Randomisation en PMNS: arithmetic, redundancy and equality test

Nadia EL MRABET

SAS, Ecoles des Mines de Saint Etienne

WRACH 2025
Roscoff, April 23 2025

Introduction

Context:

- Main goal: Efficient and secure modular arithmetic
- PMNS: Polynomial Modular Number System
- Main characteristic: Elements are polynomials in the PMNS
- Additional characteristic: PMNS is a redundant system

Introduction

Context:

- Main goal: Efficient and secure modular arithmetic
- PMNS: Polynomial Modular Number System
- Main characteristic: Elements are polynomials in the PMNS
- Additional characteristic: PMNS is a redundant system

Goals:

- Improve and extend PMNS generation
- Study and control the redundancy in the PMNS
- Perform equality test within the system

Presentation based on: https://eprint.iacr.org/2023/1231

Plan

PMNS: Polynomial Modular Number System

Let $p \geqslant 3$, be an odd integer. We want to represent elements of $\mathbb{Z}/p\mathbb{Z}$.

A PMNS is a subset of $\mathbb{Z}[X]$, defined by a tuple $\mathcal{B} = (p, n, \gamma, \rho, E)$.

- $n \in \mathbb{N}$: elements are represented with n coefficients.
- $\gamma \in \mathbb{Z}/p\mathbb{Z}$: $T \in \mathcal{B}$ represents the integer $t = T(\gamma) \pmod p$
- $\rho \in \mathbb{N}$: $\|T\|_\infty < \rho, \forall T \in \mathcal{B}$
- E: a monic polynomial $\in \mathbb{Z}_n[X]$, such that $E(\gamma) \equiv 0 \pmod p$.

where $0 < \gamma < p$ and $\rho \approx \sqrt[n]{p}$.

Example: $\mathcal{B} = (p,\ n,\ \gamma,\ \rho,\ E) = (19,\ 3,\ 7,\ 2,\ X^3 - 1)$

0	1	2	3	4
0	1	$-X^2 - X + 1$	$X^2 - X - 1$	$X^2 - X$

5	6	7	8	9
$X^2 - X + 1$	$X - 1$	X	$X + 1$	$-X^2 + 1$

10	11	12	13	14
$\boldsymbol{X^2 - 1}$	X^2	$X^2 + 1$	$-X + 1$	$-X^2 + X - 1$

15	16	17	18
$-X^2 + X$	$-X^2 + X + 1$	$X^2 + X - 1$	-1

$(X^2 - 1) \equiv 10_{\mathcal{B}}$, since $7^2 - 1 = 48 \equiv 10 \pmod{19}$.

A redundant system: $(-X - 1) \equiv 11_{\mathcal{B}}$.

$$(X^2 + X + 1) \equiv 0_{\mathcal{B}}.$$

Main operations and reductions

Let $A, B \in \mathcal{B}$. There are two main operations:

- Addition: $S = A + B$
- Multiplication: $C = A \times B$

We have:

- $\deg(S) < n$, but $\|S\|_\infty < 2\rho$
- $\deg(C) < 2n - 1$, and $\|C\|_\infty < n\rho^2$

So, we need to:

- reduce $\deg(C)$ $\Rightarrow$ **External reduction**
- reduce $\|C\|_\infty$ and $\|S\|_\infty$ $\Rightarrow$ **Internal reduction**

Multiplication example for $\mathcal{B} = (19, 3, 7, 2, X^3 - 1)$

Remember that: $p = 19,\ n = 3,\ \gamma = 7,\ \rho = 2,\ E(X) = X^3 - 1$.

- Let $a = 8$; $A \equiv a_{\mathcal{B}}$, with $A(X) = X + 1$
- Let $b = 12$; $B \equiv b_{\mathcal{B}}$, with $B(X) = X^2 + 1$

Multiplication example for $\mathcal{B} = (19, 3, 7, 2, X^3 - 1)$

Remember that: $p = 19,\ n = 3,\ \gamma = 7,\ \rho = 2,\ E(X) = X^3 - 1$.

- Let $a = 8$; $A \equiv a_{\mathcal{B}}$, with $A(X) = X + 1$
- Let $b = 12$; $B \equiv b_{\mathcal{B}}$, with $B(X) = X^2 + 1$

- $C = AB = X^3 + X^2 + X + 1$
- $C(7) \bmod 19 = 1 = ab \pmod{19} = 1$, but $C \notin \mathcal{B}$

Multiplication example for $\mathcal{B} = (19, 3, 7, 2, X^3 - 1)$

Remember that: $p = 19,\ n = 3,\ \gamma = 7,\ \rho = 2,\ E(X) = X^3 - 1$.

- Let $a = 8$; $A \equiv a_{\mathcal{B}}$, with $A(X) = X + 1$
- Let $b = 12$; $B \equiv b_{\mathcal{B}}$, with $B(X) = X^2 + 1$

- $C = AB = X^3 + X^2 + X + 1$
- $C(7) \bmod 19 = 1 = ab \pmod{19} = 1$, but $C \notin \mathcal{B}$
- $R = C \bmod E = X^2 + X + 2$
- $R(7) \bmod 19 = 1$ and $\deg(R) < 3$, but $R \notin \mathcal{B}$.

Multiplication example for $\mathcal{B} = (19, 3, 7, 2, X^3 - 1)$

Remember that: $p = 19,\ n = 3,\ \gamma = 7,\ \rho = 2,\ E(X) = X^3 - 1$.

- Let $a = 8$; $A \equiv a_{\mathcal{B}}$, with $A(X) = X + 1$
- Let $b = 12$; $B \equiv b_{\mathcal{B}}$, with $B(X) = X^2 + 1$

- $C = AB = X^3 + X^2 + X + 1$
- $C(7) \bmod 19 = 1 = ab \pmod{19} = 1$, but $C \notin \mathcal{B}$
- $R = C \bmod E = X^2 + X + 2$
- $R(7) \bmod 19 = 1$ and $\deg(R) < 3$, but $R \notin \mathcal{B}$.

Internal reduction:

- Let $T(X) = X^2 + X + 1$.
  $T(7) \equiv 0 \pmod{19}$ and $S = R - T = 1 \in \mathcal{B}$
- How to find such a polynomial T?
  $\Rightarrow$ the **internal reduction** process

The internal reduction

Let $R \in \mathbb{Z}_{n-1}[X]$, with possibly $\|R\|_\infty \geqslant \rho$.

The Goal:

find $S \in \mathbb{Z}_{n-1}[X]$, such that: $\|S\|_\infty < \rho$ and $S(\gamma) \equiv R(\gamma) \pmod{p}$

Equivalent to compute:

$T \in \mathbb{Z}_{n-1}[X]$, such that: $T(\gamma) \equiv 0 \pmod{p}$ and $\|S\|_\infty = \|R - T\|_\infty < \rho$

The internal reduction

Let $R \in \mathbb{Z}_{n-1}[X]$, with possibly $\|R\|_\infty \geqslant \rho$.

The Goal:

find $S \in \mathbb{Z}_{n-1}[X]$, such that: $\|S\|_\infty < \rho$ and $S(\gamma) \equiv R(\gamma) \pmod{p}$

Equivalent to compute:

$T \in \mathbb{Z}_{n-1}[X]$, such that: $T(\gamma) \equiv 0 \pmod{p}$ and $\|S\|_\infty = \|R - T\|_\infty < \rho$

Many methods to do this reduction DDEMV'19:

- **Montgomery-like method DDEMV'19**
- Barrett-like method
- Babaï-based approaches
- 'Direct' approaches

Internal reduction: the Montgomery-like approach

By Christophe Negre and Thomas Plantard (2008).

Introduces an integer ϕ and two polynomials $M, M' \in \mathbb{Z}_{n-1}[X]$, such that:

- $\phi \geqslant 2$
- $M(\gamma) \equiv 0 \pmod{p}$
- $M' = -M^{-1} \bmod (E, \phi)$

Mont-like:

1: **Input :** $R \in \mathbb{Z}_{n-1}[X]$
2: **Output :** $S \in \mathbb{Z}_{n-1}[X]$, with $S(\gamma) \equiv R(\gamma)\phi^{-1} \pmod{p}$
3: $Q \leftarrow R \times M' \bmod (E, \phi)$
4: $T \leftarrow Q \times M \bmod E$
5: $S \leftarrow (R + T)/\phi$ # exact divisions
6: **return** S

Generation of M: a lattice of zeros

To a PMNS $\mathcal{B}$, one associates the following lattice:

$$\mathcal{L}_{\mathcal{B}} = \{\boldsymbol{A} \in \mathbb{Z}_{\boldsymbol{n}-1}[\boldsymbol{X}] \mid \boldsymbol{A}(\gamma) \equiv 0 \pmod{\boldsymbol{p}}\}$$

- $\mathcal{L}_{\mathcal{B}}$ is a n-dimensional full-rank Euclidean lattice;
- a basis of $\mathcal{L}_{\mathcal{B}}$ is:

$$\mathrm{B} = \begin{pmatrix} p & 0 & 0 & \dots & 0 & 0 \\ t_1 & 1 & 0 & \dots & 0 & 0 \\ t_2 & 0 & 1 & \dots & 0 & 0 \\ \vdots & & & \ddots & & \vdots \\ t_{n-2} & 0 & 0 & \dots & 1 & 0 \\ t_{n-1} & 0 & 0 & \dots & 0 & 1 \end{pmatrix} \begin{matrix} \leftarrow p \\ \leftarrow X + t_1 \\ \leftarrow X^2 + t_2 \\ \\ \leftarrow X^{n-2} + t_{n-2} \\ \leftarrow X^{n-1} + t_{n-1} \end{matrix}$$

where $t_i = (-\gamma)^i \bmod p$.

Note: each line i of B represents the polynomial $X^i + t_i$.

Generation of M: a lattice of zeros

- Let $\mathcal{W}$ be a reduced basis of $\mathcal{L}_{\mathcal{B}}$;
- i.e. $\mathcal{W} = LLL(\mathrm{B}) = BKZ(\mathrm{B}) = HKZ(\mathrm{B})$, ...

Let's assume that ϕ is a power of two (best choice for efficiency).

Fundamental result: (Didier, Dosso, Véron, JCEN-2020)

There always exists $(\alpha_0, \ldots, \alpha_{n-1}) \in \{0,1\}^n$, such that:

$$M = \sum_{i=0}^{n-1} \alpha_i \mathcal{W}_i \quad \text{and} \quad M' = -M^{-1} \bmod (E, \phi) \text{ exists.}$$

Note:

- we need Resultant(E, M) to be odd for M' to exist.
- we take $\rho \approx \|M\|_\infty$, hence a reduced basis $\mathcal{W}$.

So, to find a suitable polynomial M, a search is done in a space of size 2^n.

Application to cryptographic examples

Classical cryptography

- RSA 2048-4096 bits;
- ECC: scalar multiplication, pairings 256-512 bits.

Application to cryptographic examples

Classical cryptography

- RSA 2048-4096 bits;
- ECC: scalar multiplication, pairings 256-512 bits.

PQC

- SQiSign 256-512 bits.
- CSidh 256-512 bits.
- Racoon $q = (2^{24} - 2^{18} + 1) \times (2^{25} - 2^{18} + 1)$
- Dilithium 23 bits...

Software implementation

Brainpool, pairings DDE'25

Processor: Intel `11th Gen Intel Core` $i5\text{-}1135G7@2.40GHz \times 8$

Memory: 16 GiB of RAM

OS: Ubuntu 20.04.6 LTS (64 bits)

Our C implementations of PMNS can be found in this GitHub repository:

`https://github.com/PMNS-APPLICATION/`

Software implementation: brainpool

Table: Clock cycle number comparisons of Modular Multiplication for **brainpool** curve moduli.

Modulus	PMNS	OpenSSL	
		Bloc-Mont	Std
brainpoolP256r1	177	181	718
brainpoolP384r1	267	294	1071
brainpoolP512r1	405	347	1385

Software implementation: pairings

Table: Clock cycle number comparisons of Modular Multiplication for pairing-friendly base fields

Modulus	PMNS	GMP		
		Bloc-Mont	Low-lvl	Std
KSS16-330	225	248	494	541
BN-462	349	368	709	762
BLS12-381	275	249	496	547

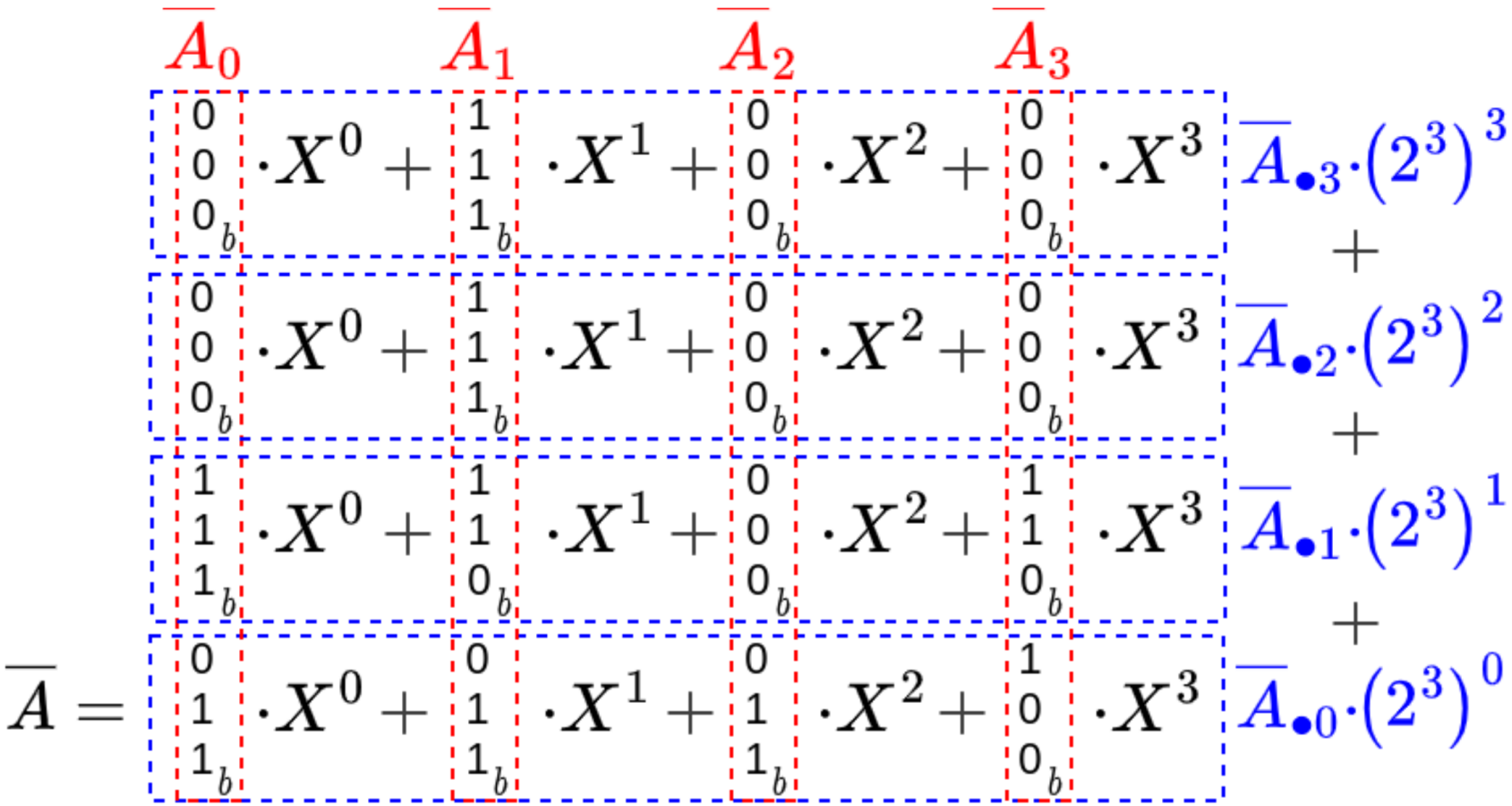
Hardware implementation: Block slicing

$$A = 59 - 13 \cdot X + 3 \cdot X^2 + 52 \cdot X^3$$

$$\overline{A} = \begin{matrix} \overline{A_0} \\ \begin{matrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1_b \end{matrix} \end{matrix} \cdot X^0 + \begin{matrix} \overline{A_1} \\ \begin{matrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1_b \end{matrix} \end{matrix} \cdot X^1 + \begin{matrix} \overline{A_2} \\ \begin{matrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1_b \end{matrix} \end{matrix} \cdot X^2 + \begin{matrix} \overline{A_3} \\ \begin{matrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0_b \end{matrix} \end{matrix} \cdot X^3$$

Hardware implementation

$$A = 59 - 13 \cdot X + 3 \cdot X^2 + 52 \cdot X^3$$
$$w = 3, s = 4, \overline{A} = \sum_{i=0}^{s-1} \overline{A}_{\bullet i} \cdot (2^w)^i, \phi = 2^{sw}$$



Hardware implementation

FPGAs devices : quick prototyping and design space exploration
Modern Xilinx Ultrascale FPGA family used in [8]
DSP48E2 arithmetic accelerator components feature:

- A 17x17 bits multiplier
- A 3-input 48-bit adder which can be used to add the result of a multiplication, accumulate data (possibly right shifted by 17 bits) and add external data in a single clock cycle

$w = 17$ bits slicing of operands.
Primary goals: performance and scalability to any number of coefficients and size of coefficients.

Hardware implementation

$\tilde{A}, \tilde{B}$: polynomial blocks such that $||A||_\infty < 2^{17}$ and $||B||_\infty < 2^{17}$, $N = 5$

Target	Cycle	Coefficients				
		X^0	X^1	X^2	X^3	X^4
$\tilde{A}$		$\tilde{A}_0$	$\tilde{A}_1$	$\tilde{A}_2$	$\tilde{A}_3$	$\tilde{A}_4$
$\tilde{B}$		$\tilde{B}_0$	$\tilde{B}_1$	$\tilde{B}_2$	$\tilde{B}_3$	$\tilde{B}_4$

		Congested Scheduling				
$\tilde{A} \cdot \tilde{B}[E]$	1	$\tilde{A}_0\tilde{B}_0$	$\tilde{A}_0\tilde{B}_1$	$\tilde{A}_0\tilde{B}_2$	$\tilde{A}_0\tilde{B}_3$	$\tilde{A}_0\tilde{B}_4$
	2	$+\lambda\tilde{A}_1\tilde{B}_4$	$+\ \tilde{A}_1\tilde{B}_0$	$+\ \tilde{A}_1\tilde{B}_1$	$+\ \tilde{A}_1\tilde{B}_2$	$+\tilde{A}_1\tilde{B}_3$

$\vdots$

		Relaxed Scheduling (N odd)				
$\tilde{A} \cdot \tilde{B}[E]$	1	$\tilde{A}_0\tilde{B}_0$	$\lambda\tilde{A}_3\tilde{B}_3$	$\tilde{A}_1\tilde{B}_1$	$\lambda\tilde{A}_4\tilde{B}_4$	$\tilde{A}_2\tilde{B}_2$
	2	$+\lambda\tilde{A}_2\tilde{B}_3$	$+\ \tilde{A}_0\tilde{B}_1$	$+\lambda\tilde{A}_3\tilde{B}_4$	$+\ \tilde{A}_1\tilde{B}_2$	$+\tilde{A}_4\tilde{B}_0$

$\vdots$

Hardware implementation

Design parameters	Freq (MHz)	Latency (cc)	DSP/LUT/FF	Time (μs)	DSP/LUT/FF AT (resource.μs)
width = 256					
CA0D2C1E [8]	625	140	16/1759/3365	0.224	3.58/394/754
AMNS [3]	200	33	120/2728/-	0.165	19.8/450/-
AMNS [3]	194	47	91/1718/-	0.242	22.0/415/-
RNS [1]	-	-	248/9450/-	0.0852	21.13/805/-
N = 3, s = 6	**625**	**111**	**18/4156/5145**	**0.178**	**3.20/738/914**
width = 512					
CA0D2C1E [8]	625	275	31/3443/6602	0.440	13.6/1510/2900
AMNS [3]	162	33	188/29985/-	0.204	38.4/6120/-
AMNS [3]	182	47	176/37138/-	0.258	45.4/9580/-
N = 7, s = 5	**550**	**199**	**35/8124/10128**	**0.362**	**12.7/2940/3660**
width = 2048					
CA0D2C1E [8]	625	1085	121/13487/22602	1.74	210/23400/39000
N = 5, s = 25	**500**	**785**	**125/29182/35008**	**1.57**	**196/45800/54900**
width = 4096					
CA0D2C1E [8]	625	2174	242/26978/44806	3.48	842/93800/156000
N = 5, s = 25	**525**	**1553**	**245/58161/66740**	**2.96**	**725/172000/197000**

Hardware implementation ENPV'24

Open-Source project

`https://github.com/LOUISNOYEZ/AMNS_MM`

A summary: the pro

- High parallelization capability (no carry propagation nor conditional branching)
- It is always possible to generate efficient PMNS given any prime: Efficient modular operations using the adapted modular number system (JCEN-2020)
- PMNS has been proven competitive for both hardware and software implementations:
  - PMNS for Efficient Arithmetic and Small Memory Cost (TETC-2022)
  - Modular Multiplication in the AMNS representation: Hardware Implementation (SAC-2024)
- PMNS is redundant: it allows easy and efficient randomisation. See: Randomization of Arithmetic over Polynomial Modular Number System (ARITH-26/2019).

A summary: the cons

When n becomes big:

- The generation of the parameter M could be very long; the search is done in a space of size 2^n.

PMNS is redundant:

- More memory is needed to represent elements (compared to a non-redundant system).
- **Trivial equality test is not possible.**

Our goals in the remaining:

- Define and control redundancy in the PMNS.
- Make equality test possible within the PMNS (even when the system is chosen very redundant).

Redundancy in the PMNS: first attempt to secure against SCA

Redundancy in the PMNS: first attempt to secure against SCA

Reductions are guilty

The internal and external reductions are well defined, so well that we define a canonical representation in PMNS.

Redundancy in the PMNS: first attempt to secure against SCA

Reductions are guilty

The internal and external reductions are well defined, so well that we define a canonical representation in PMNS.

The end of PMNS vs SCA?

Of course not!

Redundancy in the PMNS

Limitations:

- It is not precisely defined.
- We can only choose the **minimum** number of distinct representations for $\mathbb{Z}/p\mathbb{Z}$ elements in the PMNS.
  See: Randomization of Arithmetic over PMNS (ARITH-26).

Motivations:

Precisely control the redundancy for:

- smaller memory requirement to represent element,
- a more reliable randomisation.

A new tool: the set $\mathcal{D}_j$

Sub-lattice $\mathcal{L}$ of zeros: some fundamental regions

Let $\mathcal{G}$ be a basis of $\mathcal{L}$.

Let $\mathcal{H}$ be the fundamental domain of $\mathcal{L}$:

$$\mathcal{H} = \{t \in \mathbb{R}^n \mid t = \sum_{i=0}^{n-1} \mu_i \mathcal{G}_i \quad \text{and} \quad 0 \leqslant \mu_i < 1\}$$

And $\mathcal{H}'$ be the fundamental region:

$$\mathcal{H}' = \{t \in \mathbb{R}^n \mid t = \sum_{i=0}^{n-1} \mu_i \mathcal{G}_i \quad \text{and} \quad -\frac{1}{2} \leqslant \mu_i < \frac{1}{2}\}$$

Remarks:

- If $V \in \mathcal{H}$, then $\|V\|_\infty < \|\mathcal{G}\|_1$.
- If $V \in \mathcal{H}'$, then $\|V\|_\infty \leqslant \frac{1}{2}\|\mathcal{G}\|_1$.

A representation of $\mathcal{H}$ and $\mathcal{H}'$, for $n = 2$

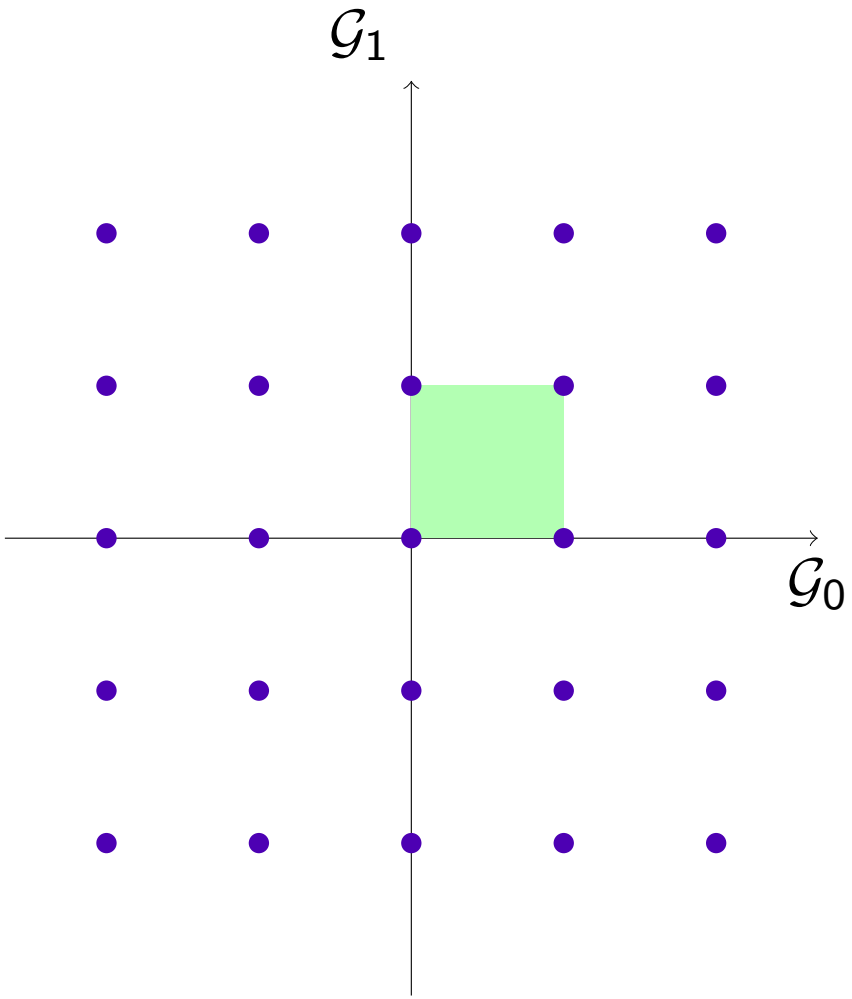


Figure: $\mathcal{H}$

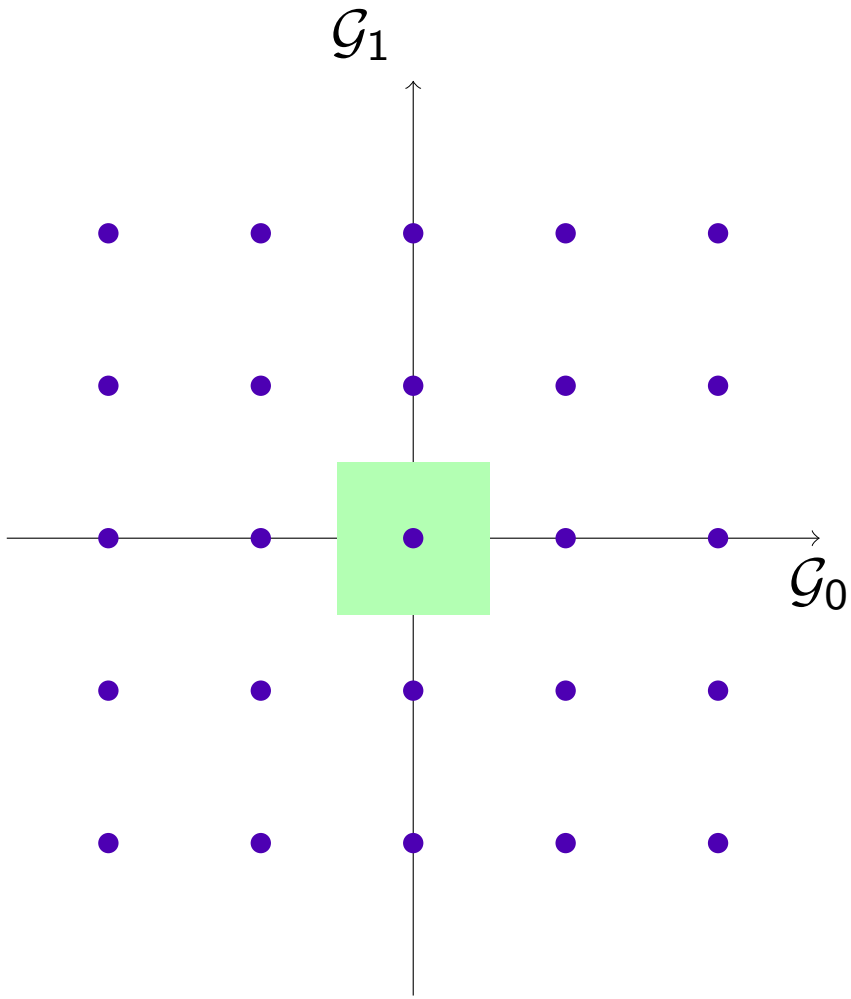


Figure: $\mathcal{H}'$

$\mathcal{D}_j$: the Domain j

Let $j \geqslant 1$ be an integer.

We define the set $\mathcal{D}_j$ as:

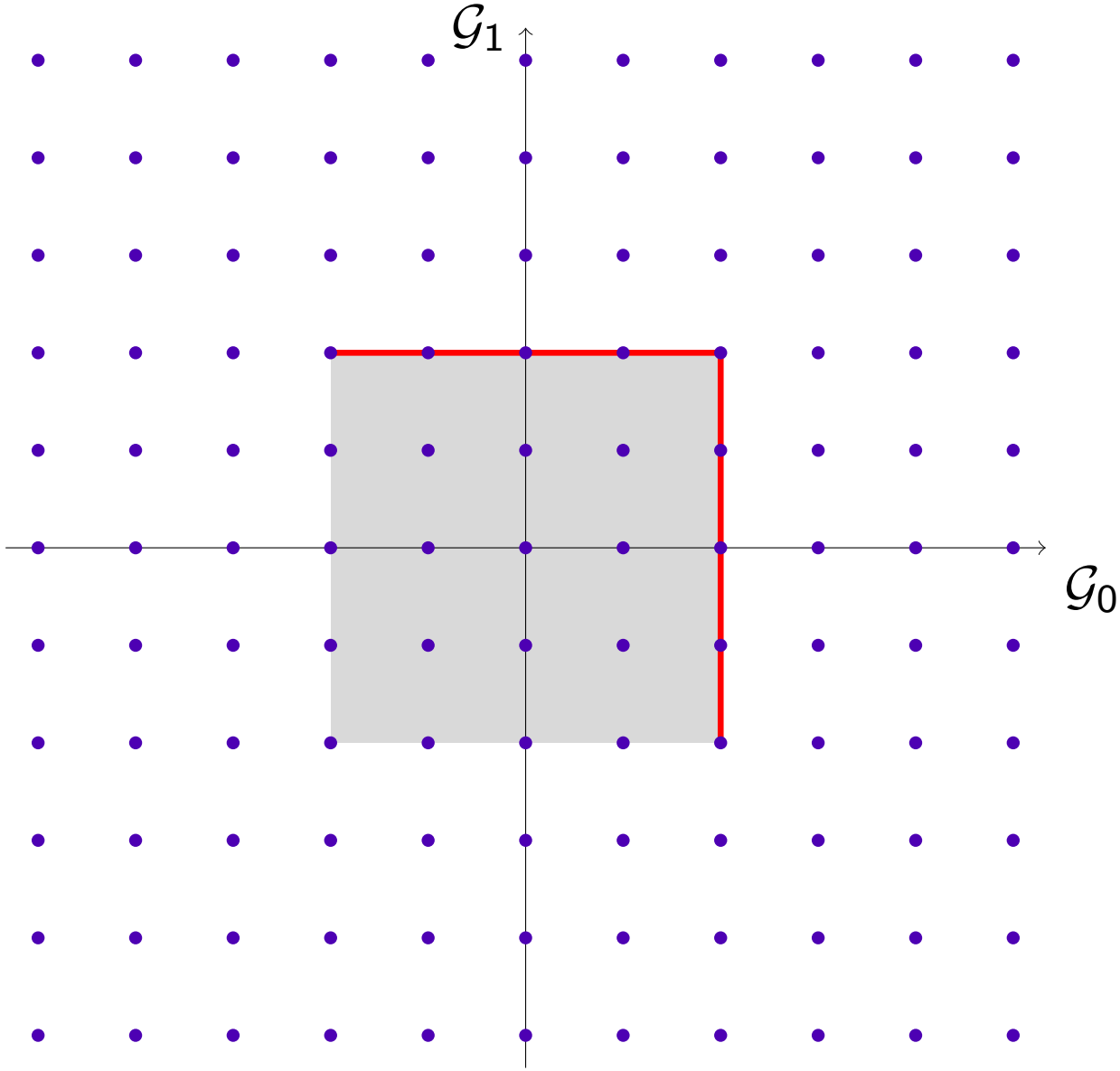
$$\mathcal{D}_j = \{t \in \mathbb{R}^n \mid t = \sum_{i=0}^{n-1} \mu_i \mathcal{G}_i \quad \text{and} \quad -j \leqslant \mu_i < j\}$$

This can be seen as an extension of the fundamental region $\mathcal{H}'$.

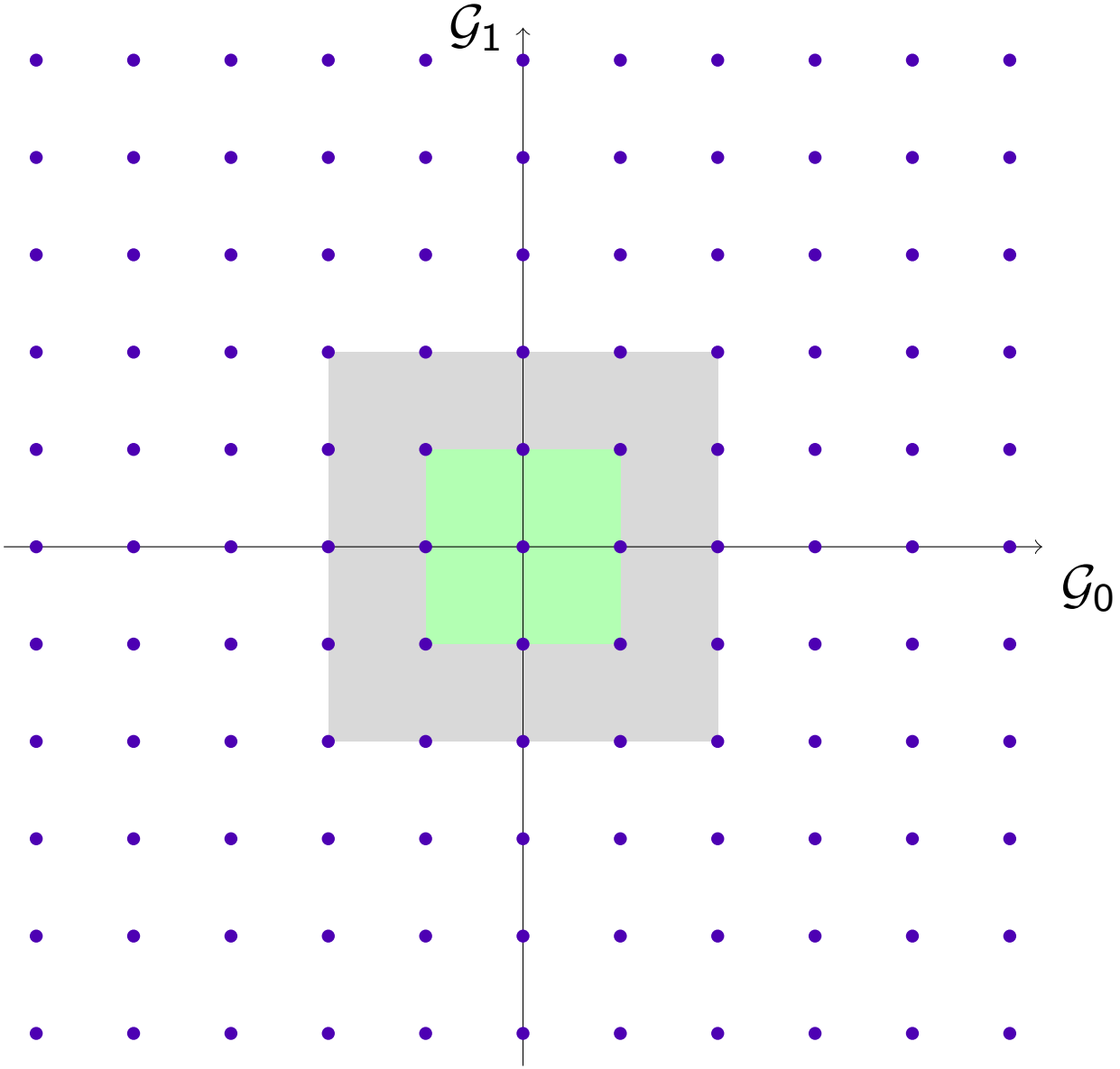
Remark

If $A \in \mathcal{D}_j$, then: $\|A\|_\infty \leqslant j\|\mathcal{G}\|_1$.

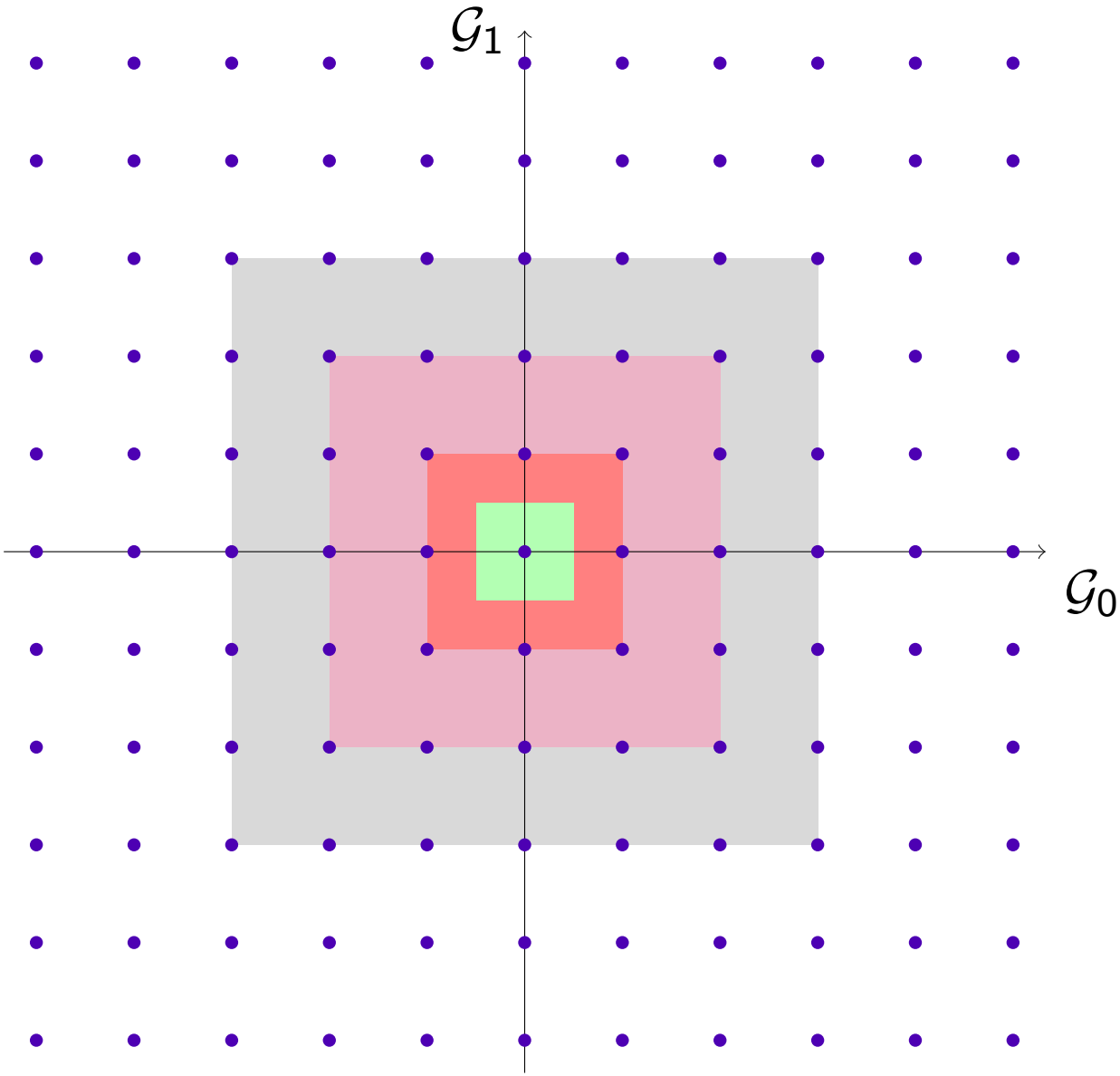
A representation of $\mathcal{D}_2$, for $n = 2$



Domain $\mathcal{D}_1$ vs $\mathcal{D}_2$, for $n = 2$



A representation of $\mathcal{H}'$, $\mathcal{D}_1$, $\mathcal{D}_2$ and $\mathcal{D}_3$, for $n = 2$



Redundancy in the PMNS

Fundamental result:

The set $\mathcal{D}_j$ contains exactly $(2j)^n$ times the set $\mathcal{H}$.

Property:

If $\mathcal{L} = \mathcal{L}_{\mathcal{B}}$, then each $a \in \mathbb{Z}/p\mathbb{Z}$ has **exactly one representation** in $\mathcal{H}$.

Redundancy in the PMNS

Fundamental result:

The set $\mathcal{D}_j$ contains exactly $(2j)^n$ times the set $\mathcal{H}$.

Property:

If $\mathcal{L} = \mathcal{L}_{\mathcal{B}}$, then each $a \in \mathbb{Z}/p\mathbb{Z}$ has **exactly one representation** in $\mathcal{H}$.

Consequence:

If $\mathcal{L} = \mathcal{L}_{\mathcal{B}}$, then:

each $a \in \mathbb{Z}/p\mathbb{Z}$ has **exactly** $(2j)^n$ **representation** in $\mathcal{D}_j$.

Redundancy in the PMNS

Let $a \in \mathbb{Z}/p\mathbb{Z}$.

The set of representations

Let's define the set $\mathcal{R}_j(a)$ as:

$$\mathcal{R}_j(a) = \{A \in \mathcal{D}_j \cap \mathbb{Z}^n \mid a = A(\gamma) \pmod{p}\}$$

Redundancy in the PMNS

Let $a \in \mathbb{Z}/p\mathbb{Z}$.

The set of representations

Let's define the set $\mathcal{R}_j(a)$ as:

$$\mathcal{R}_j(a) = \{A \in \mathcal{D}_j \cap \mathbb{Z}^n \mid a = A(\gamma) \pmod{p}\}$$

Property:

If $\mathcal{L} = \mathcal{L}_{\mathcal{B}}$, then:

$$\#\mathcal{R}_j(a) = (2j)^n$$

In particular, $\#\mathcal{R}_1(a) = 2^n$.

Easy to compute: the representations of zeros in $\mathcal{D}_j$

It corresponds to the lattice points in $\mathcal{D}_j$.

$$\mathcal{R}_j(0) = \{(\alpha_0, \ldots, \alpha_{n-1})\mathcal{G}, \text{ with } \alpha_i \in \mathbb{Z} \cap [-j, j[\,\}.$$

Redundancy in the PMNS

Property:

Let us assume that $\mathcal{L} = \mathcal{L}_{\mathcal{B}}$.
Let $a \in \mathbb{Z}/p\mathbb{Z}$. If A is its unique representation in $\mathcal{H}$, then:

$$\mathcal{R}_j(a) = \{A + J \mid J \in \mathcal{R}_j(0)\}.$$

Questions:

- How to compute a representation in $\mathcal{H}$?
- How to make PMNS elements live in a set $\mathcal{D}_j$?

Let us first focus on $\mathcal{D}_1$.

An interesting comparison: $\mathcal{D}_1$ vs $\mathcal{H}$

Comparison 1:

If $\mathcal{L} = \mathcal{L}_{\mathcal{B}}$, then:

- each $a \in \mathbb{Z}/p\mathbb{Z}$ has **exactly one representation** in $\mathcal{H}$.
- each $a \in \mathbb{Z}/p\mathbb{Z}$ has **exactly** 2^n **representation** in $\mathcal{D}_1$.

Comparison 2:

- If $A \in \mathcal{H}$, then $\|A\|_\infty < \|\mathcal{G}\|_1$.
- If $A \in \mathcal{D}_1$, then $\|A\|_\infty \leqslant \|\mathcal{G}\|_1$.

So, same memory requirement to represent their elements.
But, different redundancies.

Internal reduction to $\mathcal{D}_1$

Let $A \in \mathbb{Z}_{n-1}[X]$, with $A = (\alpha_0, \alpha_1, \ldots, \alpha_{n-1})\mathcal{G}$.

Fundamental property:

If $\forall i \in \{0, ..., n-1\},\ -\phi \leqslant \alpha_i \leqslant 0$, then:

$$\textbf{GMont-like}(A) \in \mathcal{D}_1.$$

Question:

How to make all the coordinates of an element negative?

Answer:

Using **the translation vector**.

The translation vector (a simplified version)

Let $A, B \in \mathcal{B}$ and $C = A \times B \bmod E$.

Property:

$C = \alpha\mathcal{G}$, with $\alpha = (\alpha_0, \ldots, \alpha_{n-1}) \in \mathbb{R}^n$ such that:

$$\|\alpha\|_\infty \leqslant w(\rho - 1)^2 \|\mathcal{G}^{-1}\|_1 \,.$$

- Let $u = \lceil w(\rho - 1)^2 \|\mathcal{G}^{-1}\|_1 \rceil$.
- The **translation vector** $\mathcal{T}$ is defined as follows:

$$\mathcal{T} = (-u, \ldots, -u)\mathcal{G} \,.$$

Important: note that $\mathcal{T} \in \mathcal{L}$.

The translation vector (a simplified version)

Let $A, B \in \mathcal{B}$ and $C = A \times B \bmod E$.

Property:

$C = \alpha\mathcal{G}$, with $\alpha = (\alpha_0, \ldots, \alpha_{n-1}) \in \mathbb{R}^n$ such that:

$$\|\alpha\|_\infty \leqslant w(\rho - 1)^2 \|\mathcal{G}^{-1}\|_1 .$$

- Let $u = \lceil w(\rho - 1)^2 \|\mathcal{G}^{-1}\|_1 \rceil$.
- The **translation vector** $\mathcal{T}$ is defined as follows:

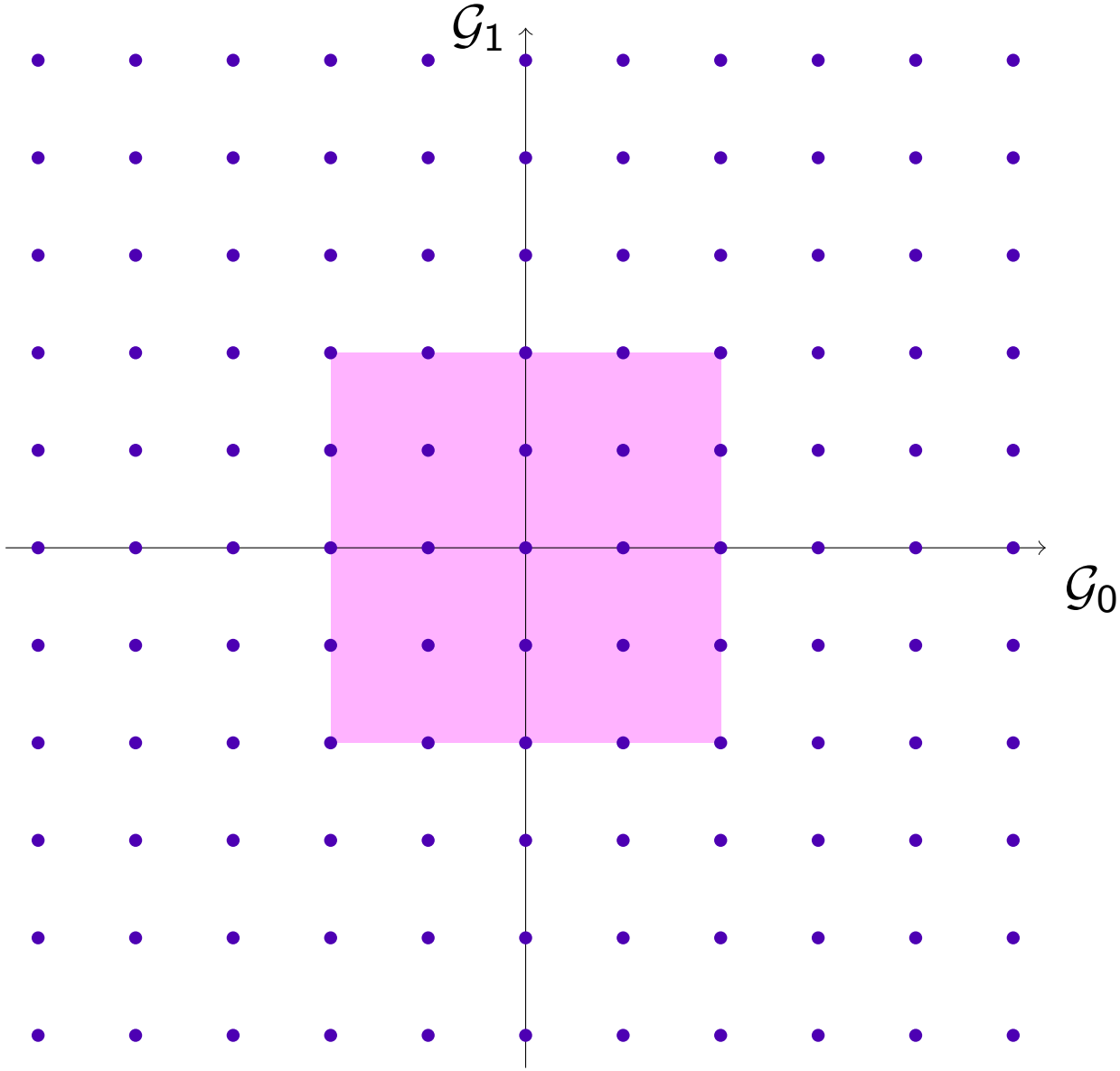
$$\mathcal{T} = (-u, \ldots, -u)\mathcal{G} .$$

Important: note that $\mathcal{T} \in \mathcal{L}$.

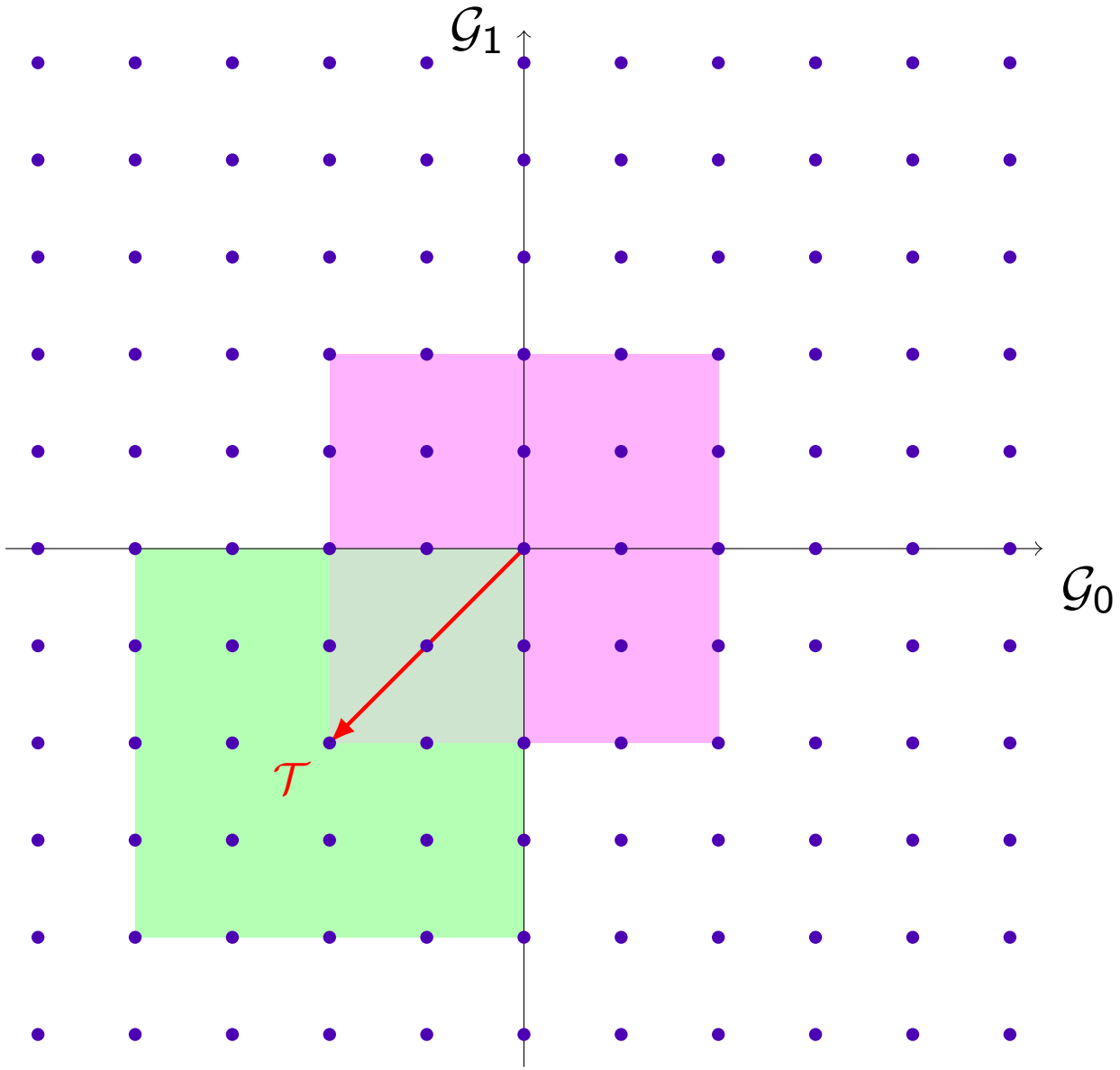
Consequence:

- $C + \mathcal{T} = \beta\mathcal{G}$, with $-2u \leqslant \beta_i \leqslant 0$.
- Thus, if $\phi \geqslant 2\boldsymbol{u}$, then **GMont-like**$(C + \mathcal{T}) \in \mathcal{D}_1$.

The translation vector: example for $\phi = 4$, with $u = 2$



The translation vector: example for $\phi = 4$, with $u = 2$



About the bounds

Note: For simplicity, the parameter δ for 'free' additions is not included. See https://eprint.iacr.org/2023/1231 for full formulas and details.

Old bounds on ρ and ϕ:

$$\rho \geqslant 2\|\mathcal{G}\|_1\,,$$

$$\phi \geqslant 2w\rho\,.$$

New bounds for reduction in $\mathcal{D}_1$, using $\mathcal{T}$:

$$\rho = \|\mathcal{G}\|_1 + 1\,,$$

$$\phi \geqslant 2u\,,$$

with $u = \lceil w\|\mathcal{G}\|_1^2\|\mathcal{G}^{-1}\|_1 \rceil$.

Equality test in the PMNS

Let $A, B \in \mathcal{B}$.

Goal:

Check if $A(\gamma) \equiv B(\gamma) \pmod{p}$, **without** conversion out of the PMNS.

Fundamental property:

Let $\boldsymbol{A} \in \boldsymbol{\mathcal{L}}$, such that: $A = \alpha\mathcal{G}$. So $\alpha \in \mathbb{Z}^n$.

If $\forall i \in \{0, ..., n-1\}$, $-\phi < \alpha_i \leqslant 0$, then:

$$\textbf{GMont-like}(A) = 0$$

Equality test in the PMNS

We assume that $\phi \geqslant 2u \geqslant 4$, with $u = \lceil w(\rho-1)^2 \|\mathcal{G}^{-1}\|_1 \rceil$.

A fact:

If $A, B \in \mathcal{B}$, then: $A - B = \nu\mathcal{G}$, with $\|\nu\|_\infty \leqslant 2 < \phi$.

So, the previous property applies.

The check:

$$A \equiv B \iff \textbf{GMont-like}((A-B)+\mathcal{T}) = 0$$

Remark:

- Works regardless of PMNS redundancy.
- Does not require that $\mathcal{L} = \mathcal{L}_\mathcal{B}$.

Implementations

Codes to generate PMNS, study its redundancy, perform equality test (with examples) and much more are available at:

https://github.com/arithPMNS/PMNS-and-redundancy

The associated GitHub account also contains repositories that provide C code generators from PMNS parameters.

Open problems

- PMNS and Side Channel attacks.
- Security proof of PMNS randomisation.
- Statistical analysis of the randomisation operations
- PMNS for PQC
- Improve software and hardware implementations
- ...

$\Rightarrow$ The answers during Wrach 2027 organized by the ANR MAERA project?