An Algorithmic Reduction Theory for Binary Codes: LLL and More

Léo Ducas (CWI), Thomas Debris-Alazard (Inria),
Wessel van Woerden (CWI).

Centrum Wiskunde & Informatica

This work

Propose *analogues* from lattices to binary codes (Defs, Algs, Bounds).

Speed-up cryptanalytic algorithms for code-based cryptography. ?

Overview

This work

Propose *analogues* from lattices to binary codes (Defs, Algs, Bounds).

Speed-up cryptanalytic algorithms for code-based cryptography. ?

This talk

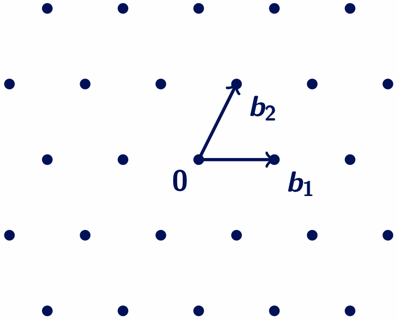
- Recall the LLL algorithm for lattices.
- Adapt it to codes.

What notion of orthogonality for binary codewords?

Lattice

$\mathcal{L}(B) := \{\sum_i x_i b_i : x \in \mathbb{Z}^k\} \subset \mathbb{R}^n$

Euclidean

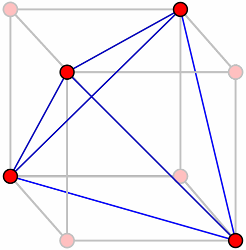


$\mathcal{L} = b_1\mathbb{Z} + b_2\mathbb{Z}$

Binary Code

$\mathcal{C}(B) := \{\sum_i x_i b_i : x \in \mathbb{F}_2^k\} \subset \mathbb{F}_2^n$

Hamming

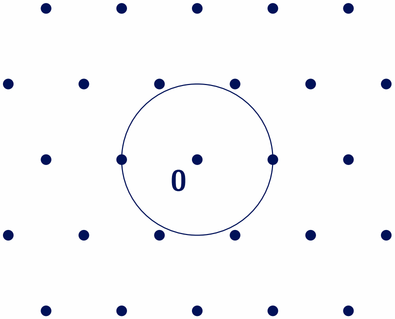


$\mathcal{C} = \{000, 011, 101, 110\}$

Lattice

$$\lambda_1(\mathcal{L}) := \min_{x \in \mathcal{L} \setminus \{0\}} \|x\|_2$$

Euclidean

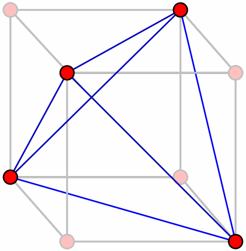


$$\mathcal{L} = b_1\mathbb{Z} + b_2\mathbb{Z}$$

Binary Code

$$d_{\min}(\mathcal{C}) := \min_{x \in \mathcal{C} \setminus \{0\}} |x|$$

Hamming

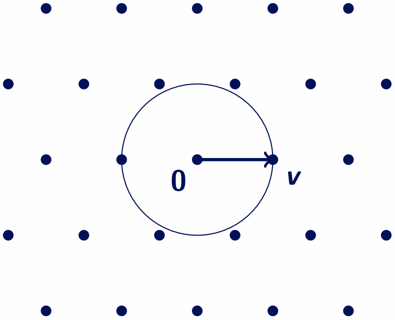


$$\mathcal{C} = \{000, 011, 101, 110\}$$

Hard Problems

Lattice

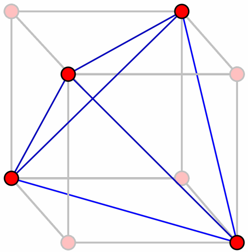
Find a *short* nonzero vector $\boldsymbol{v} \in \mathcal{L}(\boldsymbol{B})$.



$$\mathcal{L} = \boldsymbol{b}_1\mathbb{Z} + \boldsymbol{b}_2\mathbb{Z}$$

Binary Code

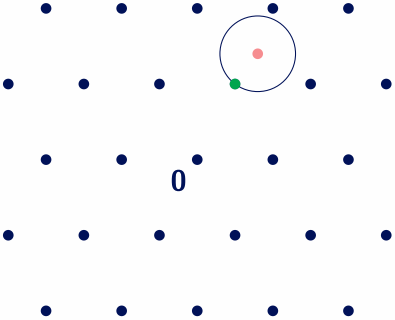
Find a *short* nonzero codeword $\boldsymbol{v} \in \mathcal{C}(\boldsymbol{B})$.



$$\mathcal{C} = \{000, 011, 101, 110\}$$

Lattice

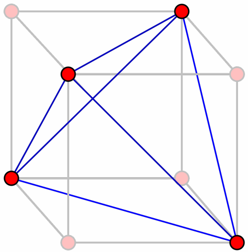
Given a target $\boldsymbol{t} \in \mathbb{R}^n$ find a *close* vector $\boldsymbol{v} \in \mathcal{L}(\boldsymbol{B})$.



$$\mathcal{L} = \boldsymbol{b}_1\mathbb{Z} + \boldsymbol{b}_2\mathbb{Z}$$

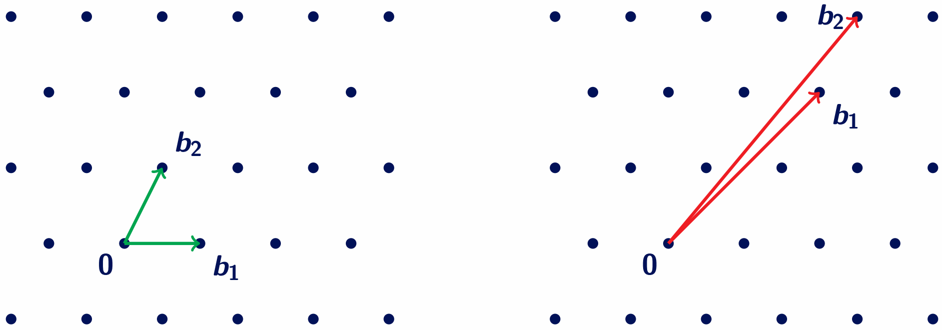
Binary Code

Given a target $\boldsymbol{t} \in \mathbb{F}_2^n$ find a *close* codeword $\boldsymbol{c} \in \mathcal{C}(\boldsymbol{B})$.



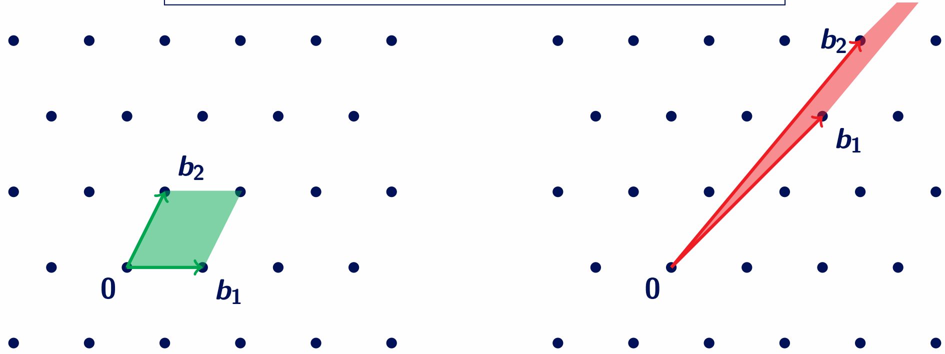
$$\mathcal{C} = \{000, 011, 101, 110\}$$

$B' := B \cdot U$ is a basis of $\mathcal{L}(B)$ if and only if $U \in \mathrm{GL}_d(\mathbb{Z})$.



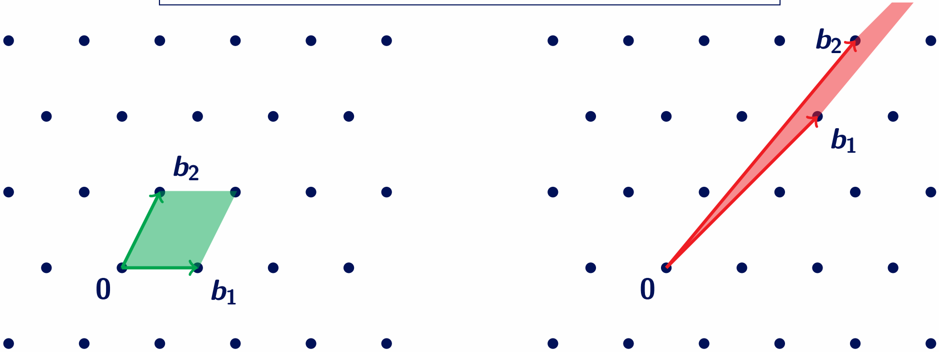
$B' := B \cdot U$ is a basis of $\mathcal{L}(B)$ if and only if $U \in \mathrm{GL}_d(\mathbb{Z})$.

Invariant: $\det(\mathcal{L}) := \mathrm{Vol}(\mathbb{R}^n/\mathcal{L}) = \det(B)$.



$\boldsymbol{B}' := \boldsymbol{B} \cdot \boldsymbol{U}$ is a basis of $\mathcal{L}(\boldsymbol{B})$ if and only if $\boldsymbol{U} \in \mathrm{GL}_d(\mathbb{Z})$.

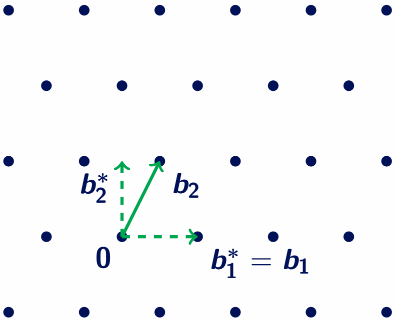
Invariant: $\det(\mathcal{L}) := \mathrm{Vol}(\mathbb{R}^n/\mathcal{L}) = \det(\boldsymbol{B})$.

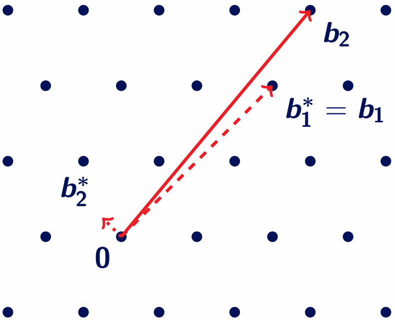


Find a ‘*good*’ lattice basis of $\mathcal{L}$.

Short and somewhat orthogonal.

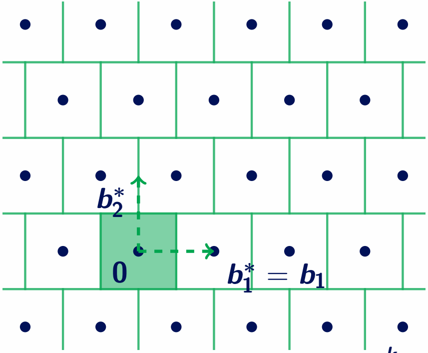
$$b_i^* := \underbrace{\pi_{(b_1,\ldots,b_{i-1})^\perp}}_{\pi_i}(b_i)$$

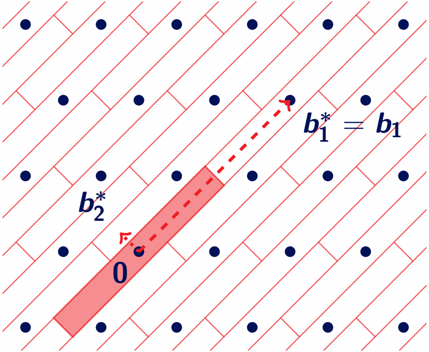




$$b_i^* := \underbrace{\pi_{(b_1,\ldots,b_{i-1})^\perp}}_{\pi_i}(b_i)$$

$$\prod_{i=1}^{k} \left\| b_i^* \right\| = \det(\mathcal{L})$$



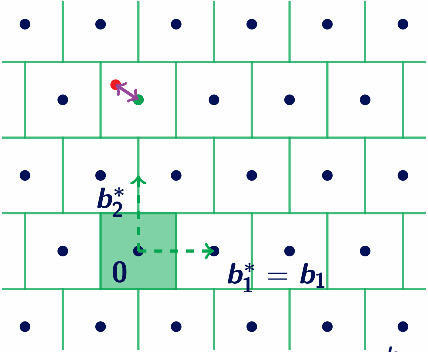


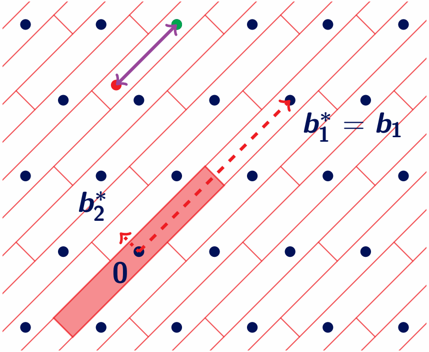
Fundamental Area: $\mathcal{F}_{B^*} := \prod_{i=1}^{k} \left[-\frac{1}{2} b_i^*, \frac{1}{2} b_i^* \right)$

Gram-Schmidt Orthogonalisation

$$b_i^* := \underbrace{\pi_{(b_1,\dots,b_{i-1})^\perp}}_{\pi_i}(b_i)$$

$$\prod_{i=1}^{k} \left\| b_i^* \right\| = \det(\mathcal{L})$$





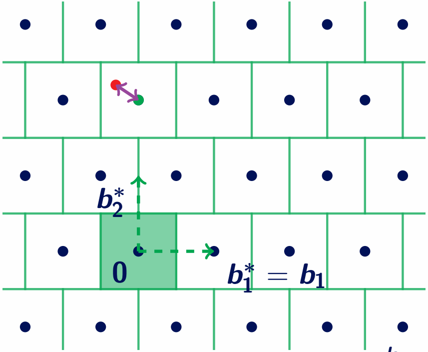
Fundamental Area: $\mathcal{F}_{B^*} := \prod_{i=1}^{k} \left[-\frac{1}{2} b_i^*, \frac{1}{2} b_i^*\right)$

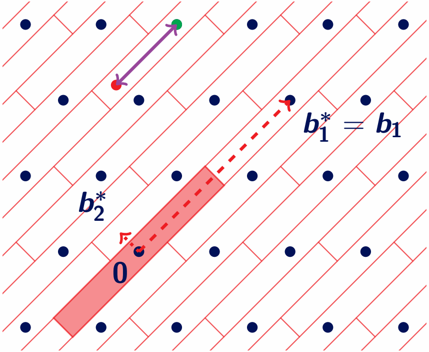
Babai Decoding: $e := t - v \in \mathcal{F}_{B^*}$

Gram-Schmidt Orthogonalisation

$$b_i^* := \underbrace{\pi_{(b_1,\ldots,b_{i-1})^\perp}}_{\pi_i}(b_i)$$

$$\prod_{i=1}^{k} \|b_i^*\| = \det(\mathcal{L})$$



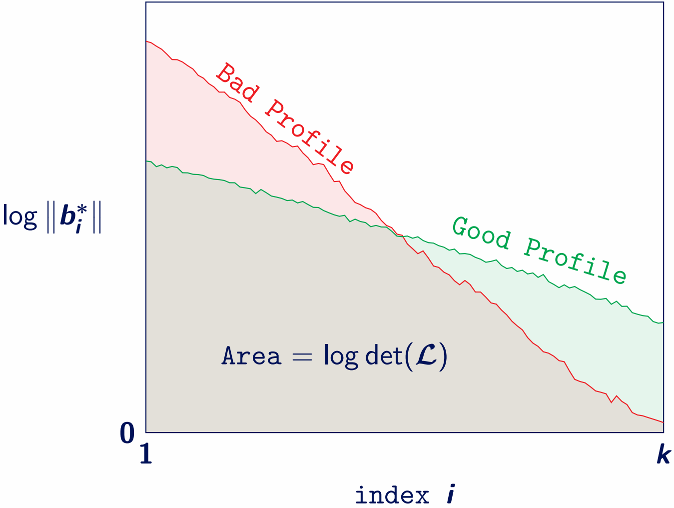


Fundamental Area: $\mathcal{F}_{B^*} := \prod_{i=1}^{k} \left[-\frac{1}{2}b_i^*, \frac{1}{2}b_i^*\right)$

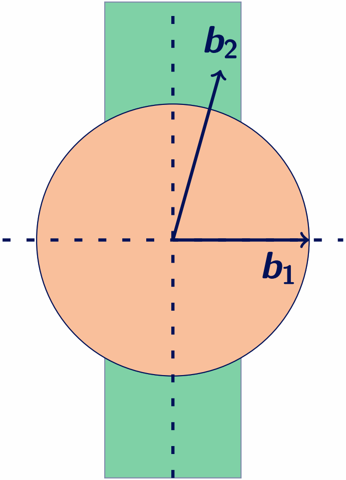
Babai Decoding: $e := t - v \in \mathcal{F}_{B^*}$

Good Basis:
$\|b_1^*\| \approx \ldots \approx \|b_k^*\|$

Profile



Bad Profile
Good Profile
Area = log det($\mathcal{L}$)
$\log \|b_i^*\|$
0
1
k
index i



Wristwatch Lemma

For any lattice $\mathcal{L}$ of rank 2
there exists a basis (b_1, b_2) s.t.

$$\|b_1\| \leq \|b_2\|$$

$$|\langle b_1, b_2 \rangle| \leq \frac{1}{2} \|b_1\|$$

$$\Downarrow$$

$$\boxed{\|b_1^*\| \leq \sqrt{\frac{4}{3}} \cdot \|b_2^*\|}$$

Definition

A basis $\mathbf{B}$ of $\mathcal{L}$ is LLL-reduced if $(\pi_i(\mathbf{b}_i), \pi_i(\mathbf{b}_{i+1}))$ is Lagrange Reduced for all $i < k$.

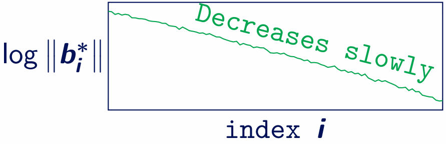
LLL Reduction

Definition

A basis B of $\mathcal{L}$ is LLL-reduced if $(\pi_i(b_i), \pi_i(b_{i+1}))$ is Lagrange Reduced for all $i < k$.

$$\Downarrow$$

$$\forall i < k,\ \left\|b_i^*\right\| \leq \sqrt{4/3} \cdot \left\|b_{i+1}^*\right\|$$



Definition

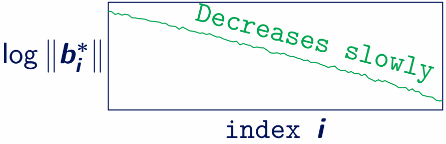
A basis B of $\mathcal{L}$ is LLL-reduced if $(\pi_i(b_i), \pi_i(b_{i+1}))$ is Lagrange Reduced for all $i < k$.

$$\Downarrow$$

$$\forall i < k,\ \|b_i^*\| \le \sqrt{4/3} \cdot \|b_{i+1}^*\|$$

$$\Downarrow$$

$$\|b_1\| \le \sqrt{4/3}^{\frac{k-1}{2}} \cdot \det(\mathcal{L})^{1/k}$$



Definition

A basis $\boldsymbol{B}$ of $\mathcal{L}$ is LLL-reduced if $(\pi_i(\boldsymbol{b}_i), \pi_i(\boldsymbol{b}_{i+1}))$ is Lagrange Reduced for all $i < k$.

Algorithm

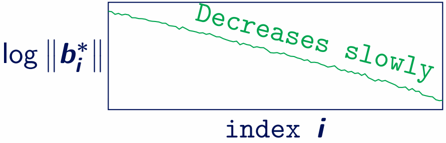
While $\exists i$ s.t. $(\pi_i(\boldsymbol{b}_i), \pi_i(\boldsymbol{b}_{i+1}))$ is *not* Lagrange Reduced, Langrange Reduce it.

$$\Downarrow$$

$$\forall i < k,\ \left\|\boldsymbol{b}_i^*\right\| \leq \sqrt{4/3} \cdot \left\|\boldsymbol{b}_{i+1}^*\right\|$$

$$\Downarrow$$

$$\|\boldsymbol{b}_1\| \leq \sqrt{4/3}^{\frac{k-1}{2}} \cdot \det(\mathcal{L})^{1/k}$$



LLL Reduction

Definition

A basis B of $\mathcal{L}$ is LLL-reduced if $(\pi_i(b_i), \pi_i(b_{i+1}))$ is Lagrange Reduced for all $i < k$.

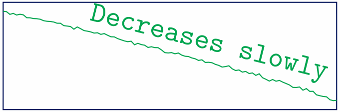
$$\Downarrow$$

$$\forall i < k, \ \|b_i^*\| \leq \sqrt{4/3} \cdot \|b_{i+1}^*\|$$

$$\Downarrow$$

$$\|b_1\| \leq \sqrt{4/3}^{\frac{k-1}{2}} \cdot \det(\mathcal{L})^{1/k}$$

$\log \|b_i^*\|$



index i

Algorithm

While $\exists i$ s.t. $(\pi_i(b_i), \pi_i(b_{i+1}))$ is *not* Lagrange Reduced, Langrange Reduce it.

Termination in poly-time:

Requires a slight relaxation. (ϵ-Lagrange Reduced)

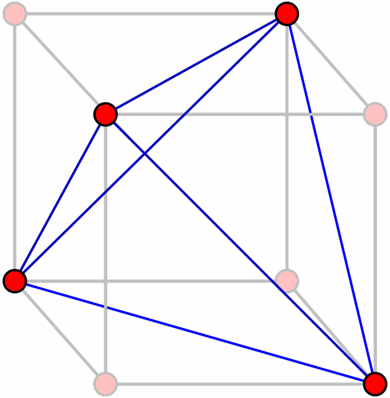
Proof argument:

$$P = \sum_{i \leq k} (n + 1 - i) \cdot \log \|b_i^*\|$$

Decreases by ϵ at each step and is lower-bounded.

$$\mathcal{C}(\boldsymbol{B}) := \{\sum_i x_i \boldsymbol{b}_i : \boldsymbol{x} \in \mathbb{F}_2^k\} \subset \mathbb{F}_2^n$$

k-dimensional subspace of $\mathbb{F}_2^n$,
endowed with the Hamming metric.



Orthopodality

To mimic LLL we need
a notion of orthogonality
for codewords.

To mimic LLL we need
a notion of orthogonality
for codewords.

Inner product
$\langle x, y \rangle = \sum x_i y_i \bmod 2$
gives no relations
on $|x|, |y|, |x \oplus y|$.

Orthopodality

To mimic LLL we need
a notion of orthogonality
for codewords.

Inner product
$\langle x, y \rangle = \sum x_i y_i \bmod 2$
gives no relations
on $|x|, |y|, |x \oplus y|$.

Definition
$\mathrm{Supp}(x) := \{i : x_i \neq 0\}$.

Orthopodality
$x \perp y$ if
$\mathrm{Supp}(x) \cap \mathrm{Supp}(y) = \emptyset$.

$|x \oplus y| = |x| + |y|$
if $x \perp y$

x	0	0	1	0	1	$\mathrm{Supp}(x) = \{3, 5\}$
y	1	0	0	1	0	$\mathrm{Supp}(y) = \{1, 4\}$

To mimic LLL we need
a notion of orthogonality
for codewords.

Inner product
$\langle x, y\rangle = \sum x_i y_i \bmod 2$
gives no relations
on $|x|, |y|, |x \oplus y|$.

Definition
$\mathrm{Supp}(x) := \{i : x_i \neq 0\}$.

Orthopodality
$x \perp y$ if
$\mathrm{Supp}(x) \cap \mathrm{Supp}(y) = \emptyset$.

$|x \oplus y| = |x| + |y|$
if $x \perp y$

x	0	0	1	0	1	$\mathrm{Supp}(x) = \{3, 5\}$
y	1	0	1	1	0	$\mathrm{Supp}(y) = \{1, 3, 4\}$

To mimic LLL we need
a notion of orthogonality
for codewords.

Inner product
$\langle x, y\rangle = \sum x_i y_i \bmod 2$
gives no relations
on $|x|, |y|, |x \oplus y|$.

Definition
$\mathrm{Supp}(x) := \{i : x_i \neq 0\}$.

Orthopodality
$x \perp y$ if
$\mathrm{Supp}(x) \cap \mathrm{Supp}(y) = \emptyset$.

$|x \oplus y| = |x| + |y|$
if $x \perp y$

x	0	0	1	0	1	$\mathrm{Supp}(x) = \{3, 5\}$
$\pi_{x^\perp}(y)$	1	0	0	1	0	$\mathrm{Supp}(y) = \{1, 3, 4\}$

To mimic LLL we need
a notion of orthogonality
for codewords.

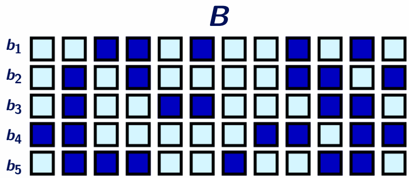
Inner product
$\langle x, y \rangle = \sum x_i y_i \bmod 2$
gives no relations
on $|x|, |y|, |x \oplus y|$.

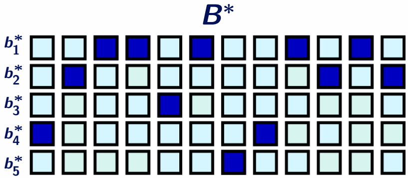
Definition
$\mathrm{Supp}(x) := \{i : x_i \neq 0\}$.

Orthopodality
$x \perp y$ if
$\mathrm{Supp}(x) \cap \mathrm{Supp}(y) = \emptyset$.

$|x \oplus y| = |x| + |y|$
if $x \perp y$

Gram-Schmidt-like
orthopodalisation





To mimic LLL we need
a notion of orthogonality
for codewords.

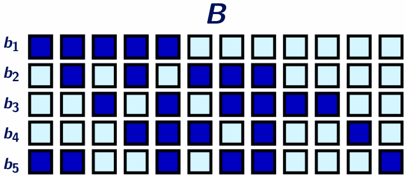
Inner product
$\langle x, y \rangle = \sum x_i y_i \bmod 2$
gives no relations
on $|x|, |y|, |x \oplus y|$.

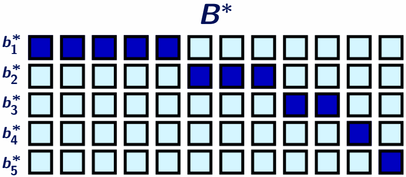
Definition
$\mathrm{Supp}(x) := \{i : x_i \neq 0\}$.

Orthopodality
$x \perp y$ if
$\mathrm{Supp}(x) \cap \mathrm{Supp}(y) = \emptyset$.

$|x \oplus y| = |x| + |y|$
if $x \perp y$

Gram-Schmidt-like
orthopodalisation





Orthopodality

To mimic LLL we need
a notion of <u>orthogonality</u>
for codewords.

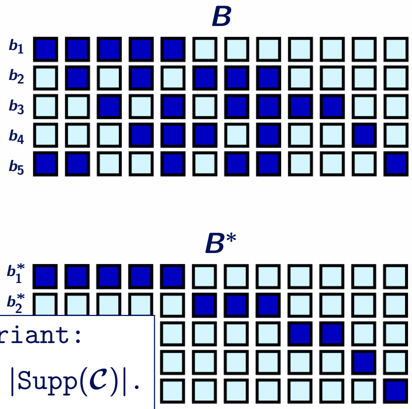
Inner product
$\langle x, y\rangle = \sum x_i y_i \bmod 2$
gives no relations
on $|x|, |y|, |x \oplus y|$.

<u>Definition</u>
$\mathrm{Supp}(x) := \{i : x_i \neq 0\}$.

<u>Orthopodality</u>
$x \perp y$ if
$\mathrm{Supp}(x) \cap \mathrm{Supp}(y) = \emptyset$.

$|x \oplus y| = |x| + |y|$
if $x \perp y$

Gram-Schmidt-like
<u>orthopodalisation</u>



Invariant:
$\sum_{i=1}^{k} |b_i^*| = |\mathrm{Supp}(\mathcal{C})|$.

Orthopodality

To mimic LLL we need
a notion of orthogonality
for codewords.

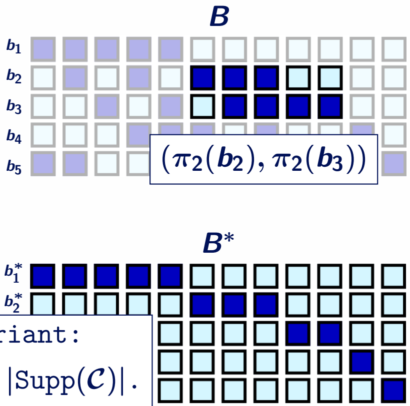
Inner product
$\langle x, y \rangle = \sum x_i y_i \bmod 2$
gives no relations
on $|x|, |y|, |x \oplus y|$.

Definition
$\mathrm{Supp}(x) := \{i : x_i \neq 0\}$.

Orthopodality
$x \perp y$ if
$\mathrm{Supp}(x) \cap \mathrm{Supp}(y) = \emptyset$.

$|x \oplus y| = |x| + |y|$
if $x \perp y$

Gram-Schmidt-like
orthopodalisation



Invariant:
$\sum_{i=1}^{k} |b_i^*| = |\mathrm{Supp}(\mathcal{C})|$.

Langrange Reduction (for codes)

<u>Lemma</u>

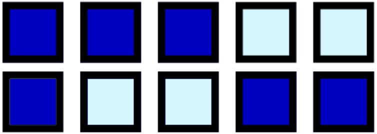
For any code $\mathcal{C}$ of support size $n = |\mathrm{Supp}(\mathcal{C})|$, and rank $k = 2$, there exists a basis b_1, b_2 s.t.

$$|b_1| \leq |b_2|, \qquad |\mathrm{Supp}(b_1) \cap \mathrm{Supp}(b_2)| \leq \frac{1}{2} \cdot |b_1|.$$

$$\Downarrow$$

$$|b_1^*| \leq 2 \cdot |b_2^*|$$

(Lattice case: $\|b_1^*\| \leq \sqrt{4/3}\,\|b_2^*\|$)



LLL Reduction (for codes)

Algorithm

While $\exists \boldsymbol{i}$ s.t. $(\pi_{\boldsymbol{i}}(\boldsymbol{b_i}), \pi_{\boldsymbol{i}}(\boldsymbol{b_{i+1}}))$ is *not* Lagrange Reduced,
Langrange Reduce it.

LLL Reduction (for codes)

Algorithm

While $\exists \boldsymbol{i}$ s.t. $(\pi_i(\boldsymbol{b}_i), \pi_i(\boldsymbol{b}_{i+1}))$ is *not* Lagrange Reduced,
Langrange Reduce it.

- Runs in polynomial time.
- No need for an ϵ-relaxation.
- Same potential argument works.

LLL Reduction (for codes)

Algorithm

While $\exists \boldsymbol{i}$ s.t. $(\pi_{\boldsymbol{i}}(\boldsymbol{b}_{\boldsymbol{i}}), \pi_{\boldsymbol{i}}(\boldsymbol{b}_{\boldsymbol{i}+1}))$ is *not* Lagrange Reduced,
Langrange Reduce it.

- Runs in polynomial time.
- No need for an ϵ-relaxation.
- Same potential argument works.

$$|\boldsymbol{b}_{\boldsymbol{i}}^*| \leq 2 \cdot |\boldsymbol{b}_{\boldsymbol{i}+1}^*|, \qquad |\boldsymbol{b}_{\boldsymbol{i}}^*| \geq 1$$

LLL Reduction (for codes)

Algorithm

While $\exists i$ s.t. $(\pi_i(b_i), \pi_i(b_{i+1}))$ is *not* Lagrange Reduced,
Langrange Reduce it.

- Runs in polynomial time.
- No need for an ϵ-relaxation.
- Same potential argument works.

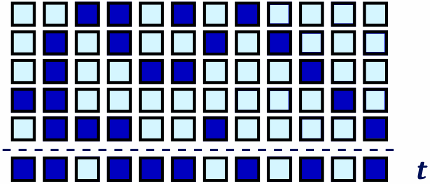
$$|b_i^*| \leq 2 \cdot |b_{i+1}^*|, \qquad |b_i^*| \geq 1$$

$$\Downarrow$$

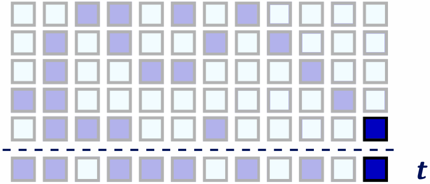
$$|b_1| - \frac{\lceil \log_2 |b_1| \rceil}{2} \leq \frac{n-k}{2} + 1$$

Griesmer bound!

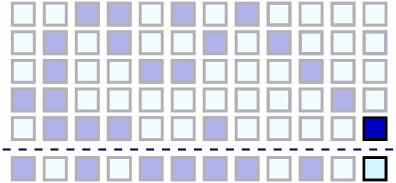
Prange Decoding



Prange Decoding

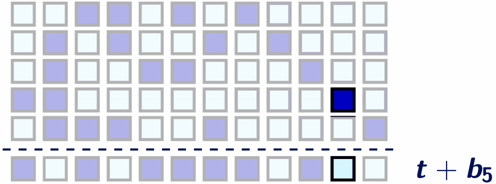


Prange Decoding

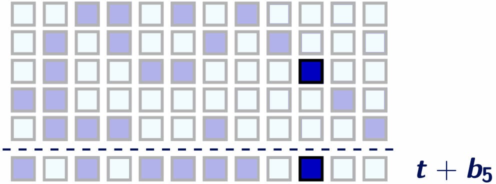


$t + b_5$

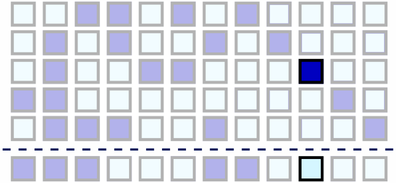
Prange Decoding



Prange Decoding

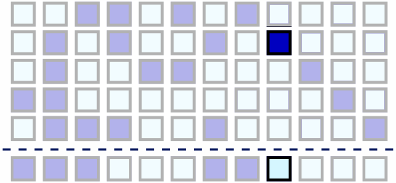


Prange Decoding



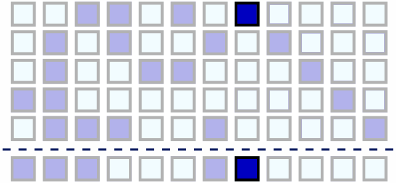
$\boldsymbol{t} + \boldsymbol{b}_5 + \boldsymbol{b}_3$

Prange Decoding



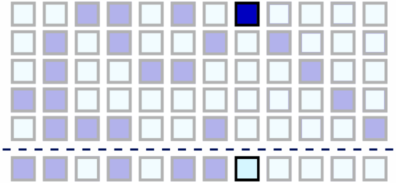
$\boldsymbol{t} + \boldsymbol{b}_5 + \boldsymbol{b}_3$

Prange Decoding



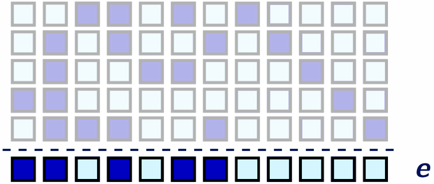
$t + b_5 + b_3$

Prange Decoding

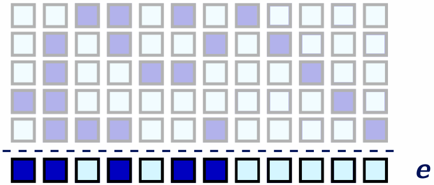


$t + b_5 + b_3 + b_1$

Prange Decoding



Prange Decoding



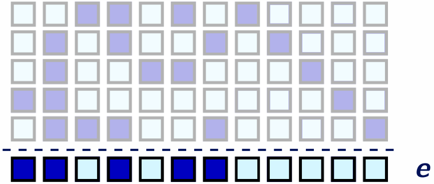
Fun. Domain: $e \in \mathcal{F} := \mathbb{F}_2^{n-k} \times \{0\}^k$

Worst-case: $|e| \leq n - k$

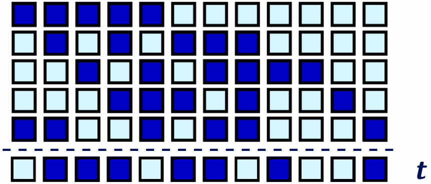
Average-case: $\mathbb{E}[|e|] = \frac{n-k}{2}$

Babai Decoding (for codes)

Prange Decoding



Babai Decoding



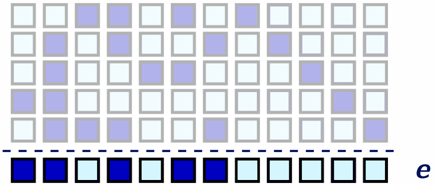
Fun. Domain: $e \in \mathcal{F} := \mathbb{F}_2^{n-k} \times \{0\}^k$

Worst-case: $|e| \leq n - k$

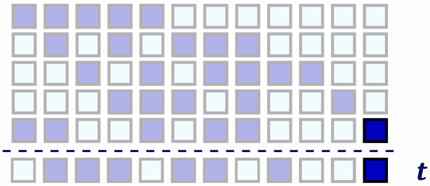
Average-case: $\mathbb{E}[|e|] = \frac{n-k}{2}$

Babai Decoding (for codes)

Prange Decoding



Babai Decoding

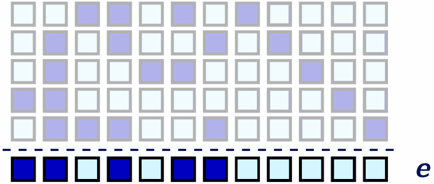


Fun. Domain: $e \in \mathcal{F} := \mathbb{F}_2^{n-k} \times \{0\}^k$

Worst-case: $|e| \leq n - k$

Average-case: $\mathbb{E}[|e|] = \frac{n-k}{2}$

Prange Decoding

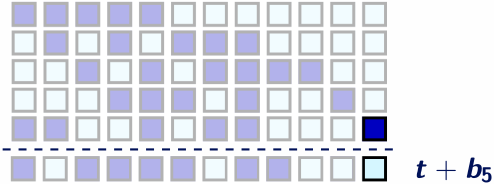


Fun. Domain: $e \in \mathcal{F} := \mathbb{F}_2^{n-k} \times \{0\}^k$

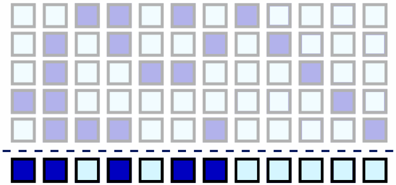
Worst-case: $|e| \leq n - k$

Average-case: $\mathbb{E}[|e|] = \frac{n-k}{2}$

Babai Decoding



Prange Decoding

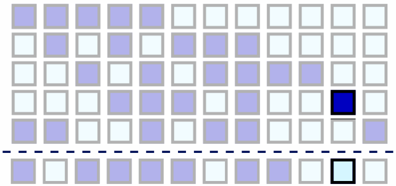


Fun. Domain: $e \in \mathcal{F} := \mathbb{F}_2^{n-k} \times \{0\}^k$

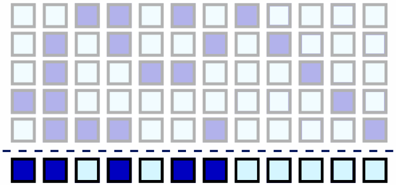
Worst-case: $|e| \leq n - k$

Average-case: $\mathbb{E}[|e|] = \frac{n-k}{2}$

Babai Decoding



Prange Decoding



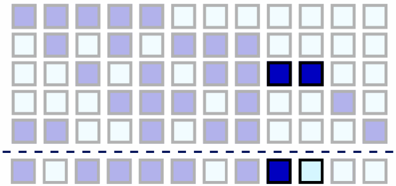
e

Fun. Domain: $e \in \mathcal{F} := \mathbb{F}_2^{n-k} \times \{0\}^k$

Worst-case: $|e| \le n - k$

Average-case: $\mathbb{E}[|e|] = \frac{n-k}{2}$

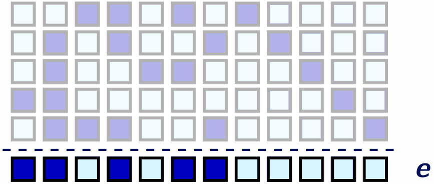
Babai Decoding



$t + b_5$

Babai Decoding (for codes)

Prange Decoding

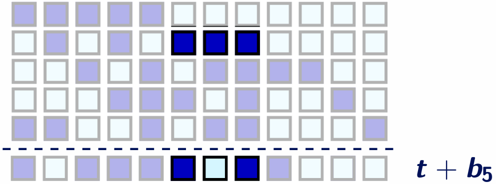


Fun. Domain: $e \in \mathcal{F} := \mathbb{F}_2^{n-k} \times \{0\}^k$

Worst-case: $|e| \leq n - k$

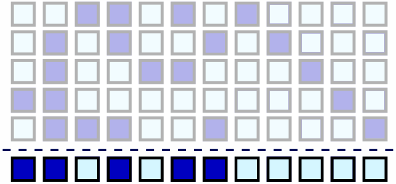
Average-case: $\mathbb{E}[|e|] = \frac{n-k}{2}$

Babai Decoding



Babai Decoding (for codes)

Prange Decoding



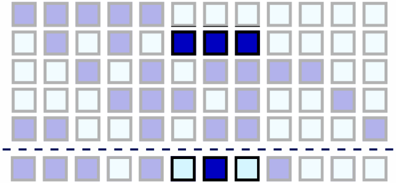
e

Fun. Domain: $e \in \mathcal{F} := \mathbb{F}_2^{n-k} \times \{0\}^k$

Worst-case: $|e| \leq n - k$

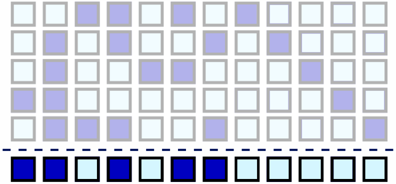
Average-case: $\mathbb{E}[|e|] = \frac{n-k}{2}$

Babai Decoding



$t + b_5 + b_2$

Prange Decoding



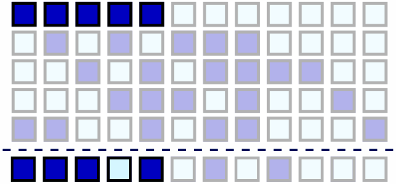
e

Fun. Domain: $e \in \mathcal{F} := \mathbb{F}_2^{n-k} \times \{0\}^k$

Worst-case: $|e| \leq n - k$

Average-case: $\mathbb{E}[|e|] = \frac{n-k}{2}$

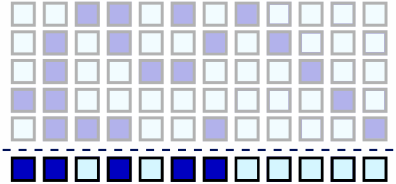
Babai Decoding



$t + b_5 + b_2$

Babai Decoding (for codes)

Prange Decoding



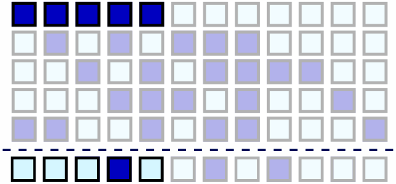
e

Fun. Domain: $\boldsymbol{e} \in \mathcal{F} := \mathbb{F}_2^{n-k} \times \{0\}^k$

Worst-case: $|\boldsymbol{e}| \leq n - k$

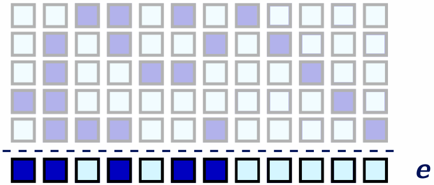
Average-case: $\mathbb{E}[|\boldsymbol{e}|] = \frac{n-k}{2}$

Babai Decoding

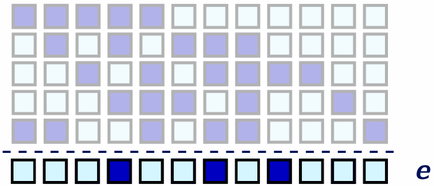


$\boldsymbol{t} + \boldsymbol{b}_5 + \boldsymbol{b}_2 + \boldsymbol{b}_1$

Prange Decoding



Babai Decoding

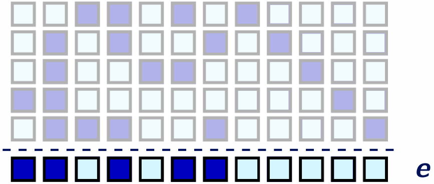


Fun. Domain: $e \in \mathcal{F} := \mathbb{F}_2^{n-k} \times \{0\}^k$

Worst-case: $|e| \leq n - k$

Average-case: $\mathbb{E}[|e|] = \frac{n-k}{2}$

Prange Decoding

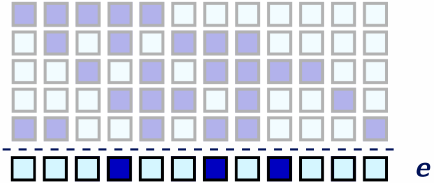


Fun. Domain: $e \in \mathcal{F} := \mathbb{F}_2^{n-k} \times \{0\}^k$

Worst-case: $|e| \leq n - k$

Average-case: $\mathbb{E}[|e|] = \frac{n-k}{2}$

Babai Decoding



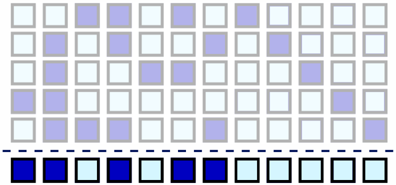
Fun. Domain(*): $e \in \mathcal{F}_{B^*} := \prod_{i=1}^{k} \mathcal{B}^{|b_i^*|}_{\lfloor |b_i^*|/2 \rfloor}$

Worst-case: $|e| \leq \sum_{i=1}^{n} \lfloor |b_i^*|/2 \rfloor \ll n - k$

Average-case: $\mathbb{E}[|e|] \leq \frac{n-k}{2}$

Babai Decoding (for codes)

Prange Decoding



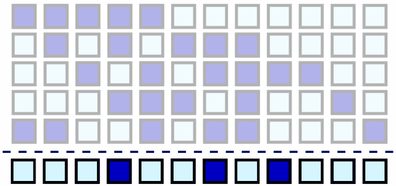
e

Fun. Domain: $e \in \mathcal{F} := \mathbb{F}_2^{n-k} \times \{0\}^k$

Worst-case: $|e| \leq n - k$

Average-case: $\mathbb{E}[|e|] = \frac{n-k}{2}$

Babai Decoding



e

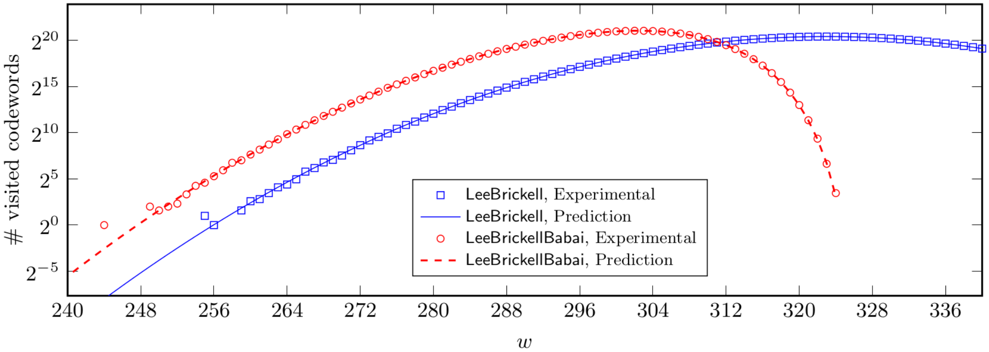
Fun. Domain(*): $e \in \mathcal{F}_{B^*} := \prod_{i=1}^{k} \mathcal{B}_{\lfloor |b_i^*|/2 \rfloor}^{|b_i^*|}$

Worst-case: $|e| \leq \sum_{i=1}^{n} \lfloor |b_i^*|/2 \rfloor \ll n - k$

Average-case: $\mathbb{E}[|e|] \leq \frac{n-k}{2}$

Better when more reduced!

Improved hybrid algorithms



$\Theta(n^{0.717}/\log(n))$ heuristic speed-up over standard Lee Brickell.

Compatible with more advanced algorithms.

Thank you!

Paper:
eprint.iacr.org/2020/869

Code & Experiments:
github.com/lducas/CodeRed

Open-source!