# RÉSEAUX EUCLIDIENS EN CRYPTOGRAPHIE 2023 – TD 3

ALEXANDRE WALLET, QUYEN NGUYEN

Le but de ce TD est d'étudier des algorithmes pour résoudre $\gamma$-CVP[1] :
*Soit $\mathcal{L} = \mathcal{L}(\mathbf{B}) \subset \mathbb{R}^m$ un réseau, et $t \in \mathbb{R}^m$. Trouver $v \in \mathcal{L}$ tel que $\|v - t\| \leq \gamma \cdot d(t, \mathcal{L})$.*

**Exercice 1.** (1) Montrer comment résoudre 1-CVP si $\mathcal{L}$ est de rang 1.

(2) On appelle *arrondi* de $t$ le vecteur $s = \mathbf{B}\lceil \mathbf{B}^{-1} t \rfloor$, où $\lceil \cdot \rfloor$ désigne l'entier le plus proche (pris coordonnée par coordonnée). Montrer que $\|t - s\| \leq \frac{n}{2} \cdot \max \|b_i\|$, où les $b_i$ sont les colonnes de $\mathbf{B}$.

(3) Si les $b_i$ sont de plus orthogonaux deux à deux, montrer que l'arrondi est le plus proche vecteur de $t$ dans $\mathcal{L}$.

**Exercice 2.** Voici l'algorithme *Nearest Plane*, attribué à Babai. On donne à gauche une version récursive utilisant le fait qu'on sait résoudre CVP en dimension 1, et à droite une version itérative, équivalente. Si $b_1, \ldots, b_n$ est une base du réseau considéré, on notera $V_i = \mathrm{Vect}_{\mathbb{R}}(b_1, \ldots, b_i)$ et $b_i^*$ les orthogonalisés de Gram-Schmidt correspondants.

**Nearest Plane :**

Entrées : $(b_1, \ldots, b_n)$ une base de $\mathcal{L}$, $t \in \mathbb{R}^n$
Sortie : $s \in \mathcal{L}$ "proche" de $t$.

(1) Projeter $t$ sur $t'$ dans l'espace ambiant de $\mathcal{L}$ ;
(2) Si on est en dimension 1, renvoyer le plus proche vecteur de $t'$ ;
Sinon, trouver $V_{n-1} + y$, avec $y \in \mathcal{L}$, le plus proche de $t'$ ;
(3) Se ramener "dans un réseau" : $t' \leftarrow t' - y$ ;
(4) Rappeler sur $t'$ et $b_1, \ldots, b_{n-1}$, ceci donne $s'$ ;
(5) On corrige la translation : renvoyer $s' + y$ ;

(1) $s \leftarrow 0, e_{n+1} \leftarrow t$ ;
(2) Pour $i = n$ à 1, faire :
— $c_i \leftarrow \lceil \frac{\langle e_{i+1}, b_i^* \rangle}{\|b_i^*\|^2} \rfloor$ ;
— $s \leftarrow s + c_i b_i$ ;
— $e_i \leftarrow e_{i+1} - c_i b_i$ ;
(3) Renvoyer $s$

L'exercice est d'abord de comprendre le comportement de l'algorithme, puis de démontrer que combiné à LLL, il permet de résoudre $2^{O(n)}$-CVP.

(1) Fixons $i \geq 1$, et supposons $t \in V_{i+1}$. Si $y = \lceil t_{i+1} \rfloor b_{i+1}$, montrer que $V_i + y$ est l'hyperplan affine de $V_{i+1}$ le plus proche[2] de $t$ parmi les translatés de $V_i$ par des vecteurs du réseau.

(2) Lorsqu'on n'est pas en dimension 1, que peut-il se passer à l'étape (2) de la version récursive ? *Indication : Raisonner sur l'emplacement du plus proche vecteur $u \in \mathcal{L}$ de $t$ par rapport à l'hyperplan de la question précédente.*

(3) Supposons que $u \notin V_i + y$. Montrer qu'à cette étape, on a alors $\|t - u\| \geq \|b_{i+1}^*\|/2$.

(4) On suppose maintenant $u \in V_i + y$. Montrer qu'on peut se ramener à une autre instance du problème CVP, mais en dimension $i$.

On va maintenant quantifier les distances.

1. Pour Closest Vector Problem, comme vu en cours.
2. C'est de là que vient son nom.

(5) Monter que Nearest Plane renvoie $s$ tel que $\|t-s\|^2 \leq \frac{1}{4}\sum_i \|b_i^*\|^2$. *Indication : à l'aide de la version itérative, montrer que* $e := t - s \in \{\sum_i x_i b_i^* \ : \ |x_i| \leq \frac{1}{2}\}$.

Comme LLL s'exécute en temps polynomial, on peut supposer que $b_1, \ldots, b_n$ est LLL-réduite car ceci n'influe pas sur la classe de complexité du problème CVP.

(6) Montrer dans ce cas que si $t \in V_{i+1}$, alors le $s$ renvoyé par Nearest Plane appelé dans $V_{i+1}$ satisfait $\|t-s\| \leq 2^{(i-1)/2}\|b_{i+1}^*\|$.

(7) Reprendre la question (3), en déduire que si l'algorithme ne prend pas le bon hyperplan dans $V_{i+1}$, alors $\|t-s\| \leq 2^{(i+1)/2} \cdot d(t, \mathcal{L})$.

(8) Conclure en procédant par récurrence.