appel récursif décalé`
Retourner $V_2 \cdot V_1$

Similairement à la preuve du chapitre précédent, $V_2 V_1$ est une base de F_{p^σ}.

Question
Dans quelle mesure le produit de matrices LLL-réduites reste-t-il lui-même LLL-réduit ? Peut-on quantifier cette propriété ?

Si V_1 est une matrice orthogonale, c'est-à-dire dont les lignes sont orthonormées, alors $V_2 V_1$ est LLL-réduite.
On rappelle que $V_1 = U_1 V_1^*$ d'après le procédé de Gram-Schmidt. En écrivant $V_1 V_2 = V_1 V_2^* ((V_2^*)^{-1} U_2 V_2^*)$, peut-être pourront nous mieux contrôler le résultat du produit.

5.3 *Comprendre le rôle du shift dans le cas euclidien*

On rappel que calculer le degré de ligne décalé d'une matrice revient à calculer le degré de ligne de cette matrice multipliée à gauche par une matrice diagonale.
On pourrait ici définir une notion de décalage avec une matrice ligne-orthogonale. Cela a donné lieu à l'algorithme suivant que j'ai écris et dont j'ai prouvé la correction par moi-même.

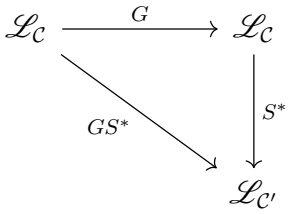
Algorithme 12 : *SHIFTLLL*

Entrée : Une base G de $\mathscr{L}$, S^* une matrice ligne-orthogonale
Sortie : Une base B de $\mathscr{L}$ tel que BS^* soit LLL-réduite

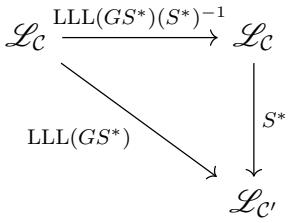
Retourner $\text{LLL}(GS^*) \cdot (S^*)^{-1}$

Théorème 5.3. L'algorithme *shiftLLL* calcule correctement une base B du réseau $\mathscr{L}$ telle que BS^* soit LLL-réduite.

Preuve. On a le diagramme commutatif suivant en voyant S^* comme la matrice de passage de la base canonique $\mathcal{C}$ à la base $\mathcal{C}' \coloneqq S^*$. L'écriture $\mathscr{L}_\mathcal{C}$ où $\mathcal{C}$ est une base représente $\mathscr{L}$ exprimée dans la base $\mathcal{C}$.



Comme $LLL(GS^*)$ est une base de $\mathscr{L}_{\mathcal{C}'}$, on en déduit le diagramme commutatif suivant et donc que $LLL(GS^*)(S^*)^{-1}$ est une base de $\mathscr{L}_\mathcal{C}$.



On a par construction de LLL que $LLL(GS^*)(S^*)^{-1}S^*$ est LLL-réduite. ■

L'algorithme termine car LLL termine et sa complexité est de l'ordre de la complexité de LLL.

En revenant à la décomposition précédente, en calculant une base telle que $V_1V_2^*$ est LLL-réduite, c'est-à-dire V_1 est LLL-réduite pour le décalage V_2^*. On peut se poser les questions suivantes qui sont des pistes à explorer :

Est-ce que $(V_2^*)^{-1}U_2V_2$ est "quasi" LLL-réduite ?

> *Hypothèse*
> $(V_2^*)^{-1}U_2V_2$ est une base propre, c'est-à-dire qu'en écrivant sa décomposition de Gram-Schmidt, les coefficients correspondant ne sont pas trop grand.

La condition de Lovász peut être interprétée comme une forme de 2-quasi-croissance. Une question naturelle serait alors de se demander si, par analogie, le produit calculé par LLL pourrait satisfaire une propriété de 4-quasi-croissance.

Conclusion et perspectives

Ce stage a été l'occasion de m'immerger pleinement dans une thématique à la frontière de plusieurs domaines, en explorant les liens entre la réduction de bases dans les réseaux polynomiaux et euclidiens. J'ai appris beaucoup de choses sur la cryptographie. L'objectif initial, comprendre dans quelle mesure certaines techniques exactes issues du cas polynomial peuvent être transposées au cas entier, m'a conduit à une réflexion profonde sur les structures internes de ces deux cadres, leurs obstacles respectifs, et c'est difficile, j'ai perdu littéralement des cheveux.

Après une première phase dédiée à la compréhension fine de l'algorithme de Lenstra–Lenstra–Lovász (LLL), de sa preuve et de ses variantes, j'ai étudié les réseaux d'approximation dans le cas polynomial, en particulier à travers les algorithmes de type Basis, M-Basis et PM-Basis. Cela m'a permis de me familiariser avec des notions comme les fonctions potentielles, les décalages, et les stratégies de réduction modulaire.

Dans un second temps, j'ai tenté d'adapter certaines de ces idées au cadre euclidien. J'ai notamment proposé une version p-adique de réduction de type LLL, fondée sur une stratégie récursive par précision croissante, et prouvé la validité d'un algorithme que j'ai nommé shiftLLL, inspiré d'une vision géométrique du changement de base. Ces constructions, bien qu'exploratoires, constituent des pistes sérieuses pour une adaptation plus fine des techniques de réduction exactes aux contraintes du calcul numérique.

Ce stage m'a permis d'approfondir mes compétences en mathématiques théoriques et en algorithmique constructive, tout en me confrontant à des problématiques de mise en œuvre concrète, notamment via SageMath. J'ai également eu l'opportunité de présenter certains résultats à l'oral, d'échanger avec des chercheurs du domaine, et de développer une plus grande autonomie dans la conduite d'un projet de recherche. Ce travail ouvre naturellement la voie à de nombreuses questions, que ce soit du point de vue théorique (analyse fine de la stabilité des produits de bases réduites) ou algorithmique (optimisation, généralisation, complexité).

Plusieurs directions naturelles s'ouvrent à la suite de ce travail. Un premier axe concerne l'analyse fine du comportement de la fonction potentielle D dans le cadre euclidien : mieux comprendre ses bornes effectives, son lien avec la qualité de la base en sortie de LLL, et la possibilité d'adapter dynamiquement ses paramètres à des situations spécifiques.

Un second axe réside dans l'étude approfondie des algorithmes de type LLL-DAC-Padique. Leur structure récursive offre un cadre intéressant pour l'optimisation par précision croissante, mais leur stabilité, leur efficacité réelle et la propagation des erreurs méritent une étude plus systématique, notamment en lien avec les bornes théoriques et le comportement empirique.

Enfin, une piste originale, initiée ici, concerne l'exploration de stratégies de réduction adaptées à un décalage donné. L'algorithme shiftLLL en constitue une première illustration. Il serait pertinent d'approfondir les conditions sous lesquelles un produit de matrices LLL-réduites reste lui-même bien réduit, et de développer des critères mesurant le degré de "quasi-réduction" d'une base obtenue par composition. Ces réflexions pourraient notamment s'articuler autour de notions telles que la k-quasi-croissance ou la structure spectrale du produit de Gram-Schmidt associé.

Lors de la présentation orale du stage, j'envisage de compléter ce travail théorique par des simulations numériques illustrant concrètement les algorithmes développés.

Je tiens à remercier vivement mon encadrant Romain Lebreton pour ses conseils, sa bienveillance et la richesse des échanges tout au long de ce stage.

ANNEXE A

Détails de preuves de LLL

Lemme A.1.

1. Soit $G, G^*, M \in \mathbb{Q}^{n\times n}$ et $H, H^*, N \in \mathbb{Q}^{n\times n}$ les matrices des $\mathbf{g}_k$, g_k^*, $\mu_{k,l}$ avant et après *Proprification partielle de* $\mathbf{g}_i$. Soit $E = I_n - \lceil \mu_{i,j} \rfloor E_{i,j} \in \mathbb{Z}^{n\times n}$, où $E_{i,j}$ désigne la matrice élémentaire. Alors
$$H = EG, \quad N = EM, \quad H^* = G^*.$$
2. Avant *Proprification partielle de* $\mathbf{g}_i$, on a :
$$|\mu_{i,l}| \leq \frac{1}{2} \quad \text{pour} \quad j < l < i.$$

Preuve.

1. - *Proprification partielle de* $\mathbf{g}_i$ se traduit matriciellement par $H = EG$.
   - La nouvelle famille est
   $$(\mathbf{g}_1, \ \ldots, \ \mathbf{g}_{i-1}, \ \mathbf{g}_i - \lceil \mu_{i,j} \rfloor \mathbf{g}_j, \ \mathbf{g}_{i+1}, \ \ldots, \ \mathbf{g}_n).$$
   On rappelle que i et j sont fixés et que $1 \leq j \leq i-1$. On a par définition du procédé de Gram-Schmidt $h_k^* = \mathbf{g}_k^*$ pour $1 \leq k \leq i-1$.
   Pour $k = i$, on a
   $$\begin{aligned} h_i^* &= h_i - \sum_{k=1}^{i-1} \frac{\langle h_i, h_k^* \rangle}{\|h_k^*\|^2} h_k^* \\ &= \mathbf{g}_j - \lceil \mu_{i,j} \rfloor \mathbf{g}_j - \sum_{k=1}^{i-1} \frac{\langle \mathbf{g}_i - \lceil \mu_{i,j} \rfloor \mathbf{g}_j, \mathbf{g}_k^* \rangle}{\|\mathbf{g}_k^*\|^2} \mathbf{g}_k^* \\ &= \mathbf{g}_i^* + \lceil \mu_{i,j} \rfloor \sum_{k=1}^{i-1} \frac{\langle \mathbf{g}_j, \mathbf{g}_k^* \rangle}{\|\mathbf{g}_k^*\|^2} \mathbf{g}_k^* - \lceil \mu_{i,j} \rfloor \mathbf{g}_j \\ &= \mathbf{g}_i^* + \lceil \mu_{i,j} \rfloor \sum_{k=1}^{j} \frac{\langle \mathbf{g}_j, \mathbf{g}_k^* \rangle}{\|\mathbf{g}_k^*\|^2} \mathbf{g}_k^* - \lceil \mu_{i,j} \rfloor \mathbf{g}_j \\ &= \mathbf{g}_i^* + \lceil \mu_{i,j} \rfloor \sum_{k=1}^{j-1} \frac{\langle \mathbf{g}_j, \mathbf{g}_k^* \rangle}{\|\mathbf{g}_k^*\|^2} \mathbf{g}_k^* + \mathbf{g}_j^* - \lceil \mu_{i,j} \rfloor \mathbf{g}_j \\ &= \mathbf{g}_i^* \end{aligned}$$
   Ainsi $h_i^* = \mathbf{g}_i^*$. En poursuivant le procédé de Gram-Schmidt , on en conclut $H^* = G^*$.
   - On a $NG^* \overset{(1)}{=} NH^* \overset{(D.1)}{=} H \overset{(1)}{=} EG \overset{(D.1)}{=} EMG^*$. En identifiant on a $N = EM$.
2. Au début de la boucle lorsque $j = i-1$ on a trivialement l'inégalité.
   La i-ème ligne de M
   $$(\mu_{i,1}, \ \cdots, \ \mu_{i,j}, \ \cdots, \ \mu_{i,i-1})$$

Devient

$$(\mu_{i,1} - \lceil \mu_{i,j} \rfloor \mu_{j,1},\ \cdots,\ \mu_{i,j} - \lceil \mu_{i,j} \rfloor \mu_{j,j},\ \cdots,\ \mu_{i,i-1} - \lceil \mu_{i,j} \rfloor \mu_{j,i-1})$$

$$(\mu_{i,1} - \lceil \mu_{i,j} \rfloor \mu_{j,1}, \cdots,\ \mu_{i,j} - \lceil \mu_{i,j} \rfloor,\ \cdots,\ \mu_{i,i-1})$$

■

Algorithme 13 : *Proprification de* $\mathbf{g}_i$

Pour $j = i-1, i-2, \ldots, 1$ **faire**
  Proprification partielle de $\mathbf{g}_i$

Lemme A.2. *Proprification de* $\mathbf{g}_i$ ne change pas G^* et à la fin on a

$$|\mu_{i,l}| \leq \frac{1}{2} \text{ pour } 1 \leq l < i.$$

Preuve. D'après le lemme 2.1, après l'exécution de *Proprification de* $\mathbf{g}_i$, G^* n'a pas été modifié et en appliquant le lemme 2.1 pour $j = 1$ on a

$$|\mu_{i,l}| \leq \frac{1}{2} \text{ pour } 1 \leq l < i.$$

■

On voit donc que la partie de mise à jour est en réalité simple à calculer et ne necessite pas de recalculer la decomposition de GS à chaque fois.
On étudie maintenant l'étape de réduction, l'idée consiste à réorganiser les vecteurs afin de garantir une progression quantifiable, qui assurera la terminaison de LLL.

Algorithme 14 : *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$

Si $i > 1$ ***et*** $\left\|\mathbf{g}_{i-1}^*\right\|^2 > 2\left\|\mathbf{g}_i^*\right\|^2$ **alors**
  Échanger $\mathbf{g}_{i-1}$ et $\mathbf{g}_i$
  Mettre à jour (B^*, U)
  $i \leftarrow i - 1$
Sinon
  $i \leftarrow i + 1$

Lemme A.3. Supposons que $\mathbf{g}_{i-1}$ et $\mathbf{g}_i$ sont échangés à l'étape *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$.
On note h_k les vecteurs après échange et h_k^* leur base orthogonale de Gram-Schmidt.
Alors

1. $h_k^* = \mathbf{g}_k^*$ pour tout $k \in \{1, \ldots, n\} \setminus \{i-1, i\}$,
2. $\|h_{i-1}^*\|^2 < \frac{3}{4}\|\mathbf{g}_{i-1}^*\|^2$,
3. $\|h_i^*\| \leq \|\mathbf{g}_{i-1}^*\|$.

Preuve.

1. La famille

$$(\mathbf{g}_1,\ \ldots,\ \mathbf{g}_{i-2},\ \mathbf{g}_{i-1},\ \mathbf{g}_i,\ \mathbf{g}_{i+1},\ \ldots,\ \mathbf{g}_n)$$

devient

$$(h_1 := \mathbf{g}_1,\ \ldots,\ h_{i-2} := \mathbf{g}_{i-2},\ h_i := \mathbf{g}_i,\ h_{i-1} := \mathbf{g}_{i-1},\ h_{i+1} := \mathbf{g}_{i+1},\ \ldots,\ h_n := \mathbf{g}_n)$$

Par construction de Gram-Schmidt on a $h_k^* = \mathbf{g}_k^*$ pour $1 \leq k \leq i-2$.
On a

$$\begin{aligned} h_{i+1}^* &= h_{i+1} - \sum_{j=1}^{i} \frac{\langle h_{i+1}, h_j^* \rangle}{\|h_j^*\|^2} h_j^* \\ &= \mathbf{g}_{i+1} - \sum_{j=1}^{i-2} \frac{\langle \mathbf{g}_{i+1}, \mathbf{g}_j^* \rangle}{\|\mathbf{g}_j^*\|^2} \mathbf{g}_j^* + \frac{\langle \mathbf{g}_{i+1}, \mathbf{g}_i^* \rangle}{\|\mathbf{g}_i^*\|^2} \mathbf{g}_i^* + \frac{\langle \mathbf{g}_{i+1}, \mathbf{g}_{i-1}^* \rangle}{\|\mathbf{g}_{i-1}^*\|^2} \mathbf{g}_{i-1}^* \\ &= \mathbf{g}_{i+1}^* \end{aligned}$$

On en déduit donc que $h_k^* = \mathbf{g}_k^*$ pour $i+1 \leq k \leq n$.

2. Le vecteur h_{i-1}^* est la composante de $\mathbf{g}_i$ orthogonale à $\sum_{1 \leq l < i-1} \mathbb{R}\mathbf{g}_l$, or $\mathbf{g}_i = \mathbf{g}_i^* + \sum_{1 \leq l < i-1} \mu_{i,l} \mathbf{g}_l^*$.
Alors

$$h_{i-1}^* = \mathbf{g}_i^* + \mu_{i,i-1} \mathbf{g}_{i-1}^*$$

et donc

$$\begin{aligned} \|h_{i-1}^*\|^2 &= \|\mathbf{g}_i^*\|^2 + \mu_{i,i-1}^2 \|\mathbf{g}_{i-1}^*\|^2 \\ &\leq \frac{1}{2} \|\mathbf{g}_{i-1}^*\|^2 + \frac{1}{4} \|\mathbf{g}_{i-1}^*\|^2 \\ &= \frac{3}{4} \|\mathbf{g}_{i-1}^*\|^2 \end{aligned}$$

3. Soit $u = \sum_{1 \leq l < i-1} \mu_{i-1,l} \mathbf{g}_l^*$ et posons $U = \sum_{1 \leq l < i-1} \mathbb{R}\mathbf{g}_l$ pour simplifier l'écriture.
Alors, le vecteur h_i^* est la composante de $\mathbf{g}_{i-1} = \mathbf{g}_{i-1}^* + u$ orthogonale à $U + \mathbb{R}\mathbf{g}_i$.
Or $u \in U \subseteq U + \mathbb{R}\mathbf{g}_i$.
Alors, le vecteur h_i^* est la composante de $\mathbf{g}_{i-1}^*$ orthogonale à $U + \mathbb{R}\mathbf{g}_i$.
Par conséquent,

$$\|h_i^*\| \leq \|\mathbf{g}_{i-1}^*\|.$$

■

Algorithme 15 : *LLL*

Tant que $i \leq n$ **faire**
  Proprification de $\mathbf{g}_i$
  Réduction de $\mathbf{g}_{i-1}, \mathbf{g}_i$

Lemme A.4. Au début de chaque itération de la boucle à l'étape *LLL*, les invariants suivants sont vérifiés :

$$|\mu_{k,l}| \leq \frac{1}{2} \quad \text{pour } 1 \leq l < k < i, \qquad \|\mathbf{g}_{k-1}^*\|^2 \leq 2\|\mathbf{g}_k^*\|^2 \quad \text{pour } 1 < k < i.$$

Preuve. Au début de l'étape *LLL*, les deux inégalités considérées sont satisfaites.
Supposons qu'elles restent vraies juste avant l'étape de *Proprification de* $\mathbf{g}_i$.
Le lemme 2.1 garantit que cette phase ne les altère pas : les inégalités demeurent donc valides après *Proprification de* $\mathbf{g}_i$.

Passons à l'étape *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$. Un échange effectué durant cette réduction ne modifie aucun coefficient $\mu_{k,l}$ avec $k < i-1$; par conséquent, la première inégalité reste satisfaite. Par ailleurs, d'après le lemme 2.3, un échange ne modifie pas $\mathbf{g}_k^*$ pour $k \notin \{i-1, i\}$, ce qui préserve la seconde inégalité.
Ainsi, à la fin de l'étape *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$ et donc au début de l'itération de *LLL* suivante les deux inégalités sont toujours vérifiées. Par récurrence sur les itérations, elles le sont à chaque instant de l'algorithme, ce qui conclut la démonstration. ■

Preuve de la correction. Le lemme 2.4 implique qu'à la fin de *LLL*, comme $i = n$, on a

$$|\mu_{k,l}| \leq \frac{1}{2} \quad \text{pour } 1 \leq l < k < n, \qquad \|\mathbf{g}_{k-1}^*\|^2 \leq 2\|\mathbf{g}_k^*\|^2 \quad \text{pour } 1 < k < n.$$

ce qui prouve que la base est réduite. ■

A.1 *Terminaison et complexité*

Théorème A.1 (Terminaison et complexité)**.** On pose $A = \max_{1 \leq i \leq n} \|\mathbf{g}_i\|$. L'algorithme *LLL* termine et utilise $\mathcal{O}(n^4 \log A)$ opérations arithmétiques sur des entiers.

La difficulté est de montrer que la boucle Tant que ne va pas s'exécuter indéfiniment.

Lemme A.5.

1. Orthogonalisation de Gram-Schmidt nécessite $\mathcal{O}(n^3)$ opérations dans $\mathbb{Z}$.
2. *Proprification de* $\mathbf{g}_i$ nécessite $\mathcal{O}(n^2)$ opérations dans $\mathbb{Z}$
3. *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$ nécessite $\mathcal{O}(n)$ opérations dans $\mathbb{Z}$

Preuve.

1. Il faut utiliser le théorème sur la complexité de l'algorithme de Gram-Schmidt.
2. Une exécution de *Proprification partielle de* $\mathbf{g}_i$ revient à faire les multiplications $H = EG$ et $N = EM$ se font en $\mathcal{O}(n)$ étapes, ainsi une exécution de *Proprification de* $\mathbf{g}_i$ nécessite $\mathcal{O}(n^2)$ opérations dans $\mathbb{Z}$.
3. Si un échange a lieu à *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$, alors seuls g_{i-1}^*, g_i^*, ainsi que les lignes et colonnes $i-1$ et i de la matrice de transition M sont modifiés, et ces éléments peuvent être mis à jour en $O(n)$ opérations.

■

Il reste à borner le nombre d'itérations de la boucle Tant que à l'étape *LLL*.
Pour tout $1 \leq k \leq n$, on pose

$$\mathbf{g}_k = \begin{pmatrix} \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_k \end{pmatrix} \in \mathbb{Z}^{k \times n}, \quad d_0 = 1, \quad d_k = \det(\mathbf{g}_k \cdot \mathbf{g}_k^T) \in \mathbb{Z}.$$

Lemme A.6. Pour tout $1 \leq k \leq n$, on a :

$$d_k = \prod_{1 \leq l \leq k} \|\mathbf{g}_l^*\|^2 > 0.$$

Preuve. Soit $1 \leq k \leq n$, on définit $\mathbf{g}_k = U_k \mathbf{g}_k^*$ la décomposition de Gram-Schmidt de la famille $(\mathbf{g}_i)_{1 \leq i \leq k}$
Alors

$$d_k = \det(\mathbf{g}_k \mathbf{g}_k^t) = \det(U_k \mathbf{g}_k^* (\mathbf{g}_k^*)^t U_k^t) = \det(\mathbf{g}_k^* (\mathbf{g}_k^*)^t) = \prod_{1 \leq l \leq k} \|\mathbf{g}_l^*\|^2 > 0$$

■

Lemme A.7.

1. *Proprification de* $\mathbf{g}_i$ ne change pas d_k pour tout $1 \leq k \leq n$.
2. Si $\mathbf{g}_{i-1}$ et $\mathbf{g}_i$ sont échangés à l'étape *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$, et si d_k^* désigne la nouvelle valeur de d_k, alors :

$$d_k^* = d_k \quad \text{pour tout } k \neq i-1, \quad \text{et} \quad d_{i-1}^* \leq \frac{3}{4} d_{i-1}.$$

Preuve.

1. D'après le lemme 2.2 *Proprification de* $\mathbf{g}_i$ ne modifie pas $\mathbf{g}_k^*$ et donc ne modifie pas d_k.
2. Pour $k \neq i-1$, une exécution de *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$ multiplie $\mathbf{g}_k$ par une matrice de permutation, donc $d_k^* = d_k$
   De plus, on a

$$d_{i-1} \overset{(2.6)}{=} \prod_{1 \leq l \leq i-1} \|\mathbf{g}_l^*\|^2 \overset{(2.3)}{\leq} \frac{3}{4} \prod_{1 \leq l \leq i-1} \|h_l^*\|^2 \overset{(2.6)}{=} \frac{3}{4} d_{i-1}^*$$

■

On pose

$$D = \prod_{1 \leq k < n} d_k, \quad A = \max_{1 \leq i \leq n} \|f_i\|$$

On désigne D_0 désigne la valeur de D au début de l'algorithme, on a $1 \leq D \in \mathbb{Z}$ et

$$\begin{aligned} D_0 &= \|g_1^*\|^{2(n-1)} \|g_2^*\|^{2(n-2)} \cdots \|g_{n-1}^*\|^2 \\ &\leq \|\mathbf{g}_1\|^{2(n-1)} \|\mathbf{g}_2\|^{2(n-2)} \cdots \|\mathbf{g}_{n-1}\|^2 \\ &\leq A^{n(n-1)} \end{aligned}$$

Puisque f_i^* est une projection de f_i pour tout i.

Lemme A.8.

1. *Proprification de* $\mathbf{g}_i$ ne modifie pas D.
2. D diminue d'au moins un facteur $3/4$ si un échange a lieu dans *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$

Preuve.

1. D'après le lemme 2.7 *Proprification de* $\mathbf{g}_i$ ne modifie pas d_k et donc ne modifie pas D.
2. Si $\mathbf{g}_{i-1}$ et $\mathbf{g}_i$ sont échangés lors de l'exécution de *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$, en notant D^* la nouvelle valeur de D, alors d'après le lemme 2.7

$$d_k^* = d_k, \quad d_{i-1}^* \leq \frac{3}{4} d_{i-1} \text{ donc } D^* \leq \frac{3}{4} D.$$

■

À tout moment de l'algorithme, soit $e \in \mathbb{N}$ le nombre d'échanges effectués jusqu'à présent, et e^* le nombre de fois où la branche alternative (le *else*) dans *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$ a été prise.

Lemme A.9. On a

$$e \leq \log_{4/3} D_0 \in \mathcal{O}(n^2 \log A)$$

Preuve. Soit D_e la valeur de D après e échanges.
On doit avoir

$$1 \ \leq \ D_e \ \leq \ \left(\frac{3}{4}\right)^e D_0 \leq \ \left(\frac{3}{4}\right)^e A^{n(n-1)}.$$

En appliquant $log_{3/4}$, aux extrémités de l'inégalité.

$$0 = \log_{3/4}(1) \ \geq \ e + \log_{3/4}(A^{n(n-1)}) = e + n(n-1)\frac{\log A}{\log(3/4)}.$$

On en déduit que $e \ \leq n(n-1)\frac{\log A}{-\log(3/4)}$ et donc $e \in \mathcal{O}(n^2 \log A)$

■

Preuve de la terminaison et la complexité.
Comme i est décrémenté de 1 lors d'un échange et incrémenté de 1 sinon l'entier $i+e-e^*$ est constant tout au long de *LLL*.
Initialement $i+e-e^* = 2$ et à la fin de *LLL* on a $n+1+e-e^* = 2$. On en déduit donc que $e+e^* = 2e+n-1 \in \mathcal{O}(n^2 \log A)$. et donc d'après le lemme 2.5 le coût total de *LLL* est $\mathcal{O}(n^2 \times n^2 \log A)$ opérations dans $\mathbb{Z}$. Ce qui acheve la preuve. ■

Théorème A.2. L'algorithme *LLL (LLL)* opère sur des entiers dont la longueur est $\mathcal{O}(n \log A)$.

Il reste à montrer la dernière partie du théorème.

Lemme A.10. Soit $\mathbf{g}_1, \ldots, \mathbf{g}_n \in \mathbb{Z}^n$, et soit G^* et M respectivement la base de Gram-Schmidt et la matrice des coefficients associés. Pour tout $1 \leq l < k \leq n$, on a :

(i) $d_{k-1}\mathbf{g}_k^* \in \mathbb{Z}^n$
(ii) $d_l \mu_{k,l} \in \mathbb{Z}$
(iii) $|\mu_{k,l}| \leq \sqrt{d_{l-1}}\|\mathbf{g}_k\|$

Preuve.

1. On écrit

$$\mathbf{g}_k^* = \mathbf{g}_k - \sum_{1 \leq l < k} \lambda_{k,l}\mathbf{g}_l, \quad \lambda \in \mathbb{R}.$$

Soit $j < k$. On a

$$0 = \langle \mathbf{g}_k^*, \mathbf{g}_j \rangle = \left\langle \mathbf{g}_k - \sum_{1 \leq l < k} \lambda_{k,l}\mathbf{g}_l, \mathbf{g}_j \right\rangle.$$

Ce qui implique

$$\langle \mathbf{g}_k, \mathbf{g}_j \rangle = \sum_{1 \leq l < k} \lambda_{k,l} \langle \mathbf{g}_l, \mathbf{g}_j \rangle$$

On a donc

$$\begin{pmatrix} \langle \mathbf{g}_1, \mathbf{g}_1 \rangle & \cdots & \langle \mathbf{g}_{k-1}, \mathbf{g}_1 \rangle \\ \vdots & & \vdots \\ \langle \mathbf{g}_1, \mathbf{g}_{k-1} \rangle & \cdots & \langle \mathbf{g}_{k-1}, \mathbf{g}_{k-1} \rangle \end{pmatrix} \begin{pmatrix} \lambda_{k,1} \\ \vdots \\ \lambda_{k,k-1} \end{pmatrix} = \begin{pmatrix} \langle \mathbf{g}_k, \mathbf{g}_1 \rangle \\ \vdots \\ \langle \mathbf{g}_k, \mathbf{g}_{k-1} \rangle \end{pmatrix}$$

D'après la règle de Cramer (à citer) on a

$$d_{k-1}\lambda_{k,1} = \frac{\begin{vmatrix} \langle \mathbf{g}_1, \mathbf{g}_1 \rangle & \cdots & \langle \mathbf{g}_{k-1}, \mathbf{g}_1 \rangle \\ \vdots & & \vdots \\ \langle \mathbf{g}_1, \mathbf{g}_{k-1} \rangle & \cdots & \langle \mathbf{g}_{k-1}, \mathbf{g}_{k-1} \rangle \end{vmatrix}}{\begin{vmatrix} \langle \mathbf{g}_1, \mathbf{g}_1 \rangle & \cdots & \langle \mathbf{g}_{k-1}, \mathbf{g}_1 \rangle \\ \vdots & & \vdots \\ \langle \mathbf{g}_1, \mathbf{g}_{k-1} \rangle & \cdots & \langle \mathbf{g}_{k-1}, \mathbf{g}_{k-1} \rangle \end{vmatrix}} det(\mathbf{g}_k \mathbf{g}_k^t) = \begin{vmatrix} \langle \mathbf{g}_1, \mathbf{g}_1 \rangle & \cdots & \langle \mathbf{g}_{k-1}, \mathbf{g}_1 \rangle \\ \vdots & & \vdots \\ \langle \mathbf{g}_1, \mathbf{g}_{k-1} \rangle & \cdots & \langle \mathbf{g}_{k-1}, \mathbf{g}_{k-1} \rangle \end{vmatrix} \in \mathbb{Z}$$

2.
$$d_l \mu_{k,l} = d_l \frac{\langle \mathbf{g}_k, \mathbf{g}_l^* \rangle}{\|\mathbf{g}_l^*\|^2} = d_l \frac{\langle \mathbf{g}_k, \mathbf{g}_l^* \rangle}{d_l / d_{l-1}} = d_{l-1} \langle \mathbf{g}_k, \mathbf{g}_l^* \rangle = \langle \mathbf{g}_k, \mathbf{g}_l^* d_{l-1} \rangle \in \mathbb{Z}$$

3.
$$|\mu_{k,l}| = \frac{\langle \mathbf{g}_k, \mathbf{g}_l^* \rangle^2}{\|\mathbf{g}_l^*\|} \leq \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_l^*\|} \leq \sqrt{d_{l-1}/d_l} \|\mathbf{g}_k\| \leq \sqrt{d_{l-1}} \|\mathbf{g}_k\|$$

■

Nous avons supposé que $\|\mathbf{g}_k\| \leq A$ pour tout k. Alors A est également une borne supérieure pour la base orthogonale de Gram-Schmidt initiale : $\|\mathbf{g}_k^*\| \leq A$ pour tout k.
On a d'après les lemmes $\max\{\|\mathbf{g}_k^*\| : 1 \leq k \leq n\}$ ne croît jamais au cours de l'algorithme. Ainsi, à tout instant et pour tout k, on a :

$$\|\mathbf{g}_k^*\| \leq A \quad \text{et} \quad d_k = \prod_{1 \leq l \leq k} \|\mathbf{g}_l^*\|^2 \leq A^{2k}.$$

Lemme A.11. Soit $1 \leq k \leq n$.

1. À tout moment de l'algorithme, sauf éventuellement à l'étape *Proprification de* $\mathbf{g}_i$ lorsque $k = i$, on a :
$$\|\mathbf{g}_k\| \leq \sqrt{n} A.$$
2. À chaque exécution de l'étape *Proprification partielle de* $\mathbf{g}_i$, on a :
$$\|\mathbf{g}_i\| \leq n(2A)^n.$$

Preuve.

1. Initialement $\|\mathbf{g}_k\| \leq A$ pour tout k. L'étape *Réduction de* $\mathbf{g}_{i-1}, \mathbf{g}_i$ ne modifie pas $\|\mathbf{g}_k\|$, il suffit donc d'examiner l'étape *Proprification de* $\mathbf{g}_i$. On a que $\mathbf{g}_k$, pour $k \neq i$, n'est pas affecté par l'étape *Proprification partielle de* $\mathbf{g}_i$.
   Soit $m_i = \max\{|\mu_{i,l}| : 1 \leq l \leq i\}$. À partir de
$$\mathbf{g}_i = \sum_{1 \leq l \leq i} \mu_{i,l} g_l^*$$

et de l'orthogonalité des vecteurs g_l^*, on obtient :

$$\|\mathbf{g}_i\|^2 = \sum_{1\le l\le i} \mu_{i,l}^2 \|g_l^*\|^2 \le n m_i^2 A^2, \quad \text{donc} \quad \|\mathbf{g}_i\| \le \sqrt{n}\, m_i A. \tag{4}$$

À la fin de *Proprification de* $\mathbf{g}_i$, on a $m_i = 1$ par le lemme 2.1.

2. Le lemme 2.10 et le point 1 impliquent qu'au début de *Proprification de* $\mathbf{g}_i$, on a

$$\begin{aligned} m_i &\le \max\left\{ d_l^{1/2} : 1 \le l \le i \right\} \cdot \|\mathbf{g}_i\| \\ &\le A^{n-2} \cdot n^{1/2} A \\ &= n^{1/2} A^{n-1}. \end{aligned}$$

Considérons maintenant le remplacement effectué à l'étape *Proprification partielle de* $\mathbf{g}_i$. Comme $m_i \ge 1$ et que $|\mu_{j,l}| \le \frac{1}{2}$ pour $1 \le l < j$, le lemme 2.8 donne :

$$\begin{aligned} |\mu_{i,l} - \lfloor \mu_{i,j} \rceil \mu_{j,l}| &\le |\mu_{i,l}| + |\lfloor \mu_{i,j} \rceil| \cdot |\mu_{j,l}| \\ &\le m_i + \left(m_i + \tfrac{1}{2}\right) \cdot \tfrac{1}{2} \\ &= \tfrac{3}{2} m_i + \tfrac{1}{4} \\ &\le 2 m_i \end{aligned}$$

pour $1 \le l < j$.

Pour $l = j$, la nouvelle valeur de $\mu_{i,j}$ est par construction au plus $\frac{1}{2}$ en valeur absolue, tout comme les valeurs de $\mu_{i,l}$ pour $l > j$, d'après le lemme 2.1.

On en déduit que pour chaque valeur de j, la valeur de m_i est au plus doublée. Ainsi, pendant *Proprification de* $\mathbf{g}_i$, la valeur de m_i est multipliée au plus par un facteur $2^{i-1} \le 2^{n-1}$.

On a donc

$$m_i \le n^{1/2} (2A)^{n-1}$$

Puis

$$\|\mathbf{g}_i\| \le n^{1/2} m_i A \le n (2A)^n.$$

■

Preuve du theoreme.

— Les dénominateurs d_l des nombres rationnels calculés pendant l'algorithme sont au plus A^{2n}, et leur taille est en $\mathcal{O}(n \log A)$.

— Les numérateurs sont majorés en valeur absolue par :

- $\|\mathbf{g}_k\|_\infty \le \|\mathbf{g}_k\| \le n(2A)^n$ d'après le lemme 2.11
- $\|d_{k-1}\mathbf{g}_k^*\|_\infty \le \|d_{k-1}\mathbf{g}_k^*\| \le A^{2k-2}A \le A^{2n}$ d'après le lemme 2.10
- $|d_l \mu_{k,l}| \le d_l d_{l-1}^{1/2} \|\mathbf{g}_l\| \le A^{2l} A^{l-1} n (2A)^n \le n(2A^4)^n$ d'après les lemmes 2.10 et 2.11

et par conséquent, leur taille est aussi en $\mathcal{O}(n \log A)$.

■

ANNEXE **B**

Rappels sur les anneaux

B.1 *Généralités*

Ces résultats sont classiques de la théorie des anneaux commutatifs. Pour plus de détails, on pourra se reporter à un manuel standard d'algèbre.

Définition B.1. Un **anneau** $(A, +, \cdot)$ est un ensemble A muni de deux lois internes :

— $(A, +)$ est un **groupe abélien** (on note 0 son élément neutre et $-a$ l'inverse de a),

— la multiplication $\cdot : A \times A \to A$, notée simplement ab, est **associative** et **distributive** par rapport à $+$, c'est-à-dire :

$$a(b+c) = ab + ac, \quad (b+c)a = ba + ca, \quad \forall a, b, c \in A,$$

et possède un **élément neutre** $1 \in A$ (on dit alors que A est un anneau **unitaire**).

Définition B.2. Un anneau A est **commutatif** si $ab = ba$ pour tout $a, b \in A$.

Dans ce mémoire et dans cette annexe, tous les anneaux seront supposés commutatifs par défaut, conformément aux usages standards, sauf indication explicite du contraire.

Définition B.3. Un anneau A est **intègre** si, pour tout $a, b \in A$, on a

$$ab = 0 \quad \Longrightarrow \quad (a = 0 \text{ ou } b = 0).$$

Remarque. Autrement dit, A est intègre si et seulement si il n'a pas de diviseurs de 0.

Définition B.4. Soit A un anneau commutatif. Un **idéal** $I \subseteq A$ est un sous-groupe additif de $(A, +)$ tel que, pour tout $a \in A$ et tout $x \in I$, on ait $ax \in I$.

Remarque. En commutatif, $ax = xa$, donc une seule condition suffit à le décrire.

Définition B.5. Un idéal I de A est dit **principal** s'il existe un unique $r \in A$ tel que $I = \langle r \rangle = \{ar : a \in A\}$. Un anneau A est **principal** si tous ses idéaux sont principaux.

Définition B.6. Un anneau A est **noethérien** si toutes ses chaînes d'idéaux (i.e. $I_1 \subseteq I_2 \subseteq \ldots$) sont stationnaires, c'est-à-dire qu'il n'existe pas de chaîne infinie strictement croissante d'idéaux.

Remarque. Cette définition est équivalente à dire que tout idéal de A est de type fini, i.e. $I = \langle x_1, \ldots, x_k \rangle$ pour un nombre fini d'éléments.

Définition B.7 (Anneau factoriel)**.** Un anneau intègre A est **factoriel** si tout élément non nul et non inversible de A admet une factorisation en éléments irréductibles unique à l'ordre près (et inversibles près).

Tout **anneau euclidien** est principal, tout **anneau principal** est factoriel et tout **anneau principal** est noethérien.

TABLE B.1 – Quelques exemples d'anneaux et leurs propriétés

Anneau	Commutatif	Intègre	Principal	Factoriel	Noethérien
$\mathbb{Z}$	✓	✓	✓	✓	✓
$\mathbb{Z}/n\mathbb{Z}$ (non premier)	✓	✗	✓	✗	✓
$\mathbb{Z}/p\mathbb{Z}$ (p premier)	✓	✓	✓	✓	✓
$\mathbb{Z}[i]$	✓	✓	✓	✓	✓
$\mathbb{Z}[\sqrt{-5}]$	✓	✓	✗	✗	✓
$\mathbb{Z} \times \mathbb{Z}$	✓	✗	✗	✗	✓
$M_n(\mathbb{K})$, $n \geq 2$	✗	✗	—	—	✓
$C^0([0,1], \mathbb{R})$	✓	✓	✗	✗	✗

B.2 *Anneaux de polynômes*

Les matrices polynomiales que nous étudierons dans ce mémoire sont construites à partir d'anneaux de polynômes, qui en forment la base algébrique.

Théorème B.1. Si A est un anneau factoriel, alors $A[x]$ est aussi factoriel.
Si A est un anneau noethérien, alors $A[x]$ est noethérien.
Si A est intègre, alors $A[x]$ est intègre.

Proposition B.1. Soit $\mathbb{K}$ un corps. Alors l'anneau de polynômes $\mathbb{K}[x]$ est **principal** et l'anneau $\mathbb{K}[x, y]$ **n'est pas** principal.

Remarque. Quand on a deux variables, la structure d'idéaux se complique et ne peut pas être engendrée par un seul polynôme dans la plupart des cas.

TABLE B.2 – Exemples d'anneaux de polynômes et leurs propriétés

Anneau	Commutatif	Intègre	Principal	Factoriel	Noethérien
$\mathbb{K}[x]$	✓	✓	✓	✓	✓
$\mathbb{K}[x, y]$	✓	✓	✗	✓	✓
$\mathbb{Z}[x]$	✓	✓	✗	✓	✓
$\mathbb{F}_p[x]$	✓	✓	✓	✓	✓
$\mathbb{F}_p[x, y]$	✓	✓	✗	✓	✓
$\mathbb{R}[x]/(x^2+1)$	✓	✓	✓	✓	✓
$\mathbb{K}[x]/(x^n)$, $n \geq 2$	✓	✗	✓	✗	✓

ANNEXE C

Rappels sur les modules

C.1 *Généralités*

Définition C.1. Soit A un anneau commutatif unitaire. Un **A-module** $(M, +, \cdot)$ est un ensemble M :

— muni d'une loi interne $+$ faisant de $(M, +)$ un groupe abélien,

— muni d'une loi externe $A \times M \to M,\ (a, m) \mapsto am$, satisfaisant pour tout $a, b \in A$ et $m, m' \in M$:

1. Distributivité : $a(m + m') = am + am'$,
2. Distributivité : $(a + b)m = am + bm$,
3. Associativité : $(ab)m = a(bm)$,
4. Neutre : $1 \cdot m = m$ (où 1 est l'élément neutre de A).

Remarque. Cette définition généralise la notion d'espace vectoriel, en remplaçant le *corps* des scalaires par un *anneau* commutatif. Dans le cas d'un corps, tout élément non nul de A est inversible, tandis qu'ici on n'exige pas cette propriété.

Exemple. Soit $n \in \mathbb{N}^*$. L'ensemble $\mathbb{Z}^n$, muni de l'addition vectorielle et de la multiplication par un scalaire entier, est un $\mathbb{Z}$-module. En effet :

— $(\mathbb{Z}^n, +)$ est un groupe abélien,

— pour tout $a \in \mathbb{Z}$ et $x = (x_1, \ldots, x_n) \in \mathbb{Z}^n$, on a $a \cdot x = (ax_1, \ldots, ax_n) \in \mathbb{Z}^n$,

— toutes les axiomes d'un module sont satisfaits (associativité, distributivité, existence d'un neutre).

C'est un module libre de type fini, de base canonique $(e_1, \ldots, e_n)$.

Contre exemple. Considérons l'ensemble $\mathbb{R}$ et l'anneau $\mathbb{C}$. On cherche à définir une structure de $\mathbb{C}$-module sur $\mathbb{R}$ via la multiplication usuelle :

$$\forall \alpha \in \mathbb{C},\ \forall r \in \mathbb{R}, \quad \alpha \cdot r := \alpha r.$$

Cependant, cette loi externe n'est pas bien définie car elle n'est pas **fermée** :

$$\text{par exemple, } \alpha = i \in \mathbb{C},\ r = 1 \in \mathbb{R} \quad \Rightarrow \quad \alpha \cdot r = i \notin \mathbb{R}.$$

Donc $\mathbb{R}$ n'est pas stable par multiplication par les scalaires complexes. Il ne peut pas être muni d'une structure de $\mathbb{C}$-module.

C.1.1 *Sous-modules, type fini et modules libres*

Définition C.2. Soit M un A-module. Un **sous-module** $N \subseteq M$ est un sous-groupe additif de $(M, +)$ qui est stable par multiplication externe, c'est-à-dire pour tout $a \in A$ et tout $x \in N$, on a $ax \in N$.

TABLE C.1 – Exemples et contre-exemples de modules

Ensemble considéré	Sur quel anneau A	Module ?
$\mathbb{Z}^n$	$\mathbb{Z}$	✓
$\mathbb{Q}^n$	$\mathbb{Q}$	✓
$\mathbb{Z}/n\mathbb{Z}$	$\mathbb{Z}$	✓
$C^0([0,1],\mathbb{R})$	$\mathbb{R}$	✓
$\mathbb{R}[x]$	$\mathbb{R}$	✓
$\mathbb{Q}$	$\mathbb{Z}$	✗
$\mathbb{Z}[\sqrt{2}]$	$\mathbb{Z}[x]$	✗
$\mathbb{R}$	$\mathbb{C}$	✗
$\mathbb{Z}^n$	$\mathbb{Q}$	✗

Définition C.3. Un A-module M est de **type fini** s'il existe un ensemble fini $S \subset M$ tel que tout élément de M s'écrive comme combinaison A-linéaire des éléments de S. On dit alors que S engendre M.

Définition C.4. Un A-module M est **libre** s'il admet une famille $(x_i)_{i\in I}$ telle que tout $x \in M$ s'écrive de manière unique sous la forme

$$x \;=\; \sum_{i\in I} \alpha_i\, x_i,$$

avec $\alpha_i \in A$. Cette famille (x_i) est appelée **base** de M.

Si M est libre et de type fini, il existe donc une base finie de M. La démonstration n'est pas triviale. Dans ce cas, toutes les bases de M ont le même nombre d'éléments, que l'on appelle le **rang** de M.

Exemple. .

— $\mathbb{Z}/n\mathbb{Z}$ est un $\mathbb{Z}$-module de type fini, mais il n'est pas libre pour $n \neq 0$, car aucun élément non nul de $\mathbb{Z}/n\mathbb{Z}$ n'est librement générateur.

— Au contraire, $\mathbb{Z}^n$ est libre de rang n (les vecteurs de la base canonique en constituent une base).

Libre	Type fini	Exemple
Oui	Oui	$\mathbb{Z}^n$
Oui	Non	$\bigoplus_{i\in\mathbb{N}} \mathbb{Z}$
Non	Oui	$\mathbb{Z}/n\mathbb{Z}$ pour $n \geq 2$
Non	Non	$\mathbb{Q}$

TABLE C.2 – Exemples de $\mathbb{Z}$-modules selon leur liberté et leur type fini

C.2 *Modules sur un anneau principal*

Dans la suite, on considère un anneau principal A, c'est-à-dire un anneau (commutatif unitaire) dans lequel *tout idéal* est principal.

Théorème C.1. Soit M un module **libre** sur un anneau principal A. Alors tout sous-module de M est également libre et son rang est inférieur ou égal à celui de M.

Exemple. Dans le $\mathbb{Z}$-module libre $\mathbb{Z}^2$, considérons le sous-ensemble

$$N \;=\; \{(a,b) \in \mathbb{Z}^2 \mid a \equiv b \pmod{10}\}.$$

On constate que N est un **sous-module** de $\mathbb{Z}^2$, et qu'il est engendré par les vecteurs $(1, 1)$ et $(0, 10)$. Ceux-ci sont A-linéairement indépendants, donc N est libre de rang 2, tout comme $\mathbb{Z}^2$ lui-même.

Module M	**Anneau A**	**Libre**	**Type fini**	**Torsion**	**Réf.**
$\mathbb{K}[x]^n$	$\mathbb{K}[x]$	Oui	Oui	Non	(1)
$\mathbb{K}[x]/(x^n)$	$\mathbb{K}[x]$	Non	Oui	Oui	(2)
$\mathbb{K}[x]^\infty$	$\mathbb{K}[x]$	Oui	Non	Non	(3)
$\mathbb{K}(x)$	$\mathbb{K}[x]$	Non	Non	Oui	(4)
$(\mathbb{K}[x])^n/(x \cdot \mathbb{K}[x])^n$	$\mathbb{K}[x]$	Non	Oui	Oui	(5)

TABLE C.3 – Exemples de modules sur l'anneau $\mathbb{K}[x]$

Commentaires sur les exemples :

1. $\mathbb{K}[x]^n$ est le module libre canonique : c'est un $\mathbb{K}[x]$-module libre de rang n. Il sert de modèle aux réseaux polynomiaux.
2. $\mathbb{K}[x]/(x^n)$ est un module de torsion : tout élément est annulé par une puissance de x. Il n'admet pas de base libre.
3. $\mathbb{K}[x]^\infty$ (somme directe infinie) est un module libre, mais non de type fini. Il possède une base infinie indexée par $\mathbb{N}$.
4. $\mathbb{K}(x)$, le corps des fractions rationnelles, est un module divisible mais non libre. Il contient des éléments sans expression unique comme combinaison de base.
5. $(\mathbb{K}[x])^n/(x \cdot \mathbb{K}[x])^n$ est un module quotient, utilisé dans les algorithmes de bases d'ordre modulo x^σ. C'est un module de torsion.