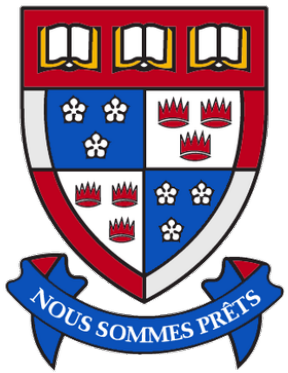
Computing over Algebraic Number Fields Modulo Primes Using Linear Algebra

Mahsa Ansari
Michael Monagan

Department of Mathematics,
Simon Fraser University,
Canada



Outline

Algebraic Numbers

Algebraic Number

A complex number $\alpha \in \mathbb{C}$ is called an **Algebraic Number** if there exists a non-zero polynomial $M(z) \in \mathbb{Q}[z]$ s.t. $M(\alpha) = 0$.
For instance, $\alpha = \sqrt{2}$ is an algebraic number since it is a root of $M(z) = 2z^2 - 4$.

Minimal Polynomial

If α is an algebraic number, there is a unique non-zero polynomial $M(z) \in \mathbb{Q}[z]$ of least degree such that

1) $M(\alpha) = 0$
2) $M(z)$ is monic
3) $M(z)$ is irreducible.

We call $M(z)$ the **Minimal Polynomial** of α.
For example, $M(z) = z^2 - 2$ is the minimal polynomial of $\alpha = \sqrt{2}$.

Algebraic Number Fields

Algebraic Number Field

Let $\alpha_1, \ldots, \alpha_n$ be algebraic numbers. The **Algebraic Number Field** $\mathbb{Q}(\alpha_1, \ldots, \alpha_n)$ is the smallest field containing $\mathbb{Q}$ and $\alpha_1, \ldots, \alpha_n$.

Question: How can we compute in $\mathbb{Q}(\alpha_1, \ldots, \alpha_n)$?

Algebraic Number Fields

Algebraic Number Field

Let $\alpha_1, \ldots, \alpha_n$ be algebraic numbers. The **Algebraic Number Field** $\mathbb{Q}(\alpha_1, \ldots, \alpha_n)$ is the smallest field containing $\mathbb{Q}$ and $\alpha_1, \ldots, \alpha_n$.

Question: How can we compute in $\mathbb{Q}(\alpha_1, \ldots, \alpha_n)$?

Answer: Let $L_0 = \mathbb{Q}$

$$L_i = L_{i-1}[z_i]/\langle M_i(z_i)\rangle \quad for\ 1 \leq i \leq n$$
$$L_n = L$$

where $M_i(z_i)$ is the minimal polynomial of α_i over L_{i-1}. Then

$$\mathbb{Q}(\alpha_1, \ldots, \alpha_n) \cong L.$$

Map the elements of $\mathbb{Q}(\alpha_1, \ldots, \alpha_n)$ to L and do the computation in L.

Notation:

We also write $L = \mathbb{Q}[z_1, \ldots, z_n]/\langle M_1, \ldots, M_n\rangle$.

- L can be specified as a $\mathbb{Q}-$vector space.
- Let $d_i = \deg(M_i(z_i), z_i)$ and $d = \prod_{i=1}^{n} d_i$. Then, $B_L = \{\prod_{i=1}^{n}(z_i)^{e_i} | \ 0 \leq e_i < d_i\}$ is a basis for L.
- The degree of L is $[L : \mathbb{Q}] =| \ B_L \ |= d$.

Example

Let $L = \mathbb{Q}[z_1, z_2]/\langle z_1^2 - 2, z_2^2 - 3\rangle$. Then $B_L = \{1, z_1, z_2, z_1 z_2\}$ is a basis for L and $[L : \mathbb{Q}] = 4$.

Primitive Element

Theorem

Let $\alpha_1, \ldots, \alpha_n$ be algebraic numbers. There exists an algebraic number γ s.t.

$$\mathbb{Q}(\alpha_1, \ldots, \alpha_n) \cong \mathbb{Q}(\gamma).$$

*We call γ a **Primitive Element**.*

Example

Let $\alpha_1 = \sqrt{2}$, $\alpha_2 = \sqrt{3}$. Then $\gamma = \sqrt{2} + \sqrt{3}$ is a primitive element of $\mathbb{Q}(\alpha_1, \alpha_2)$ and

$$\mathbb{Q}(\alpha_1, \alpha_2) \cong \mathbb{Q}(\gamma).$$

Question:

How can we speed up computation over $\mathbb{Q}(\alpha_1, \ldots, \alpha_n)$?

Question:

How can we speed up computation over $\mathbb{Q}(\alpha_1, \ldots, \alpha_n)$?

Answer:

1. First, use a modular homomorphism. Let p be a prime and $L_p = \mathbb{Z}_p[z_1, \ldots, z_n]/\langle m_1, \ldots, m_n \rangle$.
2. Then, use the **primitive element isomorphism**

$$\phi_\gamma : \mathbb{Q}(\alpha_1, \ldots, \alpha_n) \longrightarrow \mathbb{Q}(\gamma)$$

modulo p. Let $\overline{L}_p = \mathbb{Z}_p[z]/\langle M(z) \rangle$, where $M(z)$ is the minimal polynomial of γ modulo p. Then

$$\phi_\gamma : L_p \longrightarrow \overline{L}_p.$$

Why?

Computation in $\overline{L}_p$ is more efficient than computation in L_p.

The Number of Multiplications (L_p)

Let p be a prime. Then

$$\begin{aligned} L_0 &= \mathbb{Z}_p \\ L_1 &= L_0[z_1]/\langle m_1(z_1)\rangle \quad \text{s.t.} \quad d_1 = \deg(m_1, z_1) = 2 \\ L_2 &= L_1[z_2]/\langle m_2(z_2)\rangle \quad \text{s.t.} \quad d_2 = \deg(m_2, z_2) = 4. \end{aligned}$$

Let $L_p = L_2$ and let $A, B \in L_p$ s.t.

$$A = \sum_{i=0}^{3} a_i(z_1) z_2^i, \quad \text{where} \quad a_i(z_1) = a_{i0} + a_{i1} z_1 \in L_1, \quad \text{and}$$

$$B = \sum_{i=0}^{3} b_i(z_1) z_2^i, \quad \text{where} \quad b_i(z_1) = b_{i0} + b_{i1} z_1 \in L_1.$$

In the recursive dense representation, using the `recden` library in Maple, $A, B \in L_p$ are stored as nested lists:

$$A = [\underbrace{[a_{00}, a_{01}]}_{z_2^0}, \underbrace{[a_{10}, a_{11}]}_{z_2^1}, \underbrace{[a_{20}, a_{21}]}_{z_2^2}, \underbrace{[a_{30}, a_{31}]}_{z_2^3}]$$

$$B = [\underbrace{[b_{00}, b_{01}]}_{z_2^0}, \underbrace{[b_{10}, b_{11}]}_{z_2^1}, \underbrace{[b_{20}, b_{21}]}_{z_2^2}, \underbrace{[b_{30}, b_{31}]}_{z_2^3}].$$

$A \times B$ requires:

- $d_1^2 = 4$ $\times$ in $\mathbb{Z}_p$ before reduction modulo M_1. Reduction modulo M_1 costs $d_1(d_1 - 1) = 2$ additional $\times$ in $\mathbb{Z}_p$. Therefore, one $\times$ in L_1 requires at most $d_1^2 + d_1(d_1 - 1) = 6$ $\times$ in $\mathbb{Z}_p$.
- At the next level, $A \times B$ as polynomials in z_2 requires $d_2^2 = 16$ coefficient $\times$. Similarly, $\times$ and reduction at the z_2-level cost $d_2^2 + d_2(d_2 - 1) = 28$ $\times$ in L_1.
- Therefore, $A \times B$ requires $6 \cdot 28 = 168$ $\times$ in $\mathbb{Z}_p$.

The Number of Multiplications ($\overline{L}_p$)

Let $\overline{L}_p = \mathbb{Z}_p[z]/\langle M(z)\rangle \cong L_p$, where $\deg(M, z) = 8$. Let $A_\gamma, B_\gamma \in \overline{L}_p$:

$$A_\gamma = \sum_{i=0}^{7} a_i z^i, \qquad B_\gamma = \sum_{i=0}^{7} b_i z^i, \text{ with } a_i, b_i \in \mathbb{Z}_p.$$

The data representation of A_γ and B_γ in recden is presented as

$$A_\gamma = [\underbrace{a_0}_{z^0}, \underbrace{a_1}_{z^1}, \underbrace{a_2}_{z^2}, \underbrace{a_3}_{z^3}, \underbrace{a_4}_{z^4}, \underbrace{a_5}_{z^5}, \underbrace{a_6}_{z^6}, \underbrace{a_7}_{z^7}]$$
$$B_\gamma = [\underbrace{b_0}_{z^0}, \underbrace{b_1}_{z^1}, \underbrace{b_2}_{z^2}, \underbrace{b_3}_{z^3}, \underbrace{b_4}_{z^4}, \underbrace{b_5}_{z^5}, \underbrace{b_6}_{z^6}, \underbrace{b_7}_{z^7}].$$

$A_\gamma \times B_\gamma$ costs $d^2 = 8^2 = 64$ $\times$ in $\mathbb{Z}_p$. The unreduced product has degree at most $2d - 2$. Reducing it modulo M requires at most $d(d-1) = 8 \cdot 7 = 56$ $\times$ in $\mathbb{Z}_p$. Therefore, $A_\gamma \times B_\gamma$ requires at most $d^2 + d(d-1) = 64 + 56 = 120$ $\times$ in $\mathbb{Z}_p$.

The Number of Storage Allocation

- **# Storage allocations (L_p):** A $\times$ in L_p does $d_2^2 + d_2(d_2 - 1) = 28$ $\times$ in L_1. Each needs at least 2 pieces of storage, one for a product and one for a remainder. Also, it requires two pieces of storage: one for the product in L_2 and one remainder in L_2. In total, it requires 58 pieces of storage.
- **# Storage allocations ($\overline{L}_p$):** In comparison, $\times$ in $\overline{L}_p$ needs only 2 pieces of storage, one for the product of size $2d$ and one for the remainder of size d.

In summary, we have

Ring	#mults in $\mathbb{Z}_p$	#storage allocations
L_p	168	58
$\overline{L}_p$	120	2

Table: # mults and #storage

Outline

Finding a primitive element γ and its minimal polynomial $M(z)$

Let $\gamma = z_1 + C_1 z_2 + \ldots + C_{n-1} z_n$ be a candidate primitive element, where $0 \neq C_i \in \mathbb{Z}$ for $1 \leq i \leq n-1$.

Question: How can we verify whether γ is a primitive element of $\mathbb{Q}(\alpha_1, \ldots, \alpha_n)$?

Finding a primitive element γ and its minimal polynomial $M(z)$

Let $\gamma = z_1 + C_1 z_2 + \ldots + C_{n-1} z_n$ be a candidate primitive element, where $0 \neq C_i \in \mathbb{Z}$ for $1 \leq i \leq n-1$.

Question: How can we verify whether γ is a primitive element of $\mathbb{Q}(\alpha_1, \ldots, \alpha_n)$?

Answer: There are several methods that we can use:

- **Using Gröbner Bases**: Requires $O(nd^3)$ arithmetic operations in $\mathbb{Q}$.
- **Using Resultants**: Does not directly provide the isomorphism $\phi_\gamma : \mathbb{Q}(\alpha_1, \ldots, \alpha_n) \longrightarrow \mathbb{Q}(\gamma)$.
- **Using Linear Algebra**: Requires $O(d^3)$ arithmetic operations in $\mathbb{Q}$.

Finding a primitive element γ and its minimal polynomial $M(z)$

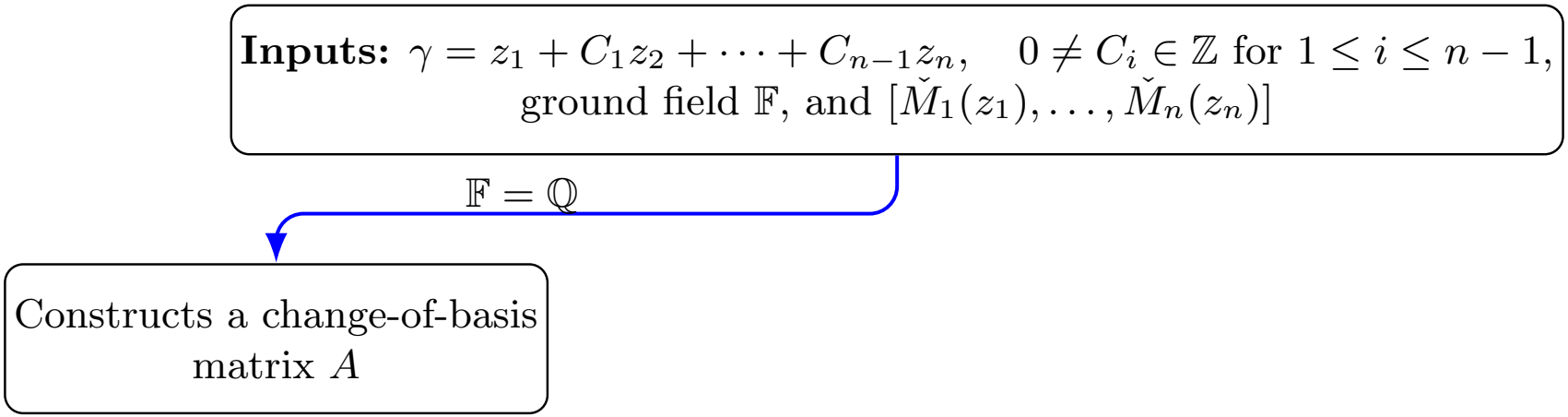
1. Let $\gamma = z_1 + C_1 z_2 + \ldots + C_{n-1} z_n$ be a candidate primitive element, where $0 \neq C_i \in \mathbb{Z}$ for $1 \leq i \leq n-1$.
2. Let $B_\gamma = \{1, \gamma, \gamma^2, \ldots, \gamma^{d-1}\}$ be a potential basis for $\mathbb{Q}(\gamma)$.
3. Let B_L be a basis for $L \cong \mathbb{Q}(\alpha_1, \ldots, \alpha_n)$.
4. We build the $d \times d$ change-of-basis matrix from B_γ to B_L

$$A = \left[\left[\begin{matrix} \vdots \\ 1 \\ \vdots \end{matrix} \right]_{B_L} \quad \ldots \quad \left[\begin{matrix} \vdots \\ \gamma^{d-1} \\ \vdots \end{matrix} \right]_{B_L} \right].$$

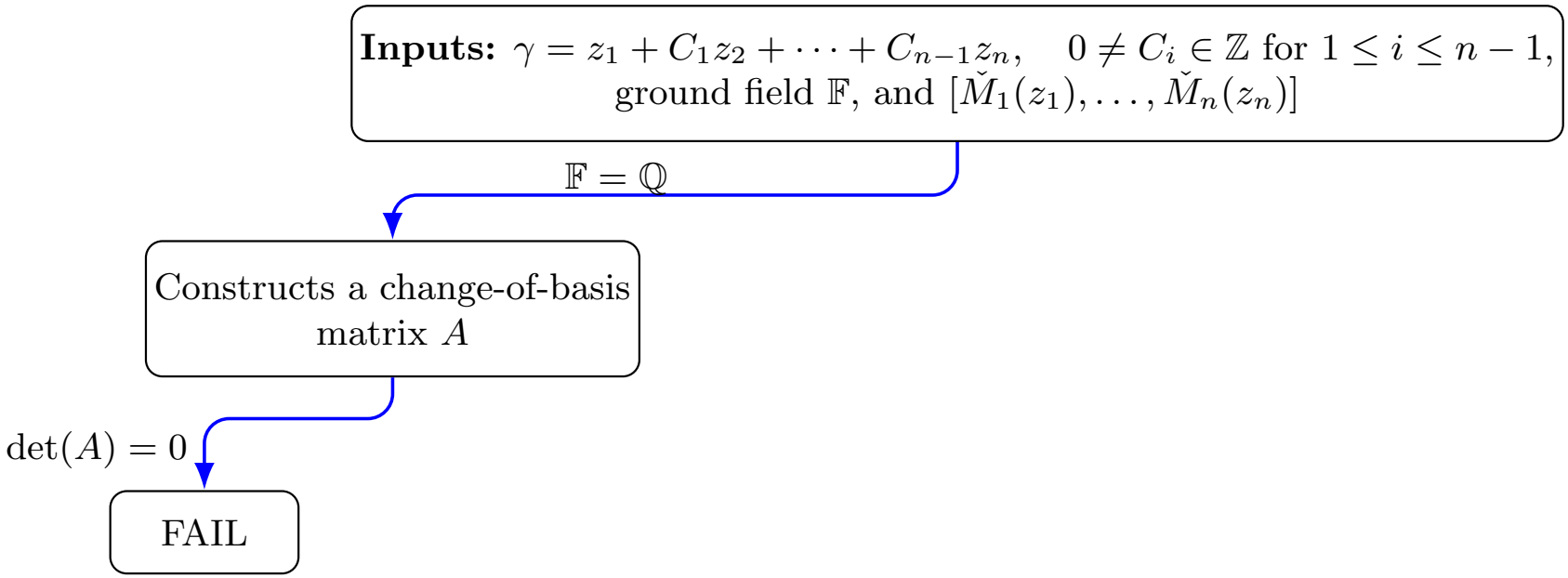
LAminpoly

Inputs: $\gamma = z_1 + C_1 z_2 + \cdots + C_{n-1} z_n, \quad 0 \neq C_i \in \mathbb{Z}$ for $1 \leq i \leq n-1$, ground field $\mathbb{F}$, and $[\check{M}_1(z_1), \ldots, \check{M}_n(z_n)]$

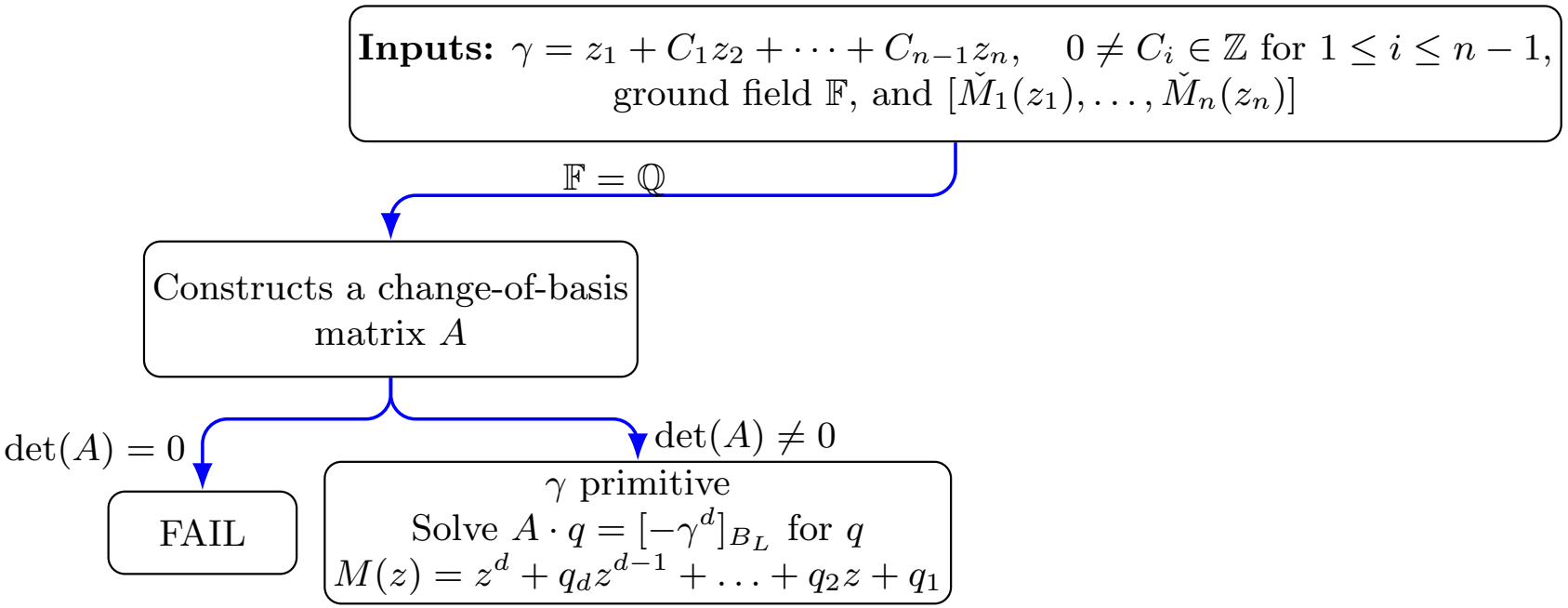
LAminpoly


Inputs: $\gamma = z_1 + C_1 z_2 + \cdots + C_{n-1} z_n, \quad 0 \neq C_i \in \mathbb{Z}$ for $1 \leq i \leq n-1$,
ground field $\mathbb{F}$, and $[\check{M}_1(z_1), \ldots, \check{M}_n(z_n)]$
$\mathbb{F} = \mathbb{Q}$
Constructs a change-of-basis matrix A

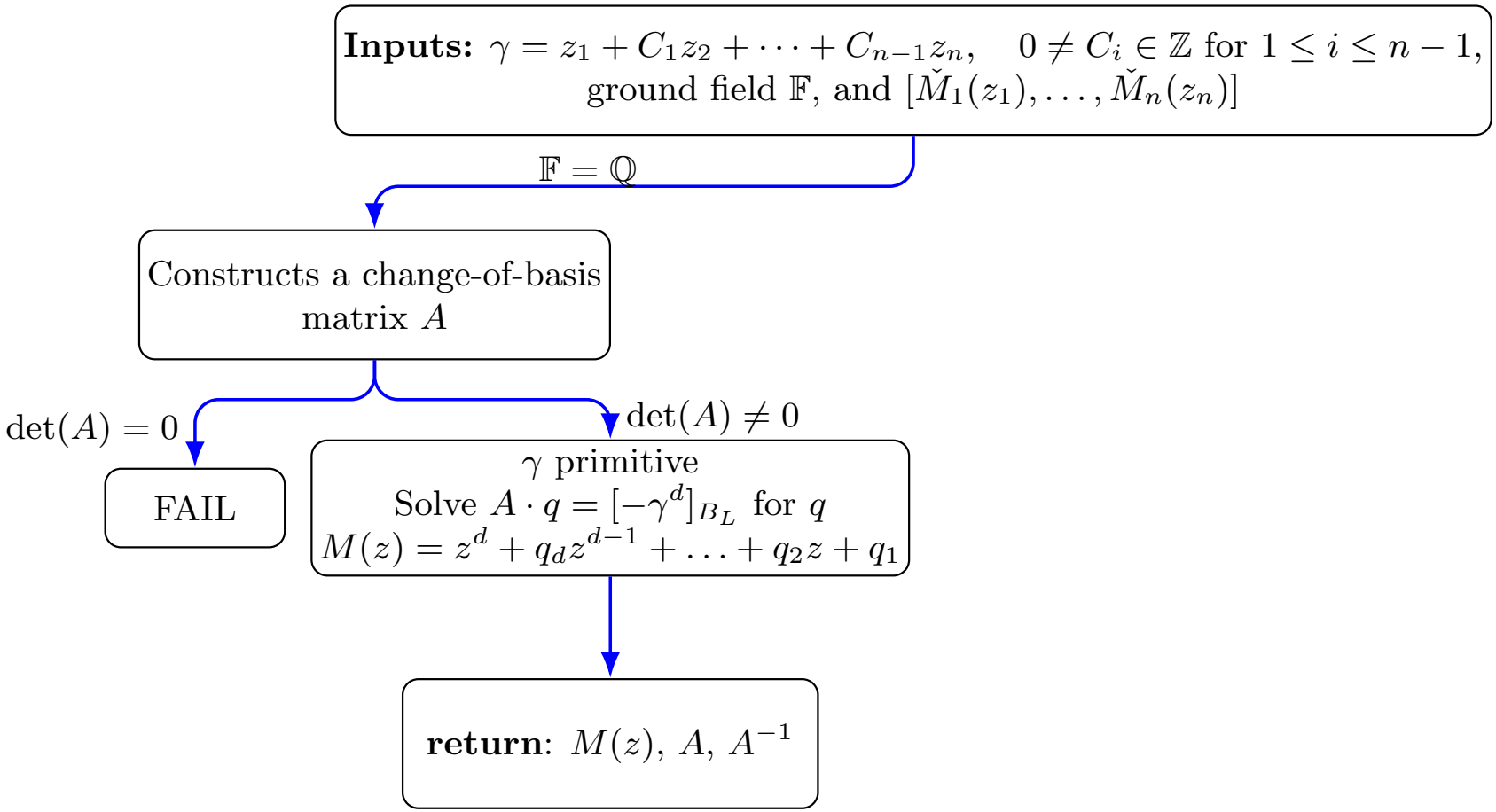
LAminpoly


Inputs: $\gamma = z_1 + C_1 z_2 + \cdots + C_{n-1} z_n, \quad 0 \neq C_i \in \mathbb{Z}$ for $1 \leq i \leq n-1$,
ground field $\mathbb{F}$, and $[\check{M}_1(z_1), \ldots, \check{M}_n(z_n)]$
$\mathbb{F} = \mathbb{Q}$
Constructs a change-of-basis matrix A
$\det(A) = 0$
FAIL

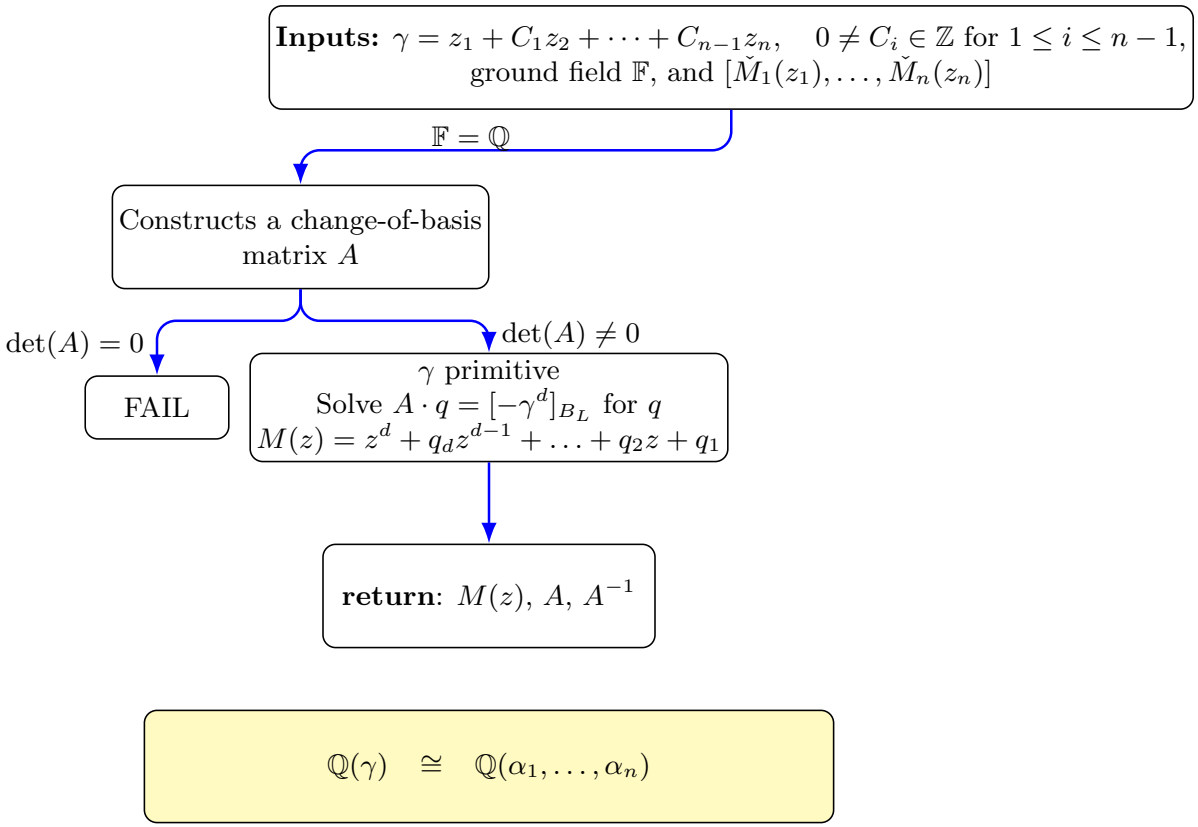
LAminpoly


Inputs: $\gamma = z_1 + C_1 z_2 + \cdots + C_{n-1} z_n, \quad 0 \neq C_i \in \mathbb{Z}$ for $1 \leq i \leq n-1$,
ground field $\mathbb{F}$, and $[\check{M}_1(z_1), \ldots, \check{M}_n(z_n)]$
$\mathbb{F} = \mathbb{Q}$
Constructs a change-of-basis matrix A
$\det(A) = 0$
$\det(A) \neq 0$
FAIL
γ primitive
Solve $A \cdot q = [-\gamma^d]_{B_L}$ for q
$M(z) = z^d + q_d z^{d-1} + \ldots + q_2 z + q_1$

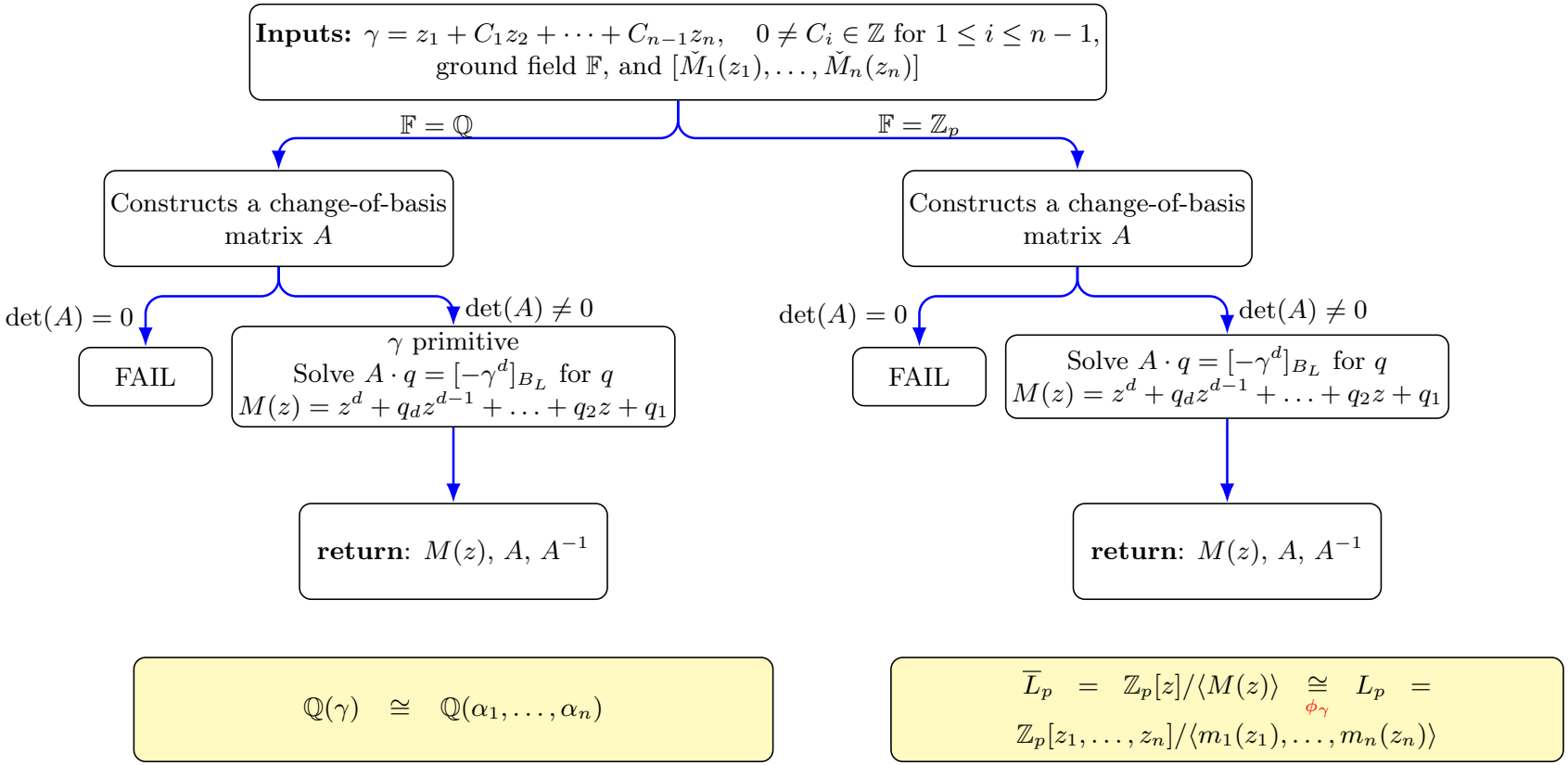
LAminpoly


Inputs: $\gamma = z_1 + C_1 z_2 + \cdots + C_{n-1} z_n, \quad 0 \neq C_i \in \mathbb{Z}$ for $1 \leq i \leq n-1$,
ground field $\mathbb{F}$, and $[\check{M}_1(z_1), \ldots, \check{M}_n(z_n)]$
$\mathbb{F} = \mathbb{Q}$
Constructs a change-of-basis matrix A
$\det(A) = 0$
$\det(A) \neq 0$
FAIL
γ primitive
Solve $A \cdot q = [-\gamma^d]_{B_L}$ for q
$M(z) = z^d + q_d z^{d-1} + \ldots + q_2 z + q_1$
return: $M(z)$, A, A^{-1}

LAminpoly


Inputs: $\gamma = z_1 + C_1 z_2 + \cdots + C_{n-1} z_n, \quad 0 \neq C_i \in \mathbb{Z}$ for $1 \leq i \leq n-1$, ground field $\mathbb{F}$, and $[\check{M}_1(z_1), \ldots, \check{M}_n(z_n)]$
$\mathbb{F} = \mathbb{Q}$
Constructs a change-of-basis matrix A
$\det(A) = 0$
$\det(A) \neq 0$
FAIL
γ primitive
Solve $A \cdot q = [-\gamma^d]_{B_L}$ for q
$M(z) = z^d + q_d z^{d-1} + \ldots + q_2 z + q_1$
return: $M(z)$, A, A^{-1}
$\mathbb{Q}(\gamma) \quad \cong \quad \mathbb{Q}(\alpha_1, \ldots, \alpha_n)$

LAminpoly



Remark

Running LAminpoly over $\mathbb{F} = \mathbb{Z}_p$:

- ✓ We avoid expression swell over $\mathbb{Q}$.
- ✗ It is likely that $m_i(z_i)$ is reducible over $\mathbb{Z}_p$ for some $1 \leq i \leq n$.

Example over $\mathbb{F} = \mathbb{Z}_p$

Let $L_7 = \mathbb{Z}_7[z_1, z_2]/\langle z_1^2 - 2, z_2^2 - 3\rangle$, where

$$z_1^2 - 2 \mod 7 = (z_1 - 3)(z_1 + 3) \in \mathbb{Z}_7[z_1].$$

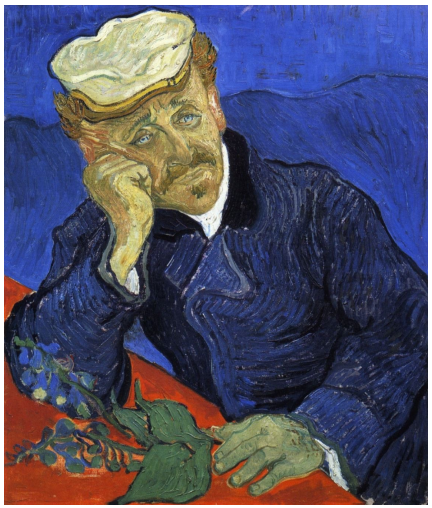
Let $B_L = \{1, z_2, z_1, z_1 z_2\}$ be a basis for L_7, $\gamma = z_1 + 2z_2$, and $B_\gamma = \{1, \gamma, \gamma^2, \gamma^3\}$. Then $A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 1 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 4 & 0 \end{bmatrix}$, with $\det(A) \mod 7 = 1 \neq 0$ so $A \cdot q = (-[\gamma^4]_{B_L}) \Rightarrow q = \begin{bmatrix} 2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ so we build $M(z) = z^4 + 2 = (z^2 + 6z + 4)(z^2 + z + 4) \in \mathbb{Z}_7[z]$ and the ring $\overline{L}_7 = \mathbb{Z}_7[z]/\langle z^4 + 2\rangle$ s.t.

$$\overline{L}_7 \underset{\phi_\gamma}{\cong} L_7.$$

Problem

Let $f_1, f_2 \in \mathbb{Q}(\alpha_1, \ldots, \alpha_n)[x_1, \ldots, x_k]$, where $n > 1$ and $k > 1$.

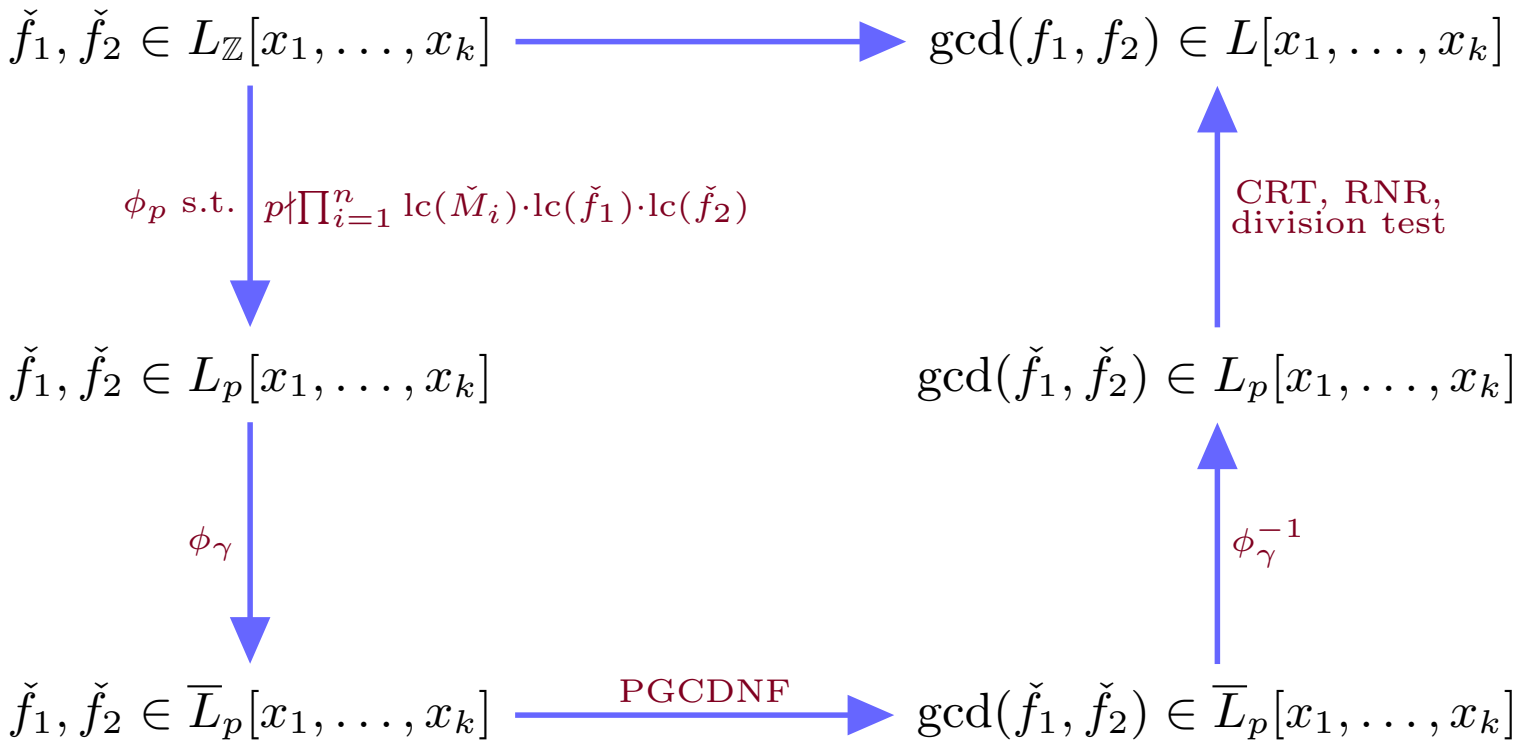
Problem: How to apply ϕ_γ to compute the monic $\gcd(f_1, f_2)$?



Outline

MGCDNF

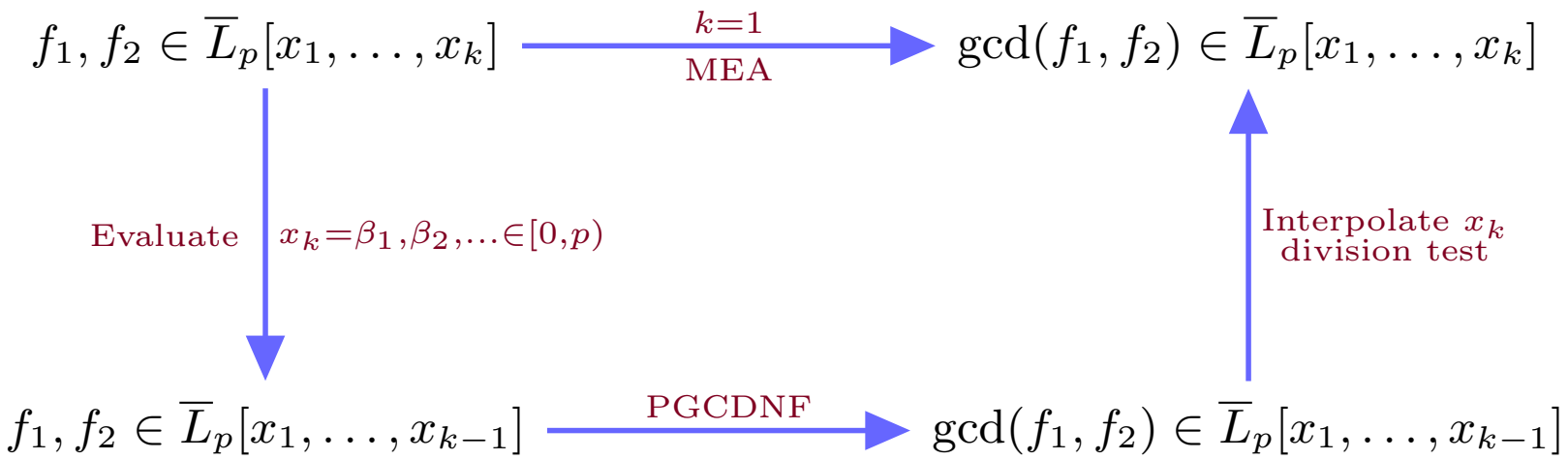
Let $f_1, f_2 \in L[x_1, \ldots, x_k]$, MGCDNF computes $g = monic \gcd(f_1, f_2)$.



$L = \mathbb{Q}[z_1, \ldots, z_n]/\langle M_1(z_1), \ldots, M_n(z_n)\rangle$
$L_{\mathbb{Z}} = \mathbb{Z}[z_1, \ldots, z_n]$
$L_p = \mathbb{Z}_p[z_1, \ldots, z_n]/\langle m_1(z_1), \ldots, m_n(z_n)\rangle$ s.t $m_i(z_i) = \check{M}_i(z_i) \mod p$
$\overline{L}_p = \mathbb{Z}_p[z]/\langle M(z)\rangle$

PGCDNF

Algorithm PGCDNF is a recursive algorithm to find the monic gcd of two polynomials f_1, f_2 in $\overline{L}_p[x_1, \ldots, x_k]$ where p is prime and $k \geq 2$.



Remark

MEA fails over $\overline{L}_p = \mathbb{Z}_p[z]/\langle M(z) \rangle$ if a zero divisor is encountered.

Lc-bad, Det-bad, Zero-divisor, and Unlucky

Question: Does every choice of prime or evaluation point lead to a successful reconstruction of the GCD?

Lc-bad, Det-bad, Zero-divisor, and Unlucky

Question: Does every choice of prime or evaluation point lead to a successful reconstruction of the GCD?
Answer: No!!!

- Lc-bad pairs (Are ruled out in advance)
- Det-bad pairs (Restart)
- Zero divisor pairs (Restart)
- Unlucky primes (Restart)
- Unlucky evaluation points (Restart)

Det-bad pairs

Definition(Det-bad pair)

Let p be a prime s.t. $p \nmid \prod_{i=1}^{n} \mathrm{lc}(\check{M}_i, z_i)$, $C = (C_1, \ldots, C_{n-1}) \in [1, p)^{n-1}$, and $\gamma = z_1 + \sum_{i=2}^{n} C_{i-1} z_i$. Let A be the change-of-basis matrix obtained from the LAminpoly over $\mathbb{F} = \mathbb{Q}$. The ordered pair (p, C) is called a **det-bad** pair if $\det(A) = 0$ or $p \mid \mathrm{numerator}(\det(A))$.

Example

Let $L = \mathbb{Q}[z_1, z_2]/\langle z_1^2 - 2, z_2^2 - 3\rangle$ and $B_L = \{1, z_1, z_2, z_1 z_2\}$. Let $C = (C_1) = (2)$, $\gamma = z_1 + C_1 z_2 = z_1 + 2z_2$, and $B_\gamma = \{1, \gamma, \gamma^2, \gamma^3\}$. Then the change-of-basis-matrix from B_γ to B_L is

$$A = \begin{bmatrix} 1 & 0 & 14 & 0 \\ 0 & 1 & 0 & 38 \\ 0 & 2 & 0 & 36 \\ 0 & 0 & 4 & 0 \end{bmatrix}, \text{ with } \det(A) = 160 = 2^5 \times 5, \text{ so the pairs}$$

$(p, C) = (2, (2))$ and $(p, C) = (5, (2))$ are det-bad pairs.

Benchmark 1: Degree of the field and number of extensions

Table: **Timings in CPU seconds for computing $\gcd(f_1, f_2)$ over an algebraic number field of degree d with n extensions, using recden.**

d	n	N	NMEA	D_f	D_g	MGCDNF 1			MGCDNF 2	
						time	LAMP	PGCDNF	time	PGCDNF
4	2	4	128	32	16	2.422	0	2.313	10.547	10.516
8	3	4	132	32	16	3.047	0	2.875	48.453	48.297
16	4	4	136	32	16	3.703	0.126	3.408	133.391	133.313
24	3	4	136	32	16	5.360	0.265	4.749	272.093	271.999
32	5	4	136	32	16	5.375	0.829	4.346	521.438	521.376
64	6	4	136	32	16	10.000	3.859	5.797	> 1000	(-)
128	7	4	136	32	16	32.422	20.951	10.781	> 1000	(-)
256	8	5	170	32	16	154.454	126.547	26.468	> 1000	(-)
512	9	4	136	32	16	542.906	491.140	48.890	> 1000	(-)
1024	10	(-)	(-)	32	(-)	> 1000	(-)	(-)	> 1000	(-)

Benchmark 2: Degree of the GCD (MGCDNF)

Table: **Timings in CPU seconds for computing $\gcd(f_1, f_2)$ over an algebraic number field of degree $d = 64$ with $n = 6$ extensions, using recden.**

d	n	N	$NMEA$	D_f	D_g	MGCDNF 1			MGCDNF 2	
						time	LAMP	PGCDNF	time	PGCDNF
64	6	3	18	4	2	3.265	3.032	0.123	7.203	7.188
64	6	4	40	8	4	4.984	3.874	0.796	51.984	51.937
64	6	4	56	12	6	5.250	3.764	1.188	62.328	62.281
64	6	4	72	16	8	5.469	3.984	1.283	274.359	274.297
64	6	4	88	20	10	6.766	3.922	2.502	321.250	321.173
64	6	4	104	24	12	7.359	4.108	2.938	542.750	542.579
64	6	4	116	28	14	9.609	4.249	4.876	> 1000	(-)
64	6	5	170	32	16	14.172	4.908	8.860	> 1000	(-)

Complexity of MGCDNF

Let $f_1, f_2 \in L[x_1, \cdots, x_k]$ and g be the monic $\gcd(f_1, f_2)$. The quantities involved in the running time of the MGCDNF algorithm are as follows:

- N is the number of good primes to reconstruct the monic g,
- $d = [L : \mathbb{Q}]$ and $D = \max_{i=1}^{k}(\deg(f_1, x_i), \deg(f_2, x_i))$,
- $H_M = \log_2 \max_{i=1}^{n} H(\check{M}_i)$, and $C = \log_2 \max(H(\check{f}_1), H(\check{f}_2))$.

Theorem

The expected time complexity of the MGCDNF algorithm is

$$\mathcal{O}\Big(\underbrace{Nd(nH_M + CD^k)}_{\text{reducing mod } p} + \underbrace{(Nd^3 + Nd^2D^k)}_{\phi_\gamma} + \underbrace{Nkd^2D^{k+1}}_{PGCDNF} + \underbrace{N^3dD^k}_{CRT/RNR} + \underbrace{(d^2D^2 + dD^k + dCD^k)}_{\text{Division test}}\Big).$$

1. M. Ansari and M. Monagan,
"Computing GCDs of multivariate polynomials over algebraic number fields presented with multiple extensions,"
Computer Algebra in Scientific Computing, LNCS 14139, pp. 1–20, Springer, 2023.

2. M. Ansari and M. Monagan,
"A failure probability analysis of a modular algorithm to compute the monic GCD of multivariate polynomials over algebraic number fields $\mathbb{Q}(\alpha_1, \ldots, \alpha_n)$,"
Computer Algebra in Scientific Computing, LNCS 16235, pp. 1–26, Springer, 2025.

Thank you!