Algorithmes d'échantillonnages haute précision sur les réseaux euclidiens et leur rôle en cryptographie

Léo Ducas

CWI, Amsterdam, the Netherlands

Seminaire RAIM, 8 Avril 2015

Outline

Outline

Avantages de la cryptographie fondée sur les réseaux

Meilleures garanties de sécurité:

- réductions pire-cas/cas-moyen
- résistances aux algorithmes quantiques
- problèmes étudiés au delà de la cryptographie.

Plus souple:

- Chiffrement homomorphe, fonctionnel, encodage multilinéaire.

Plus efficace:

- complexité en $\tilde{O}(n^2)$ voir $\tilde{O}(n)$
- parallélisable
- opérations modulo un petit entier q.

Avantages de la cryptographie fondée sur les réseaux

Meilleures garanties de sécurité:

- réductions pire-cas/cas-moyen
- résistances aux algorithmes quantiques
- problèmes étudiés au delà de la cryptographie.

Plus souple:

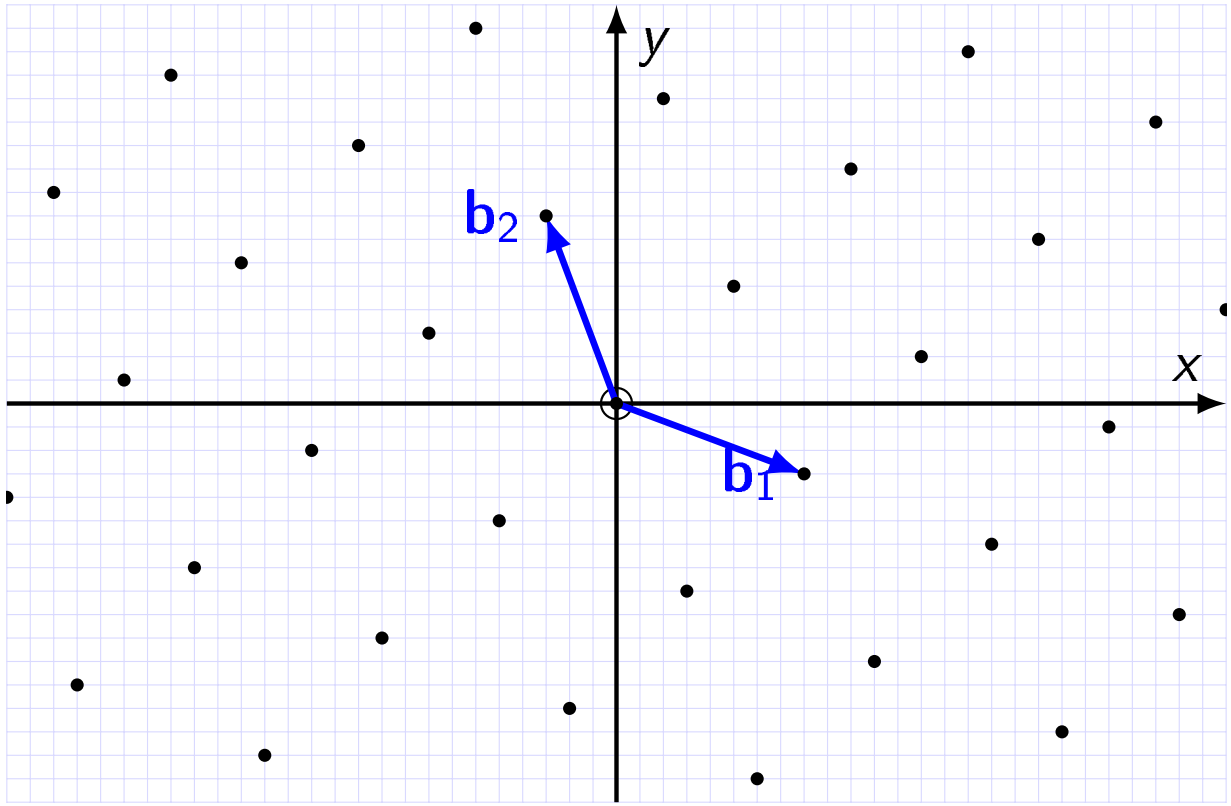
- Chiffrement homomorphe, fonctionnel, encodage multilinéaire.

Plus efficace:

- complexité en $\tilde{O}(n^2)$ voir $\tilde{O}(n)$
- parallélisable
- opérations modulo un petit entier q.

En théorie . . .

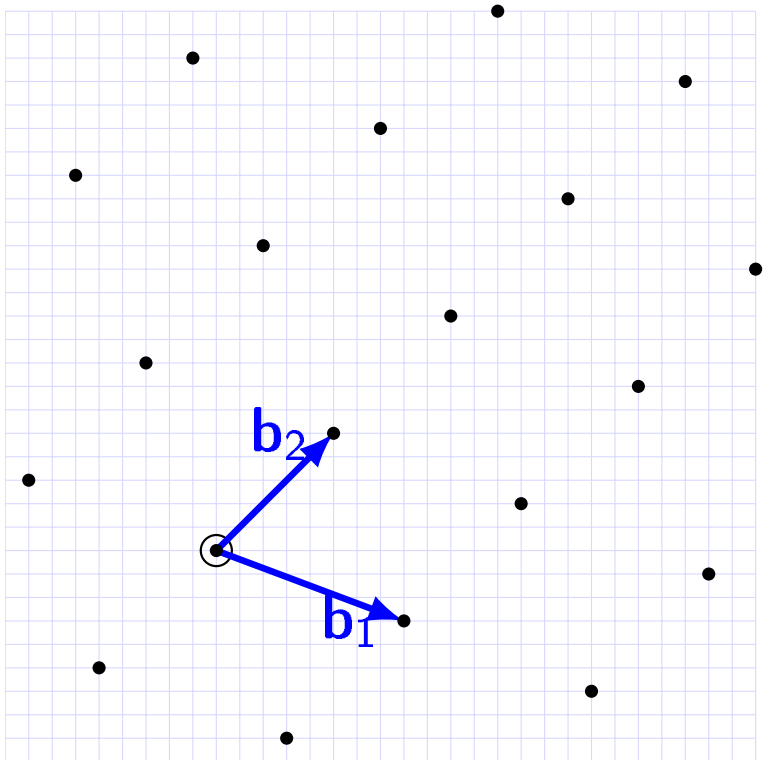
Réseaux euclidiens



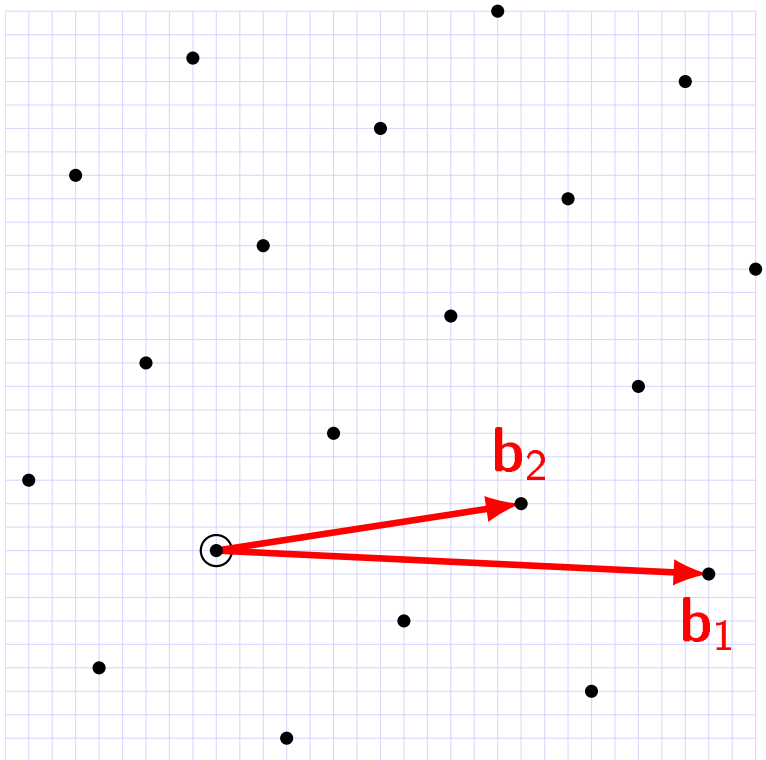
Definition (Réseau euclidien)

Un réseau euclidien L est un sous-groupe discret d'un espace vectoriel de dimension fini $\mathbb{R}^n$.

Bases d'un réseau



Bonne Base **B** de L



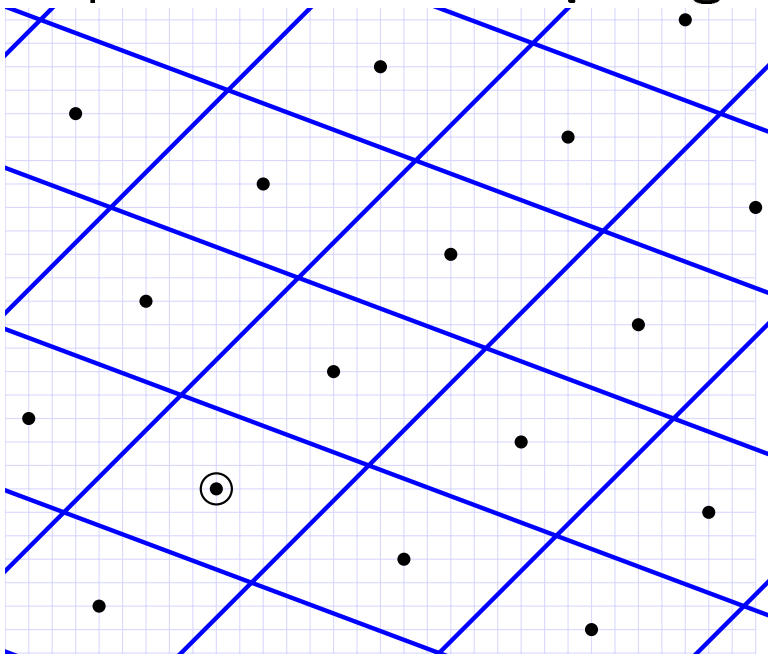
Mauvaise Base **M** de L

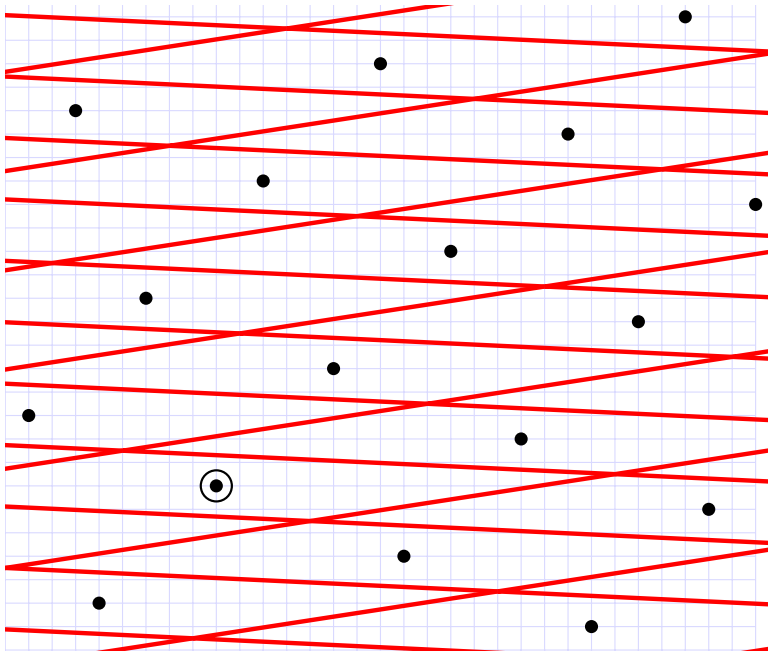
B $\rightarrow$ **M** : facile (randomisation);

M $\rightarrow$ **B** : difficile (réduction de réseau).

Bases et domaines fondamentaux

Chaque base définit un **pavage parallélépipédique**.

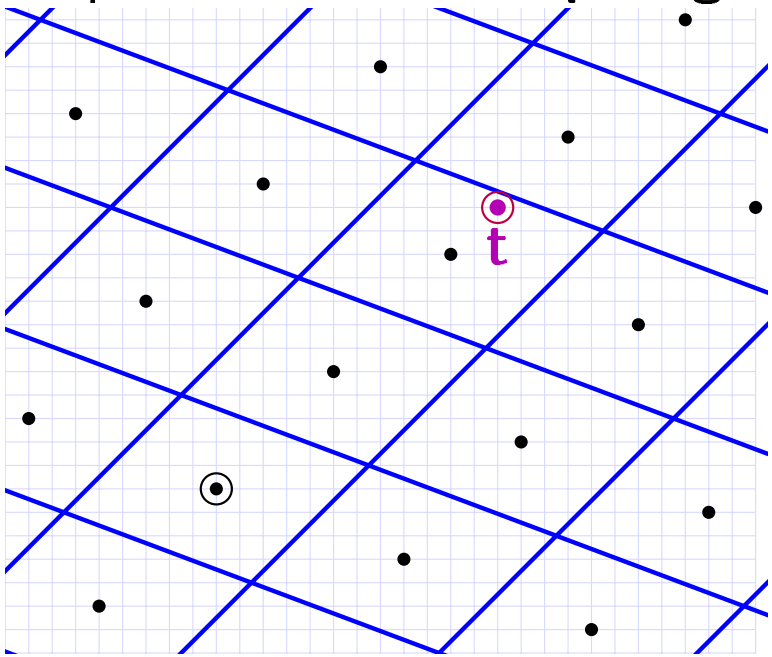


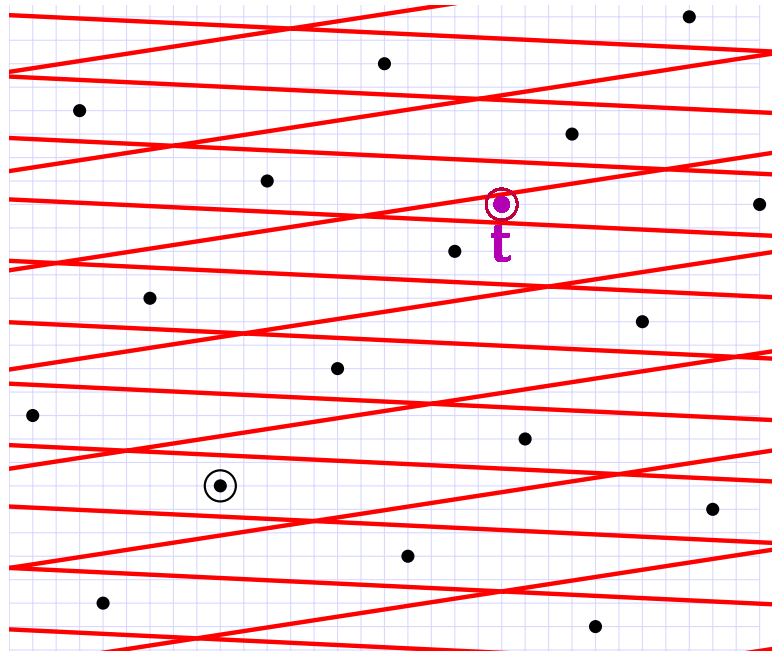


Algorithmes de Babai:

Bases et domaines fondamentaux

Chaque base définit un **pavage parallélépipédique**.



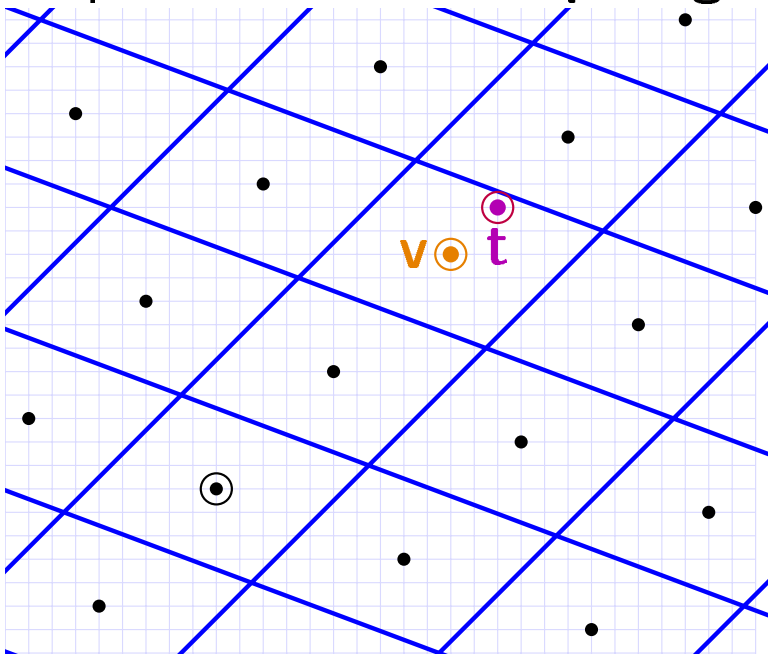


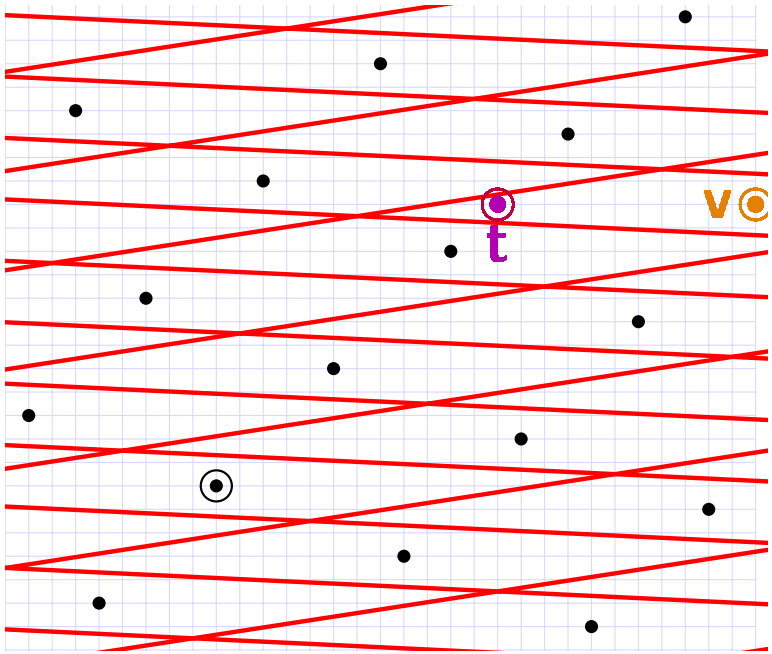
Algorithmes de Babai:

- étant donné un point **t**

Bases et domaines fondamentaux

Chaque base définit un **pavage parallélépipédique**.

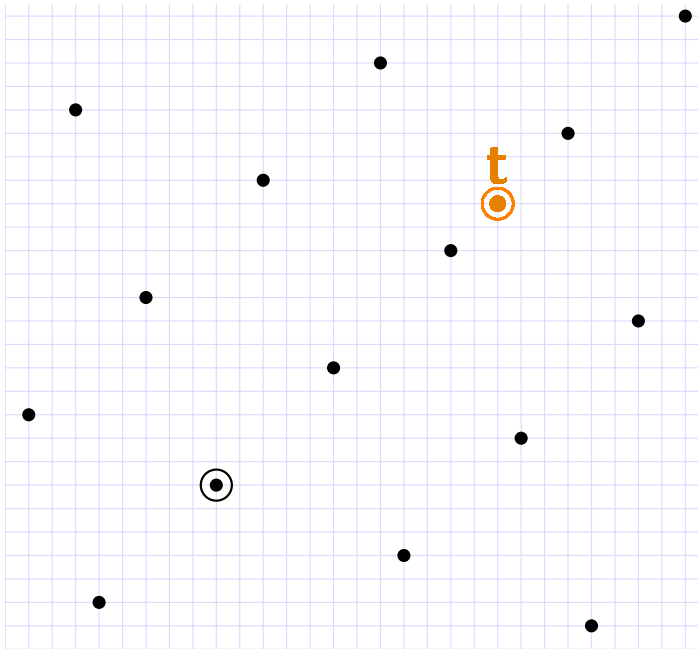


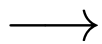


Algorithmes de Babai:

- étant donné un point $\mathbf{t}$
- trouve le point $\mathbf{v} \in L$ dans le même parallélépipède.

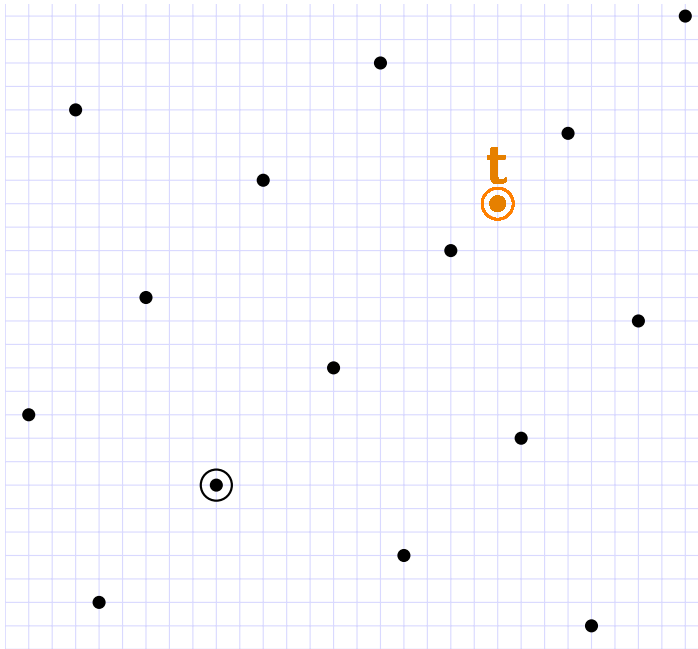
Arrondi de Babai





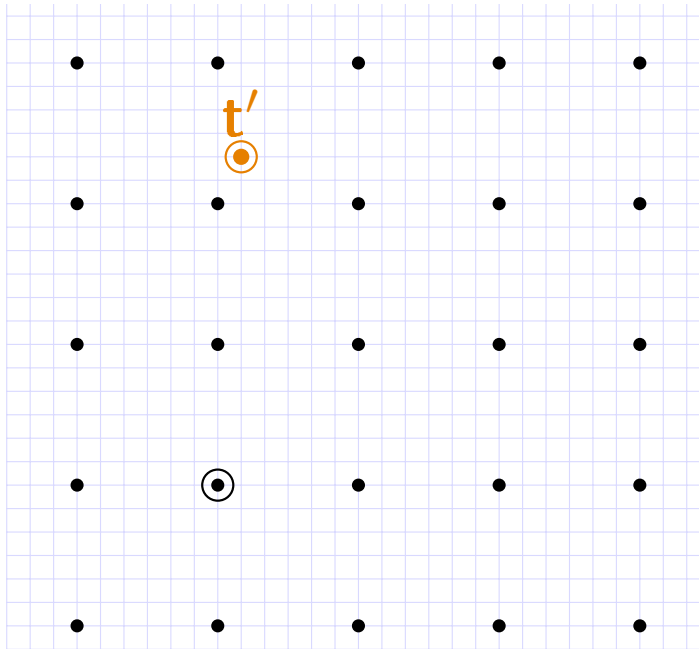
Algorithme de Babai, arrondi simple:

Arrondi de Babai



$\times \mathbf{B}^{-1}$

$\longrightarrow$

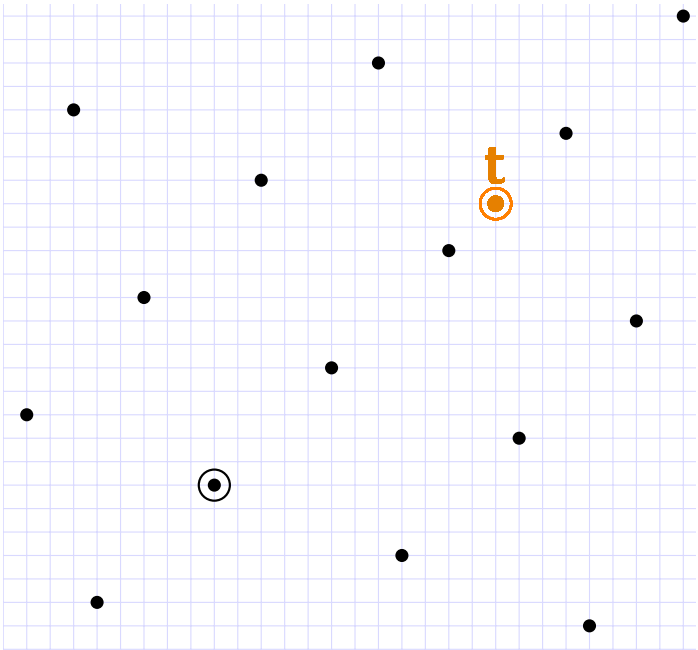


Algorithme de Babai, arrondi simple:

- utiliser la base $\mathbf{B}$ pour se ramener au réseau $\mathbb{Z}^n$ $(\times \mathbf{B}^{-1})$

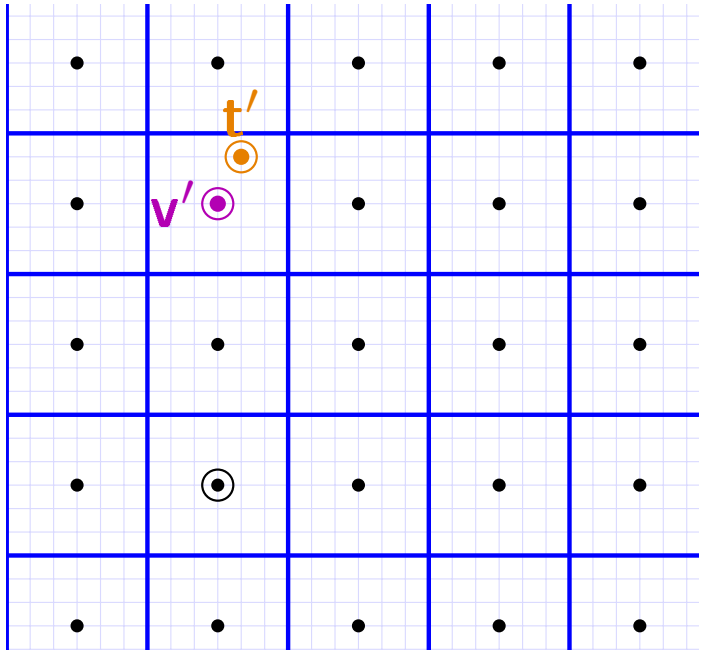
$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t};$$

Arrondi de Babai



$\times \mathbf{B}^{-1}$

$\longrightarrow$

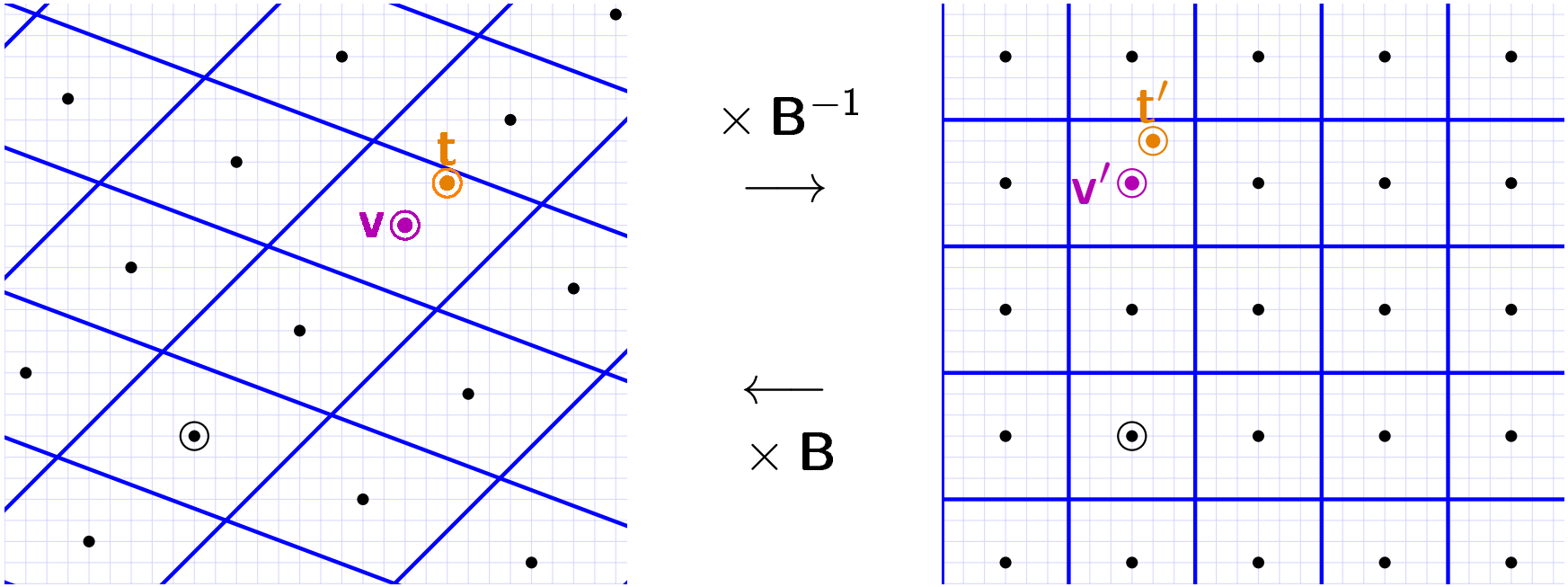


Algorithme de Babai, arrondi simple:

- utiliser la base $\mathbf{B}$ pour se ramener au réseau $\mathbb{Z}^n$ $(\times \mathbf{B}^{-1})$
- arrondir chaque coordonnée (découpage carré)

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \quad \mathbf{v}' = \lfloor \mathbf{t}' \rceil;$$

Arrondi de Babai



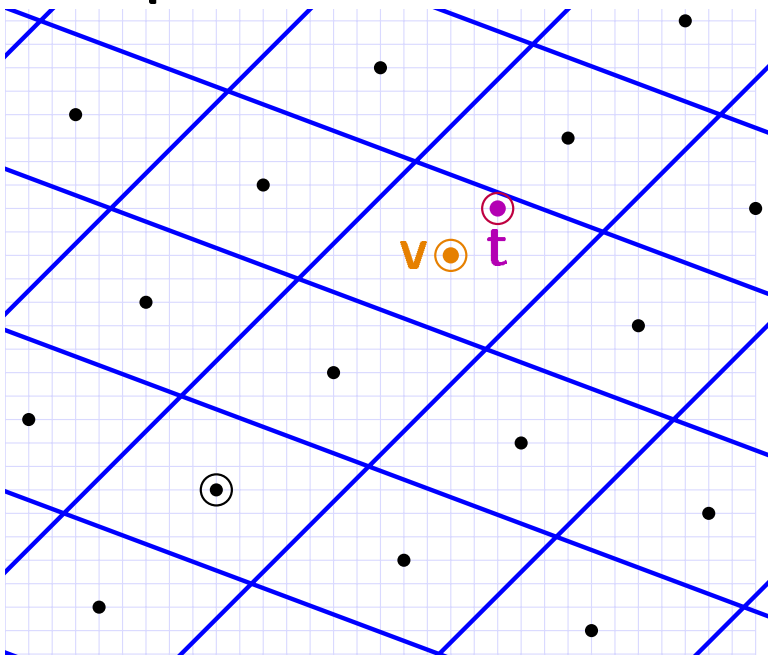
Algorithme de Babai, arrondi simple:

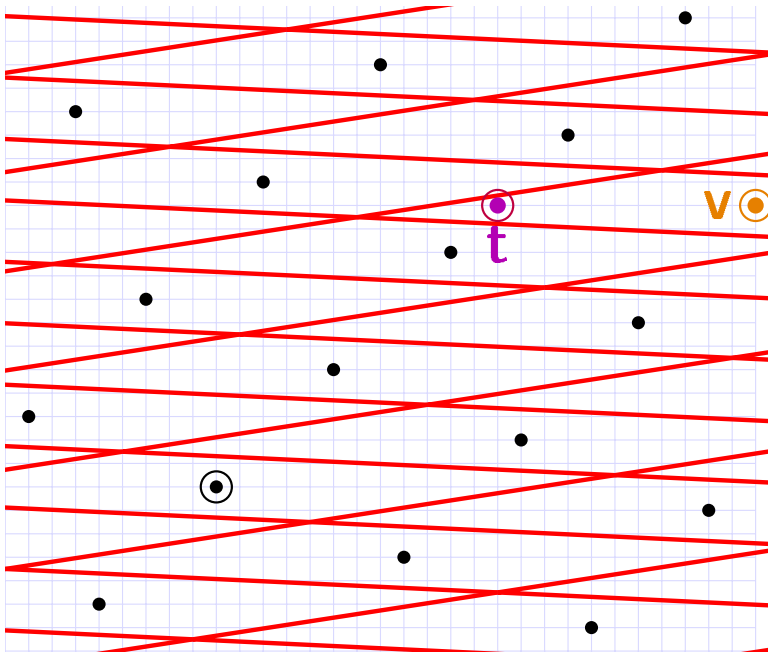
- utiliser la base $\mathbf{B}$ pour se ramener au réseau $\mathbb{Z}^n$ $(\times \mathbf{B}^{-1})$
- arrondir chaque coordonnée (découpage carré)
- revenir au réseau L $(\times \mathbf{B})$

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \quad \mathbf{v}' = \lfloor \mathbf{t}' \rceil; \quad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$

Trouver des vecteurs proches

Muni d'une bonne base **B** l'algorithme de Babai permet de trouver des vecteurs proches.
Muni seulement d'une mauvaise base **M**, trouver un vecteur proche est un **problème difficile**.





Cette différence permet de fonder la **cryptographie asymétrique**.

Outline

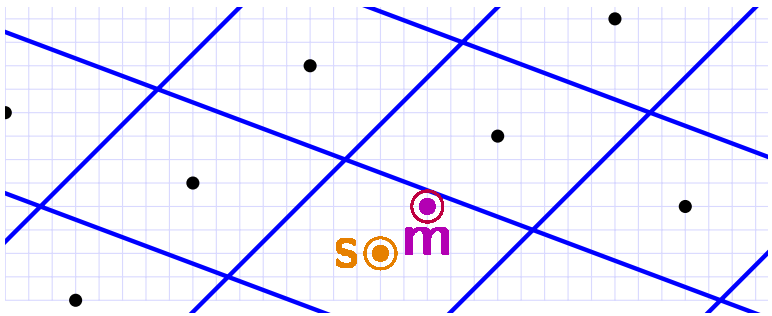
Un première tentative: Signature de [GGH97]

Signature

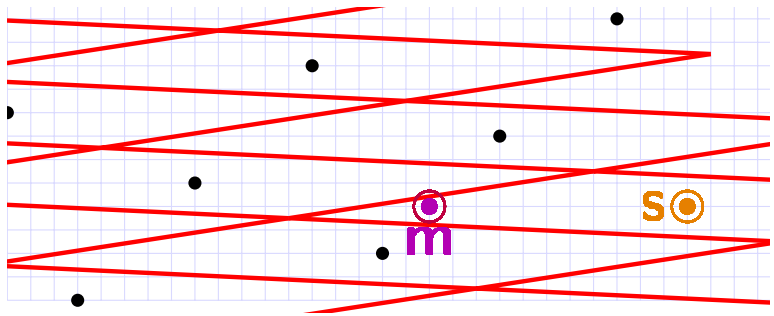
- hacher le message comme un vecteur aléatoire **m**.
- appliquer l'algorithme Babai avec **B**:
  trouver **s** $\in L$ proche de **m** .

Vérification

- vérifier que **s** $\in L$ en utilisant **M**
- et que **m** est proche de **s**.



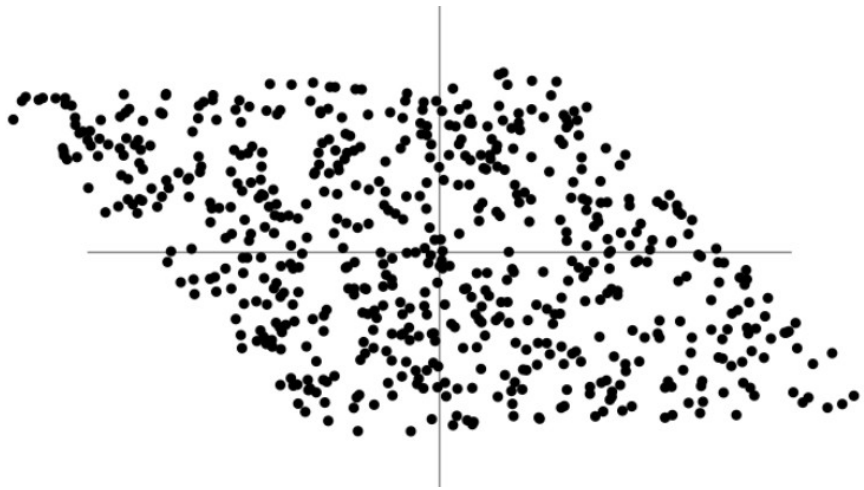
Signature correcte (proche)



Signature incorrecte (éloignée)

Une attaque statistique [NR05]

La difference $\mathbf{s} - \mathbf{m}$ est toujours à l'interieur de parrallepipèdes engendré par la bonne base $\mathbf{B}$:



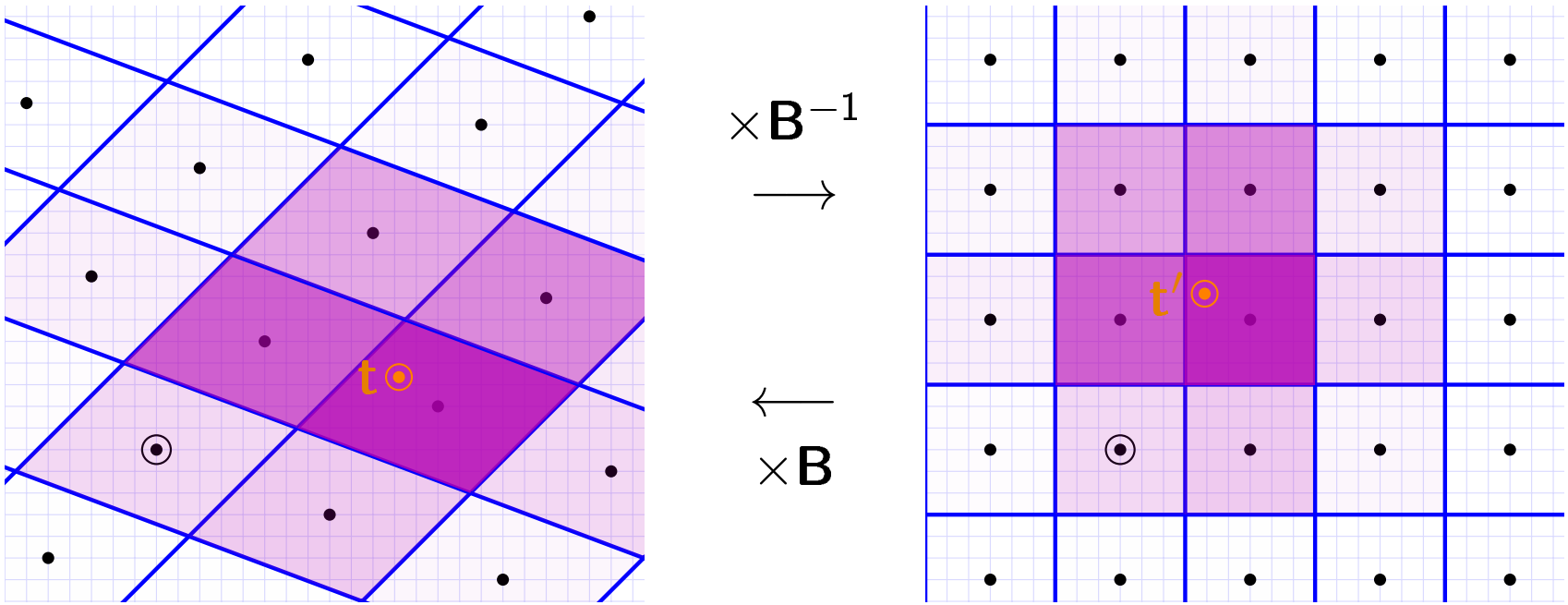
Ainsi, chaque signature $(\mathbf{s}, \mathbf{m})$ laisse fuiter de l'information su la clé secrète $\mathbf{B}$. Nguyen et Regev ont montré comment reconstruire efficacement la clé secrète, etant donné 500 signatures.

La solution: Randomiser l'Algorithme de Babai

Algorithme de Babai:

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \quad \mathbf{v}' = \lfloor \mathbf{t}' \rceil; \quad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$

Idée: Cacher le parallélépipède en randomisant l'arrondi:

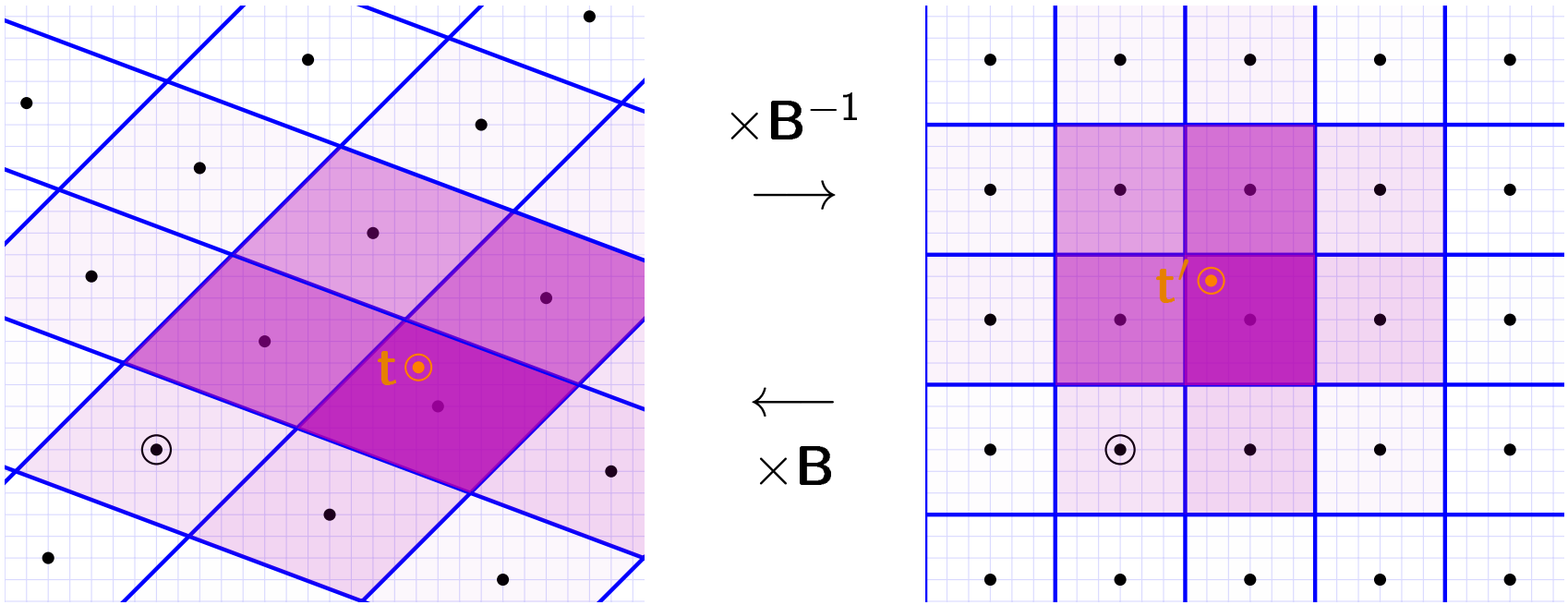


La solution: Randomiser l'Algorithme de Babai

Algorithme de Babai:

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \quad \mathbf{v}' = \lfloor \mathbf{t}' \rceil; \quad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$

Idée: Cacher le parallélépipède en randomisant l'arrondi:

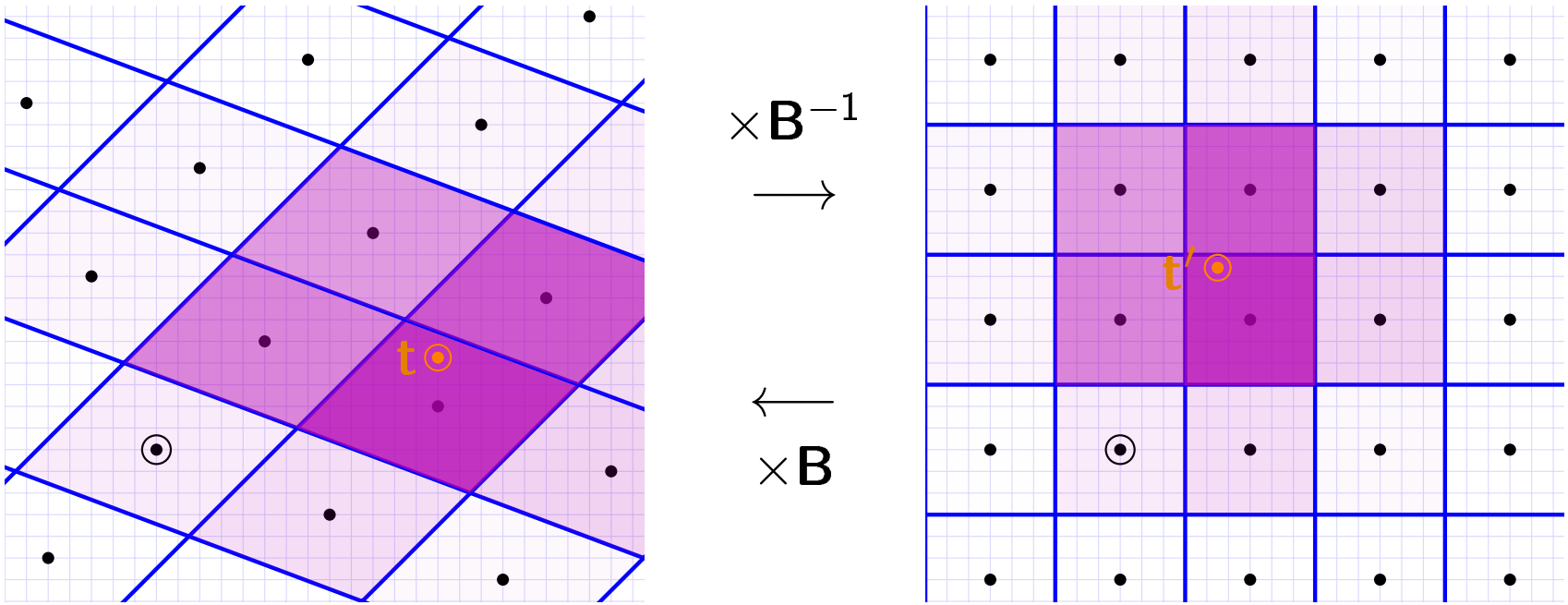


La solution: Randomiser l'Algorithme de Babai

Algorithme de Babai:

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \quad \mathbf{v}' = \lfloor \mathbf{t}' \rceil; \quad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$

Idée: Cacher le parallélépipède en randomisant l'arrondi:

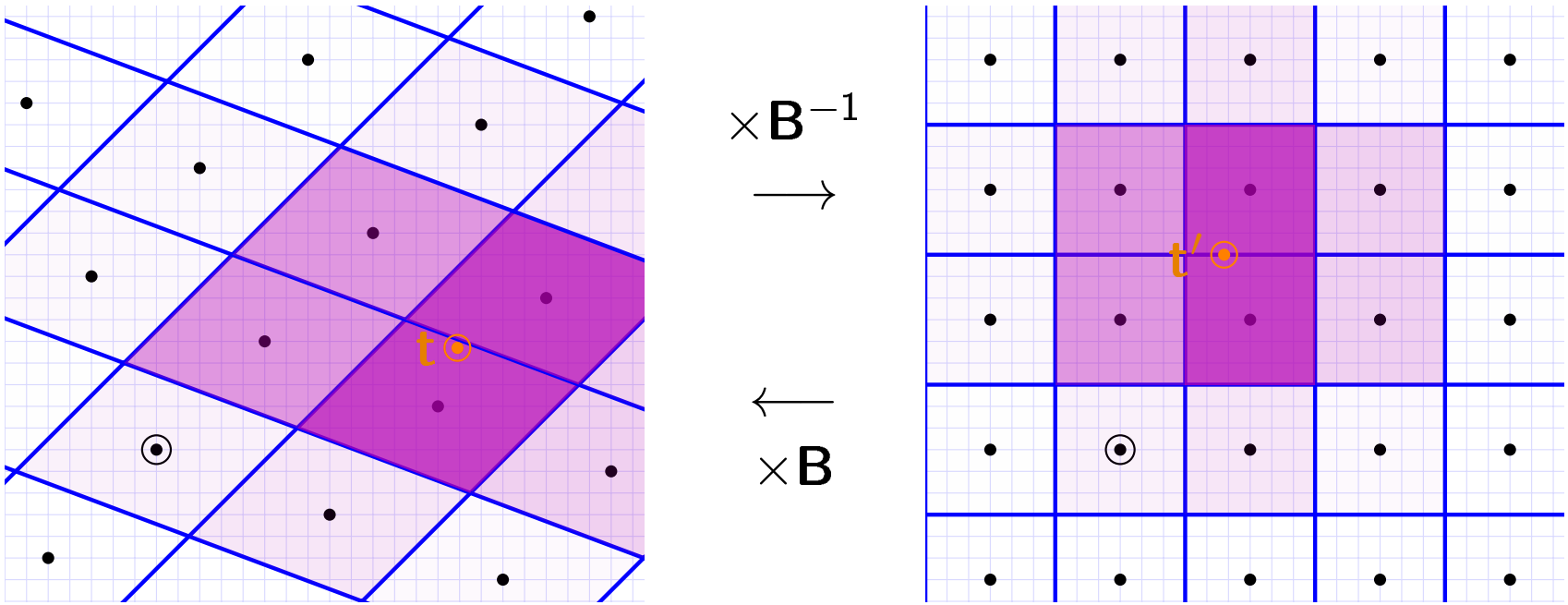


La solution: Randomiser l'Algorithme de Babai

Algorithme de Babai:

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \quad \mathbf{v}' = \lfloor \mathbf{t}' \rceil; \quad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$

Idée: Cacher le parallélépipède en randomisant l'arrondi:

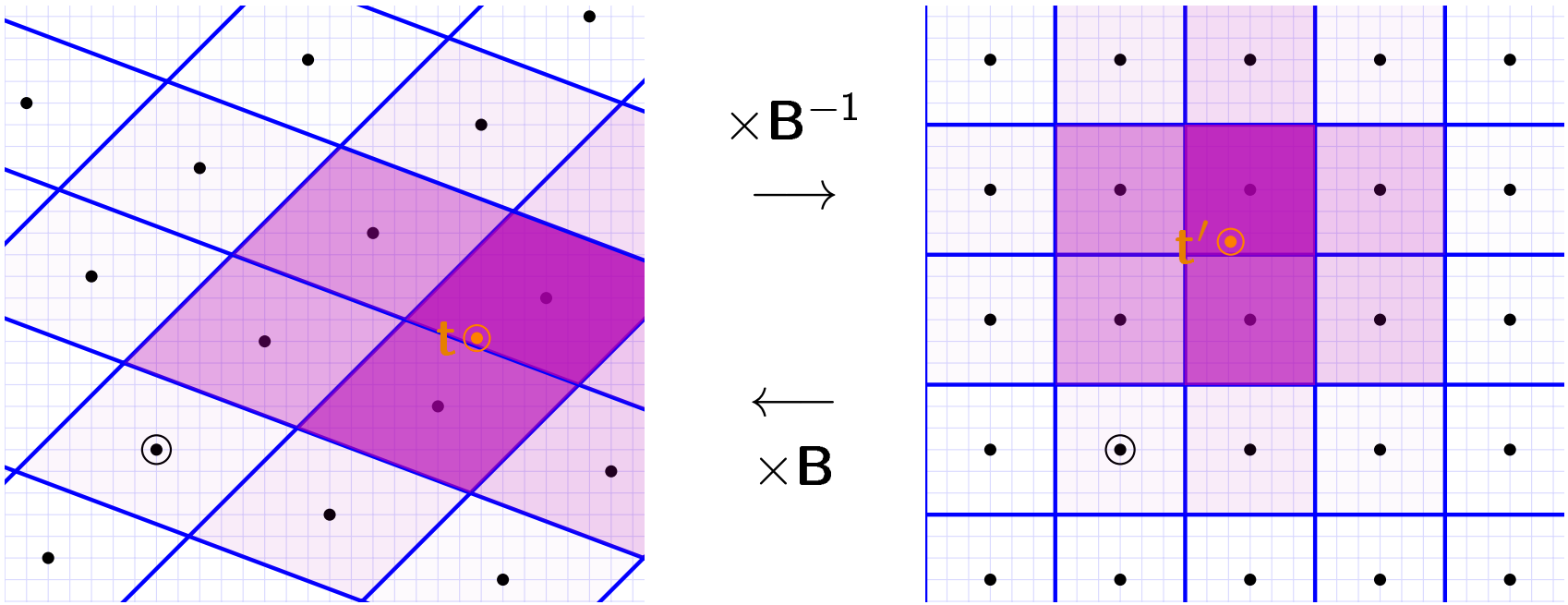


La solution: Randomiser l'Algorithme de Babai

Algorithme de Babai:

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \quad \mathbf{v}' = \lfloor \mathbf{t}' \rceil; \quad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$

Idée: Cacher le parallélépipède en randomisant l'arrondi:

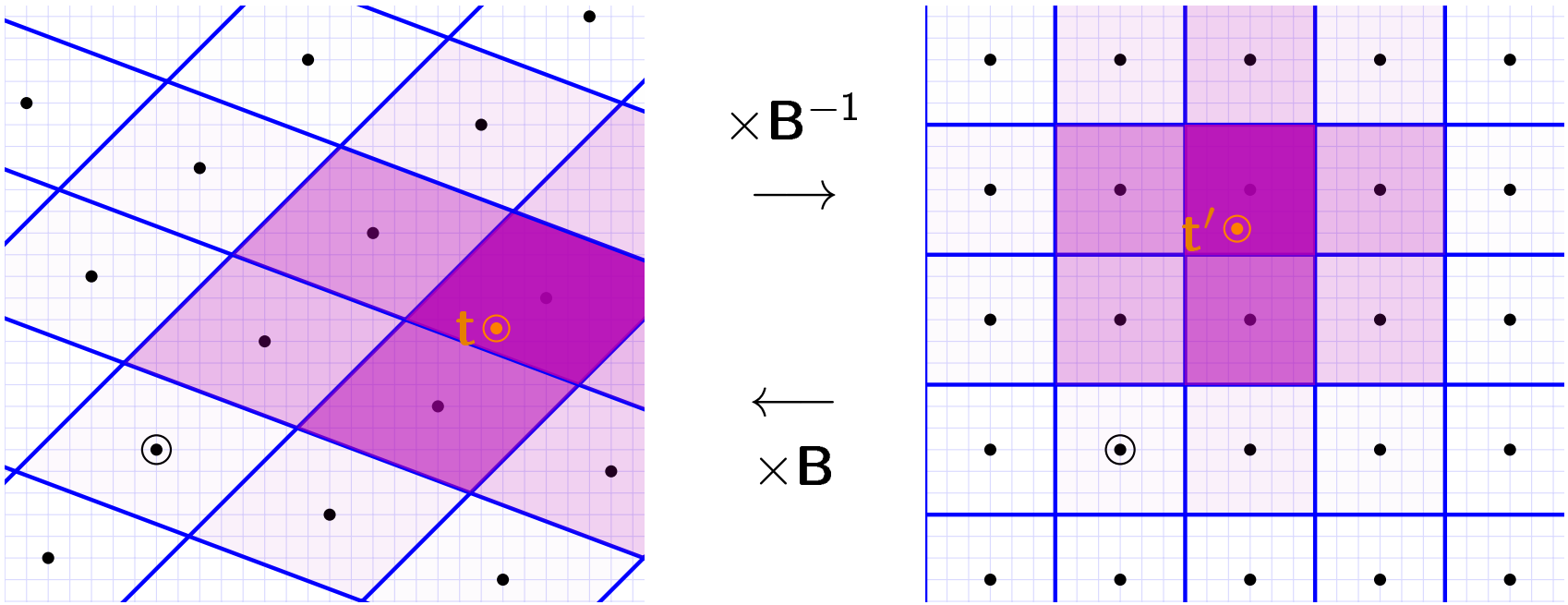


La solution: Randomiser l'Algorithme de Babai

Algorithme de Babai:

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \quad \mathbf{v}' = \lfloor \mathbf{t}' \rceil; \quad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$

Idée: Cacher le parallélépipède en randomisant l'arrondi:

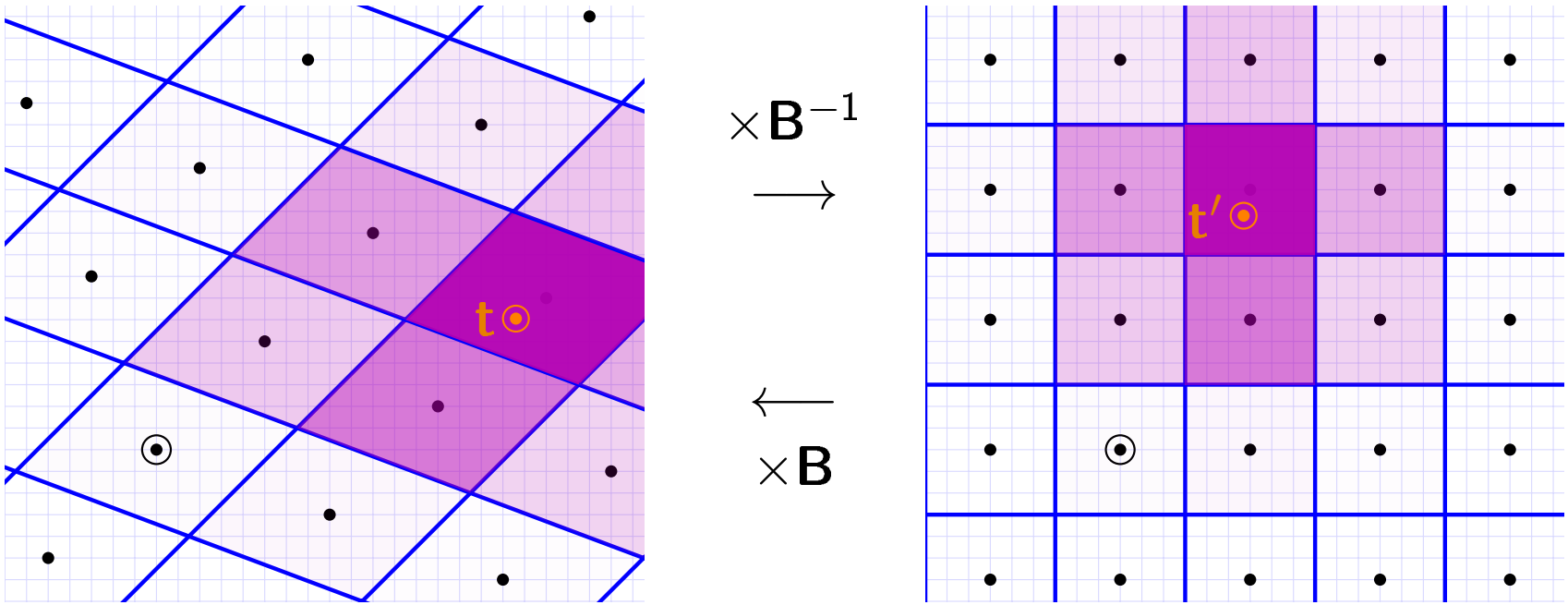


La solution: Randomiser l'Algorithme de Babai

Algorithme de Babai:

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \quad \mathbf{v}' = \lfloor \mathbf{t}' \rceil; \quad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$

Idée: Cacher le parallélépipède en randomisant l'arrondi:

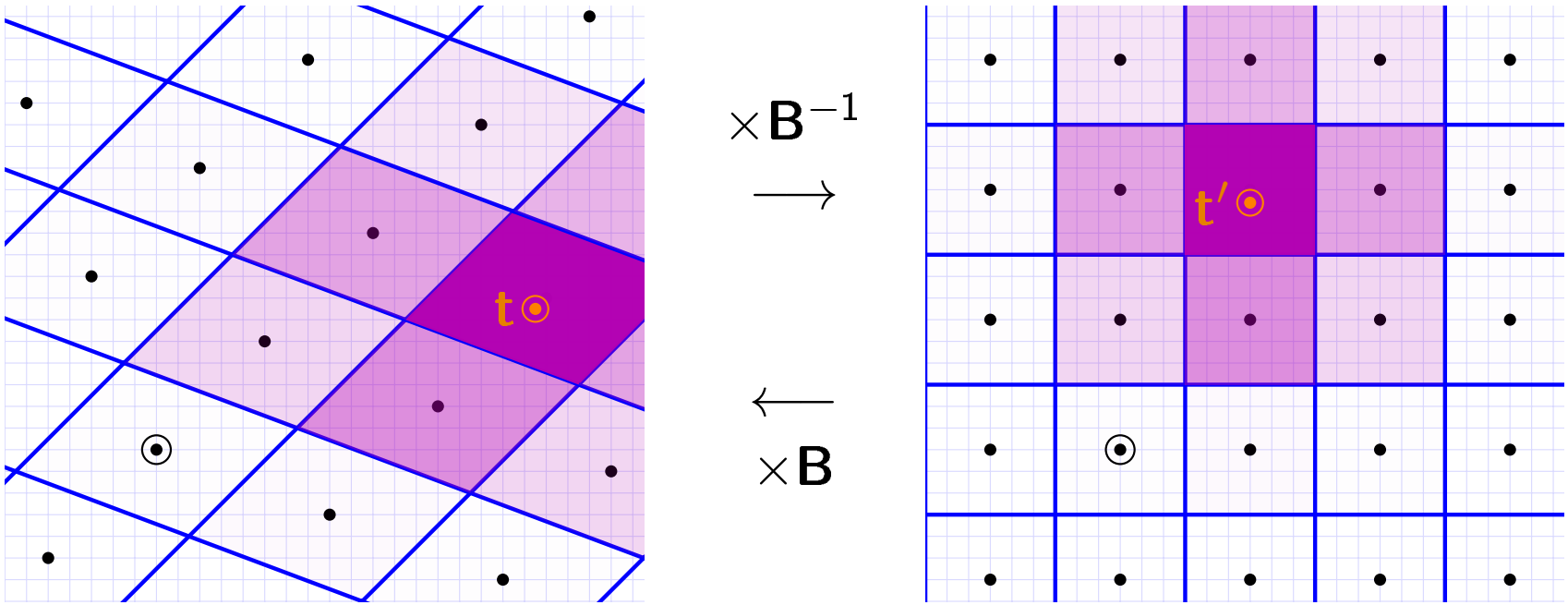


La solution: Randomiser l'Algorithme de Babai

Algorithme de Babai:

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \quad \mathbf{v}' = \lfloor \mathbf{t}' \rceil; \quad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$

Idée: Cacher le parallélépipède en randomisant l'arrondi:



Gaussienne discrète et lissage

L'arrondi est remplacé par une gaussienne discrète:

$$\mathbf{v}' \leftarrow D_{\mathbb{Z}^n,\sigma,\mathbf{t}'} : \quad \mathbf{v}' \in \mathbb{Z}^n, p_{\mathbf{v}'} \sim \exp\left(\frac{-\|\mathbf{v}' - \mathbf{t}'\|^2}{2\sigma^2}\right)$$

Pour $\sigma \to 0$ on retombe sur l'algorithme non randomisé. Par contre

Theorem (Lissage gaussien [GPV08])

Si $\sigma \geqslant O(\sqrt{\log 1/\epsilon})$, pour $\mathbf{t}'$ "uniforme" et $\mathbf{v}' \leftarrow D_{\mathbb{Z}^n,\sigma,\mathbf{t}'}$ alors la distrib. de $\mathbf{v}' - \mathbf{t}'$ est proche d'une gaussienne continue:

$$\mathbf{v}' - \mathbf{t}' \approx_{n\epsilon} D_{\mathbb{R}^n,\sigma}.$$

On prouve ainsi que cet algorithme Randomisé ne révèle **aucune information** sur la clé secrète.

Outline

Efficacité de l'échantillonnage Gaussien Discret (ÉGD)

L'algorithme précédent échantillonnent selon:

$$D_{L,\sigma,\mathbf{t}'}$$

Il opère dans $\mathbb{R} \Rightarrow$ approximation par **l'arithmétique flottante**.

Question: Quelle précision est requise ? Quelle efficacité au final ?

Efficacité de l'échantillonnage Gaussien Discret (ÉGD)

L'algorithme précédent échantillonnent selon:

$$D_{L,\sigma,\mathbf{t}'}$$

Il opère dans $\mathbb{R} \Rightarrow$ approximation par **l'arithmétique flottante**.

Question: Quelle précision est requise ? Quelle efficacité au final ?

Definition (ϵ-correction de l'ÉGD)

Un algorithme d'ÉGD sera dit ϵ-correct si sa distribution de sortie $\mathcal{D}$ est à distance statistique de $D_{\mathbb{Z}^n,\sigma,\mathbf{t}'}$ inférieure à ϵ:

$$\mathcal{D} \approx_\epsilon D_{\mathbb{Z}^n,\sigma,\mathbf{t}'}.$$

Efficacité de l'échantillonnage Gaussien Discret (ÉGD)

L'algorithme précédent échantillonnent selon:

$$D_{L,\sigma,\mathbf{t}'}$$

Il opère dans $\mathbb{R} \Rightarrow$ approximation par **l'arithmétique flottante**.

Question: Quelle précision est requise ? Quelle efficacité au final ?

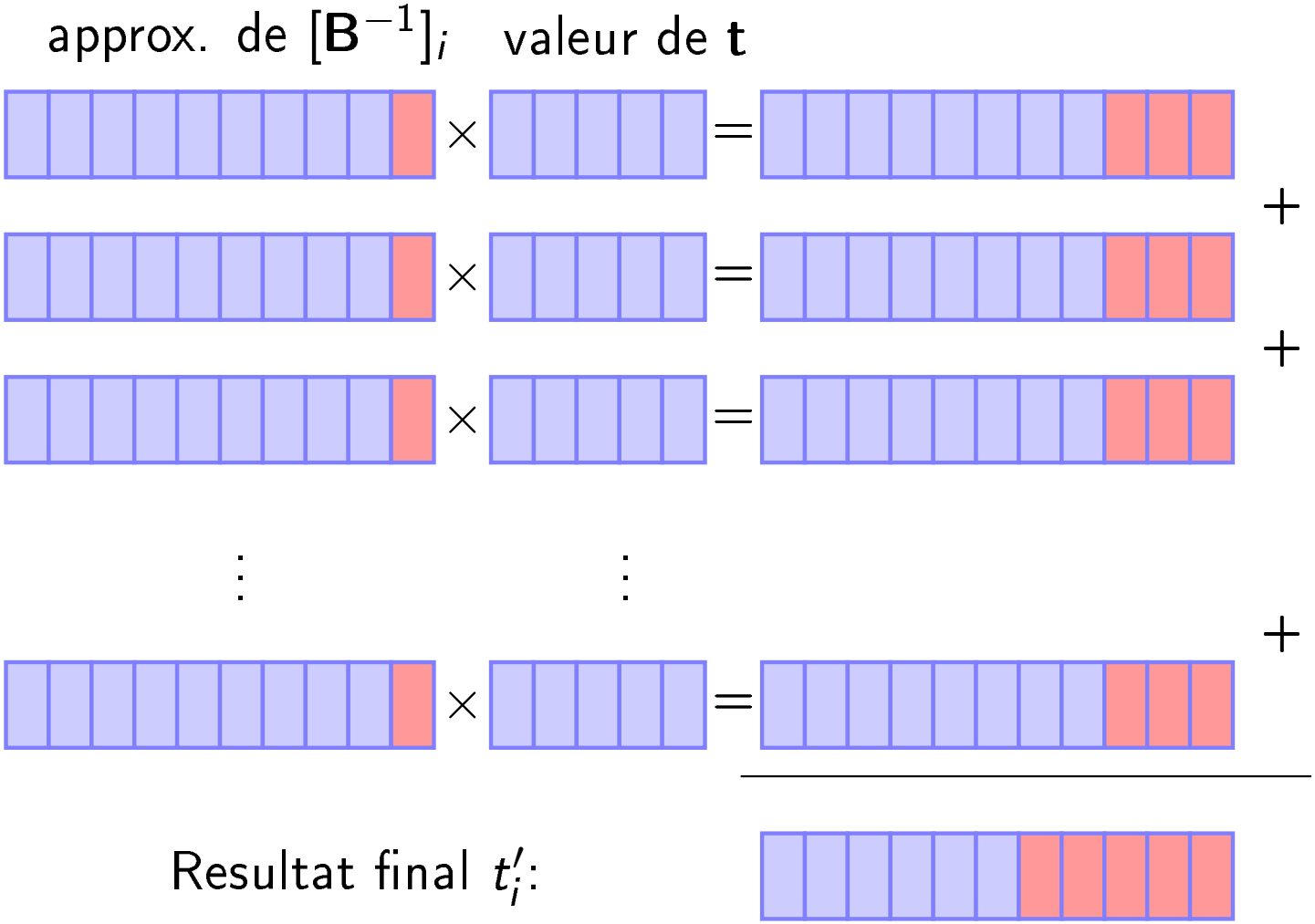
Definition (ϵ-correction de l'ÉGD)

Un algorithme d'ÉGD sera dit ϵ-correct si sa distribution de sortie $\mathcal{D}$ est à distance statistique de $D_{\mathbb{Z}^n,\sigma,\mathbf{t}'}$ inférieure à ϵ:

$$\mathcal{D} \approx_\epsilon D_{\mathbb{Z}^n,\sigma,\mathbf{t}'}.$$

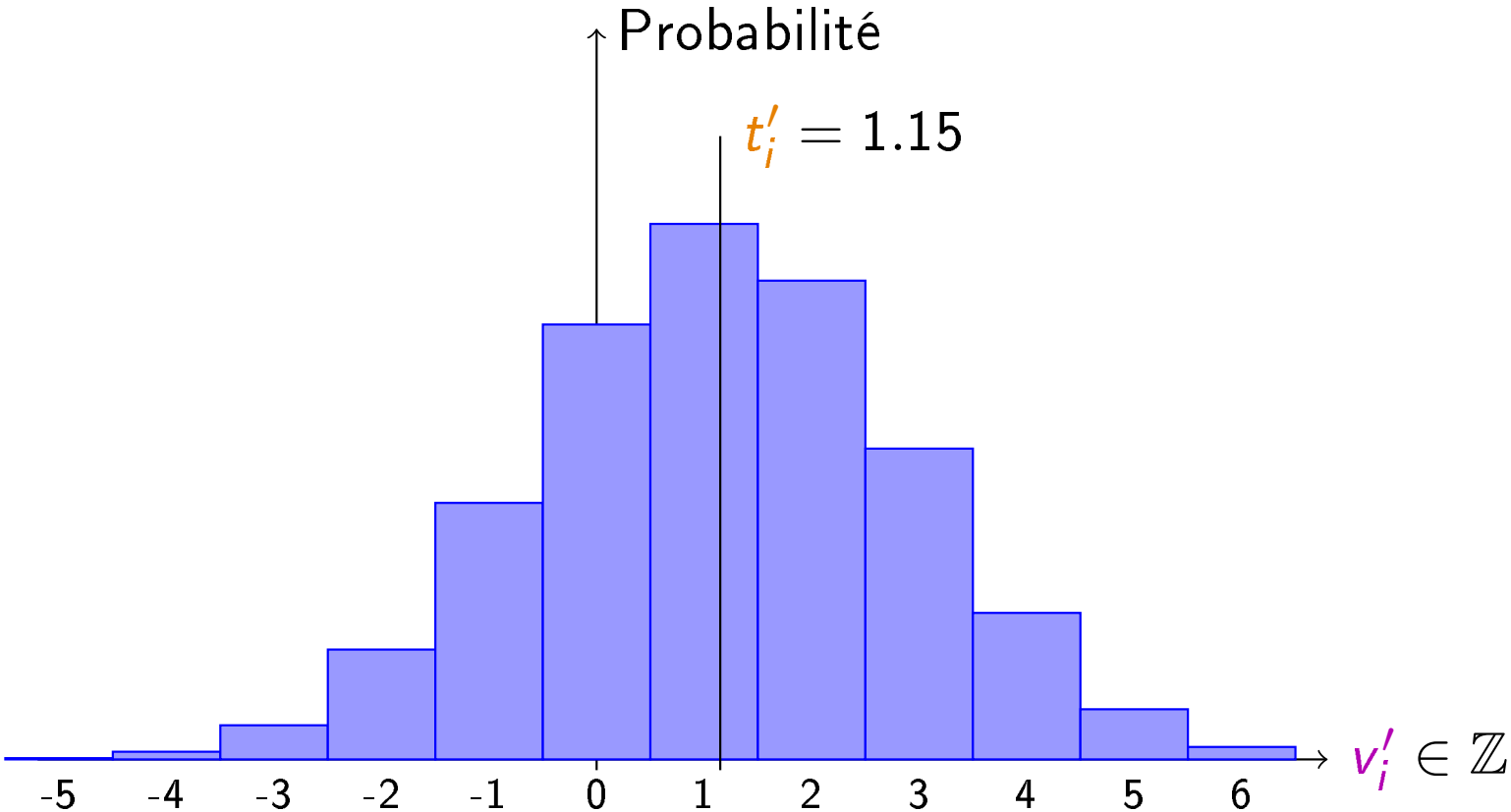
Pour les applications cryptographiques, on requiert l'ϵ-correction pour $\epsilon = 2^{-\Omega(n)}$.

Propagation d'erreur



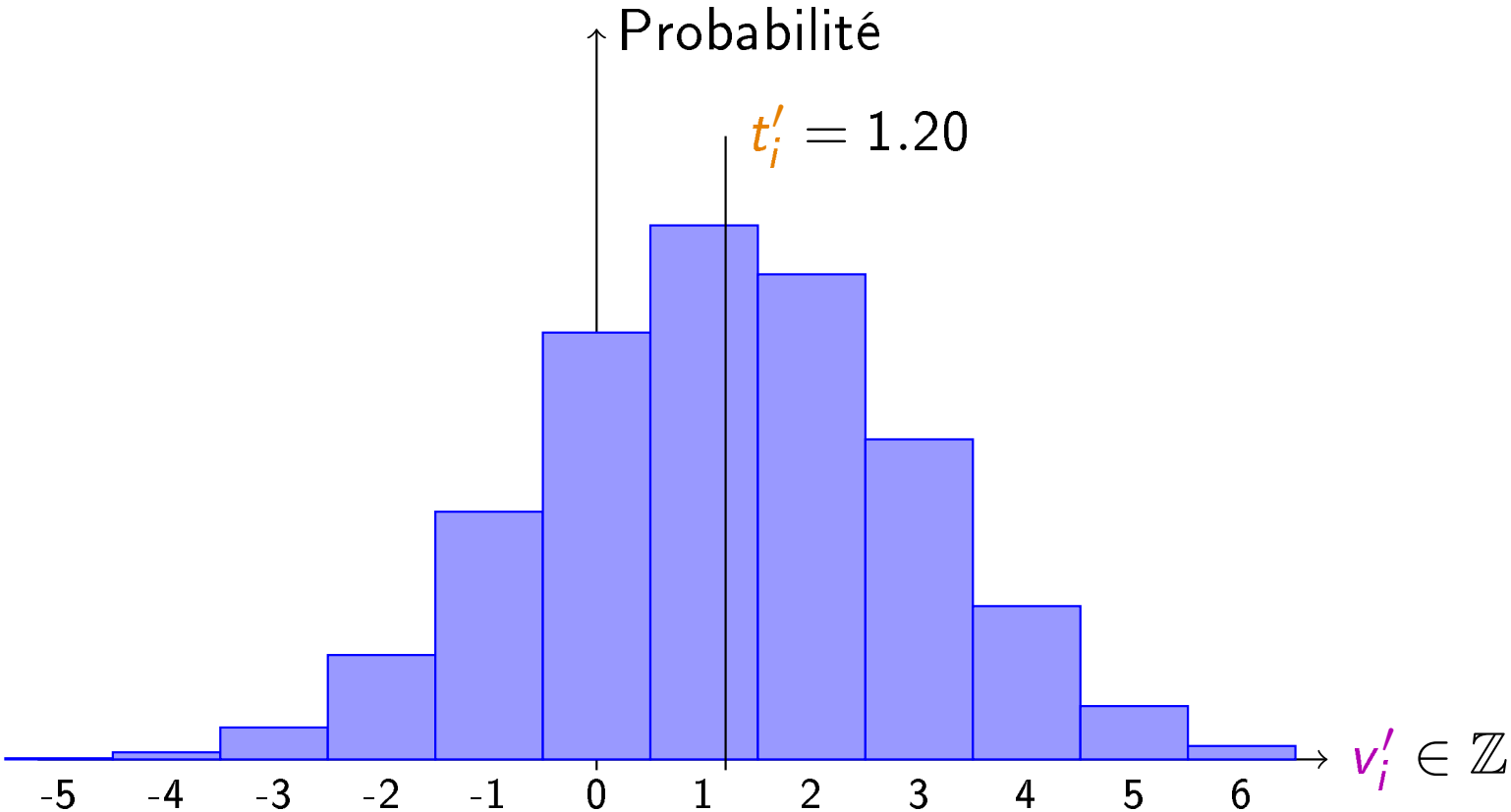
ÉGD en dimension 1: Le point clef

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \qquad \mathbf{v}' \leftarrow D_{\mathbb{Z}^n,\sigma,\mathbf{t}'}; \qquad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$



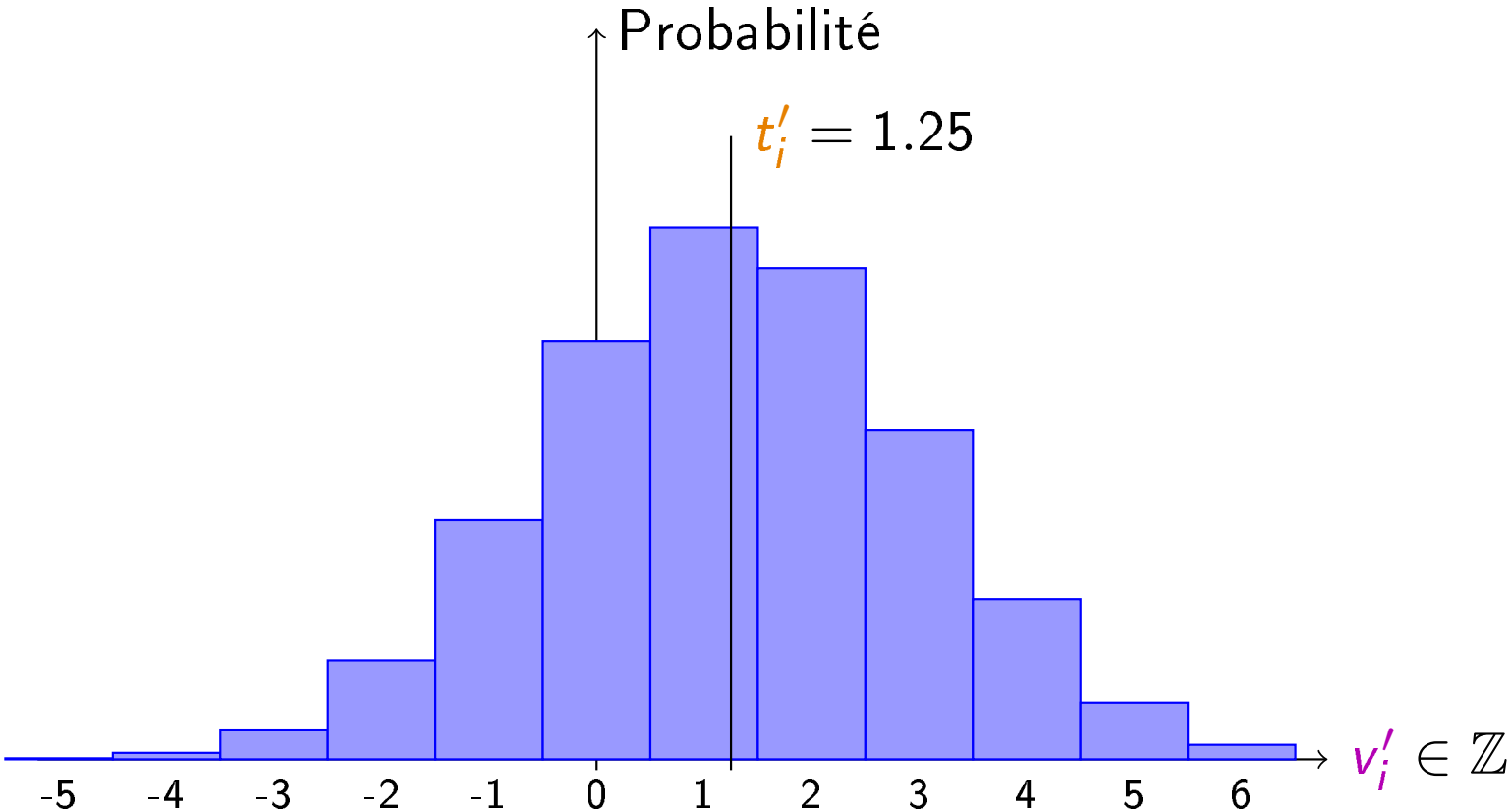
ÉGD en dimension 1: Le point clef

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \qquad \mathbf{v}' \leftarrow D_{\mathbb{Z}^n,\sigma,\mathbf{t}'}; \qquad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$



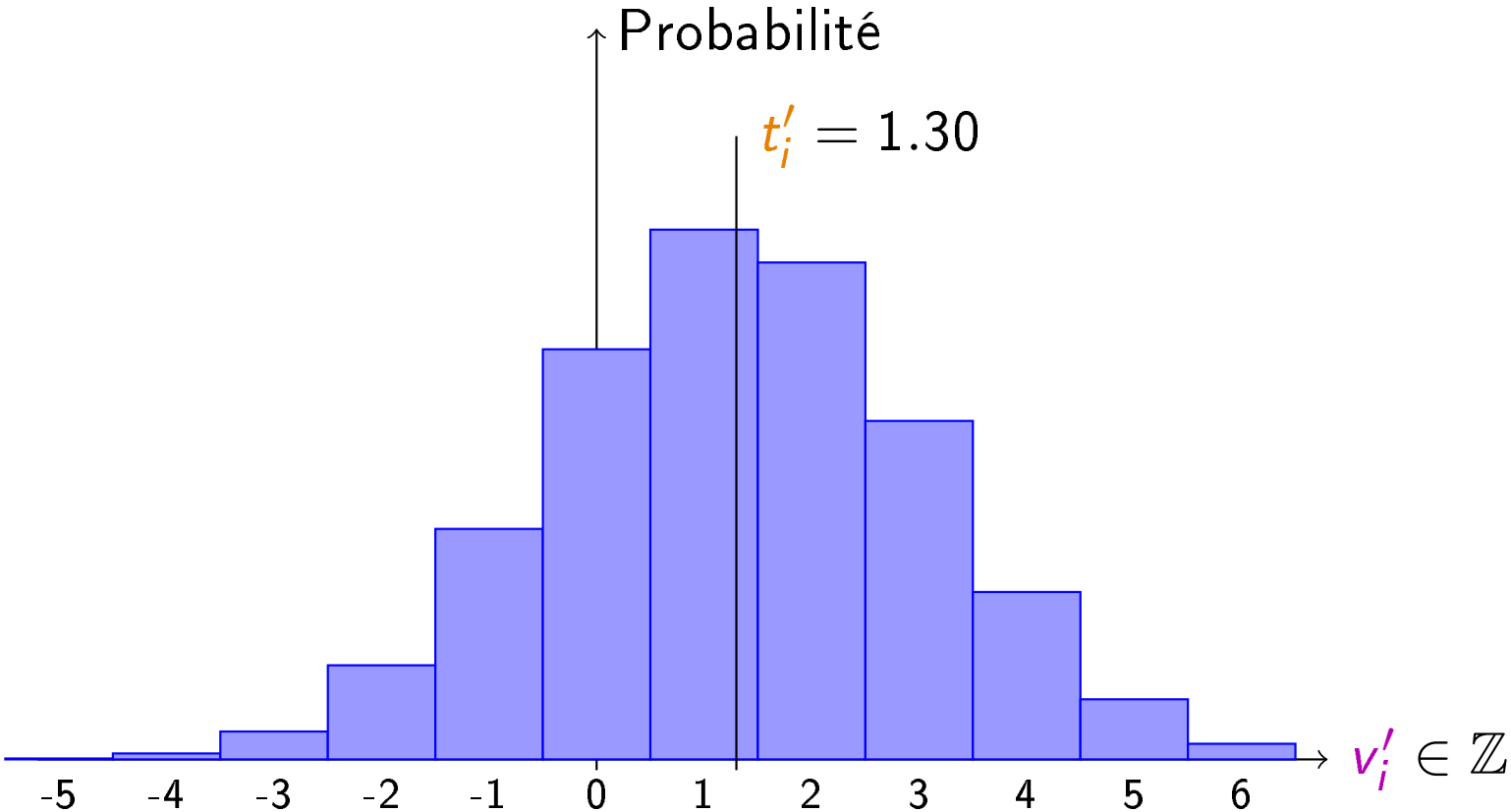
ÉGD en dimension 1: Le point clef

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \qquad \mathbf{v}' \leftarrow D_{\mathbb{Z}^n,\sigma,\mathbf{t}'}; \qquad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$



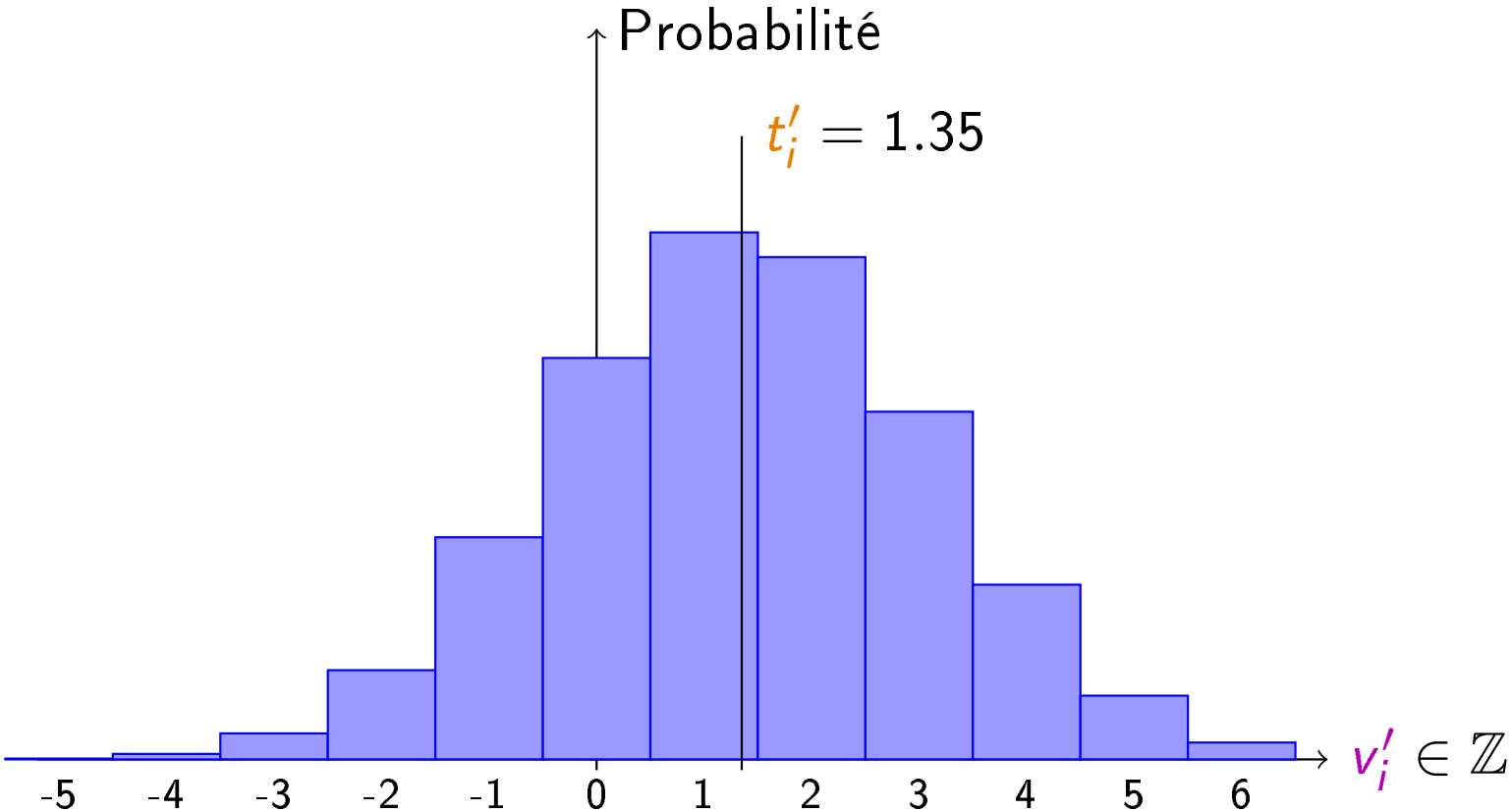
ÉGD en dimension 1: Le point clef

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \qquad \mathbf{v}' \leftarrow D_{\mathbb{Z}^n,\sigma,\mathbf{t}'}; \qquad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$



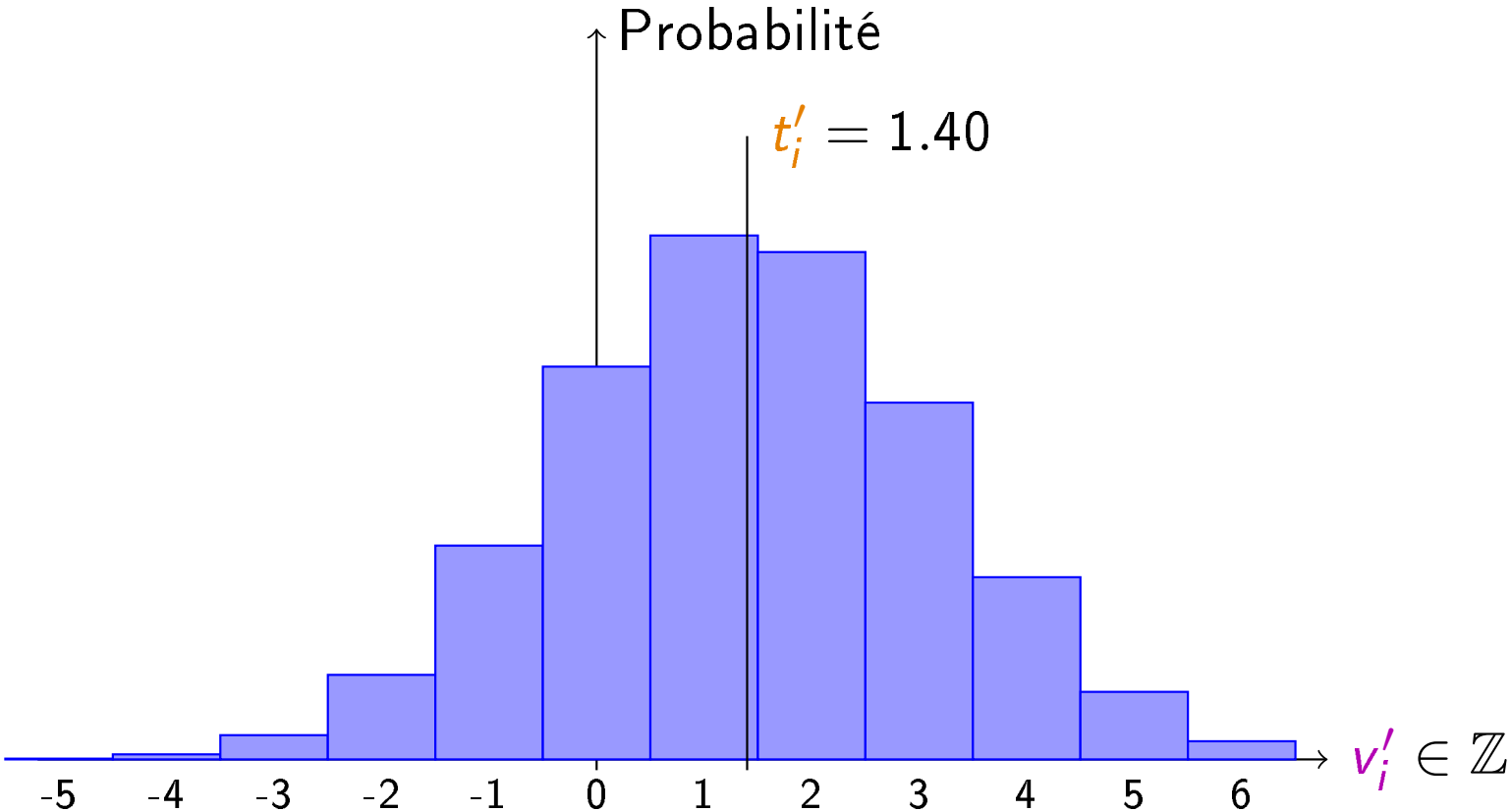
ÉGD en dimension 1: Le point clef

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \qquad \mathbf{v}' \leftarrow D_{\mathbb{Z}^n,\sigma,\mathbf{t}'}; \qquad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$



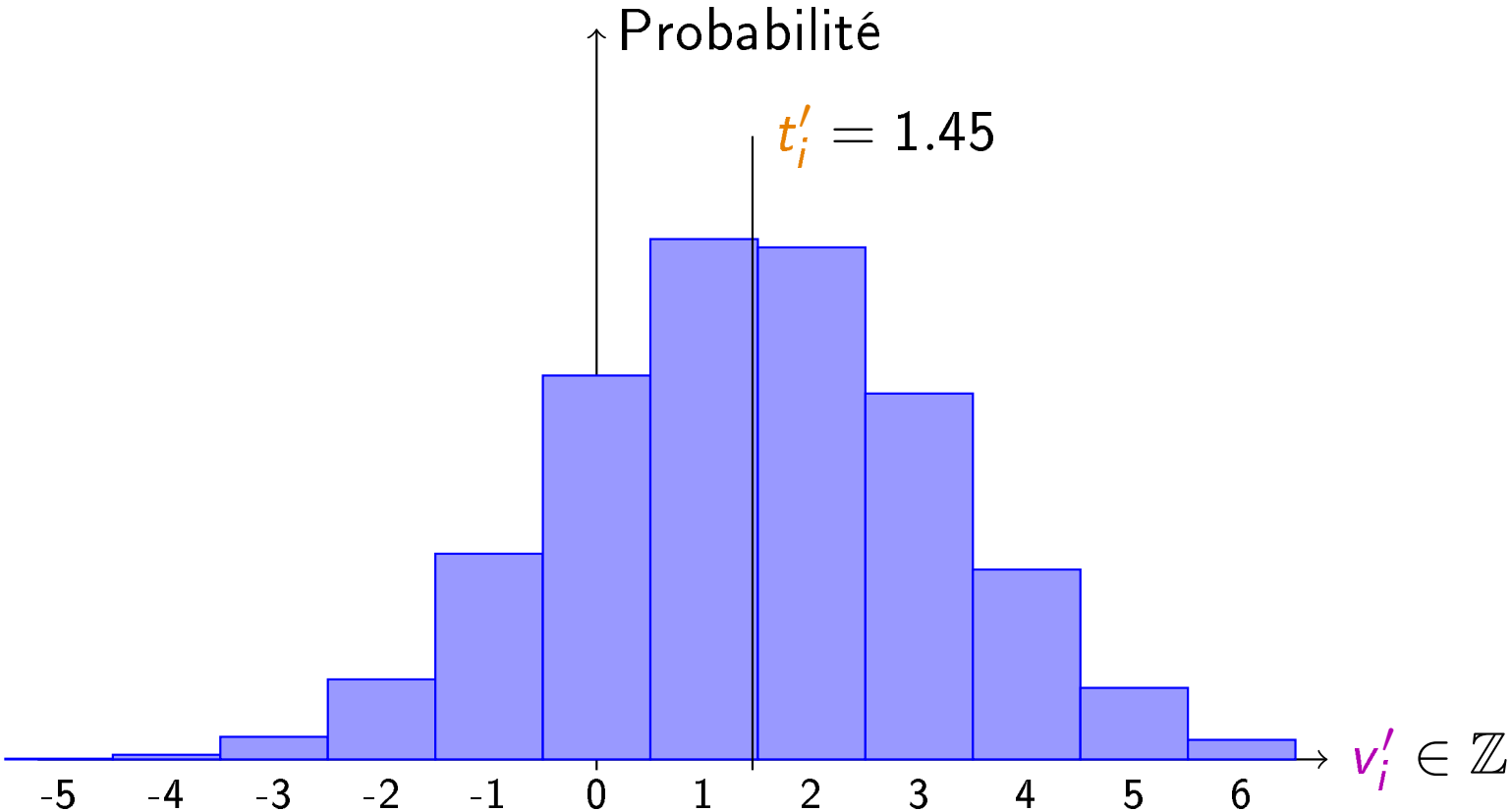
ÉGD en dimension 1: Le point clef

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \qquad \mathbf{v}' \leftarrow D_{\mathbb{Z}^n,\sigma,\mathbf{t}'}; \qquad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$



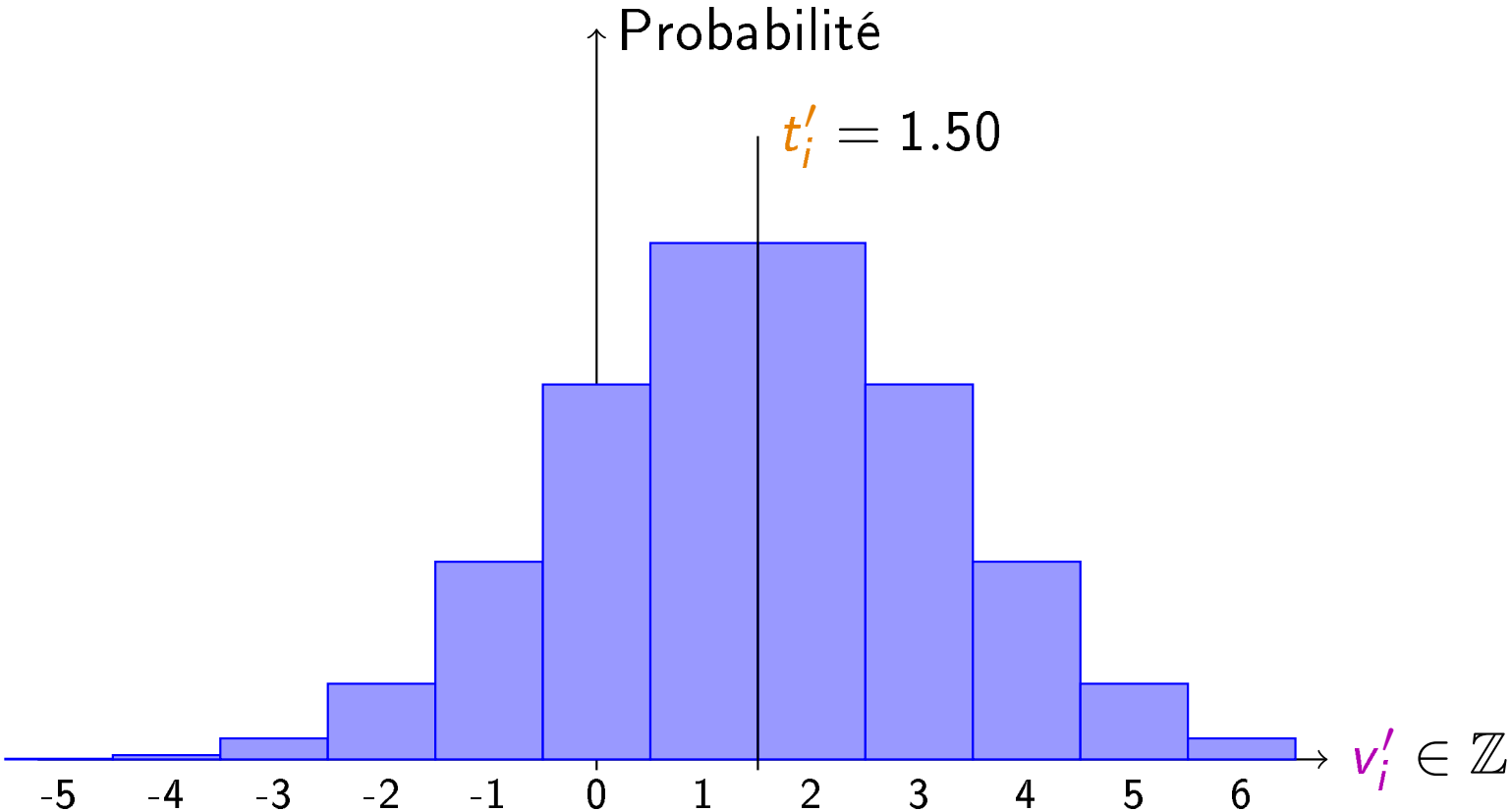
ÉGD en dimension 1: Le point clef

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \qquad \mathbf{v}' \leftarrow D_{\mathbb{Z}^n,\sigma,\mathbf{t}'}; \qquad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$



ÉGD en dimension 1: Le point clef

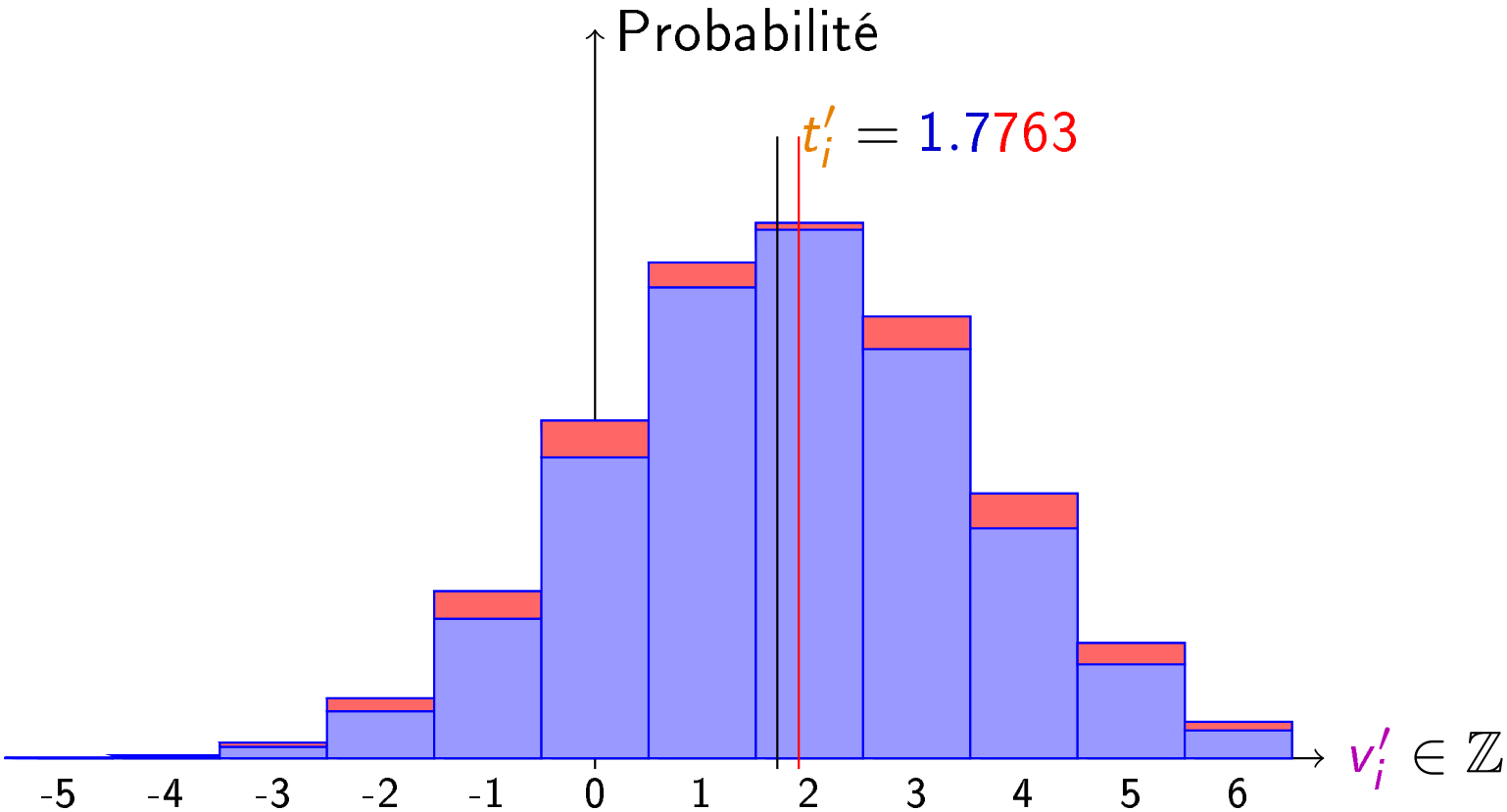
$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \qquad \mathbf{v}' \leftarrow D_{\mathbb{Z}^n,\sigma,\mathbf{t}'}; \qquad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$



ÉGD en dimension 1: Le point clef

Propagation de l'incertitude

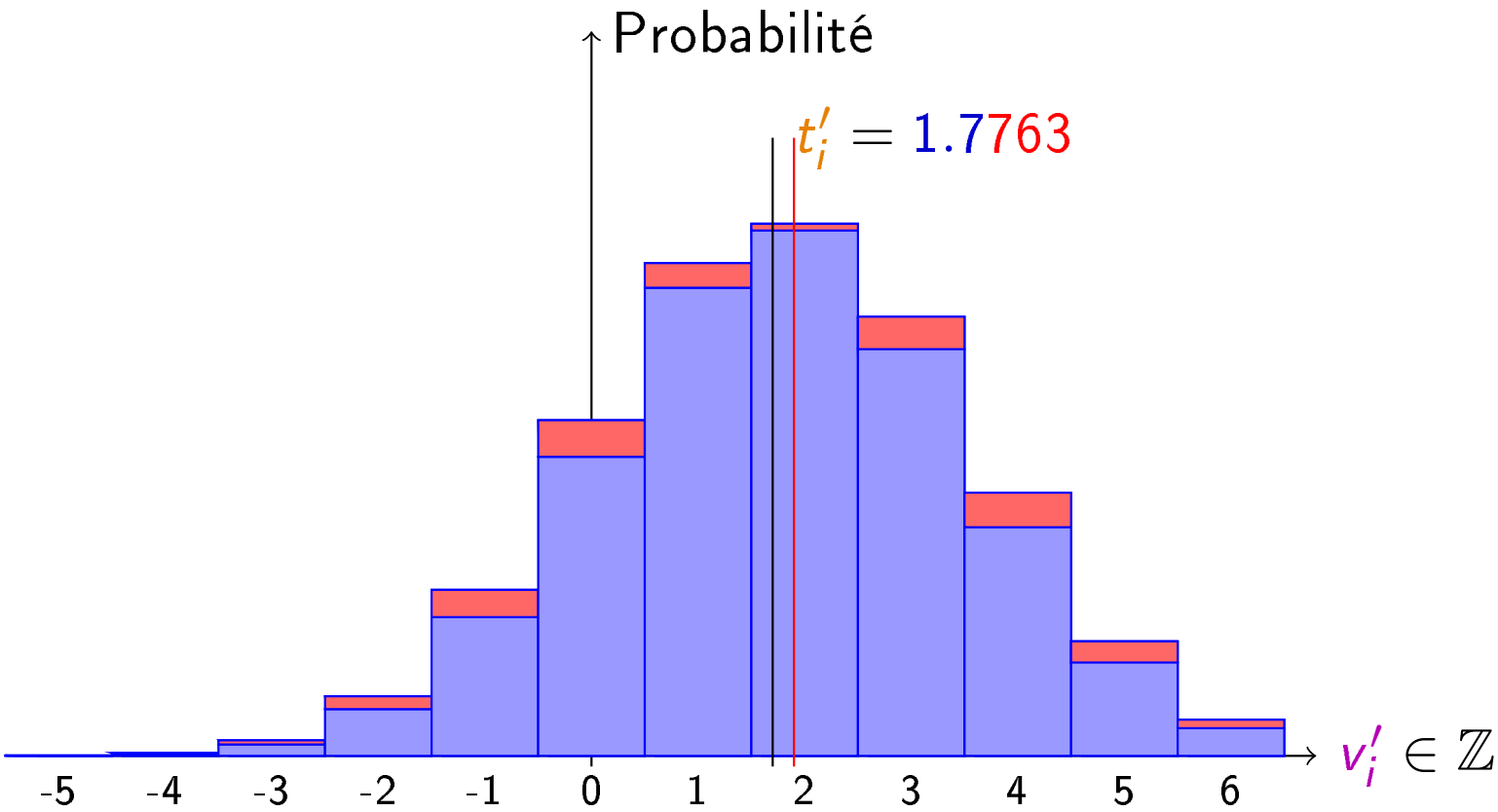
La précision de la distribution échantillonnée dépend de la précision de calcul du centre c.



ÉGD en dimension 1: Le point clef

Condition de correction

On souhaite rendre l'incertitude négligeable ($2^{-\Omega(n)}$).

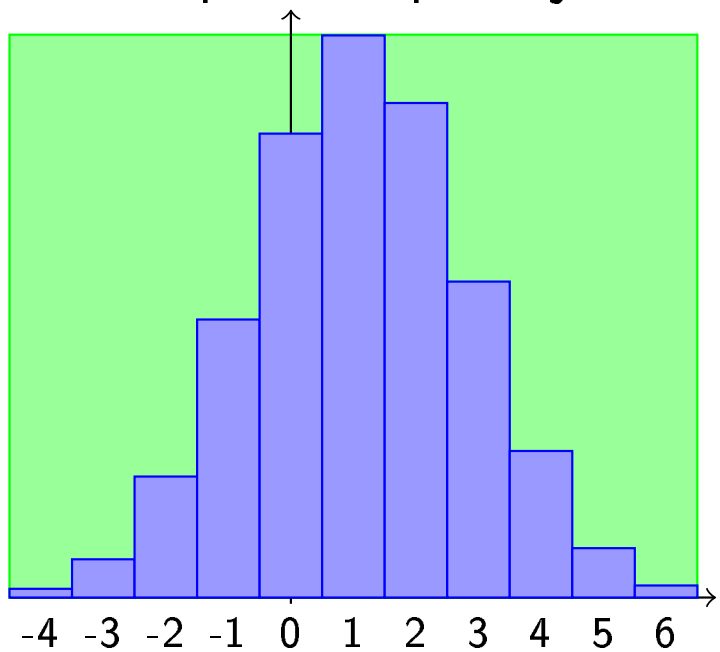


ÉGD en dimension 1: tirage avec rejet (méthode standard)

Pour échantillonner la gaussienne discrète on procède par rejet.

Tirage avec rejet:

- Échantillonner $(x, y) \in$ ■
- Si $(x, y) \in$ ■ retourner x
- Sinon, recommencer



Efficacité

L'algorithme effectue $O(n^2)$ sur $\mathbb{R}$. La précision requise est $\Omega(n)$.

Complexité totale: $O(n^3)$.

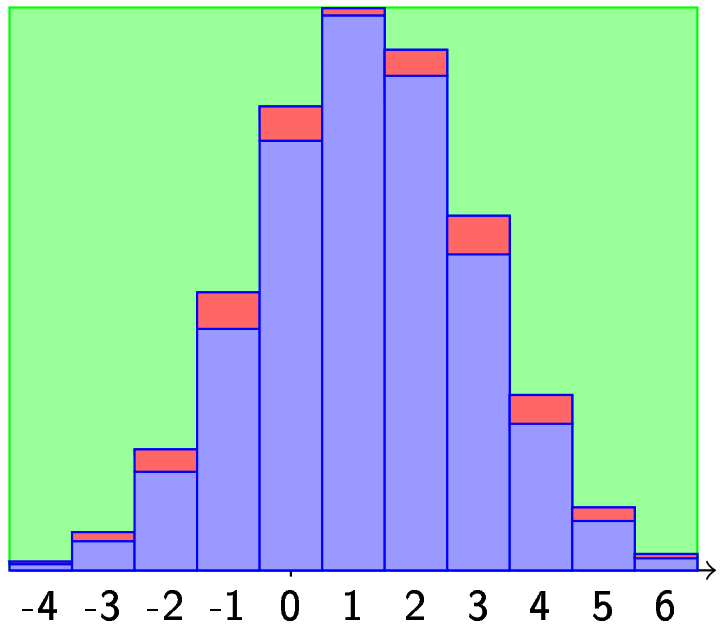
C'est bien plus cher que la promesse d'efficacité en (n^2) !
Peut-on améliorer cette complexité théorique ?

ÉGD en dimension 1: gérer l'incertitude (notre approche)

L'incertitude n'influe pas toujours sur le résultat final.
Gérer ces cas à part avec une **alerte**.

Tirage avec rejet et alerte:

- Échantillonner $(x, y) \in$ ■
- Si $(x, y) \in$ ■ **alerte**
- Si $(x, y) \in$ ■ renvoyer x
- Sinon, recommencer



Optimisation: algorithmes paresseux

Utiliser deux précisions: **haute prec.** m and **basse prec.** $m' < m$.

Haute prec. $\Rightarrow$ zone ■ négligeable: $2^{-\Theta(n)}$

Basse prec. $\Rightarrow$ zone ■ petite: $O(1/n^{\Theta(1)})$ (alertes rares)

Échantillonnage **paresseux** avec rejet:

- Calculer t'_i en basse précision
- Lancer le tirage avec rejet et alertes

Optimisation: algorithmes paresseux

Utiliser deux précisions: **haute prec.** m and **basse prec.** $m' < m$.

Haute prec. $\Rightarrow$ zone ■ négligeable: $2^{-\Theta(n)}$

Basse prec. $\Rightarrow$ zone ■ petite: $O(1/n^{\Theta(1)})$ (alertes rares)

Échantillonnage **paresseux** avec rejet:

- Calculer t'_i en basse précision
- Lancer le tirage avec rejet et alertes
- Si l'alerte est déclenchée

Optimisation: algorithmes paresseux

Utiliser deux précisions: **haute prec.** m and **basse prec.** $m' < m$.

Haute prec. $\Rightarrow$ zone ■ négligeable: $2^{-\Theta(n)}$

Basse prec. $\Rightarrow$ zone ■ petite: $O(1/n^{\Theta(1)})$ (alertes rares)

Échantillonnage **paresseux** avec rejet:

- Calculer t'_i en basse précision
- Lancer le tirage avec rejet et alertes
- Si l'alerte est déclenchée

  Recalculer t'_i en haute précision

Optimisation: algorithmes paresseux

Utiliser deux précisions: **haute prec.** m and **basse prec.** $m' < m$.

Haute prec. $\Rightarrow$ zone ■ négligeable: $2^{-\Theta(n)}$

Basse prec. $\Rightarrow$ zone ■ petite: $O(1/n^{\Theta(1)})$ (alertes rares)

Échantillonnage **paresseux** avec rejet:

- Calculer t'_i en basse précision
- Lancer le tirage avec rejet et alertes
- Si l'alerte est déclenchée

  Recalculer t'_i en haute précision

  Reprendre le tirage, avec zone d'incertitude ■ négligeable

Efficacité Théorique

Coût asymptotique

- Échantillonnage naïf: $\tilde{O}(n^3)$ ($\tilde{O}(n^2)$ sur réseaux cycliques)
- Échantillonnage paresseux: $\tilde{O}(n^2)$ ($\tilde{O}(n)$ sur réseaux cycliques)

Complexité théorique quasi-optimale ! Mais en pratique ?

Efficacité Théorique

Coût asymptotique

- Échantillonnage naïf: $\tilde{O}(n^3)$ ($\tilde{O}(n^2)$ sur réseaux cycliques)
- Échantillonnage paresseux: $\tilde{O}(n^2)$ ($\tilde{O}(n)$ sur réseaux cycliques)

Complexité théorique quasi-optimale ! Mais en pratique ?

Il faut tout de même de l'arithmétique haute précision *de temps en temps*. Ce n'est pas adaptés aux petites architectures.

Outline

Échantillonnage sans arithmétique flottante ?

L'arithmétique flottante, même à basse précision est un problème sur **petite architecture** (ex. carte-à-puce).

Question: Est-elle vraiment nécessaire ?

Échantillonnage sans arithmétique flottante ?

L'arithmétique flottante, même à basse précision est un problème sur **petite architecture** (ex. carte-à-puce).

Question: Est-elle vraiment nécessaire ?

- **Non**! au moins pour des cas simples:
  gaussienne centrée en $\mathbf{0}$ sur $\mathbb{Z}^n$ (application : BLISS)

Échantillonnage sans arithmétique flottante ?

L'arithmétique flottante, même à basse précision est un problème sur **petite architecture** (ex. carte-à-puce).

Question: Est-elle vraiment nécessaire ?

- **Non**! au moins pour des cas simples:
  gaussienne centrée en $\mathbf{0}$ sur $\mathbb{Z}^n$ (application : BLISS)
- et peux-être plus généralement [Thèse]

Méthode 1: Table Cumulative

Technique générique pour un distribution sur $0, \ldots, B$. Soit p_i la probabilité cumulative $\mathbb{P}_{x \leftarrow D}(x \leqslant i)$.

- Precalculer un table $[0, p_0, p_1, \ldots p_{B-1}, 1]$
- Tirer $u \leftarrow [0, 1]$ uniformement
- Trouver i tel que $p_i < u \leqslant p_{i+1}$
- Retourner i

Avantage: Rapide $O(\log B)$.
Contrainte: Requiert beaucoup de mémoire.

Méthode 2: Algèbre sur les distributions de Bernoulli

On cherche a échantillonner $\mathcal{B}_{\exp(-x)}$ efficacement

Idée: pré-calculer quelques biais c et combiner les variables de Bernoulli $\mathcal{B}_c$:

Note: Ayant pré-calculé c, tirer une Bernoulli $\mathcal{B}_c$ est trivial:

- Tirer $u \leftarrow [0, 1]$ uniformément
- Renvoyer 1 si $u < c$, 0 sinon

Méthode 2: Algèbre sur les distributions de Bernoulli

On cherche a échantillonner $\mathcal{B}_{\exp(-x)}$ efficacement

Idée: pré-calculer quelques biais c et combiner les variables de Bernoulli $\mathcal{B}_c$:

- $\mathcal{B}_a \wedge \mathcal{B}_b = \mathcal{B}_{ab}$
- $\mathcal{B}_a \vee \mathcal{B}_b = \mathcal{B}_{a+b-ab}$
- $\mathcal{B}_a \oplus \mathcal{B}_b = \mathcal{B}_{a+b-2ab}$

Note: Ayant pré-calculé c, tirer une Bernoulli $\mathcal{B}_c$ est trivial:

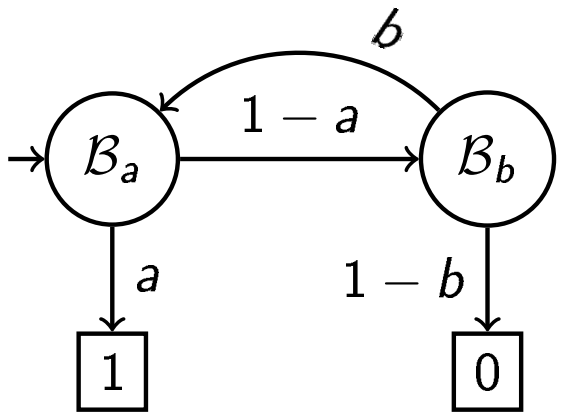
- Tirer $u \leftarrow [0, 1]$ uniformément
- Renvoyer 1 si $u < c$, 0 sinon

Méthode 2: Algèbre sur les distributions de Bernoulli

On cherche a échantillonner $\mathcal{B}_{\exp(-x)}$ efficacement

Idée: pré-calculer quelques biais c et combiner les variables de Bernoulli $\mathcal{B}_c$:

- $\mathcal{B}_a \wedge \mathcal{B}_b = \mathcal{B}_{ab}$
- $\mathcal{B}_a \vee \mathcal{B}_b = \mathcal{B}_{a+b-ab}$
- $\mathcal{B}_a \oplus \mathcal{B}_b = \mathcal{B}_{a+b-2ab}$
- $\mathcal{B}_a \oslash \mathcal{B}_b = \mathcal{B}_{a/(1-(1-a)b)}$



Calcul de $\mathcal{B}_a \oslash \mathcal{B}_b$

Note: Ayant pré-calculé c, tirer une Bernoulli $\mathcal{B}_c$ est trivial:

- Tirer $u \leftarrow [0,1]$ uniformément
- Renvoyer 1 si $u < c$, 0 sinon

Tirer une Variable de Bernoulli de Biais Exponentiel

Idée: utiliser la décomposition binaire de l'exposant:

$$x = x_0 + 2x_1 + 2^2 x_2 + \cdots + 2^\ell x_\ell$$

Ainsi,

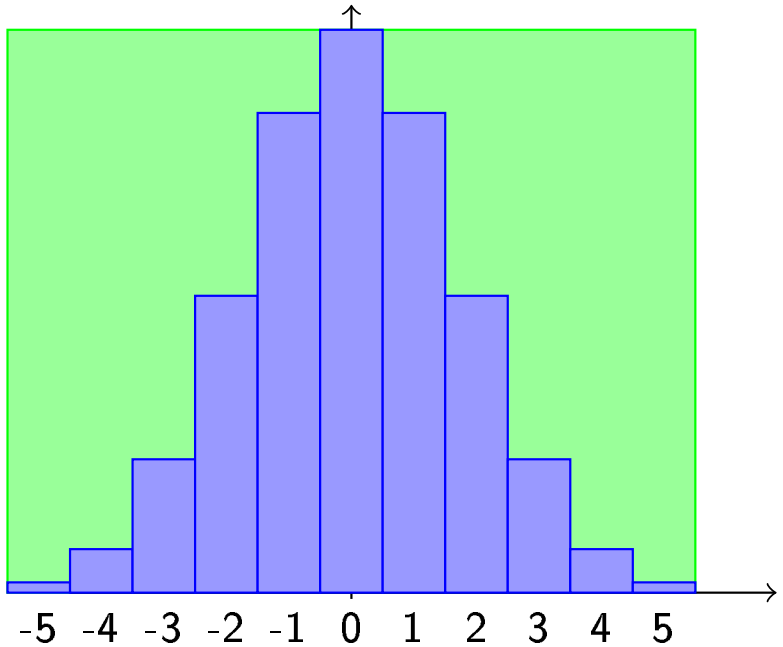
$$\begin{aligned}
\mathcal{B}_{\exp(-x)} &= \mathcal{B}_{\exp(-x_0 - 2x_1 - 2^2 x_2 - \cdots - 2^\ell x_\ell)} \\
&= \mathcal{B}_{\exp(x_0)\cdot\exp(-2x_1)\cdot\exp(-2^2 x_2)\cdot\cdots\cdot\exp(-2^\ell x_\ell)} \\
&= \mathcal{B}_{\exp(-x_0)} \wedge \mathcal{B}_{\exp(-2x_1)} \wedge \mathcal{B}_{\exp(2^2 x_2)} \wedge \cdots \wedge \mathcal{B}_{\exp(-2^\ell x_\ell)}
\end{aligned}$$

Ainsi, il suffit de pré-calculer $\ell = \log B$ biais $c_i = \exp(-2^i)$.

Méthode 2 + sampling avec rejet

Tirage avec rejet:

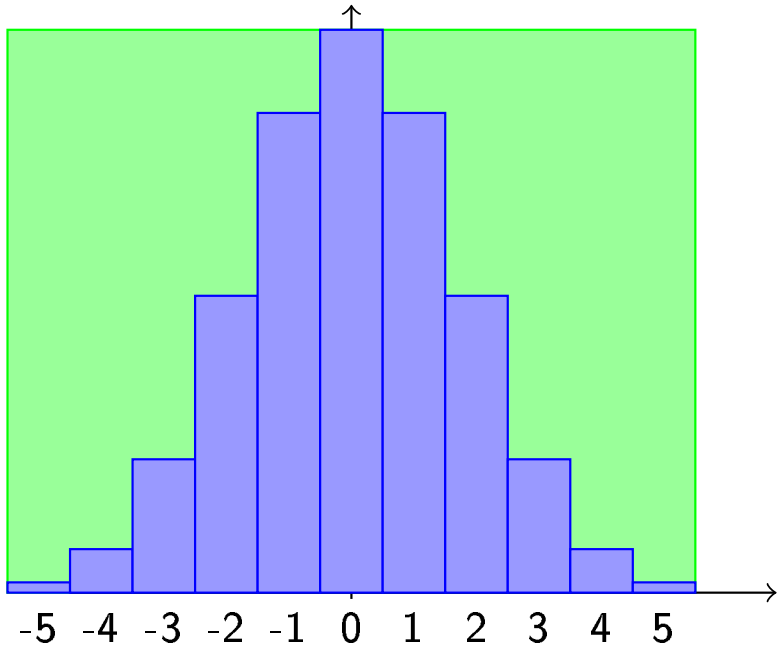
- Échantillonner $x \leftarrow \{-B \dots B\}$ (unif.)
- Tirer $b \leftarrow \mathcal{B}_{\exp(-x^2/\sigma^2)}$
- Si $b = 1$, retourner x
- Sinon, recommencer



Méthode 2 + sampling avec rejet

Tirage avec rejet:

- Échantillonner $x \leftarrow \{-B \dots B\}$ (unif.)
- Tirer $b \leftarrow \mathcal{B}_{\exp(-x^2/\sigma^2)}$
- Si $b = 1$, retourner x
- Sinon, recommencer



Défaut: il faut recommencer de nombreuses fois avant d'obtenir un sample.
Solution: truquer la distribution initiale de x (technique du Ziggourat).

Méthode 3: Série Formelle et Combinatoire

Technique classique pour tirer un distrib. exponentielle continue:

1. $\ell \leftarrow 1$
2. $x \leftarrow [0,1]$, uniforme
3. Tirer $u_1, u_2, \cdots \leftarrow [0,1]$, et déterminer le plus grand n tel que

$$x > u_1 > u_2 > \cdots > u_n$$

4. Si n est impair, incrémenter $\ell \leftarrow \ell + 1$ et retourner à 2.
5. Retourner $\ell + x$

Méthode 3: Série Formelle et Combinatoire

3. Tirer $u_1, u_2, \cdots \leftarrow [0,1]$, et determiner le plus grand n tel que

$$x > u_1 > u_2 > \cdots > u_n$$

4. Si n est impair...

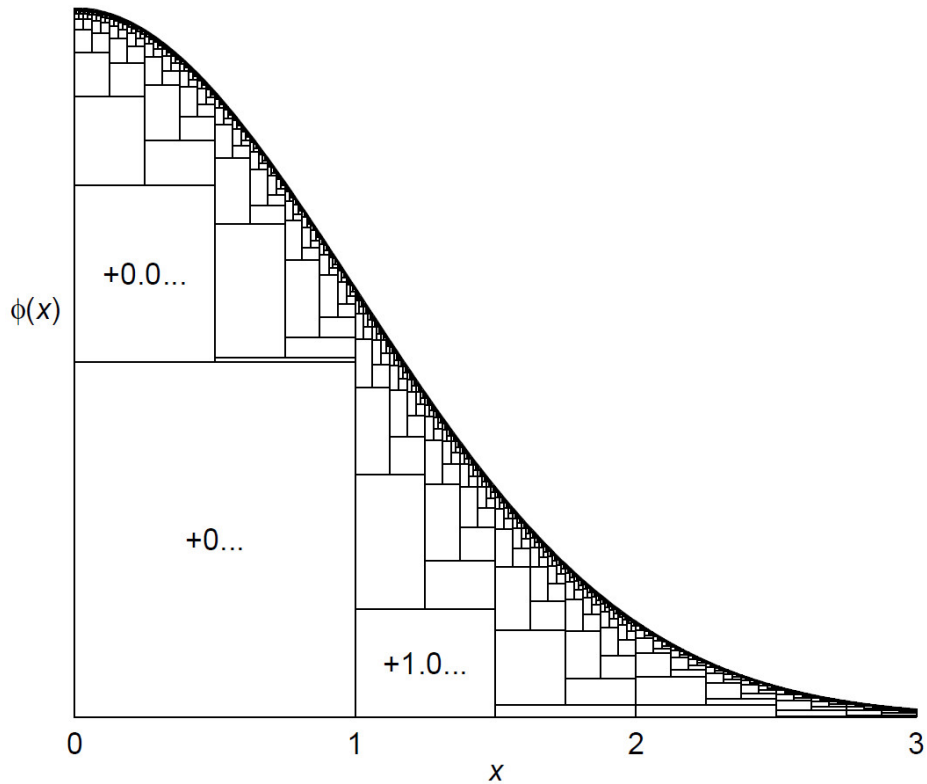
- La probabilité que $u_i < x$ pour tout $i \leqslant n$ est x^n
- La probabilité que $u_1 < u_2 < \cdots < u_n$ est $1/n!$

Ainsi, la probabilité que n soit pair est exactement:

$$1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \cdots = e^{-x}.$$

Méthode 3: Série Formelle et Combinatoire

Cette technique a été adapté pour les Gaussiennes Continues et Discrète par Karney [Kar13].



Avantage: Seuls les premiers bits coutent chers, les suivant sont uniformes.

Quelle Méthode Choisir ?

Quatre axes de mesure:

- Complexité totale
- Taille du pré-calcul
- Taille du circuit
- Consommation d'aléas

Implémentations actuelles:

- Software: méthode 1 (Table cumulative)
- Hardware: méthode 2 (Algèbre de Bernoulli)

La méthode 3 n'as pas encore été évaluée en pratique !

Outline

Conclusion

- Les réseaux euclidiens posent des problèmes d'arithmétique des ordinateurs inédit en cryptographie
- Quelques techniques ont été développées par les cryptographes

Probablement non optimales...

- Les contraintes crypto sont maintenant bien comprises:

Conclusion

- Les réseaux euclidiens posent des problèmes d'arithmétique des ordinateurs inédit en cryptographie
- Quelques techniques ont été développées par les cryptographes

Probablement non optimales...

- Les contraintes crypto sont maintenant bien comprises:

A vous de jouer !

Conclusion

- Les réseaux euclidiens posent des problèmes d'arithmétique des ordinateurs inédit en cryptographie
- Quelques techniques ont été développées par les cryptographes

Probablement non optimales...

- Les contraintes crypto sont maintenant bien comprises:

A vous de jouer !

Merci de votre attention.

Questions ?