TFHE - Chimera: How to combine fully homomorphic encryption schemes? Application: Feature selection

N. Gama[2,3], M. Georgieva [1,2]

1 ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE 2 inpher.io 2 UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES université PARIS-SACLAY

Joint work with: C. Boura, D. Jetchev, S. Carpov, J.R. Troncoso, I.Chillotti *et.al.*

Plan

1. Fully Homomorphic Encryption
2. Learning with error over the Torus
3. The framework Chimera
4. Application: feature selection

The idea

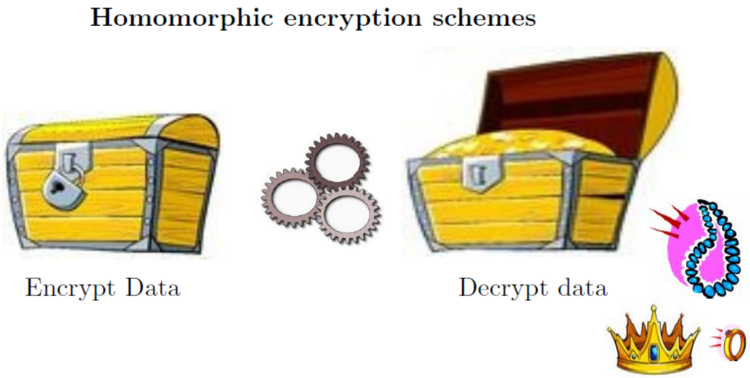
Classical encryption schemes

Encrypt Data

Decrypt data

Is it possible to manipulate the data without decryption?

The idea

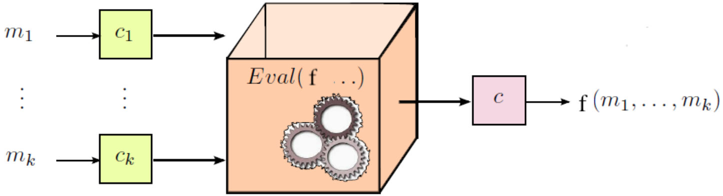

Homomorphic encryption schemes
Encrypt Data
Decrypt data

Homomorphic encryption

inpher.io

- Given $(c_1, c_2, \ldots, c_k) = (\mathcal{E}(m_1), \mathcal{E}(m_2), \ldots, \mathcal{E}(m_k))$

The homomorphic computation consists to compute $\mathcal{E}(f(m_1, m_2, \ldots, m_k))$ without decryption.



A scheme that can homomorphically evaluate all function is said **Fully Homomorphic**

Examples: homomorphic schemes

inpher.io

- Multiplicatively homomorphic : RSA

$$c_1 = m_1^e \mod N \qquad \text{et} \qquad c_2 = m_2^e \mod N$$
$$Eval(c_1, c_2) = c_1.c_2 = (m_1.m_2)^e \mod N = \mathcal{E}(m_1.m_2) \mod N$$

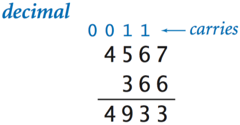
- Additively homomorphic : Paillier

$$c_1 = g^{m_1} r_1^n \mod n^2 \qquad \text{et} \qquad c_2 = g^{m_2} r_2^n \mod n^2$$
$$Eval(c_1, c_2) = c_1.c_2 = g^{m_1+m_2}(r_1.r_2)^n \mod n^2 = \mathcal{E}(m_1 + m_2) \mod n^2$$

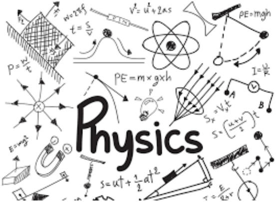
Fully homomorphic : homomorphic for both **addition and multiplication**

Model of computations

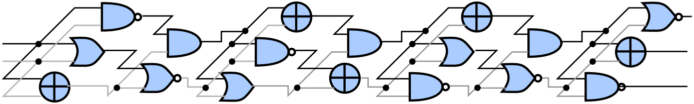
1. Integer arithmetic



2. Approximated (Fixed-point) computations



3. Binary, circuit computations



Integer arithmetic

inpher.io

Given encrypted $m_1, m_2 \in \mathbb{Z}$, compute:

$$\boxed{m_1} \quad \boxed{m_2} \longrightarrow \begin{array}{l} Dec(\boxed{m_1} +_{hom} \boxed{m_2}) = Dec(\boxed{m_1 + m_2}) \\ Dec(\boxed{m_1} *_{hom} \boxed{m_2}) = Dec(\boxed{m_1 * m_2}) \end{array}$$

Possibility to do SIMD arithmetic:

Given encrypted $m_1, m_2 \in \mathbb{Z}^N$:

- element-wise addition $m_1 + m_2$
- element-wise product $m_1 * m_2$
- permutations $\sigma(m_1)$

Usually, arbitrary precision integers are not FHE friendly:

- arithmetic modulo some mid-size p (e.g. $p = 2^{32}$, like ints in C)
- if so, be aware of overflows: (e.g. $2^{30} + 2^{30} = -2^{31}$ in C)

Floating point computations

inpher.io

In physics, we use real or complex numbers, but care only about order of magnitudes:

Example:

If the height of a person must be known $\pm$ 1cm, the radius of the earth can be given $\pm$ 10km. In both case, we just care about the 3 most significant decimal digits.

$m_1, m_2 \in \mathbb{R}$

$$\boxed{m_1} \quad \boxed{m_2} \longrightarrow \begin{array}{rcl} MSB(Dec(\boxed{m_1} +_{hom} \boxed{m_2})) & = & MSB(Dec(\boxed{m_1 + m_2})) \\ MSB(Dec(\boxed{m_1} *_{hom} \boxed{m_2})) & = & MSB(Dec(\boxed{m_1 * m_2})) \end{array}$$

There are two models: Fixed points and Floating point

inpher.io

Floating point (float, double in C):

- $x = m.2^{\tau}$, with $m \in 2^{-\rho}.\mathbb{Z}$ and $\frac{1}{2} \leq |m| < 1$
- $\tau = \lceil log_2(x) \rceil$ data dependent and not public (not FHE-friendly)
- The exponent is always in sync with the data
  ex: $(1.23 \cdot 10^{-4}) * (7.24 \cdot 10^{-4}) = (8.90 \cdot 10^{-8})$

Fixed point:

- $x = m.2^{\tau}$, with $m \in 2^{-\rho}.\mathbb{Z}$ and $0 \leq |m| < 1$,
- τ is public, thus FHE-friendly
- Risk of overflow (τ too small)
- Risk of underflow (τ too large)
  ex: $(0.000123 \cdot 10^{0}) * (0.000724 \cdot 10^{0}) = (0.000000 \cdot 10^{0})$

Plaintext parameters:

- $\rho \in \mathbb{N}$: bits of precision of the plaintext (≈ 15 bits)
- $\tau \in \mathbb{Z}$: slot exponent (order of magnitude of the complex values in each slot)

Fixed point

inpher.io

Here again, we would like:

- (possibly SIMD) Fixed point addition
- (possibly SIMD) Fixed point multiplication
- permutations

Addition is much tricker than you think!

- Given (m_1, τ_1), (m_2, τ_2), and τ.
- How do you compute $m.2^\tau = m_1.2^{\tau_1} + m_2.2^{\tau_2}$ with ρ bits of precision?
- Addition requires right shift and roundings, which are non-linear!

Fixed point

inpher.io

Here again, we would like:

- (possibly SIMD) Fixed point addition
- (possibly SIMD) Fixed point multiplication
- permutations

Addition is much tricker than you think!

- Given (m_1, τ_1), (m_2, τ_2), and τ.
- How do you compute $m.2^{\tau} = m_1.2^{\tau_1} + m_2.2^{\tau_2}$ with ρ bits of precision?
- Addition requires right shift and roundings, which are non-linear!

Circuit computations

inpher.io

$b_1, b_2 \in \{0, 1\}$

$$\boxed{b_1} \quad \boxed{b_2} \longrightarrow \begin{array}{l} Dec(\boxed{b_1} \oplus_{hom} \boxed{b_2}) = Dec(\boxed{b_1 \oplus b_2}) \\ Dec(\boxed{b_1} \wedge_{hom} \boxed{b_2}) = Dec(\boxed{b_1 \wedge b_2}) \end{array}$$

3 kind of interesting circuits:

- **boolean gates** (fully boolean): NAND, AND, OR, NOT, XOR, MUX
- **lookup tables** (mixed): given $(v_0, \ldots, v_n)$ and i, return v_i
- **decision diagrams**, or **automata** (also mixed): everytime you read a bit, you update an internal state. Return some info about the arrival state, or on the whole path.

A few examples:

- e.g. given the bits of x, compute $x \mod 7$
- e.g. given public integers (a_i) and encrypted bits of s, compute $\sum(a_i s_i) \mod 1024$

Interesting example: The comparison circuit

$$A = \; 0 \quad 1 \quad 0 \quad 1 \quad \ldots \quad 0 \quad 1$$

$$B = \; 1 \quad 0 \quad 0 \quad 1 \quad \ldots \quad 1 \quad 0$$

Interesting example: The comparison circuit

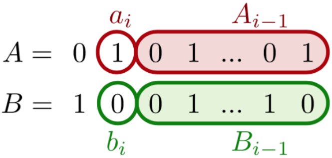
inpher.io

$$A = 0\ \underbrace{1\ 0\ 1\ \ldots\ 0\ 1}_{A_i}$$

$$B = 1\ \underbrace{0\ 0\ 1\ \ldots\ 1\ 0}_{B_i}$$

$$r_i := (A_i \leq B_i)$$

Interesting example: The comparison circuit



$$r_i := (A_i \leq B_i)$$

$$r_i = \left(\begin{array}{l} b_i \text{ when } a_i \neq b_i \\ r_{i-1} \text{ otherwise} \end{array} \right.$$

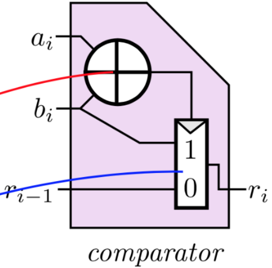
$$r_0 = \text{true}$$

$$r_n = ?$$

Interesting example: The comparison circuit

$$A = 0\ \underset{a_i}{1}\ \underbrace{0\ 1\ \ldots\ 0\ 1}_{A_{i-1}}$$

$$B = 1\ \underset{b_i}{0}\ \underbrace{0\ 1\ \ldots\ 1\ 0}_{B_{i-1}}$$



$$r_i := (A_i \le B_i)$$

$$r_i = \begin{cases} b_i & \text{when } a_i \neq b_i \\ r_{i-1} & \text{otherwise} \end{cases}$$

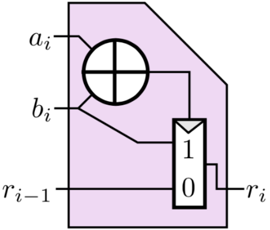
$$r_0 = \text{true}$$

$$r_n = ?$$

Interesting example: The comparison circuit

$$A = 0\ \underset{}{\overset{a_i}{(1)}}\ \overset{A_{i-1}}{(0\ 1\ \ldots\ 0\ 1)}$$

$$B = 1\ \underset{b_i}{(0)}\ \underset{B_{i-1}}{(0\ 1\ \ldots\ 1\ 0)}$$



comparator

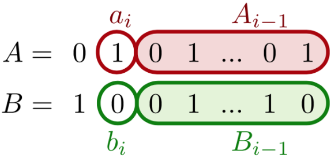
$$r_i := (A_i \leq B_i)$$

$$r_i = \left(\begin{array}{l} b_i \text{ when } a_i \neq b_i \\ r_{i-1} \text{ otherwise} \end{array} \right.$$

$$r_0 = \text{true}$$

$$r_n = ?$$

Interesting example: The comparison circuit

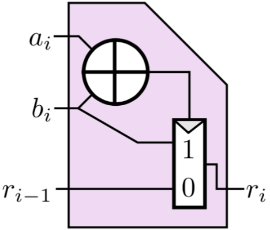


$$r_i := (A_i \leq B_i)$$

$$r_i = \begin{cases} b_i \text{ when } a_i \neq b_i \\ r_{i-1} \text{ otherwise} \end{cases}$$

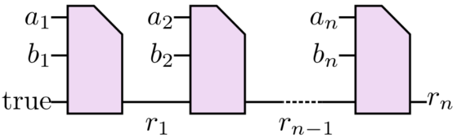
$$r_0 = \text{true}$$

$$r_n = ?$$



comparator

full circuit



Plan

inpher.io

1 Fully Homomorphic Encryption

2 Learning with error over the Torus

3 The framework Chimera

4 Application: feature selection

Reel/Complex polynomials

- $\mathbb{R}_N[X] = \mathbb{R}[X]/(X^N+1)$: the ring of polynomials with reel coefficients modulo X^N+1
- $\mathbb{C}_N[X] = \mathbb{C}[X]/(X^N+1)$: the ring of polynomials with complex coefficients modulo X^N+1

Examples: $N=2$

$(1.2+2.3X)\cdot(3.2+4.1X) = 3.84+12.28X+9.43X^2 = 12.28X-5.59 \mod (X^2+1)$

$(\mathbb{R}_N[X],+,\times)$ and $(\mathbb{C}_N[X],+,\times)$ are well defined as Ring

- ✔ $(\mathbb{R}_N[X],+)$ and $(\mathbb{C}_N[X],+)$ are groups
- ✔ It is a Ring: $x \times y$ is defined!

Coefficient and Slot packing

Coefficient packing

$$\mathbf{m} = \sum_{i=0}^{N-1} m_i \cdot X^i \qquad \sim \qquad \mathbf{m} = (m_0, m_1, \ldots, m_{N-1})$$

with $m_i \in \mathbb{C}$ for all $i = 0, 1, \ldots, N-1$

m_0	m_1	m_2	$\ldots$	m_{N-2}	m_{N-1}

Slot packing

$$X^N + 1 = \prod_{i=0}^{N-1} (X - \omega_i) \qquad \sim \qquad \mathbf{m} = (\mathbf{m}(\omega_0), \mathbf{m}(\omega_1), \ldots, \mathbf{m}(\omega_{N-1}))$$

with $\omega_i \in \mathbb{C}$ for all $i = 0, 1, \ldots, N-1$

$\mathbf{m}(\omega_0)$	$\mathbf{m}(\omega_1)$	$\mathbf{m}(\omega_2)$	$\cdots$	$\mathbf{m}(\omega_{N-2})$	$\mathbf{m}(\omega_{N-1})$

Coefficient and Slot packing

Coefficient packing

$$\mathbf{m} = \sum_{i=0}^{N-1} m_i \cdot X^i \quad \sim \quad \mathbf{m} = (m_0, m_1, \ldots, m_{N-1})$$

with $m_i \in \mathbb{C}$ for all $i = 0, 1, \ldots, N-1$

m_0	m_1	m_2	$\ldots$	m_{N-2}	m_{N-1}

Slot packing

$$X^N + 1 = \prod_{i=0}^{N-1} (X - \omega_i) \quad \sim \quad \mathbf{m} = (\mathbf{m}(\omega_0), \mathbf{m}(\omega_1), \ldots, \mathbf{m}(\omega_{N-1}))$$

with $\omega_i \in \mathbb{C}$ for all $i = 0, 1, \ldots, N-1$

$\mathbf{m}(\omega_0)$	$\mathbf{m}(\omega_1)$	$\mathbf{m}(\omega_2)$	$\ldots$	$\mathbf{m}(\omega_{N-2})$	$\mathbf{m}(\omega_{N-1})$

Morphism between coefficient and slot packing

Morphism

There exists morphism to switch between the coefficient and slot representation! (Vandermonde, DFT,...)

$$VDM = \begin{bmatrix} 1 & \omega_0^1 & \cdots & \omega_0^{N-1} \\ 1 & \omega_1^1 & \cdots & \omega_1^{N-1} \\ \vdots & \vdots & \cdots & \vdots \\ 1 & \omega_{N-1}^1 & \cdots & \omega_{N-1}^{N-1} \end{bmatrix}.$$

- A complex polynomial $\mod X^N + 1$ carries N complex slots.
- A real polynomial $\mod X^N + 1$ carries $N/2$ complex slots.

The VDM matrix is hermitian (orthonormal for the complex): slots are small $\Leftrightarrow$ coeffs are small.

Integer plaintext space

- $\mathbb{Z}_N[X] = \mathbb{Z}[X]/(X^N + 1)$: the ring of polynomials with integer coefficients module $X^N + 1$

Examples: $N = 2$

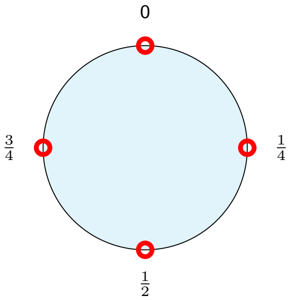
$(1 + 2X) \cdot (3 + 4X) = 3 + 10X + 8X^2 = 10X - 5 \mod (X^2 + 1)$

Attention, some additional constraints are needed to define slots

The torus $\mathbb{T}$

$(\mathbb{T}, +, \cdot) = \mathbb{R} \mod 1$ is a $\mathbb{Z}$-module ($\cdot : \mathbb{Z} \times \mathbb{T} \to \mathbb{T}$ a valid external product)

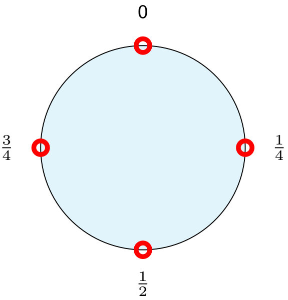
- ✔ It is a group $x + y \mod 1$, and $-x \mod 1$
- ✔ It is a $\mathbb{Z}$-module: $0 \cdot \frac{1}{2} = 0$ is defined!
- ✘ It is **not** a Ring: $0 \times \frac{1}{2}$ is **not** defined!



The torus $\mathbb{T}$

$(\mathbb{T}, +, \cdot) = \mathbb{R} \mod 1$ is a $\mathbb{Z}$-module ($\cdot : \mathbb{Z} \times \mathbb{T} \to \mathbb{T}$ a valid external product)

- ✔ It is a group $x + y \mod 1$, and $-x \mod 1$
- ✔ It is a $\mathbb{Z}$-module: $0 \cdot \frac{1}{2} = 0$ is defined!
- ✘ It is **not** a Ring: $0 \times \frac{1}{2}$ is **not** defined!

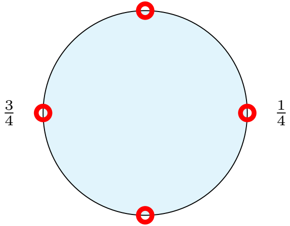


The torus $\mathbb{T}$

inpher.io

$(\mathbb{T}, +, \cdot) = \mathbb{R} \mod 1$ is a $\mathbb{Z}$-module ($\cdot : \mathbb{Z} \times \mathbb{T} \to \mathbb{T}$ a valid external product)

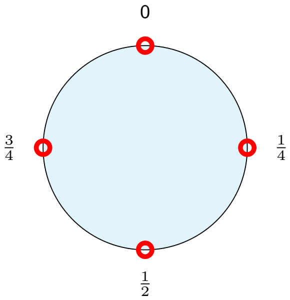
- ✔ It is a group $x + y \mod 1$, and $-x \mod 1$
- ✔ It is a $\mathbb{Z}$-module: $0 \cdot \frac{1}{2} = 0$ is defined!
- ✘ It is **not** a Ring: $0 \times \frac{1}{2}$ is **not** defined!



The torus $\mathbb{T}$

$(\mathbb{T}, +, \cdot) = \mathbb{R} \mod 1$ is a $\mathbb{Z}$-module ($\cdot : \mathbb{Z} \times \mathbb{T} \to \mathbb{T}$ a valid external product)

- ✔ It is a group $x + y \mod 1$, and $-x \mod 1$
- ✔ It is a $\mathbb{Z}$-module: $0 \cdot \frac{1}{2} = 0$ is defined!
- ✘ It is **not** a Ring: $0 \times \frac{1}{2}$ is **not** defined!



Torus polynomials $\mathbb{T}_N[X]$

$(\mathbb{T}_N[X], +, \cdot)$ is a $\mathbb{Z}_N[X]$-module

- Here, $\mathbb{Z}_N[X] = \mathbb{Z}[X] \mod (X^N + 1)$
- And $\mathbb{T}_N[X] = \mathbb{R}[X] \mod (X^N + 1) \mod 1$

Examples

- $(1 + 2X) \cdot (\frac{1}{3} + \frac{4}{7}X) = (\frac{4}{21} + \frac{5}{21}X) \mod (X^2 + 1) \mod 1$

Torus polynomials $\mathbb{T}_N[X]$

$(\mathbb{T}_N[X], +, \cdot)$ is a $\mathbb{Z}_N[X]$-module

- Here, $\mathbb{Z}_N[X] = \mathbb{Z}[X] \mod (X^N + 1)$
- And $\mathbb{T}_N[X] = \mathbb{R}[X] \mod (X^N + 1) \mod 1$

Examples

- $(1 + 2X) \cdot (\frac{1}{3} + \frac{4}{7}X) = (\frac{4}{21} + \frac{5}{21}X) \mod (X^2 + 1) \mod 1$

Torus polynomials $\mathbb{T}_N[X]$

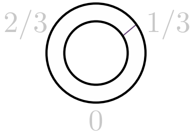
$(\mathbb{T}_N[X], +, \cdot)$ is a $\mathbb{Z}_N[X]$-module

- Here, $\mathbb{Z}_N[X] = \mathbb{Z}[X] \mod (X^N + 1)$
- And $\mathbb{T}_N[X] = \mathbb{R}[X] \mod (X^N + 1) \mod 1$

Examples

- $(1 + 2X) \cdot (\frac{1}{3} + \frac{4}{7}X) = (\frac{4}{21} + \frac{5}{21}X) \mod (X^2 + 1) \mod 1$

LWE Encryption over the torus ($\mathbb{T} = \mathbb{R}/\mathbb{Z} = \mathbb{R} \mod 1$)



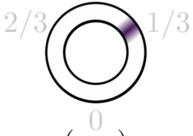
Example: $\mathcal{M} = \{0, 1/3, 2/3\}\ mod\ 1$

$\mu = 1/3\ mod\ 1 \in \mathcal{M}$

LWE Encryption over the torus ($\mathbb{T} = \mathbb{R}/\mathbb{Z} = \mathbb{R} \mod 1$)

inpher.io

	message	ciphertext	key	lin. combin.	product
TLWE	$\mathbb{T}$				



$(\;, \varphi)$

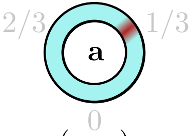
Example: $\mathcal{M} = \{0, 1/3, 2/3\}\ mod\ 1$
$\mu = 1/3\ mod\ 1 \in \mathcal{M}$

1. $\varphi = \mu +$ *Gaussian Error*
2. Random mask $\mathbf{a} \in \mathbb{T}^n$

LWE Encryption over the torus ($\mathbb{T} = \mathbb{R}/\mathbb{Z} = \mathbb{R} \mod 1$)

	message	ciphertext	key	lin. combin.	product
TLWE	$\mathbb{T}$	$\mathbb{T}^{n+1}$			

secret key: $\mathbf{s} \in \{0,1\}^n$



$(\mathbf{a}, \varphi)$

Example: $\mathcal{M} = \{0, 1/3, 2/3\} \ mod\ 1$

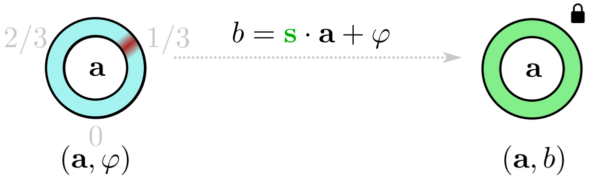
$\mu = 1/3 \ mod\ 1 \in \mathcal{M}$

1. $\varphi = \mu +$ *Gaussian Error*
2. Random mask $\mathbf{a} \in \mathbb{T}^n$

LWE Encryption over the torus ($\mathbb{T} = \mathbb{R}/\mathbb{Z} = \mathbb{R} \mod 1$)

	message	ciphertext	key	lin. combin.	product
TLWE	$\mathbb{T}$	$\mathbb{T}^{n+1}$			

secret key: $\mathbf{s} \in \{0,1\}^n$



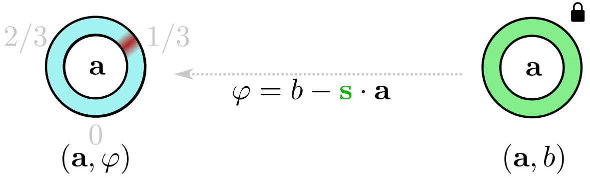
Example: $\mathcal{M} = \{0, 1/3, 2/3\}\ mod\ 1$
$\mu = 1/3\ mod\ 1 \in \mathcal{M}$

1. $\varphi = \mu +$ *Gaussian Error*
2. Random mask $\mathbf{a} \in \mathbb{T}^n$

LWE Encryption over the torus ($\mathbb{T} = \mathbb{R}/\mathbb{Z} = \mathbb{R} \mod 1$)

	message	ciphertext	key	lin. combin.	product
TLWE	$\mathbb{T}$	$\mathbb{T}^{n+1}$	$\mathbb{B}^n$		

secret key: $\mathbf{s} \in \{0,1\}^n$



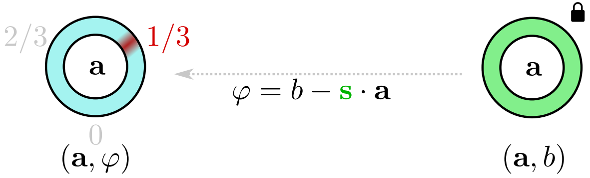
Example: $\mathcal{M} = \{0, 1/3, 2/3\}$ mod 1
$\mu = 1/3$ mod $1 \in \mathcal{M}$

1. Unlock the representation $(\mathbf{a}, \varphi)$
2. Round φ to the nearest message $\mu \in \mathcal{M}$

LWE Encryption over the torus ($\mathbb{T} = \mathbb{R}/\mathbb{Z} = \mathbb{R} \mod 1$)

	message	ciphertext	key	lin. combin.	product
TLWE	$\mathbb{T}$	$\mathbb{T}^{n+1}$	$\mathbb{B}^n$		

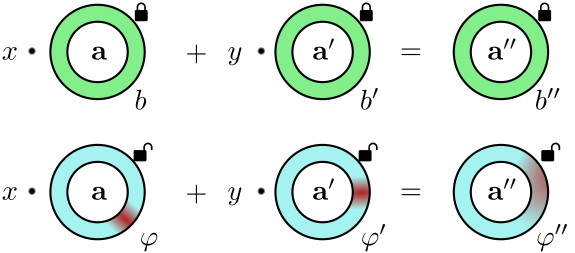
secret key: $\mathbf{s} \in \{0,1\}^n$



1. Unlock the representation $(\mathbf{a}, \varphi)$
2. Round φ to the nearest message $\mu \in \mathcal{M}$

LWE Encryption over the torus

	message	ciphertext	key	lin. combin.	product
TLWE	$\mathbb{T}$	$\mathbb{T}^{n+1}$	$\mathbb{B}^n$		
TRLWE	$\mathbb{T}_N[X]$	$\mathbb{T}_N[X]^{k+1}$	$\mathbb{B}_N[X]^k$		



$$\mathbf{a}'' = x \cdot \mathbf{a} + y \cdot \mathbf{a}'$$

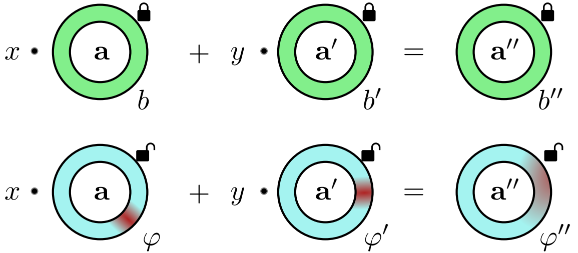
$$b'' = x \cdot b + y \cdot b'$$

$$\varphi'' = x \cdot \varphi + y \cdot \varphi'$$

$\alpha = \mathrm{stdev}(\varphi)$ $\quad \alpha'$ $\quad \alpha''$ $\quad \alpha''^2 = x^2\alpha^2 + y^2\alpha'^2$

LWE Encryption over the torus

	message	ciphertext	key	lin. combin.	product
TLWE	$\mathbb{T}$	$\mathbb{T}^{n+1}$	$\mathbb{B}^n$	✔	✘
TRLWE	$\mathbb{T}_N[X]$	$\mathbb{T}_N[X]^{k+1}$	$\mathbb{B}_N[X]^k$	✔	✘



$$\mathbf{a}'' = x \cdot \mathbf{a} + y \cdot \mathbf{a}'$$

$$b'' = x \cdot b + y \cdot b'$$

$$\varphi'' = x \cdot \varphi + y \cdot \varphi'$$

$\alpha = \mathrm{stdev}(\varphi)$ $\quad \alpha'$ $\quad \alpha''$ $\quad \alpha''^2 = x^2\alpha^2 + y^2\alpha'^2$

	message	ciphertext	key	lin. combin.	product
TLWE	$\mathbb{T}$	$\mathbb{T}^{n+1}$	$\mathbb{B}^n$	✔	✘
TRLWE	$\mathbb{T}_N[X]$	$\mathbb{T}_N[X]^{k+1}$	$\mathbb{B}_N[X]^k$	✔	✘
TRGSW	$\mathbb{Z}_N[X]$	ℓ-vector of TRLWE	$\mathbb{B}_N[X]^k$		

TR(GSW) ciphertexts of $\mu \in \mathbb{Z}_N[X]$

$$\text{TRGSW}(\mu) = \begin{pmatrix} \text{TRLWE}_K(K \cdot \frac{\mu}{2}) \\ \text{TRLWE}_K(K \cdot \frac{\mu}{4}) \\ \text{TRLWE}_K(K \cdot \frac{\mu}{8}) \\ \hline \text{TRLWE}_K(1 \cdot \frac{\mu}{2}) \\ \text{TRLWE}_K(1 \cdot \frac{\mu}{4}) \\ \text{TRLWE}_K(1 \cdot \frac{\mu}{8}) \end{pmatrix}$$

1. **Internal Product (classical)**: $\boxtimes$: TRGSW $\times$ TRGSW $\longrightarrow$ TRGSW
2. **External product (Asiacrypt 2016)**: $\boxdot$: TRGSW $\times$ TRLWE $\longrightarrow$ TRLWE

$$(\mu_A, \mu_{\mathbf{b}}) \longmapsto \mu_A \cdot \mu_{\mathbf{b}}$$
$$(\epsilon_A, \epsilon_{\mathbf{b}}) \longmapsto ||\mu_A||_1 * \epsilon_{\mathbf{b}} + O(\epsilon_A)$$

If $||\mu_A||_1 = 1$ the noise propagation is linear!

	message	ciphertext	key	lin. combin.	product
TLWE	$\mathbb{T}$	$\mathbb{T}^{n+1}$	$\mathbb{B}^n$	✔	✘
TRLWE	$\mathbb{T}_N[X]$	$\mathbb{T}_N[X]^{k+1}$	$\mathbb{B}_N[X]^k$	✔	✘
TRGSW	$\mathbb{Z}_N[X]$	ℓ-vector of TRLWE	$\mathbb{B}_N[X]^k$	✔	✔

TR(GSW) ciphertexts of $\mu \in \mathbb{Z}_N[X]$

$$\text{TRGSW}(\mu) = \begin{pmatrix} \text{TRLWE}_K(K \cdot \frac{\mu}{2}) \\ \text{TRLWE}_K(K \cdot \frac{\mu}{4}) \\ \text{TRLWE}_K(K \cdot \frac{\mu}{8}) \\ \hline \text{TRLWE}_K(1 \cdot \frac{\mu}{2}) \\ \text{TRLWE}_K(1 \cdot \frac{\mu}{4}) \\ \text{TRLWE}_K(1 \cdot \frac{\mu}{8}) \end{pmatrix}$$

1. **Internal Product (classical)**: $\boxtimes$: TRGSW $\times$ TRGSW $\longrightarrow$ TRGSW
2. **External product (Asiacrypt 2016)**: $\boxdot$: TRGSW $\times$ TRLWE $\longrightarrow$ TRLWE
   $(\mu_A, \mu_{\mathbf{b}}) \longmapsto \mu_A \cdot \mu_{\mathbf{b}}$
   $(\epsilon_A, \epsilon_{\mathbf{b}}) \longmapsto ||\mu_A||_1 * \epsilon_{\mathbf{b}} + O(\epsilon_A)$

If $||\mu_A||_1 = 1$ the noise propagation is linear!

	message	ciphertext	key	lin. combin.	product
TLWE	$\mathbb{T}$	$\mathbb{T}^{n+1}$	$\mathbb{B}^n$	✔	✘
TRLWE	$\mathbb{T}_N[X]$	$\mathbb{T}_N[X]^{k+1}$	$\mathbb{B}_N[X]^k$	✔	✘
TRGSW	$\mathbb{Z}_N[X]$	ℓ-vector of TRLWE	$\mathbb{B}_N[X]^k$	✔	✔

TR(GSW) ciphertexts of $\mu \in \mathbb{Z}_N[X]$

$$\text{TRGSW}(\mu) = \begin{pmatrix} \text{TRLWE}_K(K \cdot \frac{\mu}{2}) \\ \text{TRLWE}_K(K \cdot \frac{\mu}{4}) \\ \text{TRLWE}_K(K \cdot \frac{\mu}{8}) \\ \hline \text{TRLWE}_K(1 \cdot \frac{\mu}{2}) \\ \text{TRLWE}_K(1 \cdot \frac{\mu}{4}) \\ \text{TRLWE}_K(1 \cdot \frac{\mu}{8}) \end{pmatrix}$$

1. **Internal Product (classical)**: $\boxtimes$: TRGSW $\times$ TRGSW $\longrightarrow$ TRGSW
2. **External product (Asiacrypt 2016)**: $\boxdot$: TRGSW $\times$ TRLWE $\longrightarrow$ TRLWE

$$(\mu_A, \mu_{\mathbf{b}}) \longmapsto \mu_A \cdot \mu_{\mathbf{b}}$$
$$(\epsilon_A, \epsilon_{\mathbf{b}}) \longmapsto ||\mu_A||_1 * \epsilon_{\mathbf{b}} + O(\epsilon_A)$$

If $||\mu_A||_1 = 1$ the noise propagation is linear!

Plan

1. Fully Homomorphic Encryption
2. Learning with error over the Torus
3. The framework Chimera
4. Application: feature selection

How choose the homomorphic scheme?

inpher.io

Strengths of HE libraries

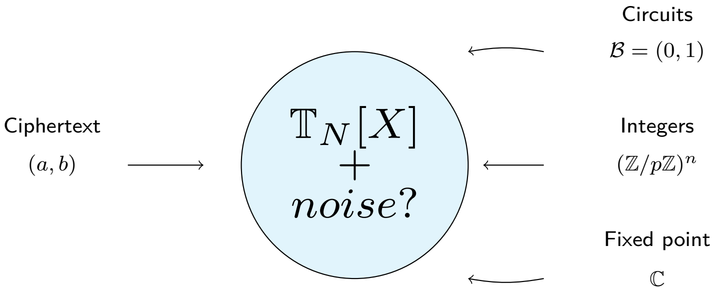
- BGV/Helib: SIMD finite field arithmetic
- B/FV, Seal: SIMD vector $\mod p$
- HEAAN: SIMD fixed point arithmetic
- TFHE: single evaluation, boolean logic, comparison, threshold, complex circuits
- etc...

How to get all the benefits without the limitations?

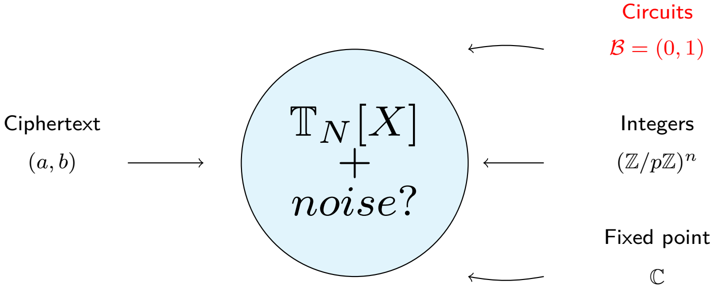
Solution: Chimera

- Unified plaintext space over the Torus
- Switch between ciphertext representations
- Implement bridges between TFHE, B/FV and HEAAN

How we can represent all plaintexts over the $\mathbb{T}_N[X]$?

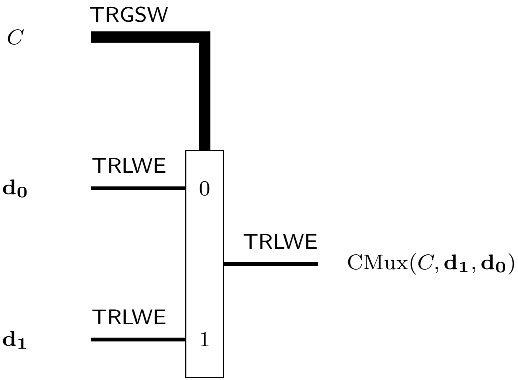


Circuit


Ciphertext
(a, b)
$\mathbb{T}_N[X]$
$+$
$noise?$
Circuits
$\mathcal{B} = (0, 1)$
Integers
$(\mathbb{Z}/p\mathbb{Z})^n$
Fixed point
$\mathbb{C}$

Circuit: CMux

$$\mathrm{CMux}(C, d_1, d_0) = C \boxdot (d_1 - d_0) + d_0$$



LUT evaluation

LookUp Tables (LUT) to evaluate arbitrary functions:

$$f : \mathbb{B}^d \longrightarrow \mathbb{T}^s$$
$$x = (x_0, \ldots, x_{d-1}) \longmapsto f(x) = (f_0(x), \ldots, f_{s-1}(x))$$

Example with $d = 3$ and $s = 2$

x_0	x_1	x_2	f_0	f_1
0	0	0	0.5	0.3
1	0	0	0.25	0.7
0	1	0	0.1	0.61
1	1	0	0.83	0.9
0	0	1	0.23	0.47
1	0	1	0.67	0.42
0	1	1	0.78	0.12
1	1	1	0.35	0.95

Evaluation via MUX tree

LUT evaluation

LookUp Tables (LUT) to evaluate arbitrary functions:

$$f : \mathbb{B}^d \longrightarrow \mathbb{T}^s$$
$$x = (x_0, \ldots, x_{d-1}) \longmapsto f(x) = (f_0(x), \ldots, f_{s-1}(x))$$

Example with $d = 3$ and $s = 2$

x_0	x_1	x_2	f_0	f_1
0	0	0	0.5	0.3
1	0	0	0.25	0.7
0	1	0	0.1	0.61
1	1	0	0.83	0.9
0	0	1	0.23	0.47
1	0	1	0.67	0.42
0	1	1	0.78	0.12
1	1	1	0.35	0.95

Evaluation via MUX tree

LUT evaluation

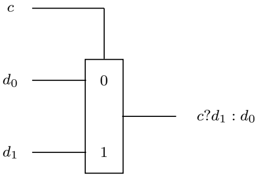
LookUp Tables (LUT) to evaluate arbitrary functions:

$$f : \mathbb{B}^d \longrightarrow \mathbb{T}^s$$
$$x = (x_0, \ldots, x_{d-1}) \longmapsto f(x) = (f_0(x), \ldots, f_{s-1}(x))$$

Example with $d = 3$ and $s = 2$

x_0	x_1	x_2	f_0	f_1
0	0	0	0.5	0.3
1	0	0	0.25	0.7
0	1	0	0.1	0.61
1	1	0	0.83	0.9
0	0	1	0.23	0.47
1	0	1	0.67	0.42
0	1	1	0.78	0.12
1	1	1	0.35	0.95

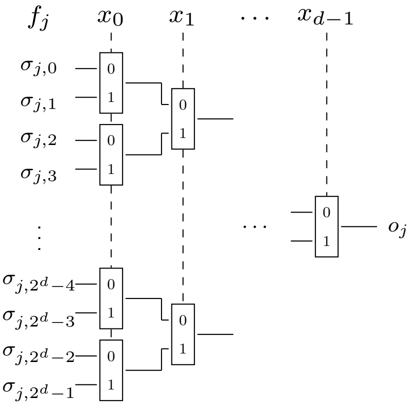
Evaluation via MUX tree



LUT evaluation

How to evaluate it?

x_0	$\cdots$	x_{d-1}	f_0	$\cdots$	f_{s-1}
0	$\cdots$	0	$\sigma_{0,0}$	$\cdots$	$\sigma_{s-1,0}$
1	$\cdots$	0	$\sigma_{0,1}$	$\cdots$	$\sigma_{s-1,1}$
0	$\cdots$	0	$\sigma_{0,2}$	$\cdots$	$\sigma_{s-1,2}$
1	$\cdots$	0	$\sigma_{0,3}$	$\cdots$	$\sigma_{s-1,3}$
$\vdots$	$\cdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
0	$\cdots$	1	$\sigma_{0,2^d-4}$	$\cdots$	$\sigma_{s-1,2^d-4}$
1	$\cdots$	1	$\sigma_{0,2^d-3}$	$\cdots$	$\sigma_{s-1,2^d-3}$
0	$\cdots$	1	$\sigma_{0,2^d-2}$	$\cdots$	$\sigma_{s-1,2^d-2}$
1	$\cdots$	1	$\sigma_{0,2^d-1}$	$\cdots$	$\sigma_{s-1,2^d-1}$



LUT evaluation: Batching and Packing

Packing data in TRLWE

- TLWE: messages $m \in \mathbb{T}$
- TRLWE: messages $\mathbf{m} \in \mathbb{T}_N[X]$

$$\mathbf{m} = \sum_{i=0}^{N-1} m_i \cdot X^i \qquad \sim \qquad \mathbf{m} = (m_0, m_1, \ldots, m_{N-1})$$

with $m_i \in \mathbb{T}$ for all $i = 0, 1, \ldots, N-1$

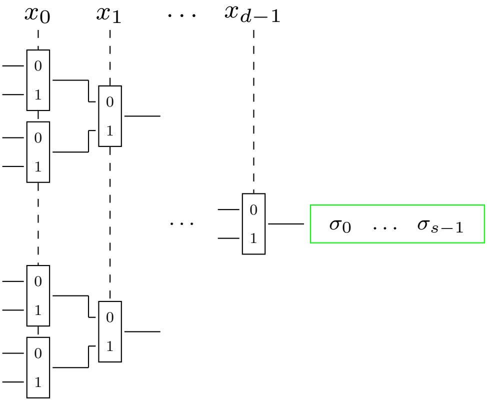
m_0	m_1	m_2	$\ldots$	m_{N-2}	m_{N-1}

LUT evaluation: Batching and Vertical Packing

Batching (Horizontal Packing)

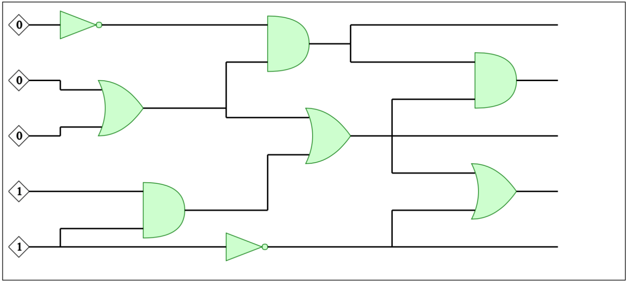
- Pack the outputs in a TRLWE ciphertext (green box)

x_0	$\cdots$	x_{d-1}	f_0	$\cdots$	f_{s-1}
0	$\cdots$	0	$\sigma_{0,0}$	$\cdots$	$\sigma_{s-1,0}$
1	$\cdots$	0	$\sigma_{0,1}$	$\cdots$	$\sigma_{s-1,1}$
0	$\cdots$	0	$\sigma_{0,2}$	$\cdots$	$\sigma_{s-1,2}$
1	$\cdots$	0	$\sigma_{0,3}$	$\cdots$	$\sigma_{s-1,3}$
$\vdots$	$\cdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
0	$\cdots$	1	$\sigma_{0,2^d-4}$	$\cdots$	$\sigma_{s-1,2^d-4}$
1	$\cdots$	1	$\sigma_{0,2^d-3}$	$\cdots$	$\sigma_{s-1,2^d-3}$
0	$\cdots$	1	$\sigma_{0,2^d-2}$	$\cdots$	$\sigma_{s-1,2^d-2}$
1	$\cdots$	1	$\sigma_{0,2^d-1}$	$\cdots$	$\sigma_{s-1,2^d-1}$



The noise in FHE

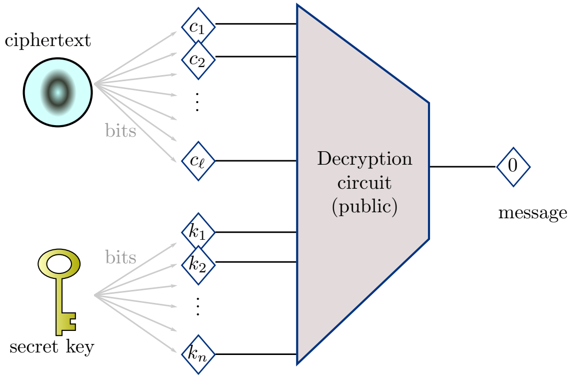
inpher.io



Animation Circuit

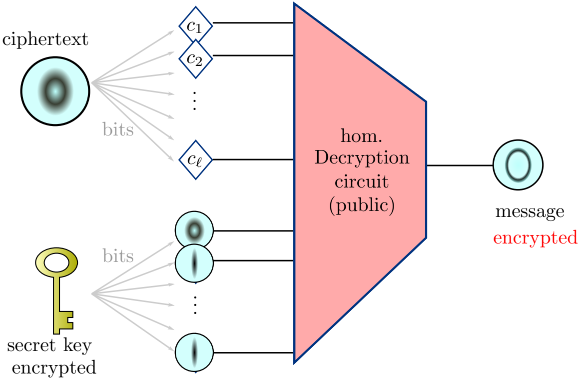
Bootstrapping

Gentry's breakthtough idea : refresh the ciphertext by evaluating the decryption circuit homomorphically (using the decryption key bits in encrypted form).



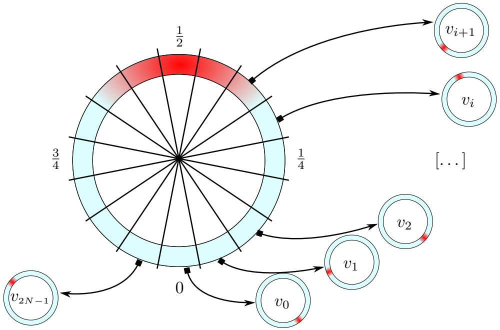
Bootstrapping

Gentry's breakthtough idea : refresh the ciphertext by evaluating the decryption circuit homomorphically (using the decryption key bits in encrypted form).



Gate Bootstrapping (TLWE to TLWE)

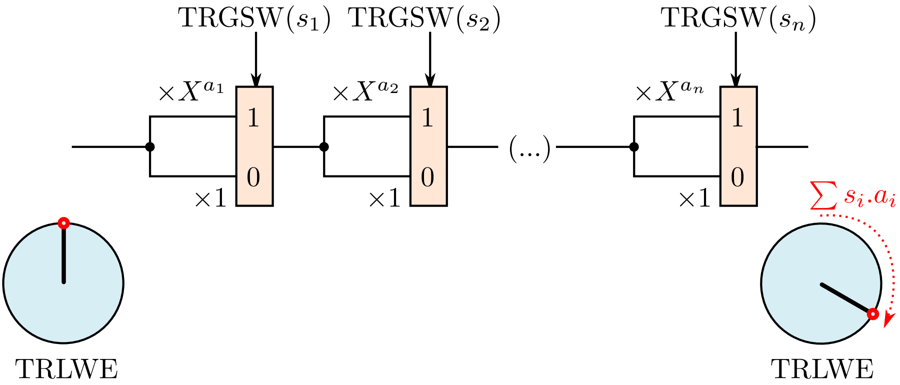
inpher.io



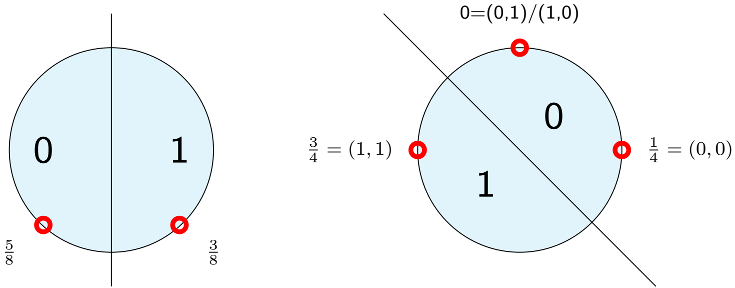
Bootstrapping algorithm of $(\mathbf{a}, b)$

1. Start from (a trivial) TRLWE ciphertext of message
   $v_0 + v_1 X + \cdots + v_{N-1} X^{N-1}$
   N coefs mod $X^N + 1$ can be viewed as $2N$ coefs mod $X^{2N} - 1$ s.t. $v_{N+i} = -v_i$
2. Rotate it by $t = -\varphi_s(\mathbf{a}, b)$ positions using external product.
3. Extract the constant term (which encrypts v_p).

Circuit: Blindrotate



Exemple AND

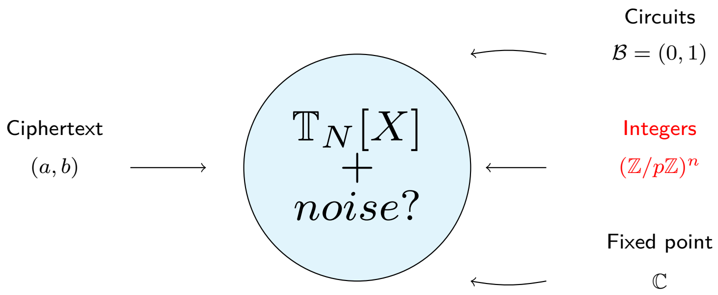


AND

Sum + BlindRotate

NAND, OR, NOT ...

Integers


Ciphertext
(a, b)
$\mathbb{T}_N[X]$
$+$
$noise?$
Circuits
$\mathcal{B} = (0, 1)$
Integers
$(\mathbb{Z}/p\mathbb{Z})^n$
Fixed point
$\mathbb{C}$

BFV scheme (encoding)

inpher.io

- $\mathbb{Z}_N[X] \mod p$: the ring of polynomials with integer $\mod p$ coefficients module $X^N + 1$
- If $X^N + 1$ has N roots mod p, $\mathbb{Z}/p\mathbb{Z}^N$ is isomorphic to $\mathbb{Z}_N[X] \mod p$ (analogue of the complex slots, but mod p).

Examples: $N = 2$, $p = 5$

- **coeffs:** $(1 + X) \cdot (3 + 4X) = 3 + 7X + 4X^2 = 4 + 2X \mod (X^2 + 1) \mod 5$
- Roots of $X^2 + 1 \mod 5$: green: X=2, blue: X=3
- **slots:** $[3, 4] \cdot [1, 0] = [3, 0] \mod 5$

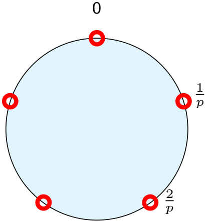
Coefficient to slot representation

- In BFV: p should verify some conditions (never power of 2)
- In BGV: any p (work in extended fields)

BFV scheme (encoding)

$$(\mathbb{Z}/p\mathbb{Z})^N \simeq \mathbb{Z}_N[X] \mod p \simeq \frac{1}{p}\mathbb{Z}_N[X] \mod 1$$

The plaintext space $\mathcal{M}$ is composed by exact multiples of $\frac{1}{p}$.



Plaintext addition $(\mu_1(X), \mu_2(X))$

$\mu_1(X) + \mu_2(X) := \mu_1(X) + \mu_2(X) \mod 1.$

Plaintext product (Montgomery) $(\mu_1(X), \mu_2(X))$

$\mu_1(X) \boxtimes_p \mu_2(X) := p \cdot \mu_1(X) \cdot \mu_2(X) \mod 1.$

Problem of lift

inpher.io

Examples: $p = 3$, $\mu_1 = \frac{1}{3}$ and $\mu_1 = \frac{2}{3}$

- Exact product: $3(I_1 + \frac{1}{3})(I_2 + \frac{2}{3}) = I + \frac{2}{3} = +\frac{2}{3} \mod 1$, for all I_1, I_2 integers
- Product with noise and small element: $3 * 5.33333 * 10.66665 = 170.6662$
- Product with noise and big element: $3 * 12345678.33333 * 7654321.66665 = -.839\ldots$

- We need a small representative of the plaintext to keep the result correct.
- We should lift the ciphertext to small representative in $\mathbb{R}[X]$ (all coefficients in $[-1/2, 1/2)$).
- $\frac{1}{p} \gg noise$

Homomorphic operations

inpher.io

Homomorphic addition $c_1 = (a_1, b_1)$, $c_2 = (a_2, b_2)$

$(a, b) = (a_1 + a_2, b_1 + b_2)$

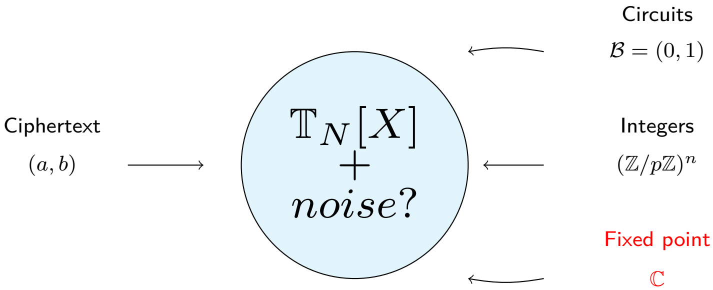
Homomorphic product $c_1 = (a_1, b_1), c_2 = (a_2, b_2)$

$$p(b_1 - s.a_1)(b_2 - s.a_2) = \underbrace{(p.b_1.b_2)}_{C_0} - s.\underbrace{(p.a_1.b_2 + p.a_2.b_1)}_{C_1} + s^2.\underbrace{(p.a_1.a_2)}_{C_2}$$
$$= (b - s.a)$$

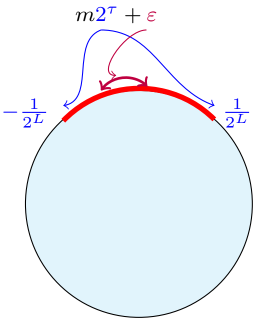
Relinearize the term $(p.a_1.a_2)s^2$ using the external product:

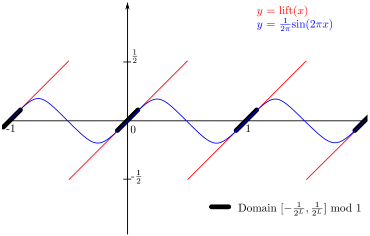
$$c_1 \boxtimes_p c_2 = (C_1, C_0) - TRGSW(s) \boxdot (C_2, 0)$$

Fixed point


Ciphertext
(a, b)
$\mathbb{T}_N[X]$
$+$
$noise?$
Circuits
$\mathcal{B} = (0, 1)$
Integers
$(\mathbb{Z}/p\mathbb{Z})^n$
Fixed point
$\mathbb{C}$

HEAAN

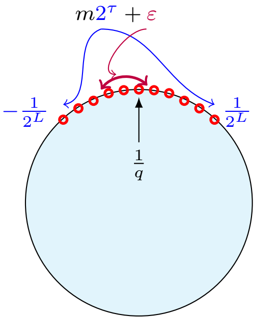




Continuous approach

- $x \times y = \text{Lift}(x) * \text{Lift}(y) \mod 1.$
- ✔ This approach can preserve (or reduce) the interval $[-\frac{1}{2^L}, \frac{1}{2^L}]$
- ✔ Lift is a periodic function: approx by sinus (or other Fourier serie) wherever it matters...
- ✘ ...but sinus can only be approx by a polynomial, which recursively requires a product.

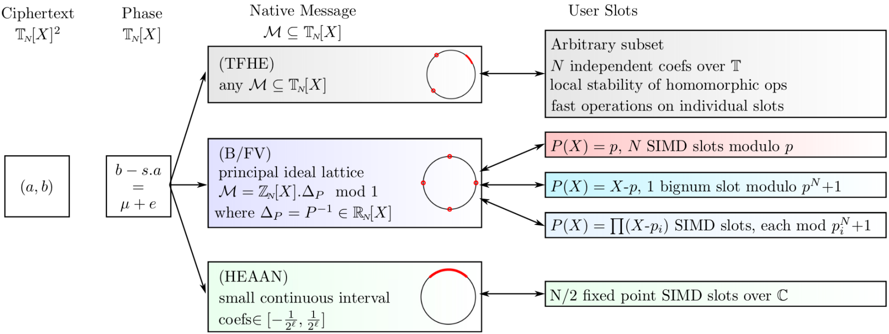
Fixed point: HEAAN



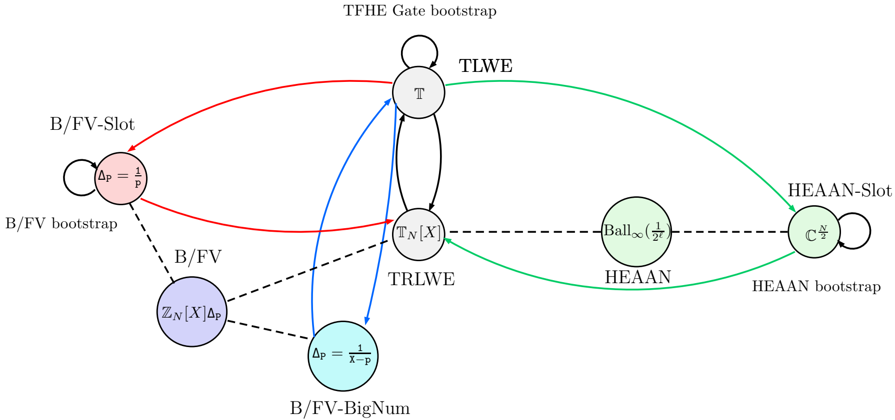
Discrete approach

- round a, b (and thus μ) on exact multiples of $\frac{1}{q}$ where $q \approx 2^{L+\rho}$.
- ✔ Brings us in the ring $\frac{1}{q}\mathbb{Z}_N[X] \mod 1$ (avoids lifting)
- ✔ Exact Montgomery product $q(b_1 - sa_1)(b_2 - sa_2)$
- ✘ Blows up the interval $[-\frac{1}{2^L}, \frac{1}{2^L}] \to [-\frac{1}{2^{L-\rho}}, \frac{1}{2^{L-\rho}}]$...
  ...works a leveled number of times.

Unifying the plaintext space in RLWE-schemes



Bridges between LWE based schemes



Plan

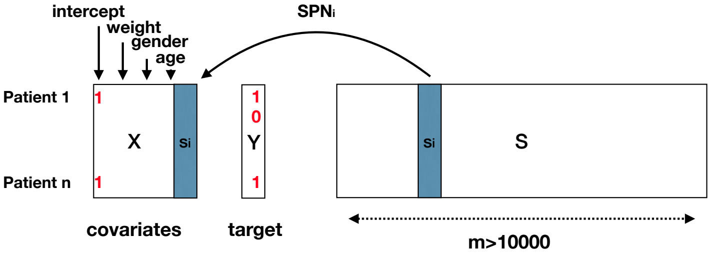
1. Fully Homomorphic Encryption
2. Learning with error over the Torus
3. The framework Chimera
4. Application: feature selection

Application Idash

inpher.io

Goal:

Develop a secure parallel outsourcing solution to compute Genome Wide Association Studies (GWAS) based on logistic regression using **homomorphically encrypted** data.



Input:

- $X \in \mathcal{M}_{n,k+1}(\mathbb{R})$ input matrix
- $y \in \mathbb{B}^n$ binary vector
- $S \in \mathcal{M}_{n,m}(\mathbb{R})$ assumed binary

Output:

- $stat \in \mathbb{R}^m$

Key points of our solution:

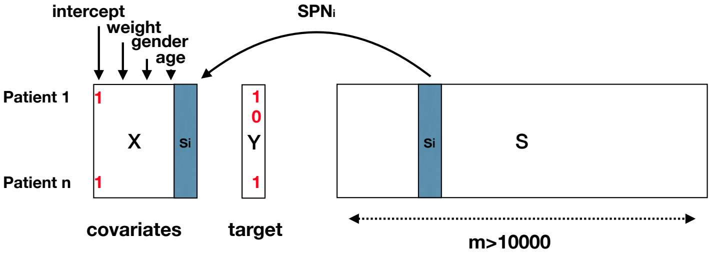
- Make plaintext algorithm FHE friendly
- Use hybrid homomorphic encryption

Application Idash

inpher.io

Goal:

Develop a secure parallel outsourcing solution to compute Genome Wide Association Studies (GWAS) based on logistic regression using **homomorphically encrypted** data.



Input:

- $X \in \mathcal{M}_{n,k+1}(\mathbb{R})$ input matrix
- $y \in \mathbb{B}^n$ binary vector
- $S \in \mathcal{M}_{n,m}(\mathbb{R})$ assumed binary

Output:

- $stat \in \mathbb{R}^m$

Key points of our solution:

- Make plaintext algorithm FHE friendly
- Use hybrid homomorphic encryption

Algorithm in plaintext

Algorithm 2 Plaintext algorithm

$\beta^{(0)} = (0, \ldots, 0)$
for $t = 1$ to iters **do**
    $\beta^{(t)} \leftarrow \beta^{(t-1)} + \text{step} \cdot X^T . \left(\mathbf{y} - \sigma(X\beta^{(t-1)})\right)$
end for ▷ logreg
$\mathbf{p} \leftarrow \sigma(X\beta^{(\text{iters})})$
$\mathbf{z}^* \leftarrow (\mathbf{y} - \mathbf{p})^T . S$ ▷ numerator
$W \leftarrow \text{diag}(p * (1 - p))$
$G \leftarrow X^T . W . X \approx \frac{1}{4} * Id$ (assumed that X orthogonal)
$A \leftarrow X^T . W . S$
$\mathbf{s}^{*\mathbf{2}} = \text{colsums}(W \cdot (S \odot S)) - \text{colsums}(A \odot G^{-1} A)$
$(\approx A[0] * \sqrt{n} - 4 * \text{colsums}(A \odot A))$ ▷ denominator
$\mathbf{r}_i = [2 \cdot \log(|\mathbf{z}^*{}_i|) - \log(|\mathbf{s}^{*\mathbf{2}}{}_i|)]$ for each $i \in [1, m]$ ▷ log of stat

$$stat_i = \frac{\mathbf{z}^*{}_i}{\sqrt{\mathbf{s}^*{}_i^2}} = \frac{1}{2}\exp(\mathbf{r}_i)$$

$$\text{p-value}_i = \text{p-Norm}(stat_i)$$

Algorithm in plaintext

Algorith

for loops
(better with fast bootstrapping)

$\beta^{(0}$

for $t = 1$ to iters **do**

$\beta^{(t)} \leftarrow \beta^{(t-1)} + \text{step} \cdot X^T . \left(\mathbf{y} - \sigma(X\beta^{(t-1)})\right)$

en ▷ logreg

continuous non-polynomial functions
(Approx numbers, or Lookup tables)

$\mathbf{p}$

$\mathbf{z}^*$ ▷ numerator

$W \leftarrow \text{diag}(p * (1 - p))$

$G \leftarrow X^T.W.X \approx \frac{1}{4} * Id$ (assumed that X orthogonal)

$A \leftarrow X^T.W.S$

$\mathbf{s}^{*\mathbf{2}} = \text{colsums}(W \cdot (S \odot S)) - \text{colsums}(A \odot G^{-1}A)$

$(\approx A[0] * \sqrt{n} - 4 * \text{colsums}(A \odot A))$ ▷ denominator

$\mathbf{r}_i = [2 \cdot \log(|\mathbf{z}^*{}_i|) - \log(|\mathbf{s}^{*\mathbf{2}}{}_i|)]$ for each $i \in [1, m]$ ▷ log of stat

$$stat_i = \frac{\mathbf{z}^*{}_i}{\sqrt{\mathbf{s}^{*2}_i}} = \frac{1}{2}\exp(\mathbf{r}_i)$$

$$\text{p-value}_i = \text{p-Norm}(stat_i)$$

Algorithm in plaintext

inpher.io

Algorit

for loops
(better with fast bootstrapping)

1: $\beta^{(0}$
2: **for** $t = 1$ to iters **do**
3: $\beta^{(t)} \leftarrow \beta^{(t-1)} + \text{step} \cdot X^T . \left(\mathbf{y} - \sigma(X\beta^{(t-1)})\right)$
4: **en**

individual non-linear operations in small dimension
(lookup tables)

continuous non-polynomial functions
(Approx numbers, or Lookup tables)

5: $\mathbf{p}$
6: $\mathbf{z}^*$ ▷ numerator
7: $W \leftarrow$ diag
8: $G \leftarrow X^T.$ nal)
9: $A \leftarrow X^T.W.S$

multiplication with fresh ciphertexts
(better with TFHE's external product)

10: $\mathbf{s}^{*\mathbf{2}} = \text{colsums}(W \cdot (S \odot S)) - \text{colsums}(A \odot G^{-1}A)$
11: $(\approx A[0] * \sqrt{n} - 4 * \text{colsums}(A \odot A))$ ▷ denominator
12: $\mathbf{r}_i = [2 \cdot \log(|\mathbf{z}^*{}_i|) - \log(|\mathbf{s}^{*\mathbf{2}}{}_i|)]$ for each $i \in [1, m]$ ▷ log of stat

$$stat_i = \frac{\mathbf{z}^*{}_i}{\sqrt{\mathbf{s}^{*2}_i}} = \frac{1}{2}\exp(\mathbf{r}_i)$$

$$\text{p-value}_i = \text{p-Norm}(stat_i)$$

Algorithm in plaintext

inpher.io

Algorit

for loops
(better with fast bootstrapping)

$\beta^{(0)}$
for $t = 1$ to iters **do**
$\quad \beta^{(t)} \leftarrow \beta^{(t-1)} + \text{step} \cdot X^T . \left(\mathbf{y} - \sigma(X\beta^{(t-1)})\right)$
end

continuous non-polynomial functions
(Approx numbers, or Lookup tables)

individual non-linear operations in small dimension
(lookup tables)

$\mathbf{p}$
$\mathbf{z}^*$ ▷ numerato
$W \leftarrow$ diag
$G \leftarrow X^T.W$ … nal)
$A \leftarrow X^T.W.S$

multiplication with fresh ciphertexts
(better with TFHE's external product)

$\mathbf{s}^{*\mathbf{2}} = \text{colsums}(W \cdot (S \odot S)) - \text{colsums}(A \odot G^{-1}A)$
$(\approx A[0] * \sqrt{n} - 4 * \text{colsums}(A \odot A))$
$\mathbf{r}_i = [2 \cdot \log(|\mathbf{z}^*{}_i|) - \log(|\mathbf{s}^{*\mathbf{2}}{}_i|)]$ for each $i \in [1, m]$ ▷ log of stat

very large dimension
(fully packed SIMD)

$$stat_i = \frac{\mathbf{z}^*{}_i}{\sqrt{\mathbf{s}^*{}_i^2}} = \frac{1}{2}\exp(\mathbf{r}_i)$$

$$\text{p-value}_i = \text{p-Norm}(stat_i)$$

continuous function batched on a large vector

Algorithm in plaintext

inpher.io

Algorith

for loops
(better with fast bootstrapping)

$\beta^{(0}$

for $t = 1$ to iters **do**

$\beta^{(t)} \leftarrow \beta^{(t-1)} + \text{step} \cdot X^T . \left(\mathbf{y} - \sigma(X\beta^{(t-1)})\right)$

en

continuous non-polynomial functions
(Approx numbers, or Lookup tables)

individual non-linear operations in small dimension
(lookup tables)

$\mathbf{p}$

$\mathbf{z}^*$ ▷ numerato

$W \leftarrow$ diag(

multiplication with fresh ciphertexts
(better with TFHE's external product)

$G \leftarrow X^T$. nal)

$A \leftarrow X^T.W.$

$\mathbf{s}^{*\mathbf{2}} = \text{colsums}(W \cdot (S \odot S)) - \text{colsums}(A \odot G^{-1}A)$

$(\approx A[0] * \sqrt{n} - 4 * \text{colsums}(A \odot A))$

very large dimension
(fully packed SIMD)

$\mathbf{r}_i = [2 \cdot \log(|\mathbf{z}^*{}_i|) - \log(|\mathbf{s}^{*\mathbf{2}}{}_i|)]$ for each $i \in [1, m]$ ▷ log of stat

$$stat_i = \frac{\mathbf{z}^*{}_i}{\sqrt{\mathbf{s}^*{}^2_i}} = \frac{1}{2}\exp(\mathbf{r}_i)$$

$$\text{p-value}_i = \text{p-Norm}(stat_i)$$

continuous function batched on a large vector

Which fully homomoprhic scheme should we choose?

Hybrid homomorphic encryption: Chimera

1. Initial Logreg on matrix X and vector y
   - adapt lib TFHE + logreg
2. Large-scale linear algebra computations
   - implement Chimera (version 2 of TFHE)
3. Batch Logarithm computation
   - adapt lib HEAAN

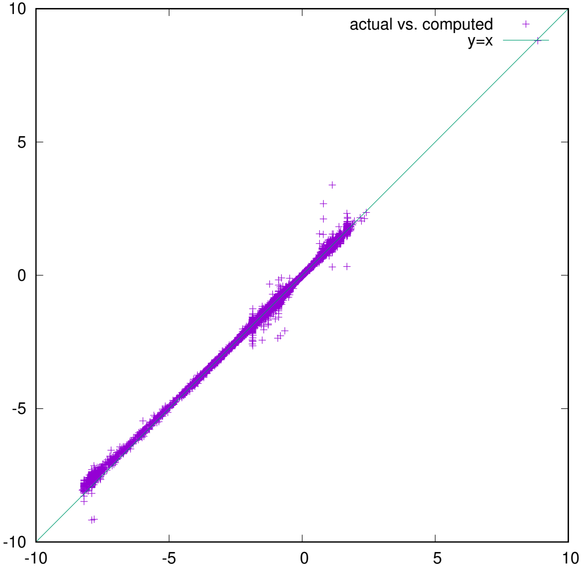
Benchmarks

inpher.io

Steps	Timing (4 cores)	Timing (96 cores)	RAM
KeyGen	5.5 mins	2.0 mins	4.4 GB
Encryption	7.2 mins	1.3 mins	8.6 GB
Cloud Computation	3h06	10.2 mins	7.8 GB

- Input ciphertext: 5GB (enc X, y, S)
- Final ciphertext: 640KB (enc numerator + denominator)

Numerical Accuracy (FHE has noise)



Questions?