CHIFFREMENT CKKS

DAMIEN STEHLÉ

20 NOVEMBRE 2025

PLAN

1- **Arithmétique CKKS**

2- Bootstrap

3- Application en biométrie

TROIS ANNEAUX

$R := \mathbb{Z}[X]/(X^N + 1)$

N est une puissance de 2 (sécurité)

Espace des **clés secrètes** s

$R_{\mathbb{R}} := \mathbb{R}[X]/(X^N + 1)$

Espace des **encodés** m,
disons à coefficients dans $[-1,1]$

$R_q := \mathbb{Z}_q[X]/(X^N + 1)$

q est premier et $q = 1\ [2N]$ (efficacité)
ou un produit de tels entiers distincts

$R_q \times R_q$ est l'espace des **chiffrés**

$$ct = (a, b) : \quad a \cdot s + b = f(m)\ \ [q]$$

L'ENCODAGE CKKS

$$ct = (a, b) : \quad a \cdot s + b = f(m) \ [q]$$

BFV : $f(m) = \frac{q}{t} \cdot m + e \ [q]$

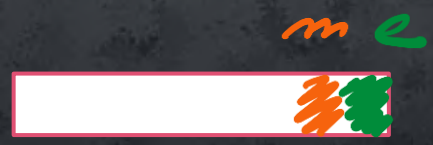
BGV : $f(m) = m + t \cdot e \ [q]$

CKKS : $f(m) \approx \Delta \cdot m$
(avec $\Delta \ll q$)

L'ENCODAGE CKKS

$$ct = (a, b) : \quad a \cdot s + b = f(m) \ [q]$$

CKKS : $f(m) \approx \Delta \cdot m$
(avec $\Delta \ll q$)



L'espace des encodés est $\mathbb{R}[X]/(X^N + 1)$, disons avec des coefficients dans $[-1,1]$.

Le **facteur d'échelle** Δ guide la precision des calculs.

L'encodé m est "mélangé" au bruit e : bruit de RLWE et bruit numérique.

ADDITIONNER ET MULTIPLIER

$$ct = (a, b) : \quad a \cdot s + b = f(m) \ [q]$$

$$ct : < ct, sk > \approx \Delta \cdot m \ [q]$$

avec $sk = (s, 1)$

$$< ct_1, sk > \approx \Delta \cdot m_1$$
$$< ct_2, sk > \approx \Delta \cdot m_2$$

$$< ct_1 + ct_2, sk > \approx \Delta \cdot (m_1 + m_2)$$

$$< ct_1, sk > \approx \Delta \cdot \ m_1$$
$$< ct_2, sk > \approx \Delta \cdot \ m_2$$

$$< ct_1 \otimes ct_2, sk \otimes sk > \ = \ < ct_1, sk > \cdot < ct_2, sk >$$
$$\approx \Delta^2 \cdot \ m_1 \cdot m_2$$

MULTIPLIER… ET CONSERVER LE FORMAT

$$< ct_1, sk > \approx \Delta \cdot m_1$$
$$< ct_2, sk > \approx \Delta \cdot m_2$$

$$< ct_1 \otimes ct_2, sk \otimes sk > \approx \Delta^2 \cdot m_1 \cdot m_2$$

Deux difficultés :

1. $ct_1 \otimes ct_2$ est de **dimension 4**
2. Le **facteur d'échelle est Δ^2**

MULTIPLIER… ET CONSERVER LE FORMAT

$$< ct_1, sk > \approx \Delta \cdot m_1$$
$$< ct_2, sk > \approx \Delta \cdot m_2$$

$$< ct_1 \otimes ct_2, sk \otimes sk > \approx \Delta^2 \cdot m_1 \cdot m_2$$

Deux difficultés :

1. $ct_1 \otimes ct_2$ est de **dimension 4**
2. Le **facteur d'échelle est Δ^2**

(1)

Supposons qu'on a une matrice C t.q.

$$C \cdot sk \approx p \cdot sk \otimes sk \; [pq]$$

avec $p \approx q$, alors :

$$\left\langle \frac{1}{p} (C^T \cdot (ct_1 \otimes ct_2) \; [pq]), sk \right\rangle \approx \Delta^2 \cdot m_1 \cdot m_2 \; [q]$$

C fait partie de la clé d'évaluation (publique)

MULTIPLIER… ET CONSERVER LE FORMAT

$$< ct_1, sk > \approx \Delta \cdot m_1$$
$$< ct_2, sk > \approx \Delta \cdot m_2$$

$$< ct_1 \otimes ct_2, sk \otimes sk > \approx \Delta^2 \cdot m_1 \cdot m_2$$

Deux difficultés :

1. $ct_1 \otimes ct_2$ est de **dimension 4**
2. Le **facteur d'échelle est Δ^2**

(1) Supposons qu'on a une matrice C t.q.

$$C \cdot sk \approx p \cdot sk \otimes sk\ [pq]$$

avec $p \approx q$, alors :

$$\left\langle \frac{1}{p}(C^T \cdot (ct_1 \otimes ct_2)\ [pq]), sk \right\rangle \approx \Delta^2 \cdot m_1 \cdot m_2\ [q]$$

C fait partie de la clé d'évaluation (publique)

(2) Si $< ct, sk > \approx \Delta^2 \cdot m\ [q]$

avec $q = q_1 q_2$ et $q_2 \approx \Delta$, alors :

$$\left\langle \frac{1}{q_2} ct\ [q_1], sk \right\rangle \approx \Delta \cdot m\ [q]$$

$\mathbb{R}[X]/(X^N + 1)$ N'EST PAS TRÈS INTÉRESSANT...

Soient $\zeta_1, \dots, \zeta_{\frac{N}{2}} \in \mathbb{C}$ des racines complexes de $X^N + 1$ non-conjuguées 2 à 2.

Alors

$$\mathbb{R}[X]/(X^N + 1) \simeq \mathbb{C}^{N/2}$$

$$m \mapsto m(\zeta_1), \dots, m(\zeta_{\frac{N}{2}})$$

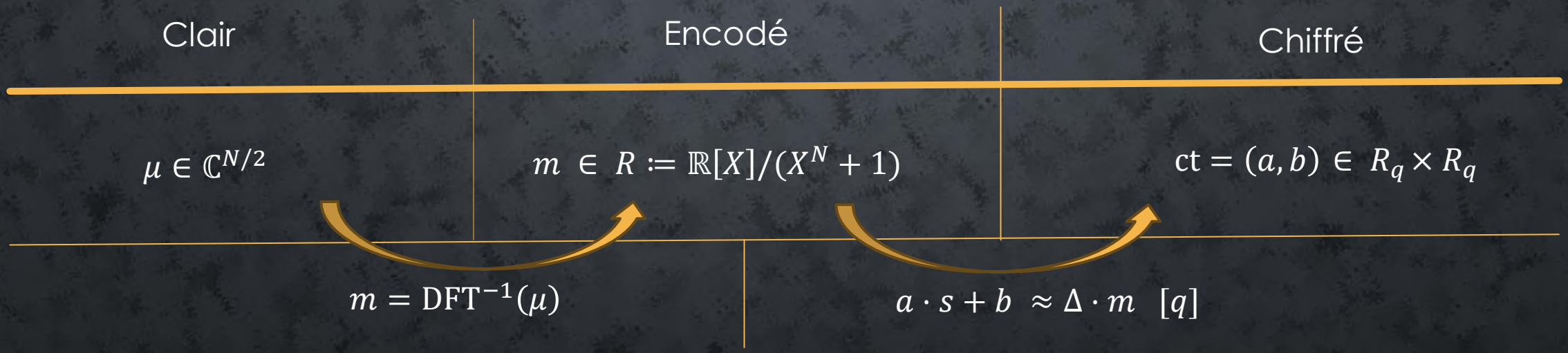
Morphisme d'anneaux :

- l'addition de polynômes correspond à l'addition de vecteurs,
- la multiplication de polynômes correspond à la multiplication (point à point) de vecteurs.

Fonction linéaire : presque la transformée de Fourier discrète.

Isométrie : la petitesse est préservée.

CLAIRS, ENCODÉS ET CHIFFRÉS



À ce stade, on sait :

- additionner homomorphiquement des vecteurs,
- multiplier homomorphiquement des vecteurs.

AUTOMORPHISMES

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

N est une puissance de 2 (sécurité)

Les automorphismes de R sont engendrés par :

- l'évaluation en X^5, i.e., $m(X) \mapsto m(X^5)$ (ordre $N/2$)
- l'évaluation en $X^{-1} = -X^{N-1}$, i.e., $m(X) \mapsto m(X^{-1})$ (ordre 2)

AUTOMORPHISMES

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

N est une puissance de 2 (sécurité)

Les automorphismes de R sont engendrés par :

- l'évaluation en X^5, i.e., $m(X) \mapsto m(X^5)$ (ordre $N/2$)
- l'évaluation en $X^{-1} = -X^{N-1}$, i.e., $m(X) \mapsto m(X^{-1})$ (ordre 2)

$$ct: < ct, sk > \approx \Delta \cdot m \ [q]$$

(avec $sk = (s, 1)$)

$$\frac{< ct, sk > \approx \Delta \cdot m}{< ct(X^5), sk(X^5) > \approx \Delta \cdot m(X^5)}$$

Pour le clair $\mu = \mathrm{DFT}(m)$, cela correspond à une **permutation des coordonnées**.

$$\frac{< ct, sk > \approx \Delta \cdot m}{< ct(X^{-1}), sk(X^{-1}) > \approx \Delta \cdot m(X^{-1})}$$

Pour le clair $\mu = \mathrm{DFT}(m)$, cela correspond à la **conjugaison des coordonnées**.

(il faut ensuite revenir à la clé $sk(X)$, comme pour la multiplication)

L'ARITHMÉTIQUE DE CKKS

Espace des clairs : vecteurs de $\mathbb{C}^{N/2}$ (à une certaine précision pilotée par Δ)

- add en //
- mult en //
- conj en //
- rotation cyclique des entrées (si on les ordonne bien)

CKKS est **étagé :**

- mult — fait baisser le module d'un facteur $\approx \Delta$ (1 étage de moins)
- add, conj & rot — ne font pas baisser le module (conserve l'étage)

L'ARITHMÉTIQUE DE CKKS

Espace des clairs : vecteurs de $\mathbb{C}^{N/2}$ (à une certaine précision pilotée par Δ)

- add en //
- mult en //
- conj en //
- rotation cyclique des entrées (si on les ordonne bien)

CKKS est **étagé :**

- mult — fait baisser le module d'un facteur $\approx \Delta$ (1 étage de moins)
- add, conj & rot — ne font pas baisser le module (conserve l'étage)

Programmer efficacement avec CKKS

Règle 1 : ne pas gaspiller le module
Règle 2 : ne pas gaspiller le parallélisme

PLAN

1- Arithmétique CKKS

2- Bootstrap

3- Application en biométrie

BOOTSTRAP

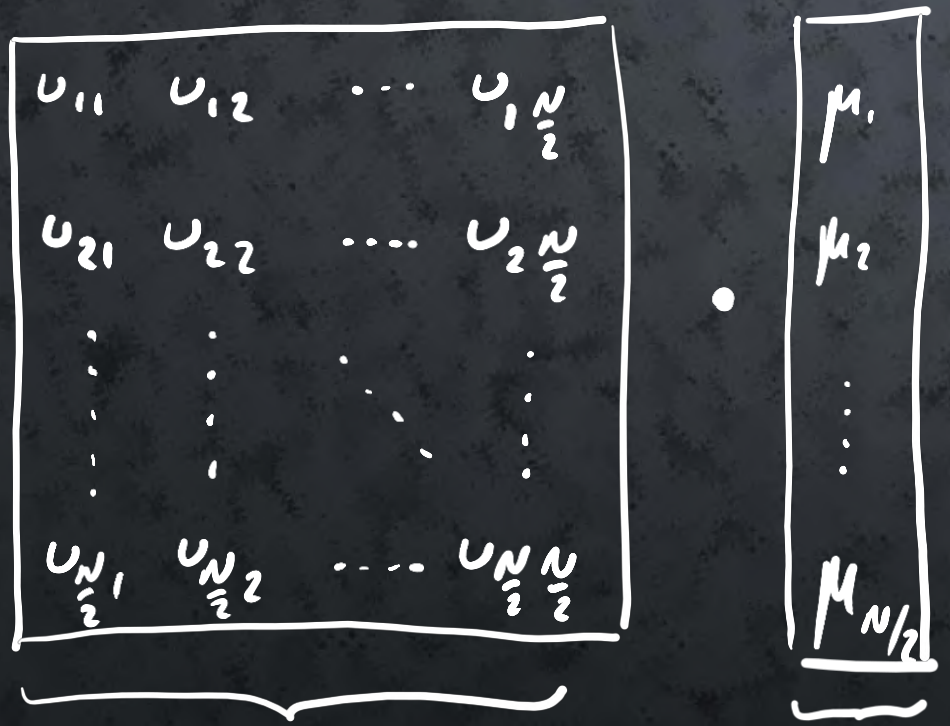
Chaque multiplication fait perdre du module (1 étage).
Quand on n'a plus assez de module… ça semble terminé.

Bootstrap : opération de maintenance permettant de regagner du module.

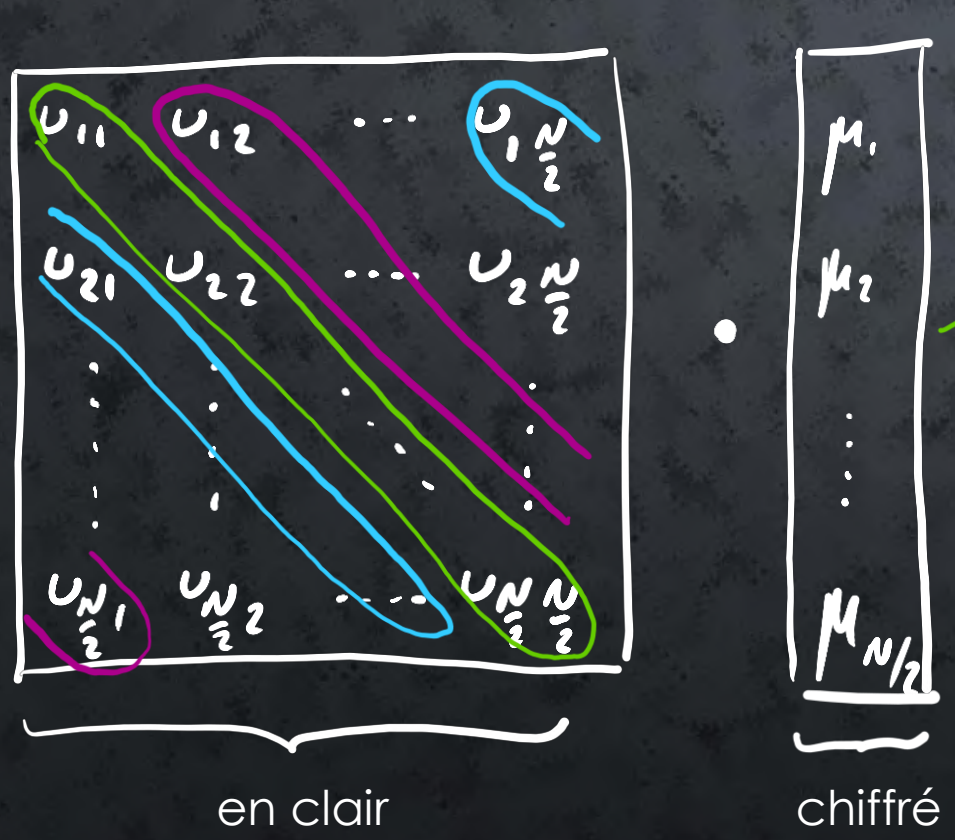
À quoi a-t-on droit pour le bootstrap ?
Aux opérations natives !

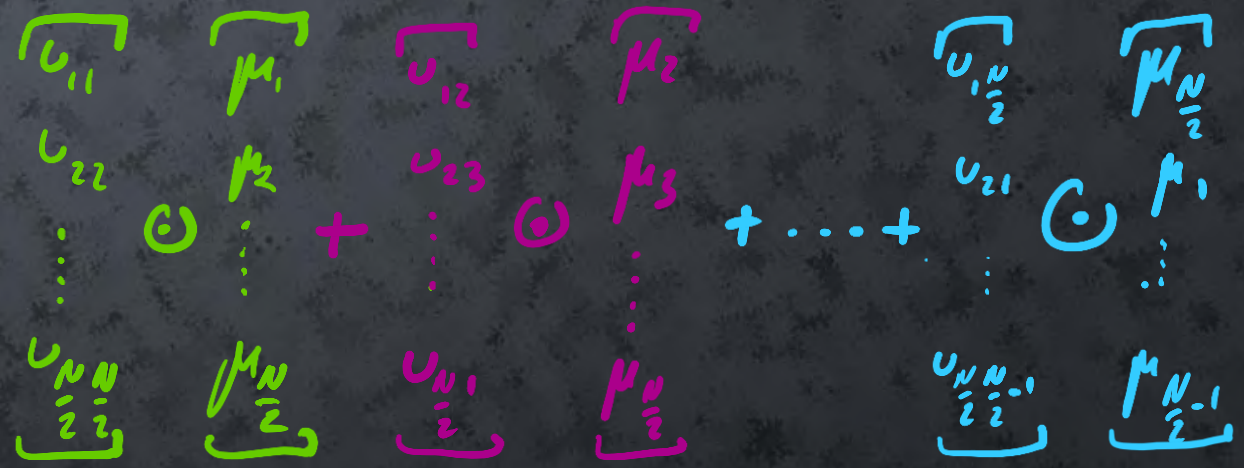
Avec le bootstrap, on pourra calculer indéfiniment.

INTERLUDE 1 : ALGÈBRE LINÉAIRE



INTERLUDE 1 : ALGÈBRE LINÉAIRE



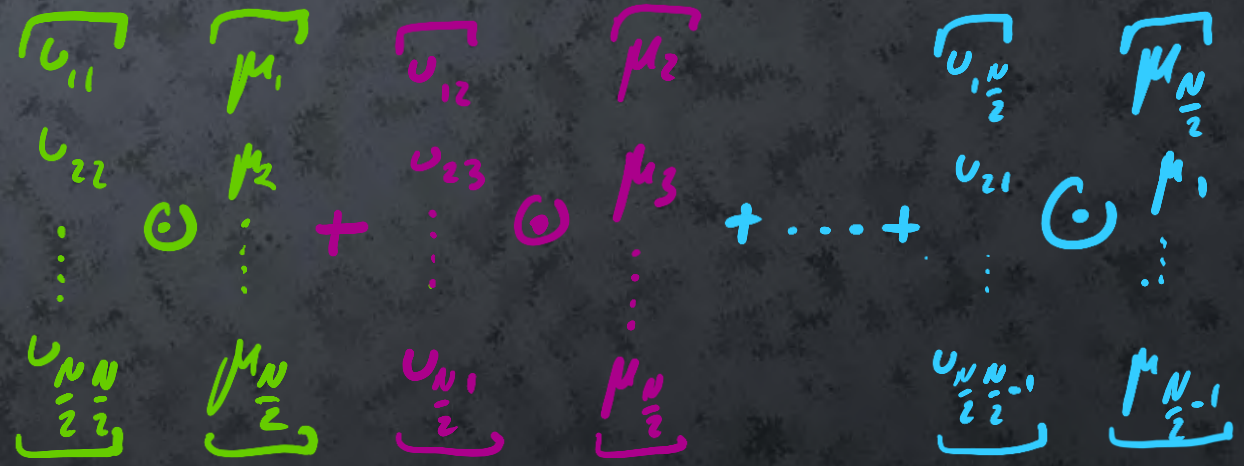


Les diagonales se décalent, le vecteur tourne !

INTERLUDE 1 : ALGÈBRE LINÉAIRE

en clair

chiffré



Les diagonales se décalent, le vecteur tourne !

- N rotations
- $N-1$ additions
- 1 étage

On peut descendre à $0(\sqrt{N})$ rotations avec une approche pas de bébé – pas de géant.

INTERLUDE 2 : ÉVALUATION POLYNOMIALE

$P(\mu) = P_0 + P_1 \cdot \mu + P_2 \cdot \mu^2 + \cdots + P_d \cdot \mu^d$ avec P en clair et μ chiffré.

1- Calculer les puissances de μ

$$\mu^{2k} = \left(\mu^k\right)^2 \quad , \quad \mu^{2k+1} = \mu^k \cdot \mu^{k+1}$$

- $2d$ multiplications
- d additions
- $\lceil \log_2 d \rceil + 1$ étages

2- Évaluer la combinaison linéaire

En fait, ça donne même les évaluations de P en $N/2$ points en parallèle.

LE BOOTSTRAP

OBJECTIF : $\mathrm{ct}\,[q_0] \rightarrow \mathrm{ct}'\ [Q]$ pour un module $Q \gg q_0$ et (à peu près) le même clair/encodé

LE BOOTSTRAP

OBJECTIF : $\mathrm{ct}\,[q_0] \rightarrow \mathrm{ct}'\ [Q]$ pour un module $Q \gg q_0$ et (à peu près) le même clair/encodé

1- ModRaise : $\langle \mathrm{ct}, \mathrm{sk} \rangle \approx \Delta \cdot m\ [q_0] \Rightarrow \langle \mathrm{ct}, \mathrm{sk} \rangle \approx \Delta \cdot m + q_0 \cdot I$ pour un certain $I \in R$

L'équation est dans R, donc modulo q pour tout q.

LE BOOTSTRAP

OBJECTIF : $\mathrm{ct}\,[q_0] \rightarrow \mathrm{ct}'\ [Q]$ pour un module $Q \gg q_0$ et (à peu près) le même clair/encodé

1- ModRaise : $\langle \mathrm{ct}, \mathrm{sk} \rangle \approx \Delta \cdot m\,[q_0] \quad \Rightarrow \quad \langle \mathrm{ct}, \mathrm{sk} \rangle \approx \Delta \cdot m + q_0 \cdot I$ pour un certain $I \in R$

L'équation est dans R, donc modulo q pour tout q.

2- CtoS : $\langle \mathrm{ct}_2, \mathrm{sk} \rangle \approx \Delta \cdot \mathrm{DFT}^{-1}(m + q_0 \cdot I)\ \ [Q_2]$ (produit matrice-vecteur)

3- EvalMod : $\langle \mathrm{ct}_3, \mathrm{sk} \rangle \approx \Delta \cdot \mathrm{DFT}^{-1}(m)\ \ [Q_3]$ (prochain transparent)

4- StoC : $\langle \mathrm{ct}', \mathrm{sk} \rangle \approx \Delta \cdot m\,[Q]$ (produit matrice-vecteur)

EVALMOD

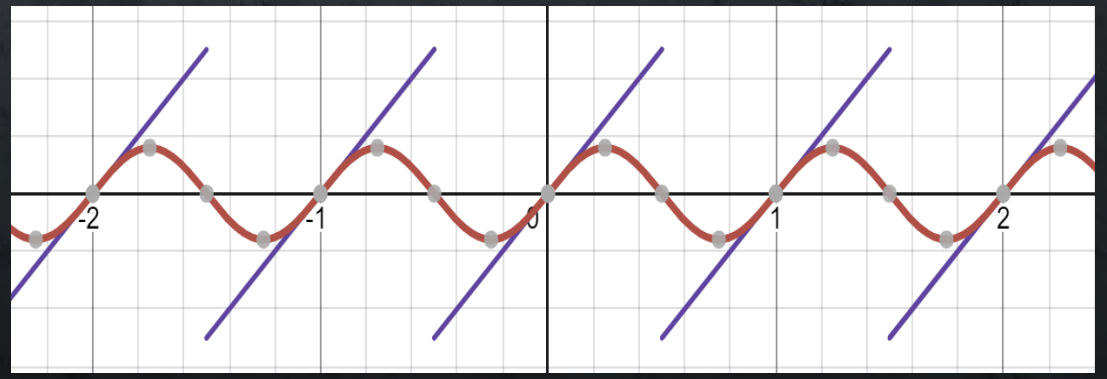
EvalMod : $\langle \mathrm{ct}_2, \mathrm{sk} \rangle \approx \Delta \cdot \mathrm{DFT}^{-1}(m + q_0 \cdot I)\ [Q_2] \quad \Rightarrow \quad \langle \mathrm{ct}', \mathrm{sk} \rangle \approx \Delta \cdot \mathrm{DFT}^{-1}(m)\ \ [Q_3]$

Après passage à l'échelle, on souhaite réduire modulo 1 : $\frac{\Delta}{q_0} m + I \ \Rightarrow \ \frac{\Delta}{q_0} m$

Mais… on ne sait qu'évaluer des polynômes.

1- $x \mapsto x \bmod 1$ ⇒ $x \mapsto \frac{1}{2\pi} \sin(2\pi x)$

2- $x \mapsto \frac{1}{2\pi} \sin(2\pi x)$ ⇒ polynôme



EFFICACITE DU BOOTSTRAP

Précision ≈ 18 bits Étages de calcul effectifs : 15 2^{15} complexes / 2^{16} réels	**CPU** Un fil d'exécution, AVX512 Intel Xeon Gold 6544Y @ 3.6GHz	**CPU** 16 fils d'exécution, AVX512 Intel Xeon Gold 6544Y @ 3.6GHz	**GPU** NVIDIA GeForce RTX 5090
	6.7 s	**1.1 s**	**14 ms**

Le GPU est une ressource raisonnable, le calcul étant délégué à un serveur.

(Ce GPU et ce CPU valent chacun environ 3000 EUR)

Estimation du débit : $2^{16} \cdot 15$ multiplications à 18 bits de précision en 14ms

=> 70M par seconde

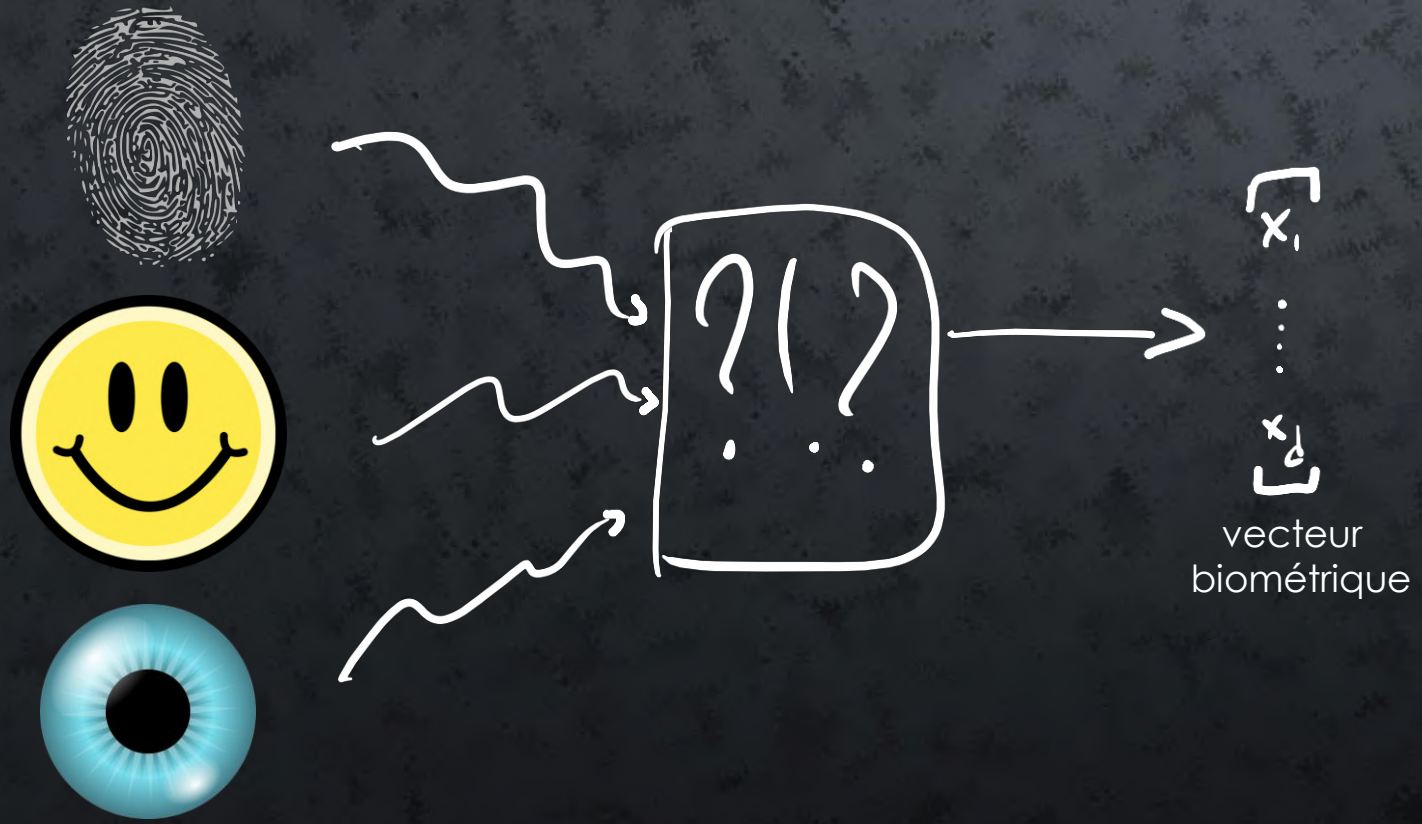
PLAN

1- Arithmétique CKKS

2- Bootstrap

3- Application en biométrie

EXTRACTION DE CARACTÉRISTIQUES BIOMÉTRIQUES



Hypothèses :

- Deux prises de la même empreinte / visage / iris donnent

$$\vec{x}, \vec{y} : \ < \vec{x}, \vec{y} > \geq \ \theta_{\text{OUI}} .$$

- Deux empreintes / visages / iris distinctes donnent

$$\vec{x}, \vec{y} : \ < \vec{x}, \vec{y} > \leq \ \theta_{\text{NON}} .$$

Avec $0 \leq \theta_{\text{NON}} < \theta_{\text{OUI}} \leq 1.$

RECONNAISSANCE 1:N

En stockant une base de données de N vecteurs biométriques, on peut :

- déterminer si un individu en fait partie,
- et/ou identifier l'individu,

en lui prenant son empreinte biométrique.

Exemples :

- Restauration scolaire (doigt)
- Accès à un bâtiment / avion (visage)
- Paiement (visage)
- Preuve d'humanité -- world.org (iris)

Calcul :

Entrées : Un vecteur et une matrice
Sortie : Est-ce qu'une entrée du produit est plus grande que θ_{OUI} ? Laquelle ?

SECURITÉ DE LA BASE DE DONNÉES

Ces vecteurs biométriques contiennent de l'information privée…
qui ne peut pas être renouvelée…

Comment la protéger ?

Chiffrement "classique"	Enclave sécurisée	Calcul multipartite sécurisé
Besoin de déchiffrer l'intégralité de la BD à chaque requête ⇒ Un attaquant ayant accès à ce serveur peut extraire la BD	Sécurité douteuse, cf rapport de l'ANSSI *"Technical Position Paper on Confidential Computing"*	La sécurité repose sur la non-collusion d'un nombre suffisant de serveurs. Ce que stocke chaque serveur doit être protégé.

SOLUTION AVEC CKKS

Chiffrer la base de données : $\mathrm{Enc}(\mathbf{BD})$

Chiffrer la requête : $\mathrm{Enc}(\vec{x})$

Algèbre linéaire homomorphe : $\mathrm{Enc}(\mathbf{BD} \cdot \vec{x})$

Comparer au seuil θ_{OUI} : $\mathrm{Enc}(\mathbf{BD} \cdot \vec{x} \geq \theta_{\mathrm{OUI}})$ // entrée par entrée

SOLUTION AVEC CKKS

Chiffrer la base de données : $\mathrm{Enc}(\mathbf{BD})$

Chiffrer la requête : $\mathrm{Enc}(\vec{x})$

Algèbre linéaire homomorphe : $\mathrm{Enc}(\mathbf{BD} \cdot \vec{x})$

Comparer au seuil θ_{OUI} : $\mathrm{Enc}(\mathbf{BD} \cdot \vec{x} \geq \theta_{\mathrm{OUI}})$ // entrée par entrée

On obtient un chiffré d'un vecteur qui contient :

- 1 dans la possible entrée qui correspond,
- et 0 ailleurs.

Ensuite, il "suffit" de déchiffrer.

(1) Y. Bae, J.H. Cheon, G. Hanrot, J.H. Park & D. Stehlé: Fast Homomorphic Linear Algebra with BLAS. Arxiv 2503.16080

REMARQUES SUR LE CALCUL

$$\mathrm{Enc}(\mathbf{BD}),\ \mathrm{Enc}(\vec{x}) \quad \mapsto \quad \mathrm{Enc}(\mathbf{BD}\cdot\vec{x} \geq \theta_{\mathrm{OUI}})$$

- Si on déchiffre $\mathrm{Enc}(\mathbf{BD}\cdot\vec{x})$ plutôt que $\mathrm{Enc}(\mathbf{BD}\cdot\vec{x} \geq \theta_{\mathrm{OUI}})$, un attaquant risque de pouvoir apprendre $\vec{x}$ et $\mathbf{BD}\cdot\vec{x}$ …. Ce qui révèle rapidement $\mathbf{BD}$.

- Pour minimiser le coût et la communication, on calcule $\mathrm{Enc}(\mathbf{BD}\cdot\vec{x})$ avec un petit module q.
  (l'algèbre linéaire chiffrée est quasiment aussi efficace qu'en clair (1).)

- La comparaison est approchée par un polynôme, évalué homomorphiquement.
  - Nécessite de bootstrapper pour avoir du module,
  - Quantité de données divisée par d (la dimension des vecteurs) .

COÛT

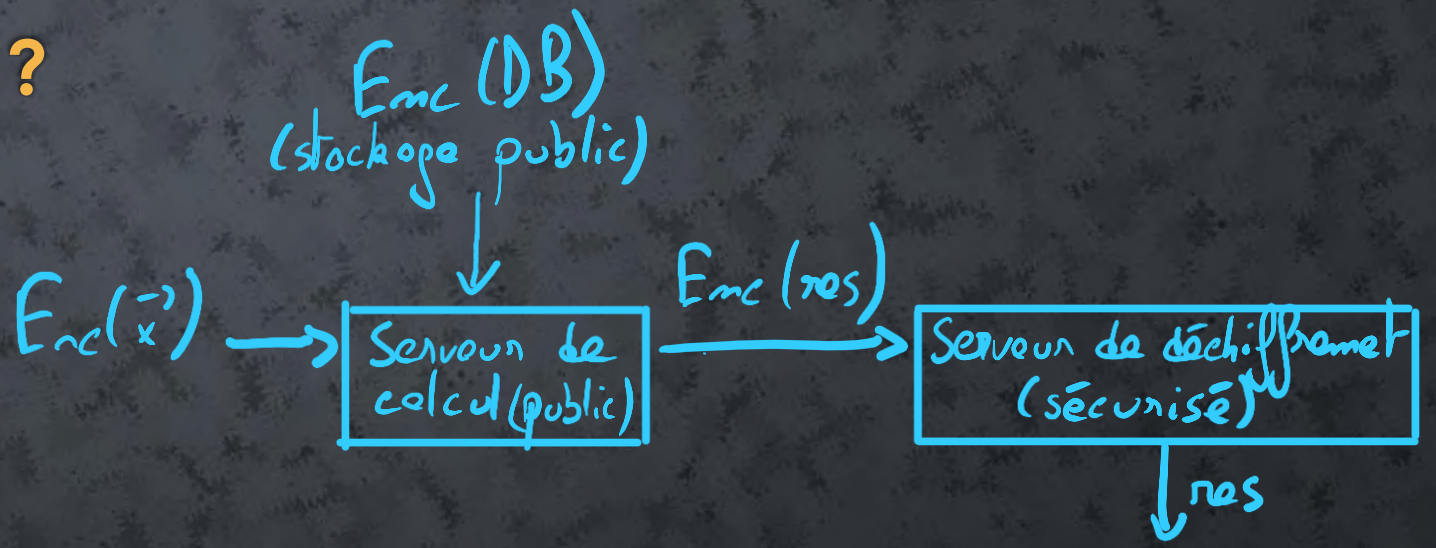
- Pour minimiser le coût et la communication, on calcule $\mathbf{Enc}(\mathbf{BD} \cdot \vec{x})$ avec un petit module q.
- La comparaison est approchée par un polynôme, évalué homomorphiquement.
  - Nécessite de bootstrapper pour avoir du module,
  - Quantité de données divisée par d (la dimension des vecteurs) .

~25ms pour 65K vecteurs dans la base de données avec 1 GPU. ⇒ Reconnaissance de 1 parmi 2.6M en 1s, par GPU.

(et il y a moyen de faire mieux ☺)

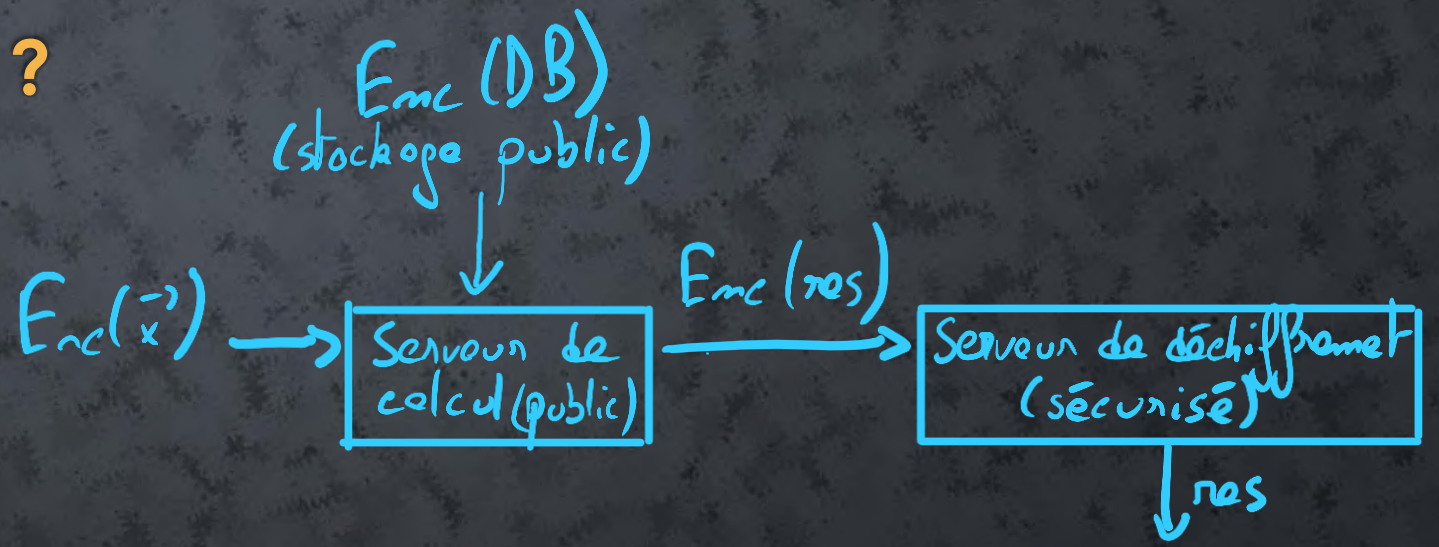
... MAIS QUI DÉCHIFFRE ?

La clé de déchiffrement peut être stockée sur un autre serveur.



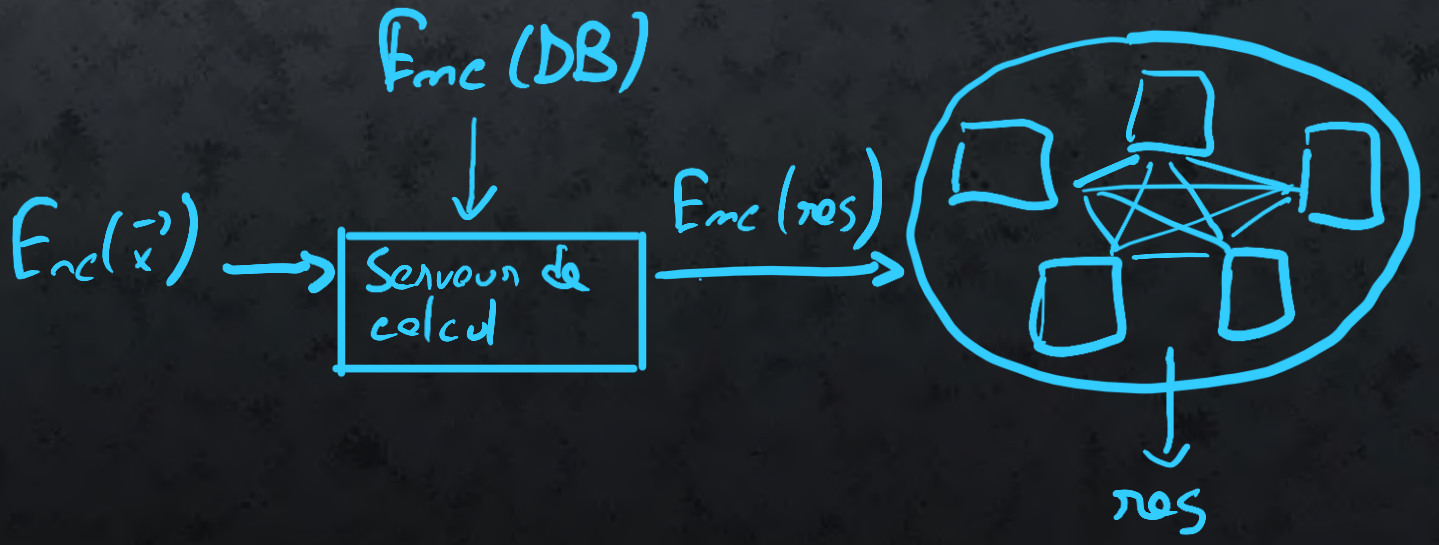
… MAIS QUI DÉCHIFFRE ?

La clé de déchiffrement peut être stockée sur un autre serveur.



On peut également partager la clé sur différents serveurs, qui doivent collaborer pour déchiffrer (cryptographie à seuil) :

- Meilleure robustesse face aux pannes
- Sécurité renforcée



Quelques liens :
https://eprint.iacr.org/2016/421
https://eprint.iacr.org/2018/153
https://heaan.io/
https://ckks.org/

QUESTIONS ?

damien.stehle@cryptolab.co.kr

HEAAN
CRYPTO LAB