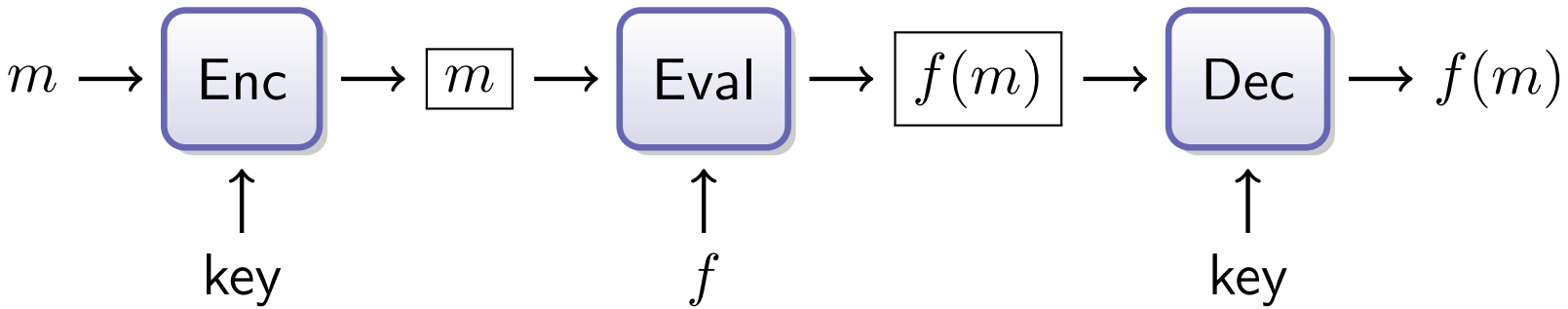
New Directions in Fully Homomorphic Encryption

Chris Peikert

University of Michigan

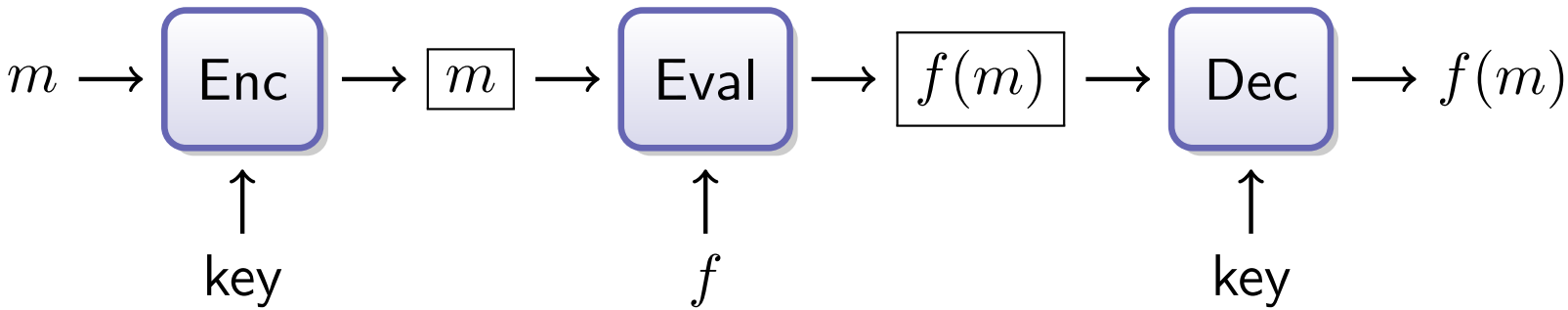
SAC Stafford Tavares Lecture, 26 Aug 2026

Fully Homomorphic Encryption [RAD'78,Gentry'09,...]



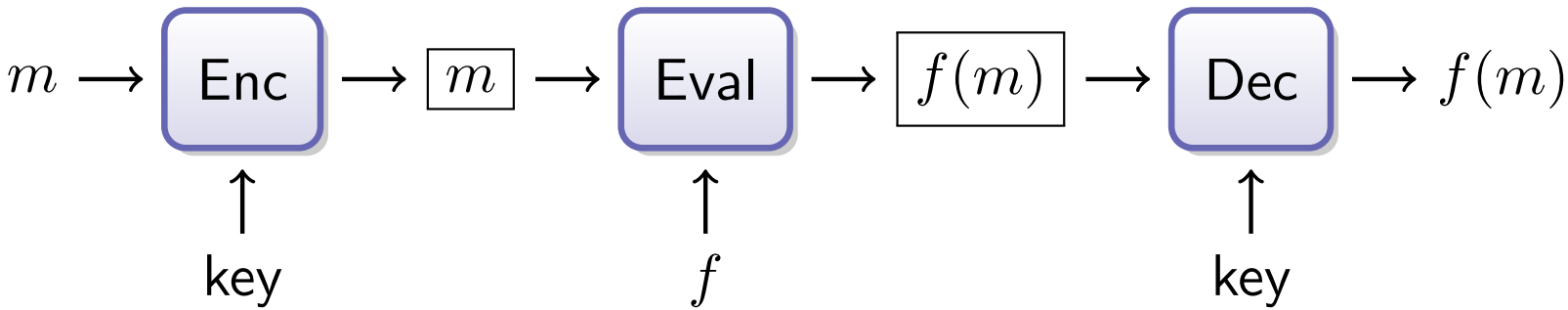
- Compute on data without seeing it

Fully Homomorphic Encryption [RAD'78,Gentry'09,...]



- Compute on data without seeing it
- For exact arithmetic, plaintexts live in a finite ring like $\mathbb{Z}_p$

Fully Homomorphic Encryption [RAD'78,Gentry'09,...]



- Compute on data without seeing it
- For exact arithmetic, plaintexts live in a finite ring like $\mathbb{Z}_p$
- Modern schemes are highly capable, but have two major pain points

Two Pain Points for Exact-Arithmetic FHE

Pain #1: **High-precision plaintexts**

Applications often have huge plaintext moduli, e.g., $p = 2^{64}$

Two Pain Points for Exact-Arithmetic FHE

Pain #1: **High-precision plaintexts**

Applications often have huge plaintext moduli, e.g., $p = 2^{64}$

- Classic schemes: noise growth $\gg p$ per multiplication

  Hence for security, sizes & runtimes at least quadratic in $\log p$

Two Pain Points for Exact-Arithmetic FHE

Pain #1: **High-precision plaintexts**

Applications often have huge plaintext moduli, e.g., $p = 2^{64}$

- Classic schemes: noise growth $\gg p$ per multiplication

  Hence for security, sizes & runtimes at least quadratic in $\log p$

- Recent work gets quasi-linear in $\log p$, but only for rare plaintext moduli of special forms, coupled to certain rings [GC'14,CLPX'18,GV'25,BFG+'26]

Two Pain Points for Exact-Arithmetic FHE

Pain #1: **High-precision plaintexts**

Applications often have huge plaintext moduli, e.g., $p = 2^{64}$

- Classic schemes: noise growth $\gg p$ per multiplication

  Hence for security, sizes & runtimes at least quadratic in $\log p$

- Recent work gets quasi-linear in $\log p$, but only for rare plaintext moduli of special forms, coupled to certain rings [GC'14,CLPX'18,GV'25,BFG+'26]

Pain #2: **Inflexible plaintext packing and SIMD**

Prior FHE work has focused almost entirely on cyclotomic fields/rings:

Two Pain Points for Exact-Arithmetic FHE

Pain #1: **High-precision plaintexts**

Applications often have huge plaintext moduli, e.g., $p = 2^{64}$

- Classic schemes: noise growth $\gg p$ per multiplication

  Hence for security, sizes & runtimes at least quadratic in $\log p$

- Recent work gets quasi-linear in $\log p$, but only for rare plaintext moduli of special forms, coupled to certain rings [GC'14,CLPX'18,GV'25,BFG+'26]

Pain #2: **Inflexible plaintext packing and SIMD**

Prior FHE work has focused almost entirely on cyclotomic fields/rings:

- ✔ computationally fast and geometrically nice

Two Pain Points for Exact-Arithmetic FHE

Pain #1: **High-precision plaintexts**

Applications often have huge plaintext moduli, e.g., $p = 2^{64}$

- Classic schemes: noise growth $\gg p$ per multiplication

  Hence for security, sizes & runtimes at least quadratic in $\log p$

- Recent work gets quasi-linear in $\log p$, but only for rare plaintext moduli of special forms, coupled to certain rings [GC'14,CLPX'18,GV'25,BFG+'26]

Pain #2: **Inflexible plaintext packing and SIMD**

Prior FHE work has focused almost entirely on cyclotomic fields/rings:

- ✔ computationally fast and geometrically nice
- ✗ difficult to get desired SIMD "slot type" at the desired security level
  - ⋆ Homomorphic AES was induced to use $\mathbb{F}_{2^{24}}$-slots: a $3\times$ capacity loss versus the natural $\mathbb{F}_{2^8}$ [GHS'12c]

Two Pain Points for Exact-Arithmetic FHE

Pain #1: **High-precision plaintexts**

Applications often have huge plaintext moduli, e.g., $p = 2^{64}$

- Classic schemes: noise growth $\gg p$ per multiplication

  Hence for security, sizes & runtimes at least quadratic in $\log p$

- Recent work gets quasi-linear in $\log p$, but only for rare plaintext moduli of special forms, coupled to certain rings [GC'14,CLPX'18,GV'25,BFG+'26]

Pain #2: **Inflexible plaintext packing and SIMD**

Prior FHE work has focused almost entirely on cyclotomic fields/rings:

- ✔ computationally fast and geometrically nice
- ✗ difficult to get desired SIMD "slot type" at the desired security level
  - ⋆ Homomorphic AES was induced to use $\mathbb{F}_{2^{24}}$-slots: a $3\times$ capacity loss versus the natural $\mathbb{F}_{2^8}$ [GHS'12c]
- ✗ difficult to get desired slot-tensor "shape" and data movement

Two New Directions

High precision via decomposition [Peikert–Zarchy–Zyskind, CRYPTO'26]

- Simple, generic technique for arbitrary modulus and ring (or no ring!)
- Parameters can match classic FHE for exponentially smaller modulus
- Sizes, runtimes can be quasi-linear in $\log p$
- Fully compatible with prior rings, packing/SIMD, and bootstrapping

Two New Directions

High precision via decomposition [Peikert–Zarchy–Zyskind, CRYPTO'26]

- Simple, generic technique for arbitrary modulus and ring (or no ring!)
- Parameters can match classic FHE for exponentially smaller modulus
- Sizes, runtimes can be quasi-linear in $\log p$
- Fully compatible with prior rings, packing/SIMD, and bootstrapping

Flexible SIMD packing via cyclotomic subfields [Peikert–Pepin, TCC'25]

- Get any desired slot type, w/ many tensor shapes and security levels
- Retain fast arithmetic, homomorphic linear algebra, and packed bootstrapping

Direction I: High-Precision Plaintexts

Decompose plaintexts, but not the computation

(Peikert–Zarchy–Zyskind ePrint 2025/2321)

Why High Precision Is Expensive

- Example apps: encrypted 64-bit ALU, large finite-field arithmetic

Why High Precision Is Expensive

- Example apps: encrypted 64-bit ALU, large finite-field arithmetic
- In BGV/BFV, each multiplication enlarges the error rate by at least p

Why High Precision Is Expensive

- Example apps: encrypted 64-bit ALU, large finite-field arithmetic
- In BGV/BFV, each multiplication enlarges the error rate by at least p
- So depth-h computation needs initial error rate $\alpha < p^{-h}$.

Why High Precision Is Expensive

- Example apps: encrypted 64-bit ALU, large finite-field arithmetic
- In BGV/BFV, each multiplication enlarges the error rate by at least p
- So depth-h computation needs initial error rate $\alpha < p^{-h}$.
- Security requires LWE dimension and integer precision $\Omega(\log(1/\alpha))$.

Why High Precision Is Expensive

- Example apps: encrypted 64-bit ALU, large finite-field arithmetic
- In BGV/BFV, each multiplication enlarges the error rate by at least p
- So depth-h computation needs initial error rate $\alpha < p^{-h}$.
- Security requires LWE dimension and integer precision $\Omega(\log(1/\alpha))$.

Resulting cost: $\Omega(h^2 \log^2 p)$ ciphertext sizes, runtimes

Why High Precision Is Expensive

- Example apps: encrypted 64-bit ALU, large finite-field arithmetic
- In BGV/BFV, each multiplication enlarges the error rate by at least p
- So depth-h computation needs initial error rate $\alpha < p^{-h}$.
- Security requires LWE dimension and integer precision $\Omega(\log(1/\alpha))$.

Resulting cost: $\Omega(h^2 \log^2 p)$ ciphertext sizes, runtimes

For $p \geq 2^{64}$, this is a substantial concrete cost!

A Fragmented Landscape

[GC'14,CLPX'18,GV'25,BK'25,K'25,K'26,CPL'26,BFGGMS'26,...]

Approach	**Main advantage**	**Main compromise(s)**
Bitwise TFHE	Arbitrary p	$\Omega(\log^2 p)$ mult, slow add
Special modulus ideals	Simple, fast	Rare moduli, coupled rings
CKKS-based methods	Arbitrary p	Complex, mult $\supseteq$ bootstr
Special non-cyclo ring	Fast for $p = 2^k$	no SIMD, slow bootstrap

A Fragmented Landscape

[GC'14,CLPX'18,GV'25,BK'25,K'25,K'26,CPL'26,BFGGMS'26,...]

Approach	**Main advantage**	**Main compromise(s)**
Bitwise TFHE	Arbitrary p	$\Omega(\log^2 p)$ mult, slow add
Special modulus ideals	Simple, fast	Rare moduli, coupled rings
CKKS-based methods	Arbitrary p	Complex, mult $\supseteq$ bootstr
Special non-cyclo ring	Fast for $p = 2^k$	no SIMD, slow bootstrap

Our Contribution

- Support arbitrary p and ring (or no ring!), with fast ordinary homomorphic arithmetic, packing/SIMD, and bootstrapping

A Fragmented Landscape

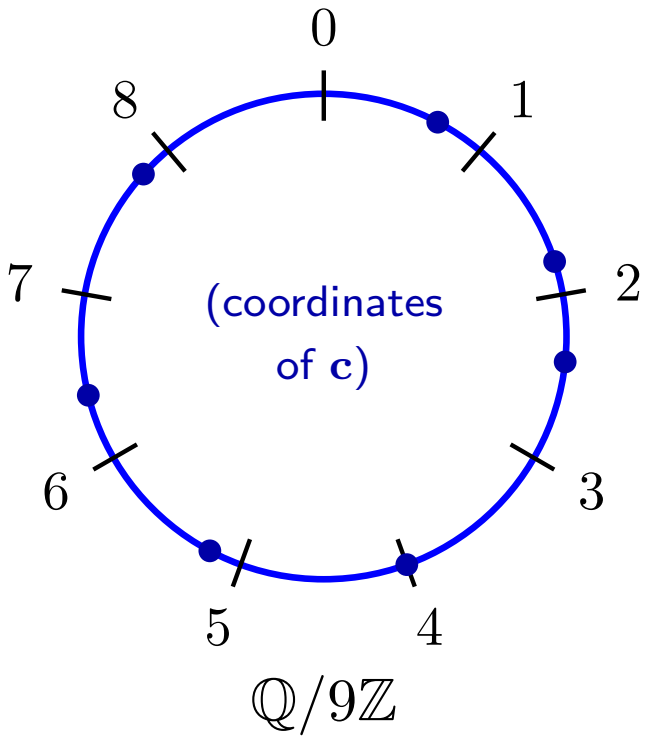
[GC'14,CLPX'18,GV'25,BK'25,K'25,K'26,CPL'26,BFGGMS'26,...]

Approach	**Main advantage**	**Main compromise(s)**
Bitwise TFHE	Arbitrary p	$\Omega(\log^2 p)$ mult, slow add
Special modulus ideals	Simple, fast	Rare moduli, coupled rings
CKKS-based methods	Arbitrary p	Complex, mult $\supseteq$ bootstr
Special non-cyclo ring	Fast for $p = 2^k$	no SIMD, slow bootstrap

Our Contribution

- Support arbitrary p and ring (or no ring!), with fast ordinary homomorphic arithmetic, packing/SIMD, and bootstrapping
- **Key Idea:** Replace $\mathbb{Z}_p$ with an isomorphic "higher-dimensional" presentation having much smaller representatives

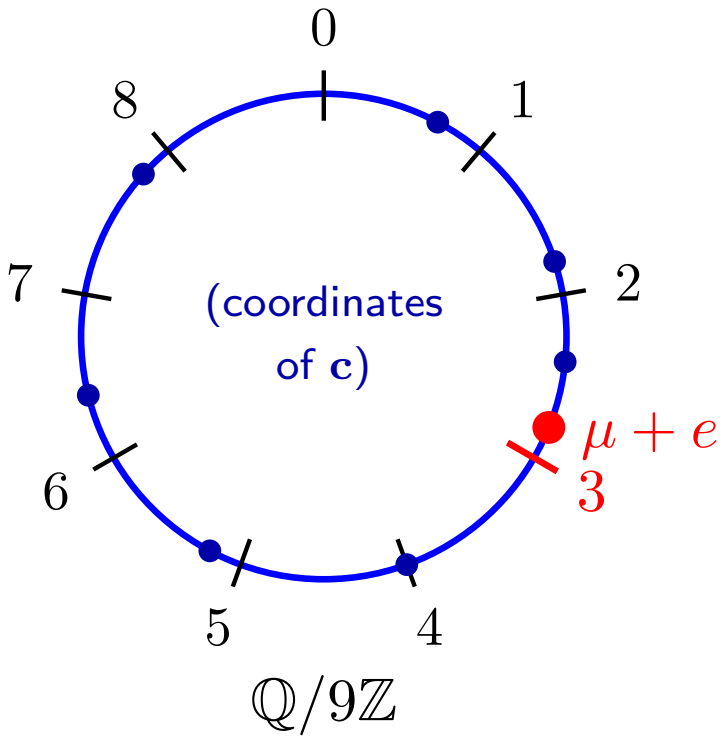
Warmup: Simplified BFV at the Right Scale



A ciphertext encrypting $\mu \in \mathbb{Z}_p$ is a (typically affine-linear) polynomial

$$\mathbf{c}(\vec{S}) \in (\mathbb{Q}/p\mathbb{Z})[\vec{S}]$$

Warmup: Simplified BFV at the Right Scale



A ciphertext encrypting $\mu \in \mathbb{Z}_p$ is a (typically affine-linear) polynomial

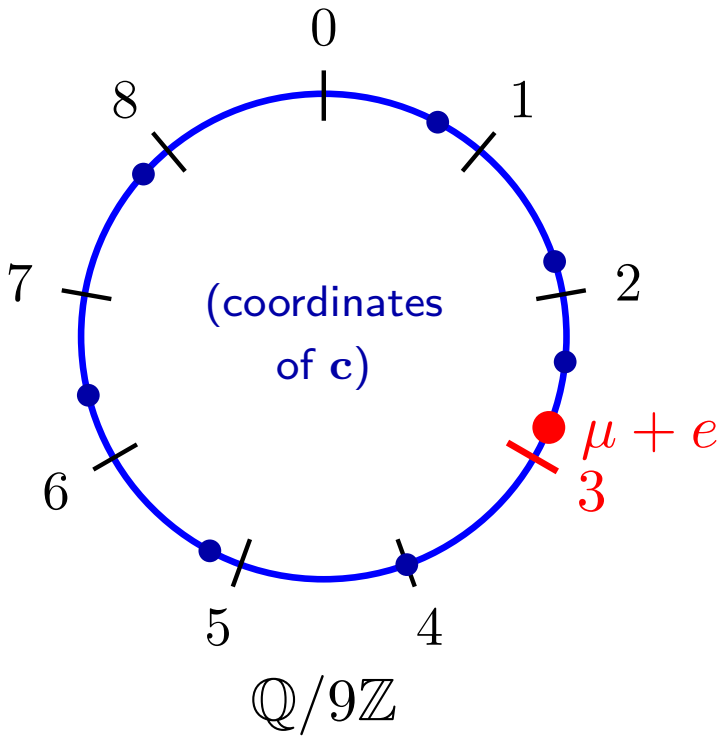
$$\mathbf{c}(\vec{S}) \in (\mathbb{Q}/p\mathbb{Z})[\vec{S}]$$

such that

$$\mathbf{c}(\mathbf{s}) = \mu + e \approx \mu \pmod{p\mathbb{Z}}, \quad |e| \ll \tfrac{1}{2}.$$

Evaluate at the short secret $\mathbf{s} \in \mathbb{Z}^n$, then round to the nearest integer modulo p.

Warmup: Simplified BFV at the Right Scale



A ciphertext encrypting $\mu \in \mathbb{Z}_p$ is a (typically affine-linear) polynomial

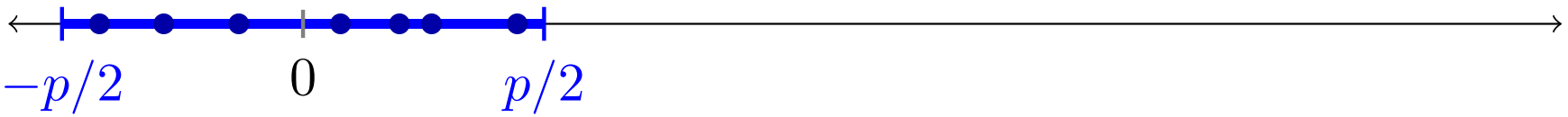
$$\mathbf{c}(\vec{S}) \in (\mathbb{Q}/p\mathbb{Z})[\vec{S}]$$

such that

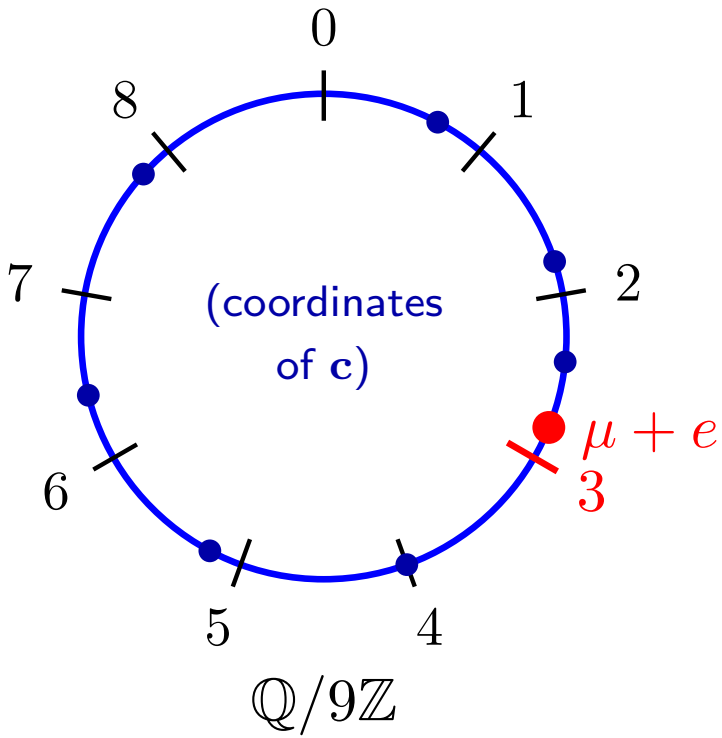
$$\mathbf{c}(\mathbf{s}) = \mu + e \approx \mu \pmod{p\mathbb{Z}}, \quad |e| \ll \tfrac{1}{2}.$$

Evaluate at the short secret $\mathbf{s} \in \mathbb{Z}^n$, then round to the nearest integer modulo p.

lifted ciphertext coordinates $\widehat{c_i} \in [-p/2, p/2)$:



Warmup: Simplified BFV at the Right Scale



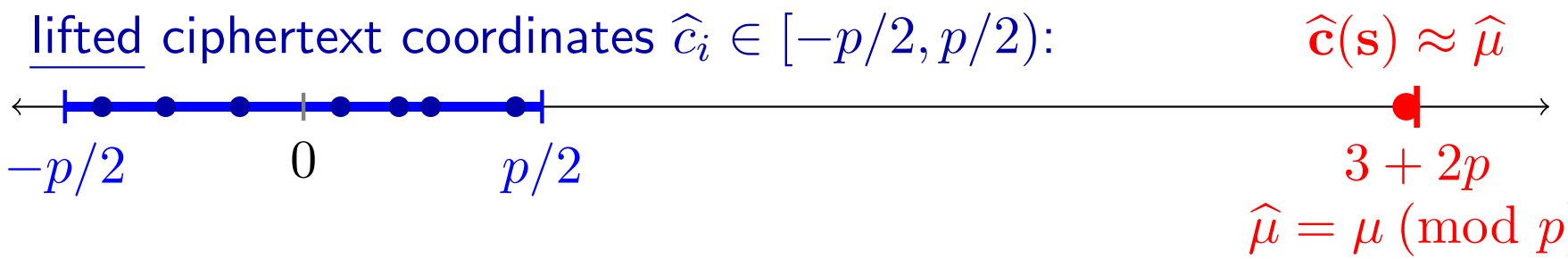
A ciphertext encrypting $\mu \in \mathbb{Z}_p$ is a (typically affine-linear) polynomial

$$\mathbf{c}(\vec{S}) \in (\mathbb{Q}/p\mathbb{Z})[\vec{S}]$$

such that

$$\mathbf{c}(\mathbf{s}) = \mu + e \approx \mu \pmod{p\mathbb{Z}}, \quad |e| \ll \tfrac{1}{2}.$$

Evaluate at the short secret $\mathbf{s} \in \mathbb{Z}^n$, then round to the nearest integer modulo p.



Warmup: BFV Homomorphic Operations

Addition: trivial

Just add the ciphertext polynomials:

$$\mathbf{c}_+(\vec{S}) := \mathbf{c}(\vec{S}) + \mathbf{c}'(\vec{S}) \in (\mathbb{Q}/p\mathbb{Z})[\vec{S}], \quad \text{so} \quad \mathbf{c}_+(\mathbf{s}) \approx \mu + \mu' \pmod{p\mathbb{Z}}.$$

Warmup: BFV Homomorphic Operations

Addition: trivial

Just add the ciphertext polynomials:

$$\mathbf{c}_+(\vec{S}) := \mathbf{c}(\vec{S}) + \mathbf{c}'(\vec{S}) \in (\mathbb{Q}/p\mathbb{Z})[\vec{S}], \quad \text{so} \quad \mathbf{c}_+(\mathbf{s}) \approx \mu + \mu' \pmod{p\mathbb{Z}}.$$

Multiplication: lift, then multiply

Lift to $\widehat{\mathbf{c}}(\vec{S}), \widehat{\mathbf{c}}'(\vec{S}) \in \mathbb{Q}[\vec{S}]$, where $\widehat{\mathbf{c}}(\mathbf{s}) = \widehat{\mu} + e$ and $\widehat{\mathbf{c}}'(\mathbf{s}) = \widehat{\mu}' + e'$, then set

$$\widehat{\mathbf{c}}_\times(\vec{S}) := \widehat{\mathbf{c}}(\vec{S}) \cdot \widehat{\mathbf{c}}'(\vec{S}).$$

Warmup: BFV Homomorphic Operations

Addition: trivial

Just add the ciphertext polynomials:

$$\mathbf{c}_+(\vec{S}) := \mathbf{c}(\vec{S}) + \mathbf{c}'(\vec{S}) \in (\mathbb{Q}/p\mathbb{Z})[\vec{S}], \quad \text{so} \quad \mathbf{c}_+(\mathbf{s}) \approx \mu + \mu' \pmod{p\mathbb{Z}}.$$

Multiplication: lift, then multiply

Lift to $\widehat{\mathbf{c}}(\vec{S}), \widehat{\mathbf{c}}'(\vec{S}) \in \mathbb{Q}[\vec{S}]$, where $\widehat{\mathbf{c}}(\mathbf{s}) = \widehat{\mu} + e$ and $\widehat{\mathbf{c}}'(\mathbf{s}) = \widehat{\mu}' + e'$, then set

$$\widehat{\mathbf{c}}_\times(\vec{S}) := \widehat{\mathbf{c}}(\vec{S}) \cdot \widehat{\mathbf{c}}'(\vec{S}).$$

Correct:
$$\begin{aligned}\widehat{\mathbf{c}}_\times(\mathbf{s}) &= (\widehat{\mu} + e) \cdot \left(\widehat{\mu}' + e'\right) \\ &= \widehat{\mu} \cdot \widehat{\mu}' + (e \cdot \widehat{\mu}' + \widehat{\mu} \cdot e' + e \cdot e') \\ &\approx \widehat{\mu} \cdot \widehat{\mu}' = \mu \cdot \mu' \pmod{p\mathbb{Z}}.\end{aligned}$$

Warmup: BFV Homomorphic Operations

Addition: trivial

Just add the ciphertext polynomials:

$$\mathbf{c}_+(\vec{S}) := \mathbf{c}(\vec{S}) + \mathbf{c}'(\vec{S}) \in (\mathbb{Q}/p\mathbb{Z})[\vec{S}], \quad \text{so} \quad \mathbf{c}_+(\mathbf{s}) \approx \mu + \mu' \pmod{p\mathbb{Z}}.$$

Multiplication: lift, then multiply

Lift to $\widehat{\mathbf{c}}(\vec{S}), \widehat{\mathbf{c}}'(\vec{S}) \in \mathbb{Q}[\vec{S}]$, where $\widehat{\mathbf{c}}(\mathbf{s}) = \widehat{\mu} + e$ and $\widehat{\mathbf{c}}'(\mathbf{s}) = \widehat{\mu}' + e'$, then set

$$\widehat{\mathbf{c}}_\times(\vec{S}) := \widehat{\mathbf{c}}(\vec{S}) \cdot \widehat{\mathbf{c}}'(\vec{S}).$$

Correct:
$$\begin{aligned}\widehat{\mathbf{c}}_\times(\mathbf{s}) &= (\widehat{\mu} + e) \cdot \left(\widehat{\mu}' + e'\right) \\ &= \widehat{\mu} \cdot \widehat{\mu}' + (e \cdot \widehat{\mu}' + \widehat{\mu} \cdot e' + e \cdot e') \\ &\approx \widehat{\mu} \cdot \widehat{\mu}' = \mu \cdot \mu' \pmod{p\mathbb{Z}}.\end{aligned}$$

Error blowup

Typically, $|\widehat{\mu}|, |\widehat{\mu}'| \approx p \cdot \|\mathbf{s}\|_2$, so multiplication amplifies noise by $\gg p$.

Base-b Decomposition: Two Approaches

- **Natural idea:** express μ as d base-b digits, where $b \approx p^{1/d}$.

Base-b Decomposition: Two Approaches

- **Natural idea:** express μ as d base-b digits, where $b \approx p^{1/d}$.

Arithmetic with Carries (e.g., TFHE)

✔ Encrypt each digit in plaintext ring $\mathbb{Z}_b$: error growth $\approx b$

Base-b Decomposition: Two Approaches

- **Natural idea:** express μ as d base-b digits, where $b \approx p^{1/d}$.

Arithmetic with Carries (e.g., TFHE)

✔ Encrypt each digit in plaintext ring $\mathbb{Z}_b$: error growth $\approx b$

✗ Addition needs nonlinear carry logic

Base-b Decomposition: Two Approaches

- **Natural idea:** express μ as d base-b digits, where $b \approx p^{1/d}$.

Arithmetic with Carries (e.g., TFHE)

- ✔ Encrypt each digit in plaintext ring $\mathbb{Z}_b$: error growth $\approx b$
- ✗ Addition needs nonlinear carry logic
- ✗ Multiplication needs $\Omega(d^2)$ $\mathbb{Z}_b$-products

Base-b Decomposition: Two Approaches

- **Natural idea:** express μ as d base-b digits, where $b \approx p^{1/d}$.

Arithmetic with Carries (e.g., TFHE)

- ✔ Encrypt each digit in plaintext ring $\mathbb{Z}_b$: error growth $\approx b$
- ✗ Addition needs nonlinear carry logic
- ✗ Multiplication needs $\Omega(d^2)$ $\mathbb{Z}_b$-products

'Relaxed' polynomial representations

- View the plaintext digits as coefficients of a polynomial:

$$\boldsymbol{\mu}(B) = \mu_0 + \mu_1 B + \cdots + \mu_{d-1} B^{d-1}, \qquad \mu = \boldsymbol{\mu}(b).$$

Base-b Decomposition: Two Approaches

- **Natural idea:** express μ as d base-b digits, where $b \approx p^{1/d}$.

Arithmetic with Carries (e.g., TFHE)

- ✔ Encrypt each digit in plaintext ring $\mathbb{Z}_b$: error growth $\approx b$
- ✗ Addition needs nonlinear carry logic
- ✗ Multiplication needs $\Omega(d^2)$ $\mathbb{Z}_b$-products

'Relaxed' polynomial representations

- View the plaintext digits as coefficients of a polynomial:

$$\boldsymbol{\mu}(B) = \mu_0 + \mu_1 B + \cdots + \mu_{d-1} B^{d-1}, \qquad \mu = \boldsymbol{\mu}(b).$$

- Add and multiply the plaintexts as polynomials

Base-b Decomposition: Two Approaches

- **Natural idea:** express μ as d base-b digits, where $b \approx p^{1/d}$.

Arithmetic with Carries (e.g., TFHE)

- ✔ Encrypt each digit in plaintext ring $\mathbb{Z}_b$: error growth $\approx b$
- ✗ Addition needs nonlinear carry logic
- ✗ Multiplication needs $\Omega(d^2)$ $\mathbb{Z}_b$-products

'Relaxed' polynomial representations

- View the plaintext digits as coefficients of a polynomial:

$$\boldsymbol{\mu}(B) = \mu_0 + \mu_1 B + \cdots + \mu_{d-1} B^{d-1}, \qquad \mu = \boldsymbol{\mu}(b).$$

- Add and multiply the plaintexts as polynomials
- Allow coefficients larger than b, but still $O(b)$; do not compute carries

Base-b Decomposition: Two Approaches

- **Natural idea:** express μ as d base-b digits, where $b \approx p^{1/d}$.

Arithmetic with Carries (e.g., TFHE)

- ✔ Encrypt each digit in plaintext ring $\mathbb{Z}_b$: error growth $\approx b$
- ✗ Addition needs nonlinear carry logic
- ✗ Multiplication needs $\Omega(d^2)$ $\mathbb{Z}_b$-products

'Relaxed' polynomial representations

- View the plaintext digits as coefficients of a polynomial:

$$\boldsymbol{\mu}(B) = \mu_0 + \mu_1 B + \cdots + \mu_{d-1} B^{d-1}, \qquad \mu = \boldsymbol{\mu}(b).$$

- Add and multiply the plaintexts as polynomials
- Allow coefficients larger than b, but still $O(b)$; do not compute carries
- (Inspired by [CCKS'23] approx FHE, but technically very different.)

Plaintext Decomposition as a Ring Isomorphism

- Fix a base b and define the $\mathbb{Z}[B]$-ideal

$$\mathcal{L} := \{\mathbf{z}(B) \in \mathbb{Z}[B] : \mathbf{z}(b) = 0 \pmod{p}\} \supseteq p\mathbb{Z}.$$

("Gadget lattice": kernel of powers-of-b vector $(1, b, b^2, \ldots)$ [MicPei12])

Plaintext Decomposition as a Ring Isomorphism

- Fix a base b and define the $\mathbb{Z}[B]$-ideal

$$\mathcal{L} := \{\mathbf{z}(B) \in \mathbb{Z}[B] : \mathbf{z}(b) = 0 \pmod{p}\} \supseteq p\mathbb{Z}.$$

("Gadget lattice": kernel of powers-of-b vector $(1, b, b^2, \ldots)$ [MicPei12])

- Evaluation at b gives a ring isomorphism:

$$\boxed{\mathbb{Z}[B]/\mathcal{L} \;\cong\; \mathbb{Z}_p}$$

$$\boldsymbol{\mu}(B) + \mathcal{L} \longmapsto \boldsymbol{\mu}(b) \bmod p,$$
$$\mu + \mathcal{L} \longmapsfrom \mu.$$

Plaintext Decomposition as a Ring Isomorphism

- Fix a base b and define the $\mathbb{Z}[B]$-ideal

$$\mathcal{L} := \{\mathbf{z}(B) \in \mathbb{Z}[B] : \mathbf{z}(b) = 0 \pmod{p}\} \supseteq p\mathbb{Z}.$$

("Gadget lattice": kernel of powers-of-b vector $(1, b, b^2, \ldots)$ [MicPei12])

- Evaluation at b gives a ring isomorphism:

$$\boxed{\mathbb{Z}[B]/\mathcal{L} \;\cong\; \mathbb{Z}_p}$$

$$\boldsymbol{\mu}(B) + \mathcal{L} \longmapsto \boldsymbol{\mu}(b) \bmod p,$$
$$\mu + \mathcal{L} \longmapsto \mu.$$

- A decomposition of μ is any representative $\boldsymbol{\mu}(B) \in \mu + \mathcal{L}$.

Plaintext Decomposition as a Ring Isomorphism

- Fix a base b and define the $\mathbb{Z}[B]$-ideal

$$\mathcal{L} := \{\mathbf{z}(B) \in \mathbb{Z}[B] : \mathbf{z}(b) = 0 \pmod{p}\} \supseteq p\mathbb{Z}.$$

("Gadget lattice": kernel of powers-of-b vector $(1, b, b^2, \ldots)$ [MicPei12])

- Evaluation at b gives a ring isomorphism:

$$\boxed{\mathbb{Z}[B]/\mathcal{L} \;\cong\; \mathbb{Z}_p} \qquad \begin{aligned} \boldsymbol{\mu}(B) + \mathcal{L} &\longmapsto \boldsymbol{\mu}(b) \bmod p, \\ \mu + \mathcal{L} &\longmapsto\!\shortmid\; \mu. \end{aligned}$$

- A decomposition of μ is any representative $\boldsymbol{\mu}(B) \in \mu + \mathcal{L}$.

The Payoff

For $d = \lceil \log_b p \rceil$, every plaintext has many decompositions

$$\boldsymbol{\mu}(B) = \mu_0 + \mu_1 B + \cdots + \mu_{d-1} B^{d-1}, \qquad |\mu_i| \approx b \approx p^{1/d}.$$

Short Representatives Modulo $\mathcal{L}$

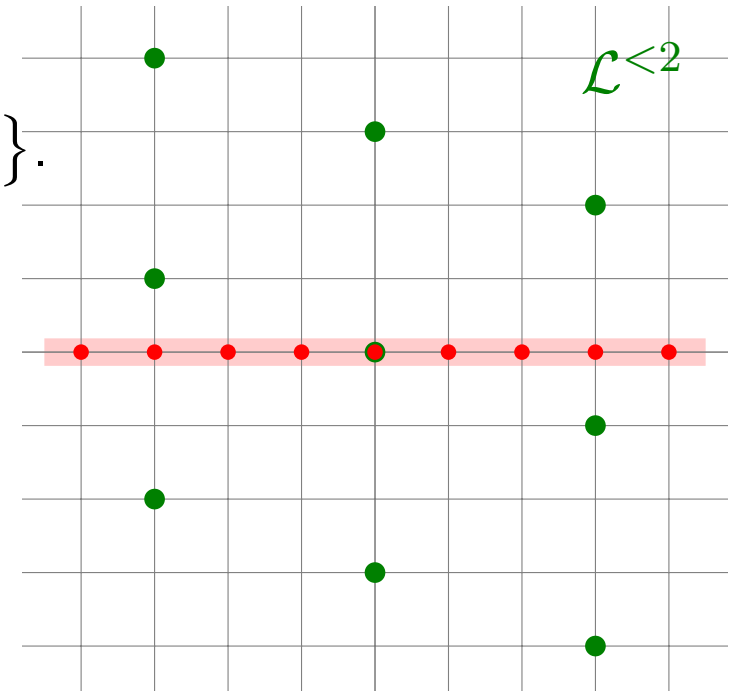
Toy example: let $b = 3$ and $p = b^2 = 9$.

$\mathbb{Q}, \mathbb{Z}$ representatives mod $p\mathbb{Z}$

Short Representatives Modulo $\mathcal{L}$

Toy example: let $b = 3$ and $p = b^2 = 9$.
Define

$$\mathcal{L}^{<2} := \left\{ z(B) \in \mathbb{Z}^{<2}[B] : z(3) = 0 \pmod 9 \right\}.$$



$\mathbb{Q}, \mathbb{Z}$ representatives mod $p\mathbb{Z}$

Short Representatives Modulo $\mathcal{L}$

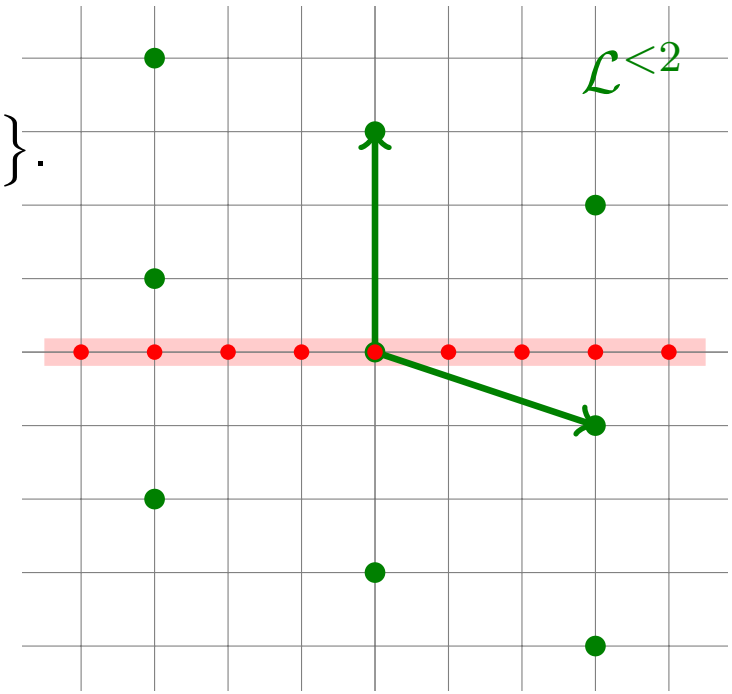
Toy example: let $b = 3$ and $p = b^2 = 9$.
Define

$$\mathcal{L}^{<2} := \left\{ z(B) \in \mathbb{Z}^{<2}[B] : z(3) = 0 \pmod 9 \right\}.$$

In coefficient form, a short basis is

$$\mathbf{b}_1 = (3, -1), \qquad \mathbf{b}_2 = (0, 3).$$

$\mathbb{Q}, \mathbb{Z}$ representatives mod $p\mathbb{Z}$



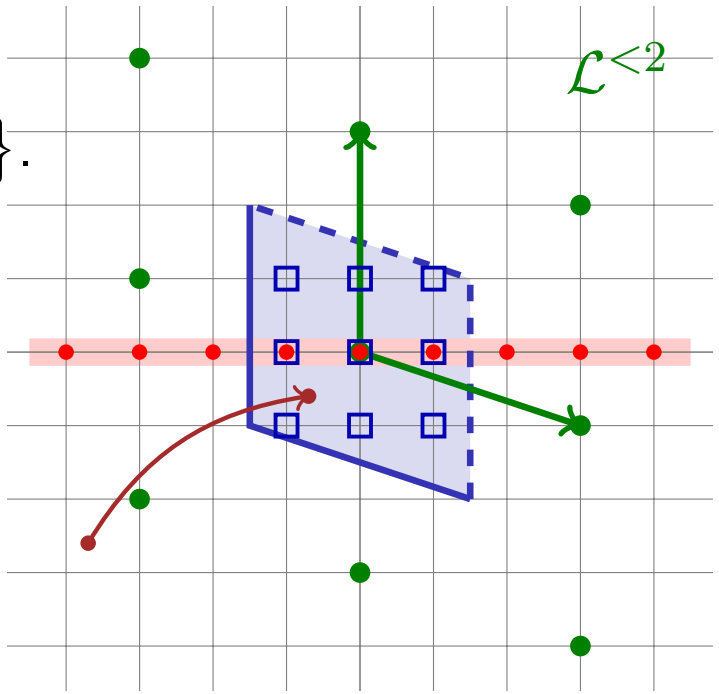
Short Representatives Modulo $\mathcal{L}$

Toy example: let $b = 3$ and $p = b^2 = 9$.
Define

$$\mathcal{L}^{<2} := \left\{ z(B) \in \mathbb{Z}^{<2}[B] : z(3) = 0 \pmod 9 \right\}.$$

In coefficient form, a short basis is

$$\mathbf{b}_1 = (3, -1), \qquad \mathbf{b}_2 = (0, 3).$$



$\mathbb{Q}, \mathbb{Z}$ representatives mod $p\mathbb{Z}$

$\mathbb{Q}^{<2}[B], \mathbb{Z}^{<2}[B]$ representatives mod $\mathcal{L}^{<2}$

The Isomorphism Extends to Noisy Short Decompositions

- Let $\widetilde{\boldsymbol{\mu}} = \boldsymbol{\mu} + \mathbf{e}$ and $\widetilde{\boldsymbol{\mu}}' = \boldsymbol{\mu}' + \mathbf{e}'$ be noisy 'short' decompositions (suppressing their argument B):

The Isomorphism Extends to Noisy Short Decompositions

- Let $\widetilde{\boldsymbol{\mu}} = \boldsymbol{\mu} + \mathbf{e}$ and $\widetilde{\boldsymbol{\mu}}' = \boldsymbol{\mu}' + \mathbf{e}'$ be noisy 'short' decompositions (suppressing their argument B):

$$\widetilde{\boldsymbol{\mu}} + \widetilde{\boldsymbol{\mu}}' = (\boldsymbol{\mu} + \boldsymbol{\mu}') + (\mathbf{e} + \mathbf{e}'),$$

$$\widetilde{\boldsymbol{\mu}} \cdot \widetilde{\boldsymbol{\mu}}' = \boldsymbol{\mu} \cdot \boldsymbol{\mu}' + (\mathbf{e}\boldsymbol{\mu}' + \boldsymbol{\mu}\mathbf{e}' + \mathbf{e}\mathbf{e}').$$

The Isomorphism Extends to Noisy Short Decompositions

- Let $\widetilde{\boldsymbol{\mu}} = \boldsymbol{\mu} + \mathbf{e}$ and $\widetilde{\boldsymbol{\mu}}' = \boldsymbol{\mu}' + \mathbf{e}'$ be noisy 'short' decompositions (suppressing their argument B):

$$\widetilde{\boldsymbol{\mu}} + \widetilde{\boldsymbol{\mu}}' = (\boldsymbol{\mu} + \boldsymbol{\mu}') + (\mathbf{e} + \mathbf{e}'),$$

$$\widetilde{\boldsymbol{\mu}} \cdot \widetilde{\boldsymbol{\mu}}' = \boldsymbol{\mu} \cdot \boldsymbol{\mu}' + (\mathbf{e}\boldsymbol{\mu}' + \boldsymbol{\mu}\mathbf{e}' + \mathbf{e}\mathbf{e}').$$

The win

- Short decompositions have size about $d\,b \approx d\,p^{1/d}$, not p.
  So multiplication grows noise by about this factor.

The Isomorphism Extends to Noisy Short Decompositions

- Let $\widetilde{\boldsymbol{\mu}} = \boldsymbol{\mu} + \mathbf{e}$ and $\widetilde{\boldsymbol{\mu}}' = \boldsymbol{\mu}' + \mathbf{e}'$ be noisy 'short' decompositions (suppressing their argument B):

$$\widetilde{\boldsymbol{\mu}} + \widetilde{\boldsymbol{\mu}}' = (\boldsymbol{\mu} + \boldsymbol{\mu}') + (\mathbf{e} + \mathbf{e}'),$$

$$\widetilde{\boldsymbol{\mu}} \cdot \widetilde{\boldsymbol{\mu}}' = \boldsymbol{\mu} \cdot \boldsymbol{\mu}' + (\mathbf{e}\boldsymbol{\mu}' + \boldsymbol{\mu}\mathbf{e}' + \mathbf{e}\mathbf{e}').$$

The win

- Short decompositions have size about $d\,b \approx d\,p^{1/d}$, not p.
  So multiplication grows noise by about this factor.
- The decompositions can be noncanonical; only shortness is needed.

Decomposed BFV (dBFV)

A dBFV ciphertext encrypting $\mu \in \mathbb{Z}_p$ is a bivariate polynomial

$$\mathbf{c}(B, \vec{S}) \in (\mathbb{Q}[B]/\mathcal{L})[\vec{S}],$$

typically affine linear in the secret-key variable $\vec{S}$,

Decomposed BFV (dBFV)

A dBFV ciphertext encrypting $\mu \in \mathbb{Z}_p$ is a bivariate polynomial

$$\mathbf{c}(B, \vec{S}) \in (\mathbb{Q}[B]/\mathcal{L})[\vec{S}],$$

typically affine linear in the secret-key variable $\vec{S}$, satisfying

$$\boxed{\mathbf{c}(B, \mathbf{s}) = \boldsymbol{\mu}(B) + \mathbf{e}(B) \approx \mu \pmod{\mathcal{L}}.}$$

Decomposed BFV (dBFV)

A dBFV ciphertext encrypting $\mu \in \mathbb{Z}_p$ is a bivariate polynomial

$$\mathbf{c}(B, \vec{S}) \in (\mathbb{Q}[B]/\mathcal{L})[\vec{S}],$$

typically affine linear in the secret-key variable $\vec{S}$, satisfying

$$\boxed{\mathbf{c}(B, \mathbf{s}) = \boldsymbol{\mu}(B) + \mathbf{e}(B) \approx \mu \pmod{\mathcal{L}}.}$$

1. Evaluate at the secret key $\mathbf{s}$ to get $\approx \boldsymbol{\mu}(B) \bmod \mathcal{L}$

Decomposed BFV (dBFV)

A dBFV ciphertext encrypting $\mu \in \mathbb{Z}_p$ is a bivariate polynomial

$$\mathbf{c}(B, \vec{S}) \in (\mathbb{Q}[B]/\mathcal{L})[\vec{S}],$$

typically affine linear in the secret-key variable $\vec{S}$, satisfying

$$\boxed{\mathbf{c}(B, \mathbf{s}) = \boldsymbol{\mu}(B) + \mathbf{e}(B) \approx \mu \pmod{\mathcal{L}}.}$$

1. Evaluate at the secret key $\mathbf{s}$ to get $\approx \boldsymbol{\mu}(B) \bmod \mathcal{L}$
2. Round the coefficients to recover $\boldsymbol{\mu}(B) \bmod \mathcal{L}$

Decomposed BFV (dBFV)

A dBFV ciphertext encrypting $\mu \in \mathbb{Z}_p$ is a bivariate polynomial

$$\mathbf{c}(B, \vec{S}) \in (\mathbb{Q}[B]/\mathcal{L})[\vec{S}],$$

typically affine linear in the secret-key variable $\vec{S}$, satisfying

$$\boxed{\mathbf{c}(B, \mathbf{s}) = \boldsymbol{\mu}(B) + \mathbf{e}(B) \approx \mu \pmod{\mathcal{L}}.}$$

1. Evaluate at the secret key $\mathbf{s}$ to get $\approx \boldsymbol{\mu}(B) \bmod \mathcal{L}$
2. Round the coefficients to recover $\boldsymbol{\mu}(B) \bmod \mathcal{L}$
3. Evaluate at $B = b$ to recover $\mu \in \mathbb{Z}_p$

Decomposed BFV (dBFV)

A dBFV ciphertext encrypting $\mu \in \mathbb{Z}_p$ is a bivariate polynomial

$$\mathbf{c}(B, \vec{S}) \in (\mathbb{Q}[B]/\mathcal{L})[\vec{S}],$$

typically affine linear in the secret-key variable $\vec{S}$, satisfying

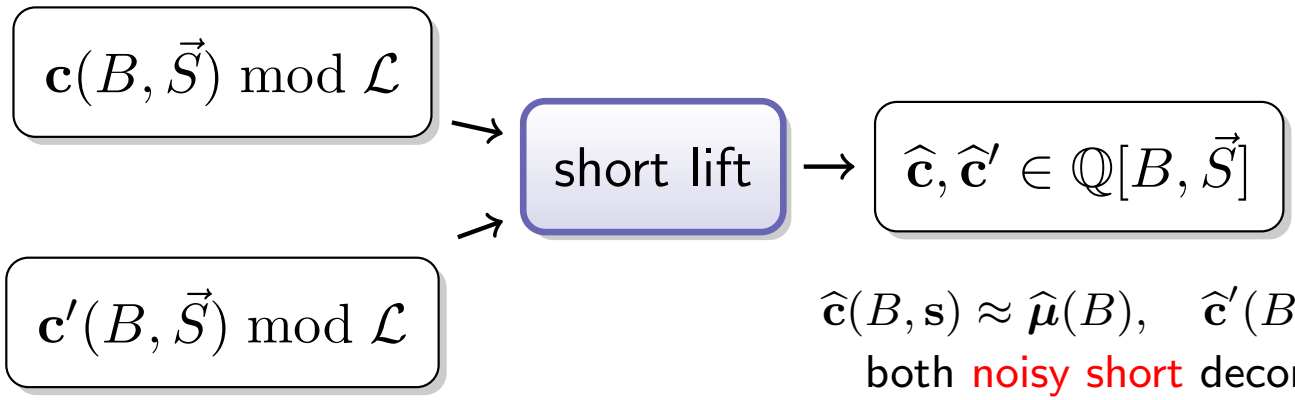
$$\boxed{\mathbf{c}(B, \mathbf{s}) = \boldsymbol{\mu}(B) + \mathbf{e}(B) \approx \mu \pmod{\mathcal{L}}.}$$

1. Evaluate at the secret key $\mathbf{s}$ to get $\approx \boldsymbol{\mu}(B) \bmod \mathcal{L}$
2. Round the coefficients to recover $\boldsymbol{\mu}(B) \bmod \mathcal{L}$
3. Evaluate at $B = b$ to recover $\mu \in \mathbb{Z}_p$

Fresh ciphertexts and security

- Fresh ciphertexts are mod-$\mathcal{L}$-LWE samples with error rate $\propto 1/p^{1/d}$.
- Hardness follows by a nearly lossless reduction from the corresponding standard plain/Module/Ring-LWE problem.

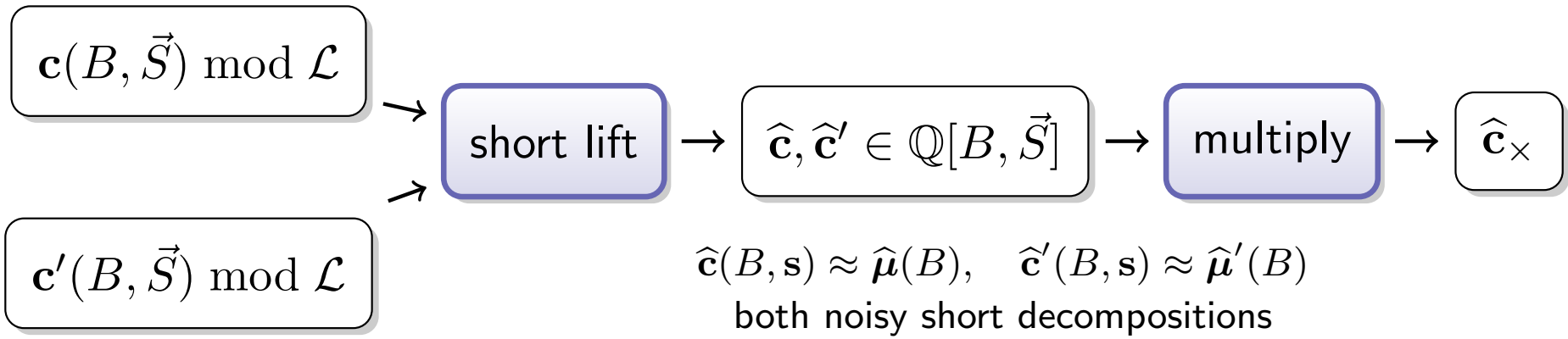
Homomorphic Multiplication in dBFV



$\widehat{\mathbf{c}}(B, \mathbf{s}) \approx \widehat{\boldsymbol{\mu}}(B), \quad \widehat{\mathbf{c}}'(B, \mathbf{s}) \approx \widehat{\boldsymbol{\mu}}'(B)$

both noisy short decompositions

Homomorphic Multiplication in dBFV

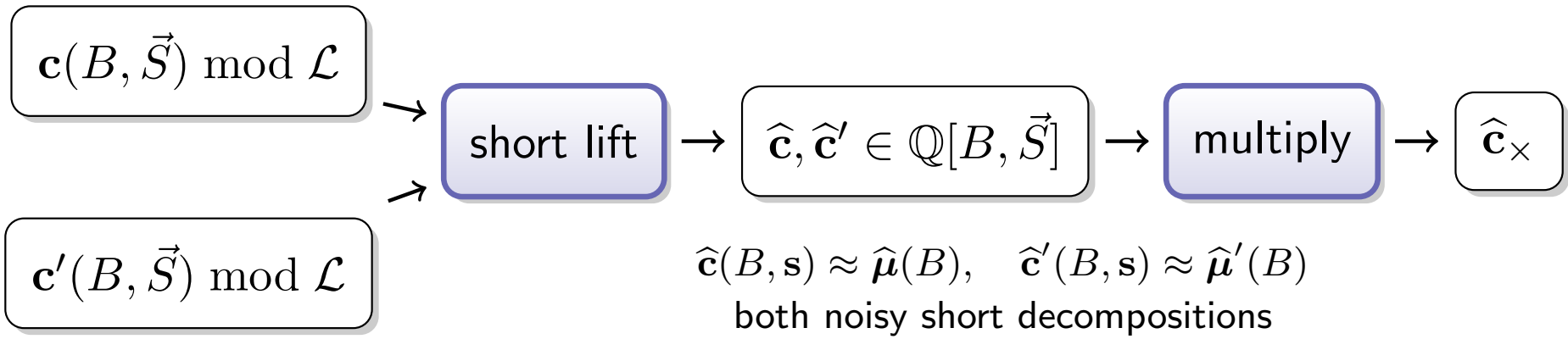


$$\widehat{\mathbf{c}}(B, \mathbf{s}) \approx \widehat{\boldsymbol{\mu}}(B), \quad \widehat{\mathbf{c}}'(B, \mathbf{s}) \approx \widehat{\boldsymbol{\mu}}'(B)$$

both noisy short decompositions

$$\widehat{\mathbf{c}}_\times(B, \mathbf{s}) \approx \widehat{\boldsymbol{\mu}}(B) \cdot \widehat{\boldsymbol{\mu}}'(B) = \mu \cdot \mu' \pmod{\mathcal{L}}$$

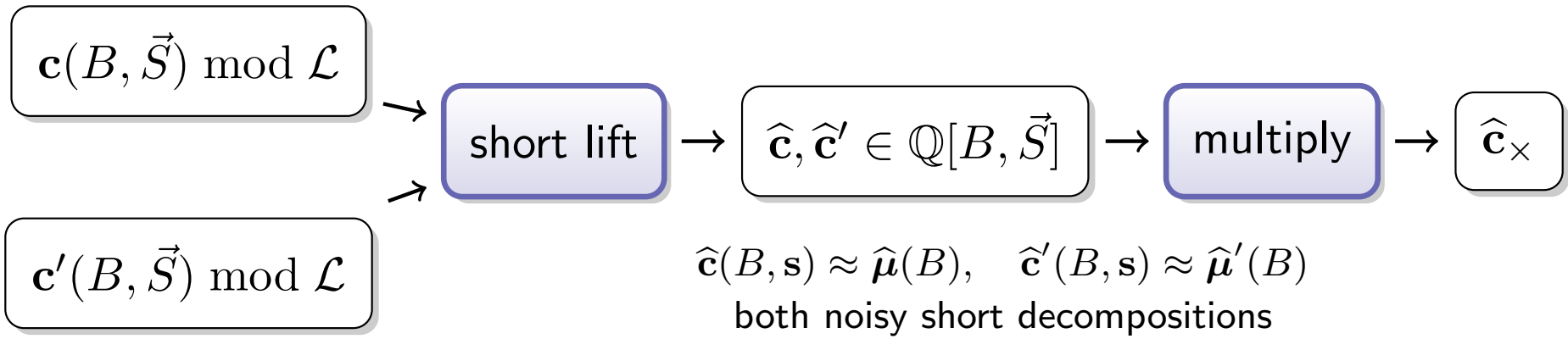
Homomorphic Multiplication in dBFV



Then restore to fresh-ciphertext form:

1. reduce modulo $\mathcal{L}$,

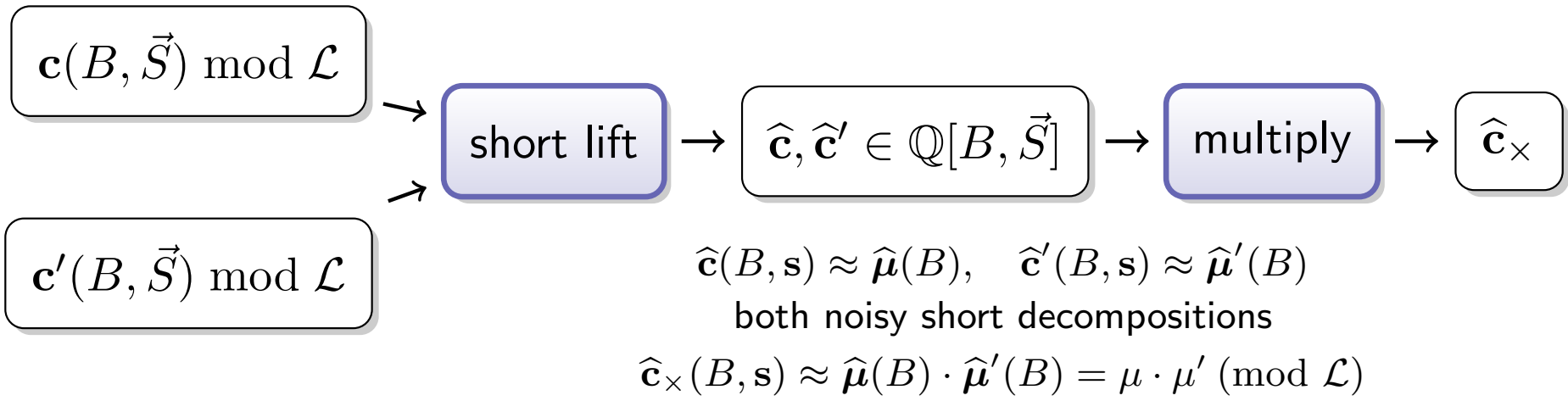
Homomorphic Multiplication in dBFV



Then restore to fresh-ciphertext form:

1. reduce modulo $\mathcal{L}$,
2. reduce degree in B,

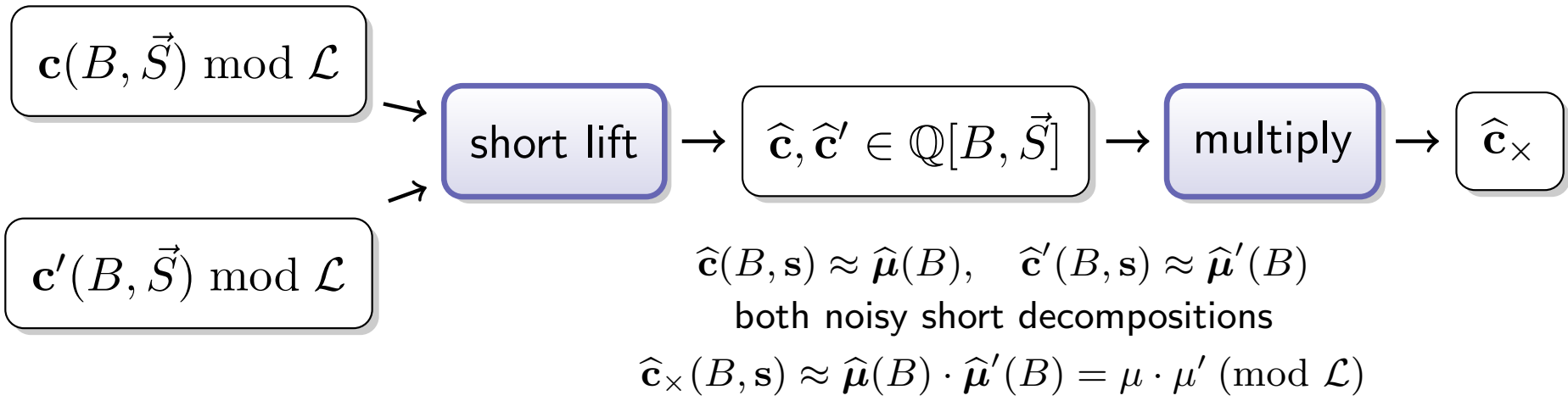
Homomorphic Multiplication in dBFV



Then restore to fresh-ciphertext form:

1. reduce modulo $\mathcal{L}$,
2. reduce degree in B,
3. relinearize in $\vec{S}$ by standard key switching.

Homomorphic Multiplication in dBFV



Then restore to fresh-ciphertext form:

1. reduce modulo $\mathcal{L}$,
2. reduce degree in B,
3. relinearize in $\vec{S}$ by standard key switching.

Degree reduction in B

- For $p = b^d$, simply use $B^d = 0 \pmod{\mathcal{L}}$.
- In general, high powers of B reduce to short combos of low powers.

Putting It All Together

Using d-digit ciphertexts with base $b \approx p^{1/d}$:

	ordinary BFV	dBFV
Fresh LWE error rate	$\approx 1/p$	$\approx 1/p^{1/d}$
Multiplicative growth	$\approx p$	$\approx d\,p^{1/d}$

Putting It All Together

Using d-digit ciphertexts with base $b \approx p^{1/d}$:

	ordinary BFV	dBFV
Fresh LWE error rate	$\approx 1/p$	$\approx 1/p^{1/d}$
Multiplicative growth	$\approx p$	$\approx d\,p^{1/d}$

Conclusions

- Taking $d \approx \log p$, dBFV's error profile matches ordinary BFV for an exponentially smaller modulus $\approx \log p$.

Putting It All Together

Using d-digit ciphertexts with base $b \approx p^{1/d}$:

	ordinary BFV	dBFV
Fresh LWE error rate	$\approx 1/p$	$\approx 1/p^{1/d}$
Multiplicative growth	$\approx p$	$\approx d\,p^{1/d}$

Conclusions

- Taking $d \approx \log p$, dBFV's error profile matches ordinary BFV for an exponentially smaller modulus $\approx \log p$.
- End-to-end sizes and times quasi-linear in $\log p$.

Putting It All Together

Using d-digit ciphertexts with base $b \approx p^{1/d}$:

	ordinary BFV	dBFV
Fresh LWE error rate	$\approx 1/p$	$\approx 1/p^{1/d}$
Multiplicative growth	$\approx p$	$\approx d\,p^{1/d}$

Conclusions

- Taking $d \approx \log p$, dBFV's error profile matches ordinary BFV for an exponentially smaller modulus $\approx \log p$.
- End-to-end sizes and times quasi-linear in $\log p$.
- Compatibility with arbitrary moduli and rings, SIMD packing, and bootstrapping.

Direction II: Flexible SIMD Packing

Use subfields of cyclotomics

(Peikert–Pepin ePrint 2025/1760)

FHE (and Lattice Cryptography) over Rings

- Use rings (in number fields) for efficiency

[HPS'96,Mic'02,PR'06,LM'06,LPR'10,...]

FHE (and Lattice Cryptography) over Rings

- Use rings (in number fields) for efficiency

  [HPS'96,Mic'02,PR'06,LM'06,LPR'10,...]

- Important FHE technique: "SIMD packing" via CRT [SV'11]

FHE (and Lattice Cryptography) over Rings

- Use rings (in number fields) for efficiency

  [HPS'96,Mic'02,PR'06,LM'06,LPR'10,...]

- Important FHE technique: "SIMD packing" via CRT [SV'11]
  - ⋆ plaintext $\mu \in \mathbb{F}_q$

FHE (and Lattice Cryptography) over Rings

- Use rings (in number fields) for efficiency

  [HPS'96,Mic'02,PR'06,LM'06,LPR'10,...]

- Important FHE technique: "SIMD packing" via CRT [SV'11]
  - ★ plaintext ~~$\mu \in \mathbb{F}_q$~~ $\vec{\mu} \in \mathbb{F}_q^n$

FHE (and Lattice Cryptography) over Rings

- Use rings (in number fields) for efficiency

  [HPS'96,Mic'02,PR'06,LM'06,LPR'10,...]

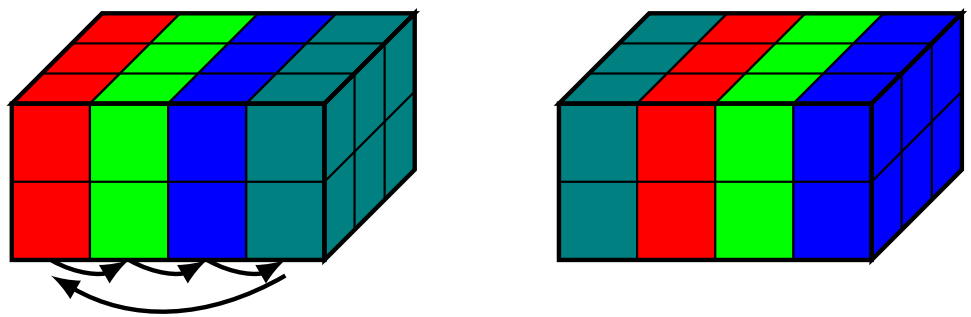
- Important FHE technique: "SIMD packing" via CRT [SV'11]
  - ★ plaintext ~~$\mu \in \mathbb{F}_q$~~ $\vec{\mu} \in \mathbb{F}_q^n$
  - ★ fast homomorphic coordinate-wise field operations

FHE (and Lattice Cryptography) over Rings

- Use rings (in number fields) for efficiency

  [HPS'96,Mic'02,PR'06,LM'06,LPR'10,...]

- Important FHE technique: "SIMD packing" via CRT [SV'11]
  - ★ plaintext ~~$\mu \in \mathbb{F}_q$~~ $\vec{\mu} \in \mathbb{F}_q^n$
  - ★ fast homomorphic coordinate-wise field operations
  - ★ certain vector permutations, for linear algebra and "packed bootstrapping" [BGV'12,GHS'12ab,HS'14]



Cyclotomics and Their Limitations

- Prior work has focused almost entirely on cyclotomic fields/rings.

Cyclotomics and Their Limitations

- Prior work has focused almost entirely on cyclotomic fields/rings.

✔ original setting for hardness of Ring-SIS/LWE (now all rings [PRS'17])

✔ computationally fast

✔ geometrically nice, for error growth & decoding

Cyclotomics and Their Limitations

- Prior work has focused almost entirely on cyclotomic fields/rings.

✔ original setting for hardness of Ring-SIS/LWE (now all rings [PRS'17])

✔ computationally fast

✔ geometrically nice, for error growth & decoding

✗ difficult to get the desired SIMD "slot type"

Cyclotomics and Their Limitations

- Prior work has focused almost entirely on cyclotomic fields/rings.

✔ original setting for hardness of Ring-SIS/LWE (now all rings [PRS'17])

✔ computationally fast

✔ geometrically nice, for error growth & decoding

✗ difficult to get the desired SIMD "slot type"

- ⋆ for homomorphic AES evaluation, [GHS'12c] was induced to use $\mathbb{F}_{2^{24}}$-slots, a $3\times$ loss in 'capacity' vs the natural $\mathbb{F}_{2^8}$

Cyclotomics and Their Limitations

- Prior work has focused almost entirely on cyclotomic fields/rings.

✔ original setting for hardness of Ring-SIS/LWE (now all rings [PRS'17])

✔ computationally fast

✔ geometrically nice, for error growth & decoding

✗ difficult to get the desired SIMD "slot type"

- ⋆ for homomorphic AES evaluation, [GHS'12c] was induced to use $\mathbb{F}_{2^{24}}$-slots, a $3\times$ loss in 'capacity' vs the natural $\mathbb{F}_{2^8}$

✗ difficult to get desired slot-tensor "shape" and data movement

Cyclotomics and Their Limitations

- Prior work has focused almost entirely on cyclotomic fields/rings.

✔ original setting for hardness of Ring-SIS/LWE (now all rings [PRS'17])

✔ computationally fast

✔ geometrically nice, for error growth & decoding

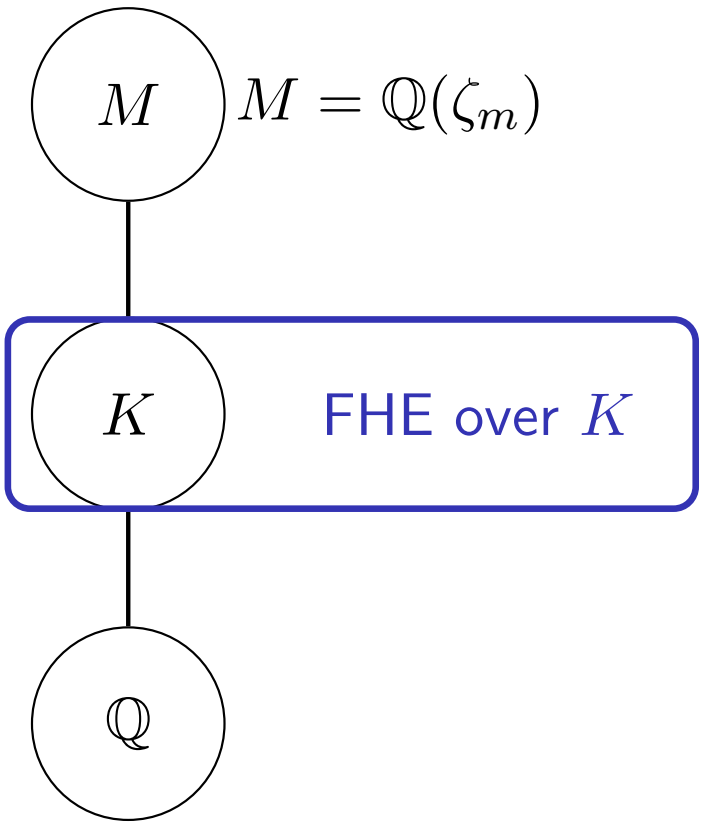
✗ difficult to get the desired SIMD "slot type"

- ⋆ for homomorphic AES evaluation, [GHS'12c] was induced to use $\mathbb{F}_{2^{24}}$-slots, a $3\times$ loss in 'capacity' vs the natural $\mathbb{F}_{2^8}$

✗ difficult to get desired slot-tensor "shape" and data movement

Use of cyclotomics entangles algebra, data layout, and security

Our Work: Beyond Cyclotomics

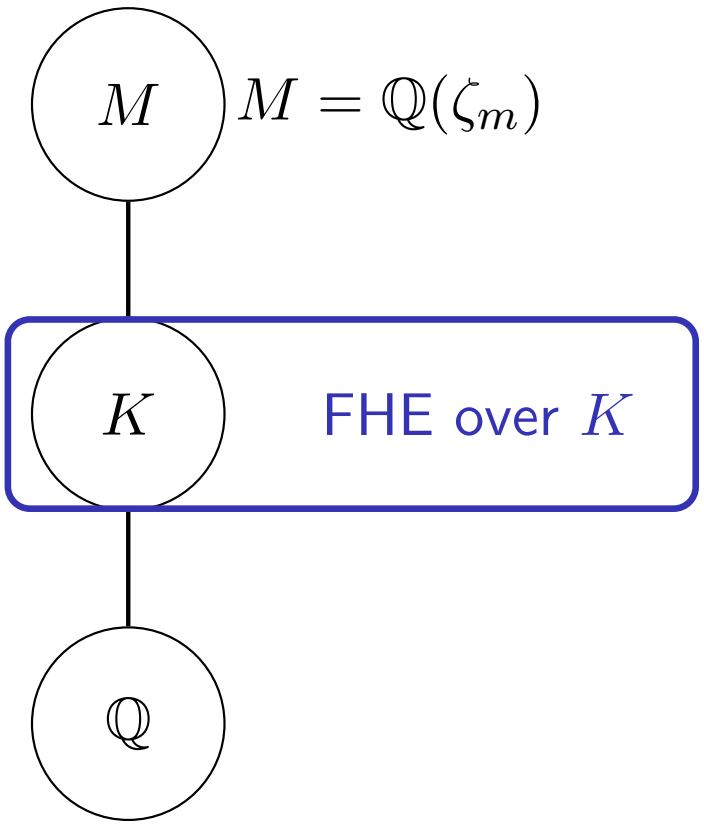


Choose the field to fit the application

Choose $K \subseteq \mathbb{Q}(\zeta_m)$ to obtain exactly the desired slot type.

Vary m and K to tune slot count, tensor shape, and security.

Our Work: Beyond Cyclotomics



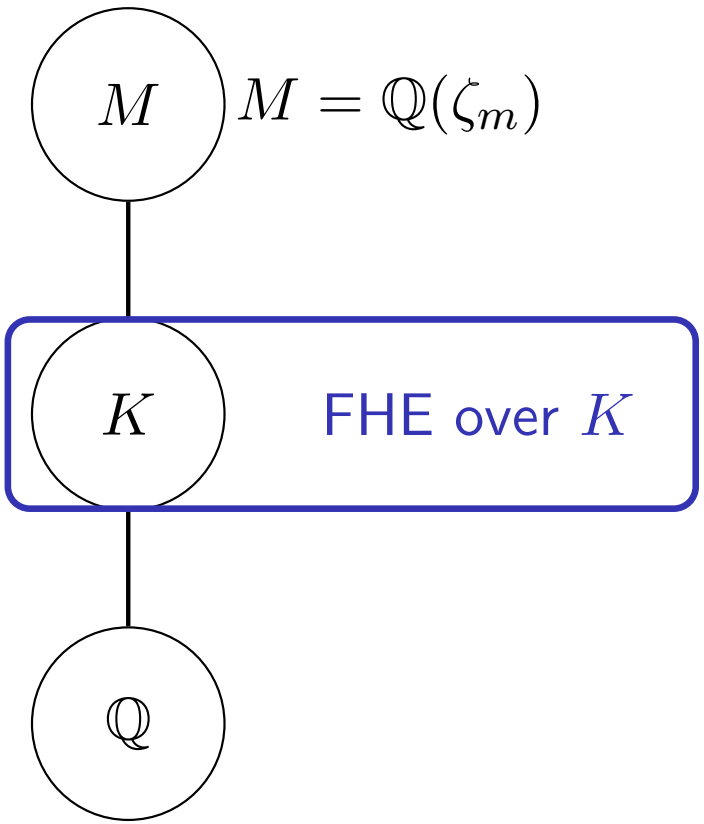
Choose the field to fit the application

Choose $K \subseteq \mathbb{Q}(\zeta_m)$ to obtain exactly the desired slot type.

Vary m and K to tune slot count, tensor shape, and security.

- These subfields are exactly the finite abelian extensions of $\mathbb{Q}$ [Kronecker–Weber]

Our Work: Beyond Cyclotomics



Choose the field to fit the application

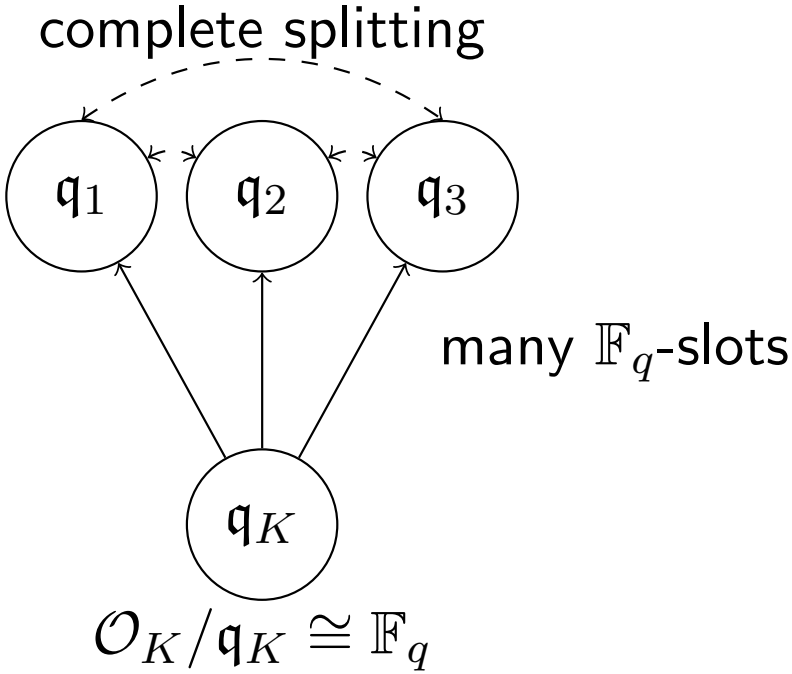
Choose $K \subseteq \mathbb{Q}(\zeta_m)$ to obtain exactly the desired slot type.

Vary m and K to tune slot count, tensor shape, and security.

- These subfields are exactly the finite abelian extensions of $\mathbb{Q}$ [Kronecker–Weber]
- Beyond prime-m decomposition subfields [AH'17]: broad class of cyclotomic subfields, with comparable geometry and algorithmic efficiency—including packed bootstrapping

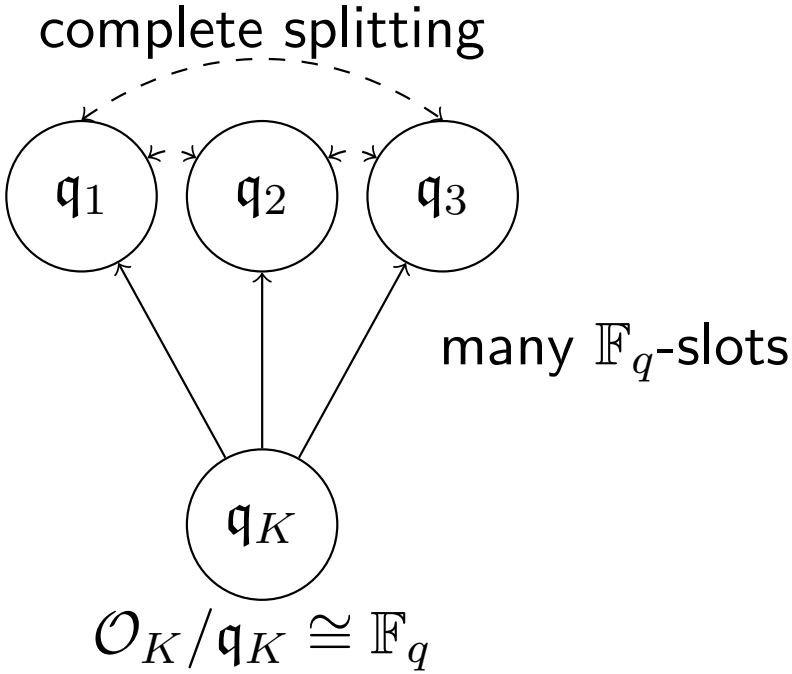
Galois Theory Controls Slot Type and Count

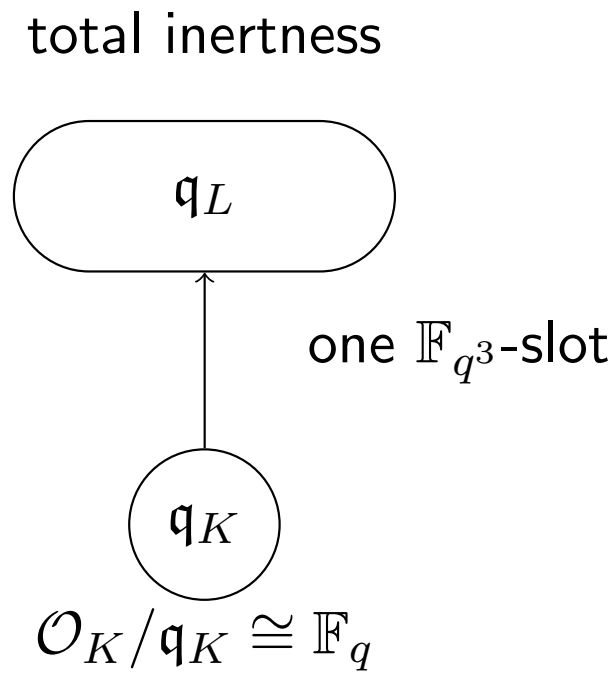
Let L/K be abelian and $\mathfrak{q}_K$ be a prime ideal in $\mathcal{O}_K$.



Galois Theory Controls Slot Type and Count

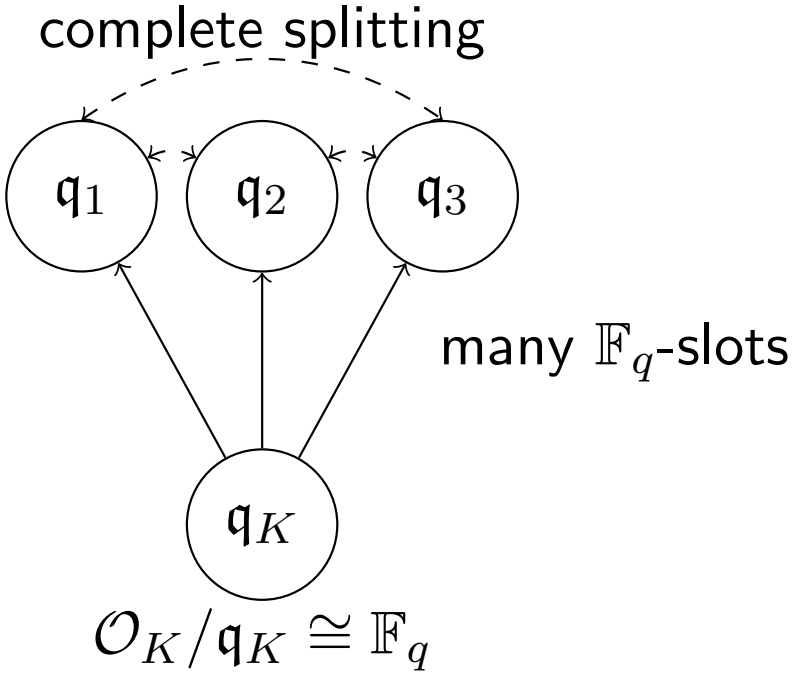
Let L/K be abelian and $\mathfrak{q}_K$ be a prime ideal in $\mathcal{O}_K$.

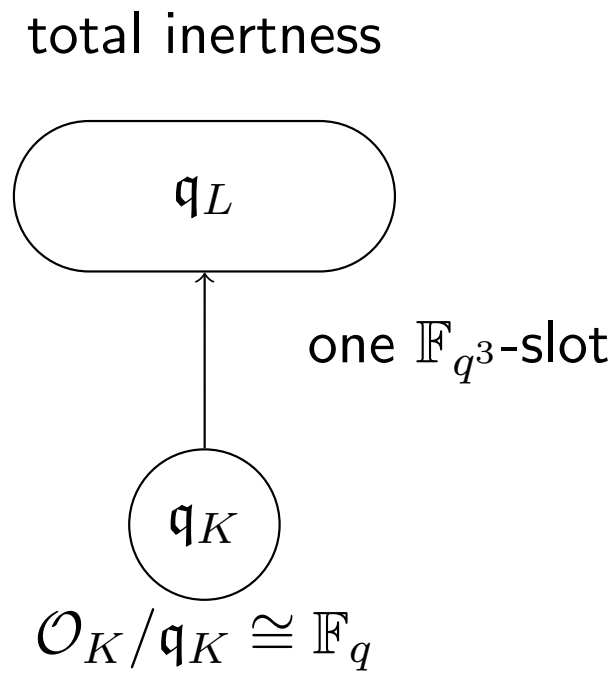




Galois Theory Controls Slot Type and Count

Let L/K be abelian and $\mathfrak{q}_K$ be a prime ideal in $\mathcal{O}_K$.



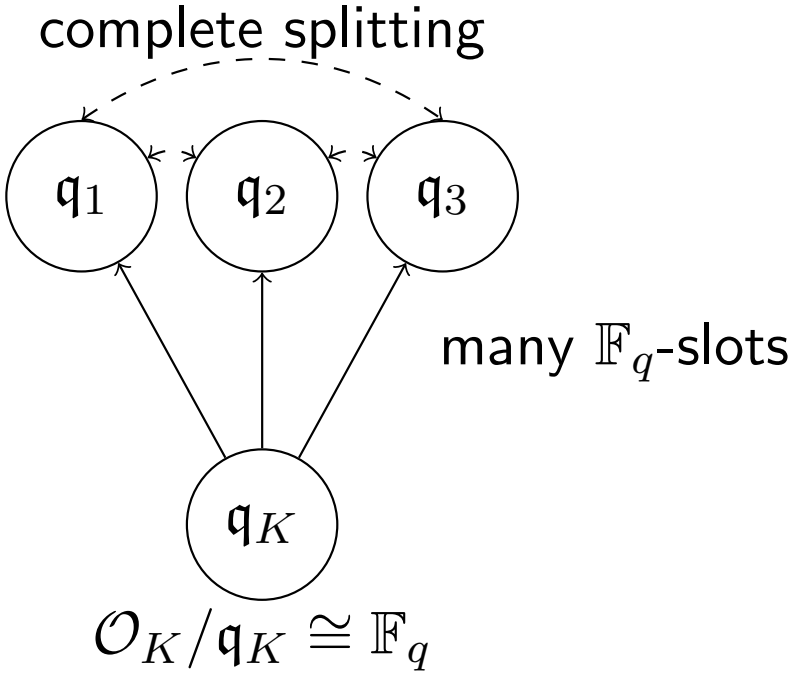


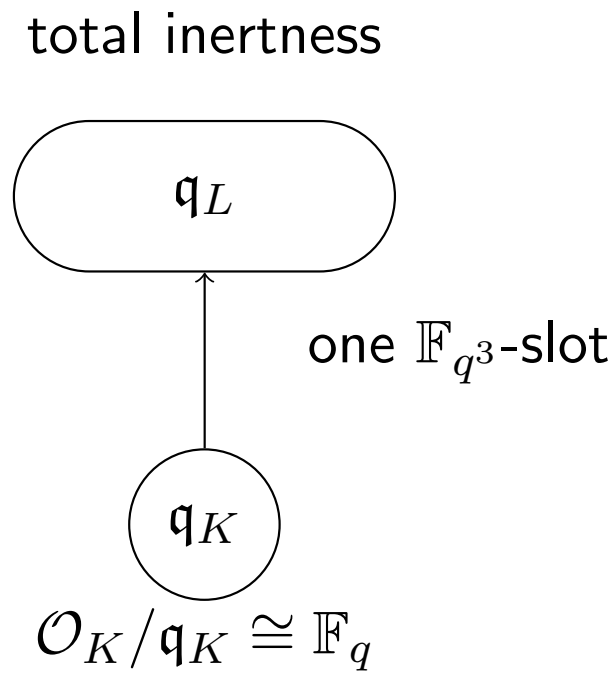
The Chinese Remainder Theorem gives the plaintext slots:

$$\mathcal{O}_L/\mathfrak{q}_K\mathcal{O}_L \cong \prod_i (\mathcal{O}_L/\mathfrak{q}_i).$$

Galois Theory Controls Slot Type and Count

Let L/K be abelian and $\mathfrak{q}_K$ be a prime ideal in $\mathcal{O}_K$.





The Chinese Remainder Theorem gives the plaintext slots:

$$\mathcal{O}_L/\mathfrak{q}_K\mathcal{O}_L \cong \prod_i (\mathcal{O}_L/\mathfrak{q}_i).$$

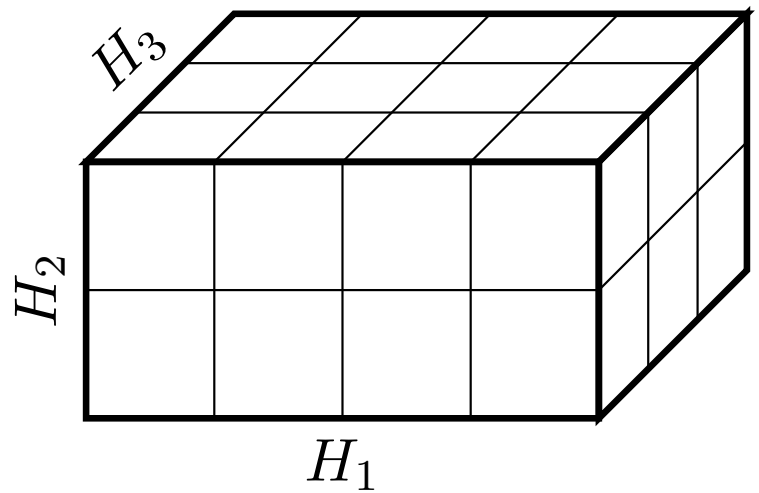
Combine splitting and inertness across towers of subfields to get desired slot field and count.

Galois Factors Give the Slot-Tensor Shape and Movement

If

$$\mathrm{Gal}(L/K) = H_1 \times H_2 \times H_3 \ ,$$

then the CRT slots form a tensor with axes indexed by the H_i.

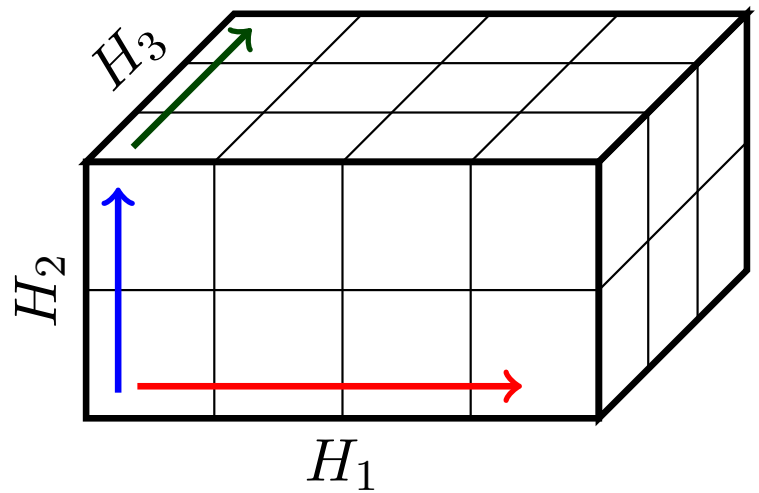


Galois Factors Give the Slot-Tensor Shape and Movement

If

$$\mathrm{Gal}(L/K) = H_1 \times H_2 \times H_3 \ ,$$

then the CRT slots form a tensor with axes indexed by the H_i.

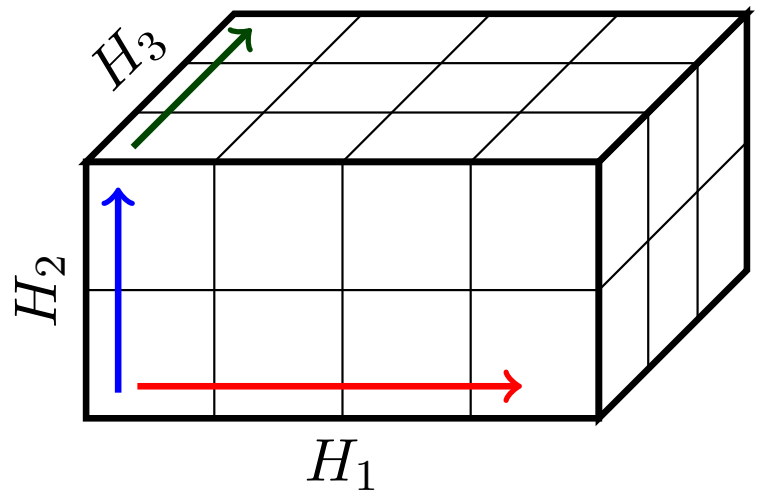


Galois Factors Give the Slot-Tensor Shape and Movement

If

$$\mathrm{Gal}(L/K) = H_1 \times H_2 \times H_3 \ ,$$

then the CRT slots form a tensor with axes indexed by the H_i.



Automorphisms from each H_i move data along just one dimension.

Retaining Cyclotomic-Like Efficiency

- FHE needs two kinds of bases:

Retaining Cyclotomic-Like Efficiency

- FHE needs two kinds of bases:
  - ⋆ a CRT basis for fast arithmetic and SIMD slots;

Retaining Cyclotomic-Like Efficiency

- FHE needs two kinds of bases:
  - ⋆ a CRT basis for fast arithmetic and SIMD slots;
  - ⋆ a short integral basis for errors, encryption, decryption, bootstrapping.

Retaining Cyclotomic-Like Efficiency

- FHE needs two kinds of bases:
  - ⋆ a CRT basis for fast arithmetic and SIMD slots;
  - ⋆ a short integral basis for errors, encryption, decryption, bootstrapping.
- Build tensor-structured bases through towers of abelian fields:

$$\mathbf{b}_{M/K} = \mathbf{b}_{M/L} \otimes \mathbf{b}_{L/K}, \qquad \mathbf{c}_{M/K} = \mathbf{c}_{M/L} \otimes \mathbf{c}_{L/K}.$$

Retaining Cyclotomic-Like Efficiency

- FHE needs two kinds of bases:
  - ⋆ a CRT basis for fast arithmetic and SIMD slots;
  - ⋆ a short integral basis for errors, encryption, decryption, bootstrapping.
- Build tensor-structured bases through towers of abelian fields:

$$\mathbf{b}_{M/K} = \mathbf{b}_{M/L} \otimes \mathbf{b}_{L/K}, \qquad \mathbf{c}_{M/K} = \mathbf{c}_{M/L} \otimes \mathbf{c}_{L/K}.$$

- Such factorizations give fast CRT transforms, both 'in the clear' and homomorphically (via two different frameworks)

Retaining Cyclotomic-Like Efficiency

- FHE needs two kinds of bases:
  - ⋆ a CRT basis for fast arithmetic and SIMD slots;
  - ⋆ a short integral basis for errors, encryption, decryption, bootstrapping.
- Build tensor-structured bases through towers of abelian fields:

$$\mathbf{b}_{M/K} = \mathbf{b}_{M/L} \otimes \mathbf{b}_{L/K}, \qquad \mathbf{c}_{M/K} = \mathbf{c}_{M/L} \otimes \mathbf{c}_{L/K}.$$

- Such factorizations give fast CRT transforms, both 'in the clear' and homomorphically (via two different frameworks)
- These transforms yield fast ring arithmetic, homomorphic linear algebra, and packed bootstrapping

Retaining Cyclotomic-Like Efficiency

- FHE needs two kinds of bases:
  - ⋆ a CRT basis for fast arithmetic and SIMD slots;
  - ⋆ a short integral basis for errors, encryption, decryption, bootstrapping.
- Build tensor-structured bases through towers of abelian fields:

$$\mathbf{b}_{M/K} = \mathbf{b}_{M/L} \otimes \mathbf{b}_{L/K}, \qquad \mathbf{c}_{M/K} = \mathbf{c}_{M/L} \otimes \mathbf{c}_{L/K}.$$

- Such factorizations give fast CRT transforms, both 'in the clear' and homomorphically (via two different frameworks)
- These transforms yield fast ring arithmetic, homomorphic linear algebra, and packed bootstrapping

Bottom line

- Subfields add design freedom, and retain the machinery that make cyclotomics useful

New Directions: Better Representations

High precision: decompose the plaintext ring

$$\mathbb{Z}_p \cong \mathbb{Z}[B]/\mathcal{L} \quad \Longrightarrow \quad \text{noise growth } \approx d\, p^{1/d}.$$

Arbitrary modulus and ring; quasi-linear dependence on $\log p$.

New Directions: Better Representations

High precision: decompose the plaintext ring

$$\mathbb{Z}_p \cong \mathbb{Z}[B]/\mathcal{L} \quad \Longrightarrow \quad \text{noise growth } \approx d\, p^{1/d}.$$

Arbitrary modulus and ring; quasi-linear dependence on $\log p$.

Flexible packing: enlarge the menu of fields

$$K \subseteq \mathbb{Q}(\zeta_m) \quad \Longrightarrow \quad \text{designed slot type, shape, and security}$$

Fast arithmetic, data movement, and bootstrapping remain available.

New Directions: Better Representations

High precision: decompose the plaintext ring

$$\mathbb{Z}_p \cong \mathbb{Z}[B]/\mathcal{L} \quad \Longrightarrow \quad \text{noise growth } \approx d\, p^{1/d}.$$

Arbitrary modulus and ring; quasi-linear dependence on $\log p$.

Flexible packing: enlarge the menu of fields

$$K \subseteq \mathbb{Q}(\zeta_m) \quad \Longrightarrow \quad \text{designed slot type, shape, and security}$$

Fast arithmetic, data movement, and bootstrapping remain available.

Thank you! Questions?