Homomorphic Encryption for Approximate Matrix Arithmetic

Jung Hee Cheon[1], Andrey Kim[1]

Seoul National University, Republic of Korea
{jhcheon, kimandrik}@snu.ac.kr

Abstract. Homomorphic Encryption for Arithmetic of Approximate Numbers (HEAAN) with its vector packing technique proved its potential in cryptographic applications. In this paper, we propose MHEAAN - a generalization of the HEAAN scheme to a multivariate case. Our design takes advantage of the HEAAN scheme, that the precision losses during the evaluation are limited by the depth of the circuit, and it exceeds no more than one bit compared to unencrypted approximate arithmetic, such as floating point operations. In addition, with a multivariate structure of the plaintext space, we suggest a general method of packing multidimensional structures as matrices and tensors in a single ciphertext. We provide a concrete two-dimensional construction and show the efficiency of our scheme on several matrix operations, such as matrix transposition, matrix multiplication, and inverse.
Keywords. Homomorphic encryption, approximate arithmetic, matrix multiplication, matrix inverse, transposition

1 Introduction

Homomorphic encryption (HE) is a scheme that allows to perform certain arithmetic operations in encrypted state. Following Gentry's blueprint [19] a numerous HE schemes have been proposed (e.g. [14, 5, 6, 3, 4, 20, 27, 2, 21, 12, 11, 16, 15, 9]). HE has plenty applications in evaluation of various algorithms on encrypted financial, medical, or genome data [29, 26, 10, 37, 25, 24].

Packing is a common technique in cryptographic schemes that allows to encrypt several messages in a single ciphertext, providing more potential to SIMD operations. Most of cryptographic schemes provide a vector packing, along with rotation operations among slots. Some HE, such as HElib [22] also provide packing messages withing a hypercube structure, however this packing technique highly depend on the structure of underlying primes factors of a modulus and is not very intuitive.

Recently Cheon et. al. [9] presented a method to construct a HE scheme for arithmetic of approximate numbers (named `HEAAN` as follows). The idea of the construction is to treat an encryption noise as part of an error occurring during approximate computations. That is, an encryption ct of a plaintext $m \in \mathcal{R}$ by a secret key sk for an underline ciphertext modulus q will have a decryption structure of the form $\langle \mathsf{ct}, \mathsf{sk} \rangle = m + e \pmod{q}$ for some small error e. `HEAAN`

showed its potential by providing the winning solution of Track 3 (Homomorphic encryption based logistic regression model learning) at iDASH privacy and security competition 2017 [24]. In their solution authors packed a matrix of inputs in a vector. Even though the authors could provide all computations using matrix to vector packing in that particular task, due to absence of row-wise matrix rotation functionality they had to circumvent and consume additional level during the computations.

Despite a diversity of HE schemes that achieve a variety of circuits evaluation, practical matrix operations such as matrix multiplication is still a problem in HE. With the growth of more complex algorithms, such as deep learning and recommendation systems, which require lots of matrix operations, the possibility of performing matrix operations becoming crucial for Homomorphic Encryption. Some works as Duong et al. [17] suggested a method for a single matrix multiplication using special packing methods but its packing structure seems to have problems with expanding to more complex computations. Another work of Hiromasa et al. [23] constructed a matrix version of GSW scheme, however it only consider binary matrix case and requires a lot of matrix multiplications, and seems not to be practical.

In this paper we suggest a generalization of `HEAAN` scheme with a new tensor packing method, along with natural rotations in various dimensions and transposition operations. Our construction (named `MHEAAN` as follows) is based on a Multivariate RLWE (m-RLWE) problem as an underline hard problem. Multivariate RLWE (m-RLWE) was introduced by Pedrouzo-Ulloa et al. [30, 31] as a multidimensional variant of RLWE problem. The hardness of m-RLWE is related to hardness problems over the tensor product of ideal lattices, and is supposed to be more appropriate for cryptographic applications dealing with multidimensional structures [32] including matrices.

Thus `MHEAAN` scheme enjoys all the benefits of `HEAAN` scheme such as rescaling procedure, which enables us to preserve the precision of the message after approximate computation and reduce the size of ciphertext significantly so the scheme can be a reasonable solution for approximate computation over the complex values. Moreover with a multivariate structure instead of packing a vector, we provide a general technique for packing any dimensional tensor in a single ciphertext. In particular we constructed a practical HE which supports matrix operations as well as standard HE operations. For example it takes only 67 ms to evaluate a transposition of a 64×64 complex matrix. For homomorphic multiplication of two 64×64 complex matrices it takes about 29 seconds, and about 4 minutes to evaluate 16-th power of a 64×64 complex matrix. Matrix inverse is a more complicated problem even in unencrypted case [36, 35]. We provided a method to evaluate inverse of a matrix for a special case. It takes about 7 minutes for a 64×64 complex matrix.

2 Preliminaries

2.1 Basic Notations

All logarithms are base 2 unless otherwise indicated. We denote vectors in bold, e.g. $\mathbf{a}$, and every vector in this paper will be a column vector. We denote by $\langle \mathbf{a}, \mathbf{b} \rangle$ the usual dot product of two vectors. By bold capital letters we denote matrices, e.g. $\mathbf{A}$. For a real number r, $\lfloor r \rceil$ denotes the nearest integer to r, rounding upwards in case of a tie. For an integer q, we identify $\mathbb{Z} \cap (-q/2, q/2]$ as a representative interval and use $\lfloor r \rceil_q$ to denote the reduction of an integer r modulo q into that interval. We use $a \leftarrow \chi$ to denote the sampling according to a distribution χ. For a finite set $\mathcal{D}$, $a \leftarrow \mathcal{D}$ denotes the sampling from the uniform distribution over $\mathcal{D}$. For matrices $\mathbf{A}, \mathbf{B} \in \mathbb{C}^{n \times m}$ denote by $\mathbf{A} \odot \mathbf{B}$ component wise product, and denote by $\mathsf{rt}(\mathbf{A}, (r, c))$ a matrix obtained from $\mathbf{A}$ by cyclic rotating by r rows and c columns.

$$\mathbf{A} = \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,m-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,m-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1,0} & a_{n-1,1} & \cdots & a_{n-1,m-1} \end{bmatrix}$$

$$\mathsf{rt}(\mathbf{A}, (r, c)) = \begin{bmatrix} a_{r,c} & a_{r,c+1} & \cdots & a_{r,c+m-1} \\ a_{r+1,c} & a_{r+1,c+1} & \cdots & a_{r+1,c+m-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{r+n-1,c} & a_{r+n-1,c+1} & \cdots & a_{r+n-1,c+m-1} \end{bmatrix}$$

where indexes are taken modulus n and m respectively. Denote the security parameter throughout the paper: all known valid attacks against the cryptographic scheme under scope should take bit operations.

2.2 The Cyclotomic Ring and Canonical Embedding

For a power of two N, let $M = 2N$ and $\Phi_M(X) = X^N + 1$ be M-th cyclotomic polynomial. Let $\mathcal{R} = \mathbb{Z}[X]/(X^N + 1)$. An element in $\mathcal{S} = \mathbb{R}[X]/(X^N + 1)$ can be represented as a polynomial $a(X) = \sum_{j=0}^{N-1} a_j X^j$ of degree less than N and identified with its coefficient vector $(a_0, \ldots, a_{N-1}) \in \mathbb{R}^N$. We define the relevant norms on the coefficient vector of a such as $\|a\|_\infty$ and $\|a\|_1$.

Let $\mathbb{Z}_M^\star = \{x \in \mathbb{Z}_M : \gcd(x, M) = 1\}$ be the multiplicative group of units in $\mathbb{Z}_M$. The canonical embedding σ_M of $a \in \mathbb{Q}[X]/(\Phi_M(X))$ into $\mathbb{C}^N$ is the vector of evaluation values of a at the roots of $\Phi_M(X)$. We naturally extend it to the set of real polynomials $\mathcal{S}$, $\sigma_M : \mathcal{S} \to \mathbb{C}^N$, so $\sigma_M(a)$ will be defined as $(a(\xi_M^j))_{j \in \mathbb{Z}_M^\star} \in \mathbb{C}^N$ for any $a \in \mathcal{R}$ where $\xi_M = \exp(-2\pi i/M)$ is a primitive M-th roots of unity. The ℓ_∞-norm of $\sigma_M(a)$ is called the *canonical embedding norm* of a, denoted by $\|a\|_\infty^{\mathsf{can}} = \|\sigma_M(a)\|_\infty$. The canonical embedding norm $\|\cdot\|_\infty^{\mathsf{can}}$ satisfies the following properties:

- For all $a, b \in \mathcal{R}$, we have $\|a \cdot b\|_\infty^{\mathsf{can}} \leq \|a\|_\infty^{\mathsf{can}} \cdot \|b\|_\infty^{\mathsf{can}}$
- For all $a \in \mathcal{R}$, we have $\|a\|_\infty^{\mathsf{can}} \leq \|a\|_1$.
- For all $a \in \mathcal{R}$, we have $\|a\|_\infty \leq \|a\|_\infty^{\mathsf{can}}$.

Refer [13] for more details.

2.3 Learning With Errors and its Variants

Learning With Errors (LWE) problem introduced by Regev [34], proved itself as a good underlying problem for many lattice-based cryptography schemes.

Definition 1 (LWE Distribution). *Let $N > 1$, $q \geq 2$. For $\mathbf{s} \in \mathbb{Z}_q$ (the "secret") and an error distribution χ over $\mathbb{Z}_q$, a sample from the LWE distribution $\mathcal{A}_{\mathbf{s},\chi}$ over $\mathbb{Z}_q^N \times \mathbb{Z}_q$ is generated by choosing $\mathbf{a} \leftarrow \mathbb{Z}_q^N$ uniformly at random, choosing $\mathbf{e} \leftarrow \chi$, and outputting $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + \mathbf{e} \pmod{\mathbb{Z}_q})$.*

Definition 2 (LWE Search). *The search version of the LWE problem is defined as follows: given access to arbitrary number of independent samples from LWE distribution $\mathcal{A}_{\mathbf{s},\chi}$ for some arbitrary $\mathbf{s} \in \mathbb{Z}_q$ find $\mathbf{s}$ (with high probability)*

Even though LWE has promising hardness assumption for cryptographic primitives [34, 33], it is not efficient enough for practical applications. The Ring LWE (RLWE) introduced by Lyubashevsky et al. [28] is an algebraic variant of LWE which provides more attractive features for cryptographic applications. We provide definition of RLWE for a power-of-two degree case.

Definition 3 (RLWE Distribution). *Let $N > 1$ be a power-of-two, $q \geq 2$, $\mathcal{R} = \mathbb{Z}[X]/(X^N + 1)$, $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$. For $\mathbf{s} \in \mathcal{R}$ (the "secret") and an error distribution χ over $\mathcal{R}_q$, a sample from the RLWE distribution $\mathcal{A}_{\mathbf{s},\chi}$ over $\mathcal{R}_q \times \mathcal{R}_q$ is generated by choosing $\mathbf{a} \leftarrow \mathcal{R}_q$ uniformly at random, choosing $\mathbf{e} \leftarrow \chi$, and outputting $(\mathbf{a}, \mathbf{a} \cdot \mathbf{s} + \mathbf{e} \pmod{\mathcal{R}_q})$.*

Definition 4 (RLWE Search). *The search version of the RLWE problem is defined as follows: given access to arbitrary number of independent samples from RLWE distribution $\mathcal{A}_{\mathbf{s},\chi}$ for some arbitrary $\mathbf{s} \in \mathcal{R}$ find $\mathbf{s}$ (with high probability)*

Multivariate RLWE (m-RLWE) was introduced by Pedrouzo-Ulloa et al. [30, 31] as a multidimensional variant of RLWE problem. m-RLWE hardness is related to hardness problems over the tensor product of ideal lattices, and is supposed to be more appropriate for cryptographic applications dealing with multidimensional structures [32].

Definition 5 (m-RLWE Distribution). *Let $N_i > 1$ be a powers-of-two for $i = 1, \ldots, l$, $q \geq 2$, $\mathcal{R} = \mathbb{Z}[X_1, \ldots, X_l]/(X_1^N + 1, \ldots, X_l^N + 1)$, $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$. For $\mathbf{s} \in \mathcal{R}$ (the "secret") and an error distribution χ over $\mathcal{R}_q$, a sample from the m-RLWE distribution $\mathcal{A}_{\mathbf{s},\chi}$ over $\mathcal{R}_q \times \mathcal{R}_q$ is generated by choosing $\mathbf{a} \leftarrow \mathcal{R}_q$ uniformly at random, choosing $\mathbf{e} \leftarrow \chi$, and outputting $(\mathbf{a}, \mathbf{a} \cdot \mathbf{s} + \mathbf{e} \pmod{\mathcal{R}_q})$.*

Definition 6 (m-RLWE Search). *The search version of the m-RLWE problem is defined as follows: given access to arbitrary number of independent samples from m-RLWE distribution $\mathcal{A}_{\mathbf{s},\chi}$ for some arbitrary $\mathbf{s} \in \mathcal{R}$ find $\mathbf{s}$ (with high probability)*

2.4 HEAAN Scheme

The following is the instantiation of the RLWE-based HEAAN scheme [9, 8]. For a power-of-two N, denote $M = 2N$, $\Phi_M(X) = (X^N + 1)$, $\mathcal{R} = \mathbb{Z}[X]/\Phi_M(X)$. For a positive integer ℓ, denote $\mathcal{R}_\ell = \mathcal{R}/2^\ell\mathcal{R} = \mathbb{Z}_{2^\ell}[X]/\Phi_M(X)$ the residue ring of $\mathcal{R}$ modulo 2^ℓ. The integer 5 has the order of $N/2$ modulo M and with (-1) spans $\mathbb{Z}_M^*$. Hence $\xi_j, \bar{\xi_j} : 0 \leq j < N/2$ forms the set of primitive M-th roots of unity for $\xi_j := \xi^{5^j}$. Define an encoding map $\phi_{N/2}$ as inverse of a variant of the complex canonical embedding map defined as $\phi_{N/2}^{-1} : m(x) \rightarrow \mathbf{z} = (z_0, \ldots, z_{N/2-1})$ such that $z_j = m(\xi_j)$. For a power-of-two $n \leq N/2$ for $X' = X^{N/(2n)}$ consider $\mathcal{R}' = \mathbb{Z}[X']/(X'^{2n} + 1) \subset \mathcal{R}$ and define in a similar way $\phi_n^{-1} : m(x') = m(x^{N/(2n)}) \rightarrow \mathbf{z} = (z_0, \ldots, z_{n-1})$ such that $z_j = m(\xi'_j)$, where $\xi'_j = \xi_j^{N/(2n)}$

- $\underline{\texttt{HEAAN.KeyGen}(1^\lambda)}$.
  - For an integer L that corresponds to the largest ciphertext modulus level, given the security parameter λ, output the ring dimension N which is a power of two.
  - Set the small distributions $\chi_{key}, \chi_{err}, \chi_{enc}$ over $\mathcal{R}$ for secret, error, and encryption, respectively.
  - Sample a secret $s \leftarrow \chi_{key}$, a random $a \leftarrow \mathcal{R}_L$ and an error $e \leftarrow \chi_{err}$. Set the secret key as $\mathsf{sk} \leftarrow (s, 1)$ and the public key as $\mathsf{pk} \leftarrow (a, b) \in \mathcal{R}_L^2$ where $b \leftarrow -as + e \pmod{2^L}$.
- $\underline{\texttt{HEAAN.KSGen}_{\mathsf{sk}}(s')}$. For $s' \in \mathcal{R}$, sample a random $a' \leftarrow \mathcal{R}_{2\cdot L}$ and an error $e' \leftarrow \chi_{err}$. Output the switching key as $\mathsf{swk} \leftarrow (a', b') \in \mathcal{R}_{2\cdot L}^2$ where $b' \leftarrow -a's + e' + 2^L s' \pmod{2^{2\cdot L}}$.
  - Set the evaluation key as $\mathsf{evk} \leftarrow \texttt{HEAAN.KSGen}_{\mathsf{sk}}(s^2)$.
- $\underline{\texttt{HEAAN.Encode}(\mathbf{z}, p)}$. For a vector $\mathbf{z} \in \mathbb{C}^n$, with of a power-of-two $n \leq N/2$ and an integer $p < L$ corresponding to precision bits, output the polynomial $m \leftarrow \phi_n(2^p \cdot \mathbf{z}) \in \mathcal{R}$.
- $\underline{\texttt{HEAAN.Decode}(m, p)}$. For a plaintext $m \in \mathcal{R}$, the encoding of a vector consisting of a power-of-two $n \leq N/2$ complex messages and precision bits p, output the vector $\mathbf{z} \leftarrow \phi_n^{-1}(m/2^p) \in \mathbb{C}^n$.
- $\underline{\texttt{HEAAN.Enc}_{\mathsf{pk}}(m)}$. For $m \in \mathcal{R}$, sample $v \leftarrow \chi_{enc}$ and $e_0, e_1 \leftarrow \chi_{err}$. Output $v \cdot \mathsf{pk} + (e_0, e_1 + m) \pmod{2^L}$.
- $\underline{\texttt{HEAAN.Dec}_{\mathsf{sk}}(\mathsf{ct})}$. For $\mathsf{ct} = (c_0, c_1) \in \mathcal{R}_\ell^2$, output $c_0 \cdot s + c_1 \pmod{2^\ell}$.
- $\underline{\texttt{HEAAN.Add}(\mathsf{ct}_1, \mathsf{ct}_2)}$. For $\mathsf{ct}_1, \mathsf{ct}_2 \in \mathcal{R}_\ell^2$, output $\mathsf{ct}_{\mathsf{add}} \leftarrow \mathsf{ct}_1 + \mathsf{ct}_2 \pmod{2^\ell}$.
- $\underline{\texttt{HEAAN.CMult}_{\mathsf{evk}}(\mathsf{ct}, \mathbf{c}, p)}$. For $\mathsf{ct} \in \mathcal{R}_\ell^2$ and $\mathbf{c} \in \mathbb{C}^n$, compute $c \leftarrow \texttt{HEAAN.Encode}(\mathbf{c}; p)$ and output $\mathsf{ct}' \leftarrow c \cdot \mathsf{ct} \pmod{2^\ell}$.
- $\underline{\texttt{HEAAN.PolyMult}_{\mathsf{evk}}(\mathsf{ct}, g, p)}$. For $\mathsf{ct} \in \mathcal{R}_\ell^2$ and $g \in \mathcal{R}_\ell$, output $\mathsf{ct}' \leftarrow g \cdot \mathsf{ct} \pmod{2^\ell}$.
- $\underline{\texttt{HEAAN.Mult}_{\mathsf{evk}}(\mathsf{ct}_1, \mathsf{ct}_2)}$. For $\mathsf{ct}_1 = (a_1, b_1), \mathsf{ct}_2 = (a_2, b_2) \in \mathcal{R}_\ell^2$, let $(d_0, d_1, d_2) = (a_1 \cdot a_2, a_1 \cdot b_2 + a_2 \cdot b_1, b_1 \cdot b_2) \pmod{2^\ell}$. Output $\mathsf{ct}_{\mathsf{mult}} \leftarrow (d_1, d_2) + \lfloor 2^{-L} \cdot d_0 \cdot \mathsf{evk} \rceil \pmod{2^\ell}$.

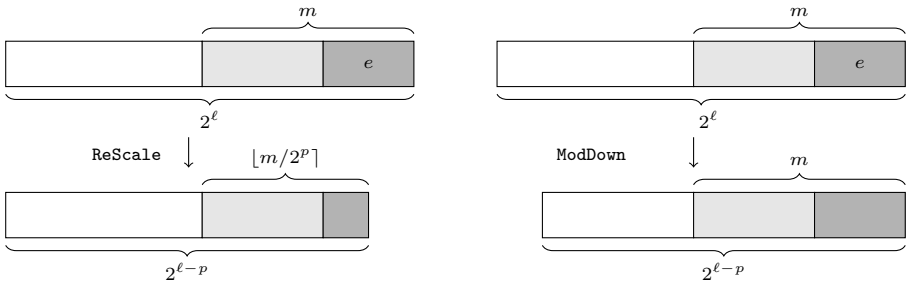

Fig. 1. Rescaling and Modulus Down

- $\underline{\texttt{HEAAN.ReScale}(\mathsf{ct}, p)}$. For a ciphertext $\mathsf{ct} \in \mathcal{R}_\ell^2$ and an integer p, output $\mathsf{ct}' \leftarrow \lfloor 2^{-p} \cdot \mathsf{ct} \rceil \pmod{2^{\ell-p}}$.
- $\underline{\texttt{HEAAN.ModDown}(\mathsf{ct}, p)}$. For a ciphertext $\mathsf{ct} \in \mathcal{R}_\ell^2$ and an integer p, output $\mathsf{ct}' \leftarrow \mathsf{ct} \pmod{2^{\ell-p}}$.

For an integer k co-prime with M, let $\kappa_k : m(X) \rightarrow m(X^k) \pmod{\Phi_M(X)}$. This transformation can be used to provide more functionalities on plaintext slots.

- $\underline{\texttt{HEAAN.Conjugate}_{\mathsf{cjk}}(\mathsf{ct})}$. Set the conjugation key as $\mathsf{cjk} \leftarrow \texttt{HEAAN.KSGen}_{\mathsf{sk}}(\kappa_{-1}(s))$. For $\mathsf{ct} = (a, b) \in \mathcal{R}_\ell^2$ encrypting vector $\mathbf{z}$, let $(a', b') = (\kappa_{-1}(a), \kappa_{-1}(b)) \pmod{2^\ell}$. Output $\mathsf{ct}_{\mathsf{cj}} \leftarrow (0, b') + \lfloor 2^{-L} \cdot a' \cdot \mathsf{cjk} \rceil \pmod{2^\ell}$. $\mathsf{ct}_{\mathsf{cj}}$ is a ciphertext encrypting $\bar{\mathbf{z}}$ - the conjugated plaintext vector of ct.
- $\underline{\texttt{HEAAN.Rotate}_{\mathsf{rtk}}(\mathsf{ct}; r)}$. Set the rotation key as $\mathsf{rtk} \leftarrow \texttt{HEAAN.KSGen}_{\mathsf{sk}}(\kappa_{5^r}(s))$. For $\mathsf{ct} = (a, b) \in \mathcal{R}_\ell^2$ encrypting vector $\mathbf{z}$, let $(a', b') = (\kappa_{5^r}(a), \kappa_{5^r}(b)) \pmod{2^\ell}$. Output $\mathsf{ct}_{\mathsf{rt}} \leftarrow (0, b') + \lfloor 2^{-L} \cdot a' \cdot \mathsf{rtk} \rceil \pmod{2^\ell}$. $\mathsf{ct}_{\mathsf{rt}}$ is a ciphertext encrypting $\mathsf{rt}(\mathbf{z}, r) = (z_r, \ldots, z_{n-1}, z_0, \ldots, z_{r-1})$ - rotated by r positions plaintext vector of ct.

Refer [9] for the technical details and noise analysis.

3 Multivariate HEAAN Scheme

3.1 Structure of MHEAAN

Encryption process in `MHEAAN` scheme can be shown in the following outline: for a powers-of-two N_i, n_i, with $n_i \leq N_i/2$, $i = 1, \ldots, l$ encode a l-dimension tensor $\mathcal{Z} \in \mathbb{C}^{n_1 \times \cdots \times n_l}$ to a l-variable polynomial $m(X_1, \ldots, X_l)$, using map $\phi_{n_1,\ldots,n_l}$ and mask the result with m-RLWE instance $\big(a(X_1, \ldots X_l), b(X_1, \ldots X_l)\big)$ in the corresponding ring $\mathcal{R}/q\mathcal{R}$.

We can treat `HEAAN` scheme as a special case of `MHEAAN` with $l = 1$:

$$\mathbf{z} = \begin{bmatrix} z_0 \\ z_1 \\ \vdots \\ z_{n-1} \end{bmatrix} \xrightarrow[\texttt{Encode}]{\phi_n} m(X) \xrightarrow[\texttt{Enc}]{\texttt{RLWE}} \mathsf{ct}$$

and for two-dimension `MHEAAN` we have:

$$\mathbf{Z} = \begin{bmatrix} z_{0,0} & z_{0,1} & \cdots & z_{0,n_y-1} \\ z_{1,0} & z_{1,1} & \cdots & z_{1,n_y-1} \\ \vdots & \vdots & \ddots & \vdots \\ z_{n_x-1,0} & z_{n_x-1,1} & \cdots & z_{n_x-1,n_y-1} \end{bmatrix} \xrightarrow[\texttt{Encode}]{\phi_{n_x,n_y}} m(X,Y) \xrightarrow[\texttt{Enc}]{2-\texttt{RLWE}} \mathsf{ct}$$

3.2 Canonical Embedding in $\mathbb{Z}[\boldsymbol{X},\boldsymbol{Y}]/(\boldsymbol{X}^{\boldsymbol{N_x}}+1,\boldsymbol{Y}^{\boldsymbol{N_y}}+1)$

In the rest of the paper we will use following notations. For a power-of-two integers N_x, N_y denote $M_x = 2N_x, M_y = 2N_y$, $\Phi_{M_x}(X) = X^{N_x}+1$, $\Phi_{M_y}(Y) = Y^{N_y}+1$, $\mathcal{R} = \mathbb{Z}[X,Y]/(\Phi_{M_x}(X), \Phi_{M_y}(Y))$, $\mathcal{S} = \mathbb{R}[X,Y]/(\Phi_{M_x}(X), \Phi_{M_y}(Y))$.

We can generalize σ_M defined in Section 2 to $\sigma_{M_x,M_y} : \mathcal{S} \to \mathbb{C}^{N_x,N_y}$, so for $a \in \mathcal{S}$, $\sigma_{M_x,M_y}(a) = (a(\xi_{M_x}^{j_x}, \xi_{M_y}^{j_y}))_{j_x \in \mathbb{Z}_{M_x}^*, j_y \in \mathbb{Z}_{M_y}^*} \in \mathbb{C}^{N_x \times N_y}$. Two dimensional canonical embedding norm is defined as $\|a\|_\infty^{\mathsf{mcan}} = \|\sigma_{M_x,M_y}(a)\|_\infty$. $\|\cdot\|_\infty^{\mathsf{mcan}}$ satisfies the following properties:

- For all $a, b \in \mathcal{R}$, we have $\|a \cdot b\|_\infty^{\mathsf{mcan}} \le \|a\|_\infty^{\mathsf{mcan}} \cdot \|b\|_\infty^{\mathsf{mcan}}$
- For all $a \in \mathcal{R}$, we have $\|a\|_\infty^{\mathsf{mcan}} \le \|a\|_1$.
- For all $a \in \mathcal{R}$, we have $\|a\|_\infty \le \|a\|_\infty^{\mathsf{mcan}}$.

For a power-of-two integers $n_x \le N_x/2$ and $n_y \le N_y/2$, let $X' = X^{N_x/(2n_x)}$, $Y' = Y^{N_y/(2n_y)}$ and $\mathcal{R}' = \mathbb{Z}[X',Y']/(X'^{2n_x}+1, Y'^{2n_y}+1)$. We can also generalize encoding map ϕ_n to a two-dimensional mapping $\phi_{n_x,n_y} : \mathbb{C}^{n_x \times n_y} \to \mathcal{R}$, defined as

$$\phi_{n_x,n_y}^{-1} : m(x',y') = m(x^{N_x/(2n_x)}, y^{N_y/(2n_y)}) \to \mathbf{Z}$$

where $\mathbf{Z} \in \mathbb{C}^{n_x \times n_y}$ with $\mathbf{Z}_{ij} = m(\xi_x'^{5^i}, \xi_y'^{5^j})$, $\xi_x' = \xi_{M_x}^{N_x/(2n_x)}$, $\xi_y' = \xi_{M_y}^{N_y/(2n_y)}$

3.3 Concrete Construction

We present a construction of 2-RLWE-based `MHEAAN` scheme. For a positive integer ℓ denote $\mathcal{R}_\ell = \mathcal{R}/2^\ell\mathcal{R} = \mathbb{Z}_{2^\ell}[X,Y]/(\Phi_{M_x}(X), \Phi_{M_y}(Y))$ the residue ring of $\mathcal{R}$ modulo 2^ℓ. For a real $\sigma > 0$, $\mathcal{DG}(\sigma^2)$ samples a polynomial in $\mathcal{R}$ by drawing its coefficient independently from the discrete Gaussian distribution of variance σ^2. For an positive integer h, $\mathcal{HWT}(h)$ is the set of signed binary matrices in $\{0,\pm1\}^{N_x \times N_y}$ whose Hamming weight is exactly h. For a real $0 \le \rho \le 1$, the distribution $\mathcal{ZO}(\rho)$ draws each entry in the matrix from $\{0,\pm1\}^{N_x \times N_y}$, with probability $\rho/2$ for each of -1 and $+1$, and probability being zero $1-\rho$.

- $\underline{\texttt{MHEAAN.KeyGen}(1^\lambda)}$.
  - For an integer L that corresponds to the largest ciphertext modulus level, given the security parameter λ, distribution parameters (ρ, σ, h), output the power-of-two ring dimension N and choose N_x, N_y such that $N_x \cdot N_y = N$
  - Set the distributions $\chi_{enc} = \mathcal{ZO}(\rho)$, $\chi_{err} = \mathcal{DG}(\sigma)$, $\chi_{key} = \mathcal{HWT}(h)$ over $\mathcal{R}$ for secret, error, and encryption, respectively.
  - Sample a secret $s \leftarrow \chi_{key}$, a random $a \leftarrow \mathcal{R}_L$ and an error $e \leftarrow \chi_{err}$. Set the secret key as $\mathsf{sk} \leftarrow (s, 1)$ and the public key as $\mathsf{pk} \leftarrow (a, b) \in \mathcal{R}_L^2$ where $b \leftarrow -a \cdot s + e \pmod{2^L}$.
- $\underline{\texttt{MHEAAN.KSGen}_{\mathsf{sk}}(s')}$. For $\mathbf{s}' \in \mathcal{R}$, sample a random $a' \leftarrow \mathcal{R}_{2 \cdot L}$ and an error $e' \leftarrow \chi_{err}$. Output the switching key as $\mathsf{swk} \leftarrow (a', b') \in \mathcal{R}_{2 \cdot L}^2$ where $b' \leftarrow -a' \cdot s + e' + 2^L s' \pmod{2^{2 \cdot L}}$.
  - Set the evaluation key as $\mathsf{evk} \leftarrow \texttt{MHEAAN.KSGen}_{\mathsf{sk}}(s^2)$.
- $\underline{\texttt{MHEAAN.Encode}(\mathbf{Z}, p)}$. For a matrix $\mathbf{Z} \in \mathbb{C}^{n_x \times n_y}$, an integer $p < L - 1$ corresponding to precision bits, output the two-degree polynomial $m \leftarrow \phi_{n_x, n_y}(2^p \cdot Z) \in \mathcal{R}$.
- $\underline{\texttt{MHEAAN.Decode}(m, p)}$. For a plaintext $m \in \mathcal{R}$, the encoding of a matrix of complex messages $\mathbf{Z} \in \mathbb{C}^{n_x \times n_y}$, precision bits p, output the matrix $Z' \leftarrow \phi^{-1}_{n_x, n_y}(m / 2^p) \in \mathbb{C}^{n_x, n_y}$.
- $\underline{\texttt{MHEAAN.Enc}_{\mathsf{pk}}(m)}$. For $m \in \mathcal{R}$, sample $v \leftarrow \chi_{enc}$ and $e_0, e_1 \leftarrow \chi_{err}$. Output $\mathsf{ct} = v \cdot \mathsf{pk} + (e_0, e_1 + m) \pmod{2^L}$.
- $\underline{\texttt{MHEAAN.Dec}_{\mathsf{sk}}(\mathsf{ct})}$. For $\mathsf{ct} = (c_0, c_1) \in \mathcal{R}_\ell^2$, output $c_0 \cdot s + c_1 \pmod{2^\ell}$.
- $\underline{\texttt{MHEAAN.Add}(\mathsf{ct}_1, \mathsf{ct}_2)}$. For $\mathsf{ct}_1, \mathsf{ct}_2 \in \mathcal{R}_\ell^2$ - encryption of matrices $Z_1, Z_2 \in \mathbb{C}^{n_x \times n_y}$ output $\mathsf{ct}_{\mathsf{add}} \leftarrow \mathsf{ct}_1 + \mathsf{ct}_2 \pmod{2^\ell}$. $\mathsf{ct}_{\mathsf{add}}$ is a ciphertext encrypting matrix $\mathbf{Z}_1 + \mathbf{Z}_2$.
- $\underline{\texttt{MHEAAN.CMult}_{\mathsf{evk}}(\mathsf{ct}, \mathbf{C}, p)}$. For $\mathsf{ct} \in \mathcal{R}_\ell^2$ - encryption of $\mathbf{Z} \in \mathbb{C}^{n_x \times n_y}$, and a constant matrix $\mathbf{C} \in \mathbb{C}^{n_x, n_y}$, compute $c \leftarrow \texttt{MHEAAN.Encode}(\mathbf{C}, p)$ the encoding of $\mathbf{C}$ and output $\mathsf{ct}_{\mathsf{cmult}} \leftarrow c \cdot \mathsf{ct} \pmod{2^\ell}$. $\mathsf{ct}_{\mathsf{cmult}}$ is a ciphertext encrypting matrix $\mathbf{Z} \odot \mathbf{C}$.
- $\underline{\texttt{MHEAAN.PolyMult}_{\mathsf{evk}}(\mathsf{ct}, g, p)}$. For $\mathsf{ct} \in \mathcal{R}_\ell^2$ - encryption of $\mathbf{Z} \in \mathbb{C}^{n_x \times n_y}$, and a constant $g \in \mathcal{R}_\ell$ output $\mathsf{ct}_{\mathsf{cmult}} \leftarrow c \cdot \mathsf{ct} \pmod{2^\ell}$. $\mathsf{ct}_{\mathsf{cmult}}$ is a ciphertext encrypting matrix $\mathbf{Z} \odot \mathbf{C}$, where $\mathbf{C}$ is decoding of g.

Multiplication by polynomial is similar to constant multiplication, however in the next section we will show why it is important to define it separately.

- $\underline{\texttt{MHEAAN.Mult}_{\mathsf{evk}}(\mathsf{ct}_1, \mathsf{ct}_2)}$. For $\mathsf{ct}_1 = (a_1, b_1), \mathsf{ct}_2 = (a_2, b_2) \in \mathcal{R}_\ell^2$ - encryptions of matrices $\mathbf{Z}_1, \mathbf{Z}_2 \in \mathbb{C}^{n_x, n_y}$, let $(d_0, d_1, d_2) = (a_1 a_2,\ a_1 b_2 + a_2 b_1, b_1 b_2) \pmod{2^\ell}$. Output $\mathsf{ct}_{\mathsf{mult}} \leftarrow (d_1, d_2) + \lfloor 2^{-L} \cdot d_0 \cdot \mathsf{evk} \rceil \pmod{2^\ell}$. $\mathsf{ct}_{\mathsf{mult}}$ is a ciphertext encrypting matrix $\mathbf{Z}_1 \odot \mathbf{Z}_2$.
- $\underline{\texttt{MHEAAN.ReScale}(\mathsf{ct}, p)}$. For a ciphertext $\mathsf{ct} \in \mathcal{R}_\ell^2$ and an integer p, output $\mathsf{ct}' \leftarrow \lfloor 2^{-p} \cdot \mathsf{ct} \rceil \pmod{2^{\ell - p}}$.

Similar to HEAAN in MHEAAN scheme for an integers k_x, k_y co-prime with M_x, M_y respectively, let

$$\kappa_{k_x,k_y} : m(X,Y) \to m(X^{k_x}, Y^{k_y}) \pmod{\Phi_{M_x}(X), \Phi_{M_y}(Y)}$$

This transformation can be used to provide conjugation, row and column rotations on plaintext matrix.

- $\underline{\texttt{MHEAAN.Conjugate}_{\mathsf{cjk}}(\mathsf{ct})}$. Set the conjugation key as

  $$\mathsf{cjk} \leftarrow \texttt{MHEAAN.KSGen}_{\mathsf{sk}}(\kappa_{-1,1}(s))$$

  For $\mathsf{ct} = (a,b) \in \mathcal{R}_\ell^2$ encrypting matrix $\mathbf{Z}$, let $(a',b') = (\kappa_{-1,1}(a), \kappa_{-1,1}(b)) \pmod{2^\ell}$. Output $\mathsf{ct}_{\mathsf{cj}} \leftarrow (0,b') + \lfloor 2^{-L} \cdot a' \cdot \mathsf{cjk} \rceil \pmod{2^\ell}$. $\mathsf{ct}_{\mathsf{cj}}$ is a ciphertext encrypting $\bar{\mathbf{Z}}$ - the conjugated plaintext matrix of ct.
- $\underline{\texttt{MHEAAN.Rotate}_{\mathsf{rtk}}(\mathsf{ct}; (r_x, r_y))}$. Set the rotation key as

  $$\mathsf{rtk} \leftarrow \texttt{MHEAAN.KSGen}_{\mathsf{sk}}(\kappa_{5^{r_x},5^{r_y}}(s))$$

  For $\mathsf{ct} = (a,b) \in \mathcal{R}_\ell^2$ encrypting matrix $\mathbf{Z}$, let $(a',b') = (\kappa_{5^{r_x},5^{r_y}}(a), \kappa_{5^{r_x},5^{r_y}}(b)) \pmod{2^\ell}$. Output $\mathsf{ct}_{\mathsf{rt}} \leftarrow (0,b') + \lfloor 2^{-L} \cdot a' \cdot \mathsf{rtk} \rceil \pmod{2^\ell}$. $\mathsf{ct}_{\mathsf{rt}}$ is a ciphertext encrypting $\mathsf{rt}(\mathbf{Z}, (r_x, r_y))$ - cyclic rotated plaintext matrix by r rows and c columns.

In case of $N_x = N_y$ let $\tau : m(X,Y) \to m(Y,X)$. This transformation can be used to provide transposition functionality on plaintext matrix.

- $\underline{\texttt{MHEAAN.Transpose}_{\mathsf{trk}}(\mathsf{ct})}$. Set the transposition key as

  $$\mathsf{trk} \leftarrow \texttt{MHEAAN.KSGen}_{\mathsf{sk}}(\tau(s))$$

  For $\mathsf{ct} = (a,b) \in \mathcal{R}_\ell^2$ encrypting matrix $\mathbf{Z}$, let $(a',b') = (\tau(a), \tau(b)) \pmod{2^\ell}$. Output $\mathsf{ct}_{\mathsf{trk}} \leftarrow (0,b') + \lfloor 2^{-L} \cdot a' \cdot \mathsf{trk} \rceil \pmod{2^\ell}$. $\mathsf{ct}_{\mathsf{trk}}$ is a ciphertext encrypting $\mathbf{Z}^T$ - transposed plaintext matrix of ct.

Throughout this paper, we use real polynomials as plaintexts for convenience of analysis. A ciphertext $\mathsf{ct} \in \mathcal{R}_\ell^2$ will be called a valid encryption of $m \in \mathcal{S}$ with the encryption noise bounded by B, and plaintext bounded by M, if $\langle \mathsf{ct}, \mathsf{sk} \rangle = m + e \pmod{2^\ell}$ for some polynomial $e \in \mathcal{S}$ with $\|e\|_\infty^{\mathsf{mcan}} < B$ and $\|m\|_\infty^{\mathsf{mcan}} < M$. We will use a corresponding tuple $(\mathsf{ct}, B, M, \ell)$ for such an encryption of m. The following lemmas give upper bounds on noise growth after encryption, rescaling and homomorphic operations. Refer to Appendix A for proofs.

Lemma 1 (Encoding & Encryption). *For $m \leftarrow$ MHEAAN.Encode(Z,p) and $\mathsf{ct} \leftarrow$ MHEAAN.Enc$_{\mathsf{pk}}(m)$ the encryption noise is bounded by $B_{\mathsf{clean}} = 8\sqrt{2} \cdot \sigma N + 6\sigma\sqrt{N} + 16\sigma\sqrt{hN}$.*

Lemma 2 (Rescaling). *Let $(\mathsf{ct}, B, M, \ell)$ be a valid encryption of m and $\mathsf{ct}' \leftarrow$ MHEAAN.ReScale(ct, p). Then $(\mathsf{ct}', B/2^p + B_{\mathsf{scale}}, M/2^p, \ell - p)$ is a valid encryption of $m/2^p$ where $B_{\mathsf{scale}} = 6\sqrt{N/12} + 16\sqrt{hN/12}$*

Remark We can slightly change the public key generation and encryption process, to obtain a ciphertext with initial noise reduced from B_{clean} to almost B_{scale}. For this we generate public key in $\mathcal{R}_{2L}^2$ instead of $\mathcal{R}_L^2$ and in encryption process we encode the plaintext m with $p + L$ precision bits, instead of p bits with the following rescaling the encryption ct of m by L bits. With a slightly slower encryption process we end up with a valid encryption in $\mathcal{R}_L^2$, with the initial noise bounded by $B_{\mathsf{clean}}/2^L + B_{\mathsf{scale}} \approx B_{\mathsf{scale}}$.

Lemma 3 (Addition & Multiplication). *Let* $(\mathsf{ct}_i, B_i, M_i, \ell)$ *be encryptions of* $m_i \in \mathcal{S}$ *and let* $\mathsf{ct}_{\mathsf{add}} \leftarrow \texttt{MHEAAN.Add}(\mathsf{ct}_1, \mathsf{ct}_2)$ *and* $\mathsf{ct}_{\mathsf{mult}} \leftarrow \texttt{MHEAAN.Mult}_{\mathsf{evk}}(\mathsf{ct}_1, \mathsf{ct}_2)$. *Then* $(\mathsf{ct}_{\mathsf{add}}, B_1 + B_2, M_1 + M_2, \ell)$ *and* $(\mathsf{ct}_{\mathsf{mult}}, M_1 \cdot B_2 + M_2 \cdot B_1 + B_1 \cdot B_2 + B_{\mathsf{mult}}, M_1 \cdot M_2, \ell)$ *are valid encryptions of* $m_1 + m_2$ *and* $m_1 \cdot m_2$*, respectively, where* $B_{\mathsf{ks}} = 8\sigma N/\sqrt{3}$ *and* $B_{\mathsf{mult}} = 2^{\ell - L} \cdot B_{\mathsf{ks}} + B_{\mathsf{scale}}$.

Lemma 4 (Conjugation, Rotation & Transposition). *Let* $(\mathsf{ct}, B, M, \ell)$ *be encryption of* $m \in \mathcal{S}$ *that encodes matrix* $\mathbf{Z}$*,* $r_x, r_y \in \mathbb{Z}$*, and let ciphertexts* $\mathsf{ct}_{\mathsf{rt}} = \texttt{MHEAAN.Rotate}_{\mathsf{rtk}}(\mathsf{ct}; (r_x, r_y))$*,* $\mathsf{ct}_{\mathsf{cj}} = \texttt{MHEAAN.Conjugate}_{\mathsf{cjk}}(\mathsf{ct})$ *and* $\mathsf{ct}_{\mathsf{tr}} = \texttt{MHEAAN.Transpose}_{\mathsf{trk}}(\mathsf{ct})$*. Then* $(\mathsf{ct}_{\mathsf{rt}}, B + B_*, M, \ell)$*,* $(\mathsf{ct}_{\mathsf{cj}}, B + B_*, M, \ell)$*,* $(\mathsf{ct}_{\mathsf{tr}}, B + B_*, M, \ell)$ *are valid encryptions of plaintexts that encode matrices* $\mathsf{rt}(\mathbf{Z}, (r_x, r_y))$*,* $\bar{\mathbf{Z}}$ *and* $\mathbf{Z}^T$ *respectively where where* $B_{\mathsf{ks}} = 8\sigma N/\sqrt{3}$ *and* $B_* = 2^{\ell - L} \cdot B_{\mathsf{ks}} + B_{\mathsf{scale}}$

Relative Error As was discussed in [9] the decryption result of a ciphertext is an approximate value of plaintext, so it needs to dynamically manage the bound of noise of ciphertext. It is sometimes convenient to consider the relative error defined by $\beta = B/M$. For addition of two ciphertexts with relative errors $\beta_i = B_i/M_i$ the output ciphertext has a relative error bounded by $\max_i(\beta_i)$. For multiplication of two ciphertexts with the following rescaling by p bits the output ciphertext has a relative error bounded by

$$\beta' = \beta_1 + \beta_2 + \beta_1\beta_2 + \frac{B_{\mathsf{mult}} + 2^{-p} \cdot B_{\mathsf{scale}}}{M_1 M_2}$$

from Lemmas 2 and 3. This relative error is close to $\beta_1 + \beta_2$ similar to the case of unencrypted floating-point multiplication under an appropriate choice of parameters.

For convenience of analysis, we will assume that for two ciphertexts with relatives errors β_1 and β_2 the relative error after multiplication and rescaling is bounded by $\beta_1 + \beta_2 + \beta^*$ for some fixed β^*

4 Homomorphic Evaluations of Matrix Operations

Along with the algorithms for homomorphic evaluation of approximate circuits, such as exponent, sigmoid, inverse, etc., described in Section 4 of [9], with the structure of `MHEAAN` we can also provide algorithms for homomorphic evaluation of approximate matrix multiplication and approximate matrix inverse.

4.1 Extracting Diagonal from a Matrix

One simple way to extract diagonal (or any other subset of values) from encrypted $n \times n$ matrix is described in Algorithm 1. We multiply the ciphertext by the identity matrix $\mathbf{I} \in \mathbb{C}^{n \times n}$ component wisely, with the following rescaling procedure. Due to rescaling procedure this method cost us p bits of a modulus.

Algorithm 1 Simple Diagonal Extraction

1: **procedure** SIMPLEDIAG($\mathtt{ct}_{\mathbf{A}}, \in \mathcal{R}_\ell^2, p$)
2: $\quad \mathtt{ct}_{\mathbf{D}} \leftarrow \mathtt{MHEAAN.CMult}(\mathtt{ct}_{\mathbf{A}}, \mathbf{I})$
3: $\quad \mathtt{MHEAAN.ReScale}(\mathtt{ct}_{\mathbf{D}}, p)$
4: $\quad$ **return** $\mathtt{ct}_{\mathbf{D}}$
5: **end procedure**

If we consider the following polynomial

$$i(X,Y) = X^{M_x - \frac{N_x}{2} + \frac{N_x}{2n}} \cdot \left(X^{\frac{N_x}{4}} + Y^{\frac{N_y}{4}}\right) \cdot \left(X^{\frac{N_x}{8}} + Y^{\frac{N_y}{8}}\right) \dots \left(X^{\frac{N_x}{2n}} + Y^{\frac{N_y}{2n}}\right)$$

we can easily check that $\phi_{n,n}^{-1}(i(X,Y)) = n \cdot \mathbf{I}$ which means $i(X,Y)$ exactly encodes the identity matrix with $\log n$ precision bits: $i(X,Y) = \mathtt{MHEAAN.Encode}(\mathbf{I}, \log n)$. So we can extract a diagonal from an encrypted matrix multiplying by the polynomial $i(X,Y)$ and rescaling by $\log n$ bits. Normally in practice p is notably larger than $\log n$.

Algorithm 2 Diagonal Extraction

1: **procedure** DIAG($\mathtt{ct}_{\mathbf{A}}, \in \mathcal{R}_\ell^2$)
2: $\quad \mathtt{ct}_{\mathbf{D}} \leftarrow \mathtt{MHEAAN.PolyMult}(\mathtt{ct}_{\mathbf{A}}, i)$
3: $\quad \mathtt{MHEAAN.ReScale}(\mathtt{ct}_{\mathbf{D}}, \log n)$
4: $\quad$ **return** $\mathtt{ct}_{\mathbf{D}}$
5: **end procedure**

By applying $\kappa_{0,N_y/2-k}$ to $i(X,Y)$ we obtain an encoding of a right shifted by k bits identity matrix. We can then extract a right shifted by k position diagonal from an encrypted matrix. We will call this procedure $\mathtt{Diag}_k$, where $\mathtt{Diag}_0$ is diagonal extraction. We will use this procedure in the following sections.

4.2 Matrix Multiplication

We adapt Fox matrix multiplication algorithm [18] to encrypted matrix multiplication. Let $\mathtt{ct}_{\mathbf{A}}, \mathtt{ct}_{\mathbf{B}}$ be encryptions of matrices $\mathbf{A}, \mathbf{B} \in \mathbb{C}^{n \times n}$ with power-of-two n. Let $\mathbf{I} \in \mathbb{C}^{n \times n}$ be the identity matrix, and $\mathtt{Diag}_k$ is procedure defined in previous section.

Algorithm 3 Matrix Multiplication

procedure MHEAAN.MatMult($\mathtt{ct}_{\mathbf{A}}, \mathtt{ct}_{\mathbf{B}} \in \mathcal{R}_\ell^2, p$)
  $\mathtt{ct}_{\mathbf{C}} \leftarrow 0$
  for $k = 0$ to $n - 1$ **do**
    $\mathtt{ct}_{\mathbf{B}_k} \leftarrow \mathtt{Diag}_k(\mathtt{ct}_{\mathbf{B}})$
    for $j = 1$ to $\log(n) - 1$ **do**
      $\mathtt{ct}_{\mathbf{B}_k} \leftarrow \mathtt{MHEAAN.Add}(\mathtt{ct}_{\mathbf{B}_k}, \mathtt{MHEAAN.Rotate}(\mathtt{ct}_{\mathbf{B}_k}, (0, 2^j))$
    end for
    $\mathtt{ct}_{\mathbf{A}_k} \leftarrow \mathtt{MHEAAN.ModDown}(\mathtt{MHEAAN.Rotate}(\mathtt{ct}_{\mathbf{A}}, (\frac{N_x}{2} - k, 0)), \log n)$
    $\mathtt{ct}_{\mathbf{C}_k} \leftarrow \mathtt{MHEAAN.Mult}(\mathtt{ct}_{\mathbf{A}_k}, \mathtt{ct}_{\mathbf{B}_k})$
    $\mathtt{ct}_{\mathbf{C}} \leftarrow \mathtt{MHEAAN.Add}(\mathtt{ct}_{\mathbf{C}}, \mathtt{ct}_{\mathbf{C}_k})$
  end for
  $\mathtt{ct}_{\mathbf{C}} \leftarrow \mathtt{MHEAAN.ReScale}(\mathtt{ct}_{\mathbf{C}}, p)$
  return $\mathtt{ct}_{\mathbf{C}}$
end procedure

Lemma 5 (Matrix Multiplication). *Let* $(\mathtt{ct}_{\mathbf{A}}, \beta_{\mathbf{A}} \cdot 2^p, 2^p, \ell)$ *and* $(\mathtt{ct}_{\mathbf{B}}, \beta_{\mathbf{B}} \cdot 2^p, 2^p, \ell)$ *be encryptions of matrices* $\mathbf{A}, \mathbf{B} \in \mathbb{C}^{n \times n}$ *respectively. The Algorithm 3 outputs* $(\mathtt{ct}_{\mathbf{C}}, \beta_{\mathbf{C}} \cdot n \cdot 2^p, n \cdot 2^p, \ell - p - \log n)$ *the valid encryption of* $\mathbf{C} = \mathbf{AB}$ *where* $\beta_{\mathbf{C}} = \beta_{\mathbf{A}} + \beta_{\mathbf{B}} + \beta^* + (\log n) \cdot \beta_*$.

Remark The plain matrix multiplication algorithm has complexity $O(n^3)$. The Algorithm 3 requires totally $O(n)$ ciphertext multiplication (each of provides multiplication in parallel of n^2 values) and $O(n \log n)$ ciphertext rotations. This is almost optimal, compare to unencrypted case.

Remark The Algorithm 3 can be generalized to multiplication of matrices with arbitrary dimensions. We will omit the details as we need to consider many cases, although they are essentially similar.

4.3 Matrix Inverse

For matrix inverse we can adapt Schulz algorithm [36] to encrypted approximate inverse circuit. However for `MHEAAN` we use a matrix version algorithm described in [7] and adopted in [9] as it more practical due to power-of-two degrees of matrix in the circuit. The algorithm is described below.

Assume that invertible square matrix $\mathbf{A}$ satisfies $\|\hat{\mathbf{A}}\| \leq \delta < 1$ for $\hat{\mathbf{A}} = \mathbf{I} - \frac{1}{2^t}\mathbf{A}$, for some $t \geq 0$ then we get

$$\frac{1}{2^t}\mathbf{A}(\mathbf{I} + \hat{\mathbf{A}})(\mathbf{I} + \hat{\mathbf{A}}^2)\dots(\mathbf{I} + \hat{\mathbf{A}}^{2^{r-1}}) = 1 - \hat{\mathbf{A}}^{2^r}$$

We can see that $\|\hat{\mathbf{A}}^{2^r}\| \leq \|\hat{\mathbf{A}}\|^{2^r} \leq \delta^{2^r}$, hence $\frac{1}{2^t}\prod_{j=0}^{r-1}(\mathbf{I} + \hat{\mathbf{A}}^{2^j}) = \mathbf{A}^{-1}(1 - \hat{\mathbf{A}^{2^r}})$ is an approximate inverse of $\mathbf{A}$ for $\delta^{2^r} \ll 1$. We will slightly strengthen the condition on δ in the following lemma:

Algorithm 4 Matrix Inverse

procedure MHEAAN.MatInv($\mathtt{ct}_{\bar{\mathbf{A}}} \in \mathcal{R}_\ell^2, r, p \in \mathbb{Z}$)
  $i = \mathtt{MHEAAN.Encode}(\mathbf{I}, p)$
  $\mathtt{ct}_{\mathbf{A}_0} \leftarrow \mathtt{ct}_{\bar{\mathbf{A}}}$
  $\mathtt{ct}_{\mathbf{V}_0} \leftarrow \mathtt{MHEAAN.ModDown}(i + \mathtt{ct}_{\bar{\mathbf{A}}}, p)$
  for $j = 0$ to $r - 1$ **do**
    $\mathtt{ct}_{\mathbf{A}_j} \leftarrow \mathtt{MHEAAN.MatMult}(\mathtt{ct}_{\mathbf{A}_{j-1}}, \mathtt{ct}_{\mathbf{A}_{j-1}})$
    $\mathtt{ct}_{\mathbf{V}_{j+1}} \leftarrow \mathtt{MHEAAN.MatMult}(\mathtt{ct}_{\mathbf{V}_j}, i + \mathtt{ct}_{\mathbf{A}_j})$
  end for
  $\mathtt{ct}_{\mathbf{V}_r} \leftarrow \mathtt{MHEAAN.ReScale}(\mathtt{ct}_{\mathbf{V}_r}, t)$
  return $\mathtt{ct}_{\mathbf{V}_r}$
end procedure

Lemma 6 (Matrix Inverse). *Let* $(\mathtt{ct}_{\bar{\mathbf{A}}}, \beta \cdot \delta 2^p/n, \delta 2^p/n, \ell)$ *be an encryption of matrix* $\bar{\mathbf{A}} \in \mathbb{C}^{n \times n}$*, and* $\|\bar{\mathbf{A}}\| = \|\mathbf{I} - \frac{1}{2^t}\mathbf{A}\| \leq \delta < \frac{n-1}{n}$ *for some* t*. The Algorithm 4 outputs* $(\mathtt{ct}_{\mathbf{V}_r}, \beta_{\mathbf{V}_r} \cdot n^{1/n} 2^{p-t}, n^{1/n} 2^{p-t}, \ell - (p + \log n)r - t)$ *the valid encryption of* $\mathbf{A}^{-1}$ *where* $\beta_{\mathbf{V}_r} = 2\beta + \beta^* + (r+1) \cdot (\beta^* + (\log n)\beta_*)$*. So we have that the output message bound is close to* 2^{p-t} *and error growth linearly in* r*.*

5 Implementation Results

In this section, we present parameters sets with experimental results. We also provide implementation results with concrete parameters. Our implementation is based on the NTL C++ library running over GMP. Every experimentation was performed on a machine with an Intel Core i5 running at 2.9 GHz processor using a parameter set with 80-bit security level. All computations are done in a single thread.

Parameters Setting The dimensions of a cyclotomic ring $\mathcal{R}$ are chosen following the security estimator of Albrecht et al. [1] for the learning with errors problem, within an additional assumption that 2-RLWE sample with dimensions N_x and N_y is indistinguishable from a sample belonging to RLWE with $N = N_x N_y$ (Prop. 1 in [30]). In the implementation, we used the Gaussian distribution of standard deviation $\sigma = 3.2$ to sample error polynomials, and set $h = 64$ as the number of nonzero coefficients in a secret key $s(X, Y)$. We skip the results of evaluation a component wise power, inverse, exponent, sigmoid functions, etc. Please refer to [9] for more details on evaluating these circuits.

Evaluation of Matrix Circuits In Table 1, we present the parameter setting and performance results for matrix transposition, multiplication, a 16-th power, and inverse. Parameter γ corresponds to the input precision bits and was calculated using Lemma 1. The average running times are only for ciphertext operations, excluding encryption and decryption procedures.

The homomorphic evaluation of the circuit $\mathbf{M}^{16}$ for an $n \times n$ matrix $\mathbf{M}$ is hard to be implemented in practice over the previous methods. Meanwhile, our

Table 1. Implementation results for $n \times n$ matrices $\mathbf{M}$, $\mathbf{M}_1$, $\mathbf{M}_2$

Function	Input params		HE params				Total time
	n	γ	N_x	N_y	L	p	
$\mathbf{M}^T$	16	17	2^7	2^7	100	30	64ms
	64						67ms
$\mathbf{M}_1\mathbf{M}_2$	16				100		6.2s
	64						29.1s
$\mathbf{M}^{16}$	16				280		44.9s
	64						3.9min
$\mathbf{M}^{-1}$	16	12			310	25	82.7s
	64						6.8min

scheme can compute this circuit by squaring a matrix 4 times and it takes about 4 minutes for a 64×64 matrix. Computing the matrix inverse homomorphically is done by evaluating a matrix polynomial up to degree 15 as was shown in Algorithm 4.

6 Conclusion

In this work, we presented `MHEAAN` scheme - a multidimensional variant of `HEAAN` homomorphic encryption scheme which supports an approximate arithmetics over ciphertexts. `MHEAAN` scheme takes advantage of the `HEAAN` scheme and provides a tensor packing method in a single ciphertext. We finally constructed a practical HE that supports standard HE operations as well as operations with matrices.

We believe that the idea of multidimensional variants could be applied to other existing schemes, with a great potential to homomorphic computations on matrices and tensors.

References

1. M. R. Albrecht, R. Player, and S. Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015.
2. J. W. Bos, K. Lauter, J. Loftus, and M. Naehrig. Improved security for a ring-based fully homomorphic encryption scheme. In *Cryptography and Coding*, pages 45–64. Springer, 2013.
3. Z. Brakerski. Fully homomorphic encryption without modulus switching from classical GapSVP. In *Advances in Cryptology–CRYPTO 2012*, pages 868–886. Springer, 2012.
4. Z. Brakerski, C. Gentry, and V. Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In *Proc. of ITCS*, pages 309–325. ACM, 2012.

5. Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, FOCS'11, pages 97–106. IEEE Computer Society, 2011.
6. Z. Brakerski and V. Vaikuntanathan. Fully homomorphic encryption from Ring-LWE and security for key dependent messages. In *Advances in Cryptology–CRYPTO 2011*, pages 505–524. Springer, 2011.
7. G. S. Çetin, Y. Doröz, B. Sunar, and W. J. Martin. An investigation of complex operations with word-size homomorphic encryption. Cryptology ePrint Archive, Report 2015/1195, 2015. `http://eprint.iacr.org/2015/1195`.
8. J. H. Cheon, A. Kim, M. Kim, and Y. Song. Implementation of HEAAN, 2016. `https://github.com/kimandrik/HEAAN`.
9. J. H. Cheon, A. Kim, M. Kim, and Y. Song. Homomorphic encryption for arithmetic of approximate numbers. In *Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Application of Cryptology and Information Security*, pages 409–437. Springer, 2017.
10. J. H. Cheon, M. Kim, and K. Lauter. Homomorphic computation of edit distance. In *International Conference on Financial Cryptography and Data Security*, pages 194–212. Springer, 2015.
11. J. H. Cheon and D. Stehlé. Fully homomophic encryption over the integers revisited. In *Advances in Cryptology–EUROCRYPT 2015*, pages 513–536. Springer, 2015.
12. J.-S. Coron, T. Lepoint, and M. Tibouchi. Scale-invariant fully homomorphic encryption over the integers. In *Public-Key Cryptography–PKC 2014*, pages 311–328. Springer, 2014.
13. I. Damgård, V. Pastro, N. Smart, and S. Zakarias. Multiparty computation from somewhat homomorphic encryption. In *Advances in Cryptology–CRYPTO 2012*, pages 643–662. Springer, 2012.
14. M. v. Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. Fully homomorphic encryption over the integers. In *Advances in Cryptology–EUROCRYPT 2010*, pages 24–43. Springer, 2010.
15. Y. Doröz, Y. Hu, and B. Sunar. Homomorphic AES evaluation using the modified LTV scheme. *Designs, Codes and Cryptography*, 80(2):333–358, 2016.
16. L. Ducas and D. Micciancio. FHEW: Bootstrapping homomorphic encryption in less than a second. In *Advances in Cryptology–EUROCRYPT 2015*, pages 617–640. Springer, 2015.
17. D. H. Duong, P. K. Mishra, and M. Yasuda. Efficient secure matrix multiplication over lwe-based homomorphic encryption. volume 67, pages 69–83. 2016.
18. G. Fox and S. Otto. Matrix algorithms on a hypercube i: Matrix multiplication *. *Parallel Computing*, 4:17–31, 1987.
19. C. Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. `http://crypto.stanford.edu/craig`.
20. C. Gentry, S. Halevi, and N. P. Smart. Homomorphic evaluation of the AES circuit. In *Advances in Cryptology–CRYPTO 2012*, pages 850–867. Springer, 2012.
21. C. Gentry, A. Sahai, and B. Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Advances in Cryptology–CRYPTO 2013*, pages 75–92. Springer, 2013.
22. S. Halevi and V. Shoup. Design and implementation of a homomorphic-encryption library. *IBM Research (Manuscript)*, 2013.

23. R. Hiromasa, M. Abe, and T. Okamoto. Packing messages and optimizing bootstrapping in gsw-fhe. In *Public-Key Cryptography – PKC 2015*, pages 699–715. Springer, 2015.
24. A. Kim, Y. Song, M. Kim, K. Lee, and J. H. Cheon. Logistic regression model training based on the approximate homomorphic encryption. *https://eprint.iacr.org/2018/254*.
25. M. Kim, Y. Song, and J. H. Cheon. Secure searching of biomarkers through hybrid homomorphic encryption scheme. *BMC medical genomics*, 10(2):42, 2017.
26. K. Lauter, A. López-Alt, and M. Naehrig. Private computation on encrypted genomic data. In *International Conference on Cryptology and Information Security in Latin America*, pages 3–27. Springer, 2014.
27. A. López-Alt, E. Tromer, and V. Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012*, pages 1219–1234. ACM, 2012.
28. V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In *Advances in Cryptology–EUROCRYPT 2010*, pages 1–23, 2010.
29. M. Naehrig, K. Lauter, and V. Vaikuntanathan. Can homomorphic encryption be practical? In *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*, pages 113–124. ACM, 2011.
30. A. Pedrouzo-Ulloa, J. R. Troncoso-Pastoriza, and F. Pérez-González. Multivariate lattices for encrypted image processing. In *IEEE ICASSP*. 2015.
31. A. Pedrouzo-Ulloa, J. R. Troncoso-Pastoriza, and F. Pérez-González. On ring learning with errors over the tensor product of number fields. 2016. `https://arxiv.org/abs/1607.05244`.
32. A. Pedrouzo-Ulloa, J. R. Troncoso-Pastoriza, and F. Pérez-González. Multivariate cryptosystems for secure processing of multidimensional signals. 2017. `https://arxiv.org/abs/1712.00848`.
33. C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem: Extended abstract. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 333–342, New York, NY, USA, 2009. ACM.
34. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, STOC '05, pages 84–93, New York, NY, USA, 2005. ACM.
35. H. Saberi Najafi and M. Shams Solary. Computational algorithms for computing the inverse of a square matrix, quasi-inverse of a nonsquare matrix and block matrices. *Appl. Math. Comput.*, 183:539–550, 2006.
36. G. Schulz. Iterative berechnung der reziproken matrix. *Zeitschrift für angewandte Mathematik und Mechanik*, 13:57–59, 1933.
37. S. Wang, Y. Zhang, W. Dai, K. Lauter, M. Kim, Y. Tang, H. Xiong, and X. Jiang. Healer: Homomorphic computation of exact logistic regression for secure rare disease variants analysis in GWAS. *Bioinformatics*, 32(2):211–218, 2016.

A Noise Estimations

We follow the heuristic approach in [20] generalizing it to two variable polynomials. Assume that a two variable polynomial $a(X,Y) \in \mathcal{R} = \mathbb{Z}[X,Y]/(\Phi_M(X), \Phi_{M_y}(Y))$

sampled from one of above distributions, so its nonzero entries are independently and identically distributed. The value $a(\xi_{M_x}, \xi_{M_y})$ can be obtained by first computing N_y inner products of vectors of coefficients of a corresponding to a power Y^j for $j = 0, \ldots, N_y - 1$ by a fixed vector $(1, \xi_{M_x}, \ldots, \xi_{M_x}^{N_x})$ of Euclidean norm $\sqrt{N_x}$ and then computing inner product of the results by a fixed vector $(1, \xi_{M_y}, \ldots, \xi_{M_y}^{N_y})$ of Euclidean norm $\sqrt{N_y}$. Then $a(\xi_{M_x}, \xi_{M_y})$ has variance $V = \sigma^2 N_x N_y = \sigma^2 N$, where σ^2 is the variance of each coefficient of a. Hence $a(\xi_{M_x}, \xi_{M_y})$ has the variances $V_U = 2^{2\ell} N/12$, $V_G = \sigma^2 N$ and $V_Z = \rho N$, when a is sampled from $\mathcal{R}_\ell$, $\mathcal{DG}(\sigma^2)$, $\mathcal{ZO}(\rho)$ respectively. In particular, $a(\xi_{M_x}, \xi_{M_y})$ has the variance $V_H = h$ when $a(X)$ is chosen from $\mathcal{HWT}(h)$. Moreover, we can assume that $a(\xi_{M_x}, \xi_{M_y})$ is distributed similarly to a Gaussian random variable over complex plane since it is a sum of many inde- pendent and identically distributed random variables. Every evaluations at roots of unity (ξ_{M_x}, ξ_{M_y}) share the same variance. Hence, we will use 6σ as a high-probability bound on the canonical embedding norm of a when each coefficient has a variance σ^2. For a multiplication of two independent random variables close to Gaussian distributions with variances σ_1^2 and σ_2^2, we will use a high-probability bound $16\sigma_1\sigma_2$

Proof of Lemma 1.

Proof. We choose $v \leftarrow \mathcal{ZO}(\rho)$, $e_0, e_1 \mathcal{DG}(\sigma)$, then set $\mathsf{ct} \leftarrow v \cdot \mathsf{pk} + (e_0, e_1 + m)$. The bound B_{clean} of encryption noise is computed by the following inequality:

$$\begin{aligned} \|\langle \mathsf{ct}, \mathsf{sk} \rangle - m \pmod{2^L}\|_\infty^{\mathsf{mcan}} &= \|v \cdot e + e_1 + e_0 \cdot s\|_\infty^{\mathsf{mcan}} \\ &\le \|v \cdot e\|_\infty^{\mathsf{mcan}} + \|e_1\|_\infty^{\mathsf{mcan}} + \|e_0 \cdot s\|_\infty^{\mathsf{mcan}} \\ &\le 8\sqrt{2} \cdot \sigma N + 6\sigma\sqrt{N} + 16\sigma\sqrt{hN}. \end{aligned}$$

□

Proof of Lemma 2.

Proof. It is satisfied that $\langle \mathsf{ct}, \mathsf{sk} \rangle = m + e \pmod{2^\ell}$ for some polynomial $e \in \mathcal{S}$ such that $\|e\|_\infty^{\mathsf{mcan}} < B$. The output ciphertext $\mathsf{ct}' \leftarrow \lfloor 2^{-p} \cdot \mathsf{ct} \rceil$ satisfies $\langle \mathsf{ct}', \mathsf{sk} \rangle = 2^{-p} \cdot (m + e) + e_{\mathsf{scale}} \pmod{2^{\ell - p}}$ for the rounding error vector $\tau = (\tau_0, \tau_1) = \mathsf{ct}' - 2^{-p} \cdot \mathsf{ct}$ and the error polynomial $e_{\mathsf{scale}} = \langle \tau, \mathsf{sk} \rangle = \tau_0 \cdot s + \tau_1$.

We may assume that each coefficient of τ_0 and τ_1 in the rounding error vector is computationally indistinguishable from the random variable in the interval $2^{-p} \cdot \mathbb{Z}_{2^p}$ with variance $\approx 1/12$. Hence, the magnitude of scale error polynomial is bounded by

$$\|e_{\mathsf{scale}}\|_\infty^{\mathsf{mcan}} \le \|\tau_0 \cdot s\|_\infty^{\mathsf{mcan}} + \|\tau_1\|_\infty^{\mathsf{mcan}} \le 6\sqrt{N/12} + 16\sqrt{hN/12}$$

as desired. □

Proof of Lemma 3.

Proof. Let $\mathsf{ct}_i = (a_i, b_i)$ for $i = 1, 2$. Then $\langle \mathsf{ct}_i, \mathsf{sk} \rangle = m_i + e_i \pmod{2^\ell}$ for some polynomials $e_i \in \mathcal{S}$ such that $\|e_i\|_\infty^{\mathsf{mcan}} \leq B_i$. Let $(d_0, d_1, d_2) = (a_1 a_2, a_1 b_2 + a_2 b_1, b_1 b_2)$. This vector can be viewed as an encryption of $m_1 \cdot m_2$ with an error $m_1 \cdot e_2 + m_2 \cdot e_1 + e_1 \cdot e_2$ with respect to the secret vector $(s^2, s, 1)$. It follows from Lemma 2 that the ciphertext $\mathsf{ct}_{\mathsf{mult}} \leftarrow (d_1, d_2) + \lfloor 2^{-L} \cdot (d_0 \cdot \mathsf{evk} \pmod{2^{\ell+L}}) \rceil$ contains an additional error $e'' = 2^{-L} \cdot d_0 e'$ and a rounding error bounded by B_{scale}. We may assume that d_0 behaves as a uniform random variable on $\mathcal{R}_\ell$, so $2^L \|e''\|_\infty^{\mathsf{can}}$ is bounded by $16\sqrt{N q_\ell^2/12}\sqrt{N\sigma^2} = 8N\sigma q_\ell/\sqrt{3} = B_{\mathsf{ks}} \cdot 2^\ell$. Therefore, $\mathsf{ct}_{\mathsf{mult}}$ is an encryption of $m_1 \cdot m_2$ with an error bounded by

$$\|m_1 e_2 + m_2 e_1 + e_1 e_2 + e''\|_\infty^{\mathsf{mcan}} + B_{\mathsf{scale}} \leq \\ M_1 B_2 + M_2 B_1 + B_1 B_2 + 2^{-L} \cdot 2^\ell \cdot B_{\mathsf{ks}} + B_{\mathsf{scale}}$$

as desired. □

Proof of Lemma 4.

Proof. Let prove the lemma for conjugation, proofs of others are the same. The vector $(a', b') = (\kappa_{-1,1}(a), \kappa_{-1,1}(b)) \pmod{2^\ell}$ can be viewed as an encryption of $\bar{\mathbf{Z}}$ with and error $\kappa_{-1,1}(e)$ with respect to the secret vector $(\kappa_{-1,1}(s), 1)$. Using proof of Lemma 3 we can get that $\mathsf{ct}_{\mathsf{cj}}$ is an encryption of $\bar{\mathcal{Z}}$ with an error bounded by

$$\|\kappa_{-1,1}(e) + e''\|_\infty^{\mathsf{mcan}} + B_{\mathsf{scale}} \leq B + 2^{-L} \cdot 2^\ell \cdot B_{\mathsf{ks}} + B_{\mathsf{scale}}$$

as desired. □

Proof of Lemma 5.

Proof. From Lemma 4 and the following remark about the relative error we can see that bound of message increase only after summations in line 10 of Algorithm 3, so the bound M of the output is equal to $n \cdot 2^p$. Note also that these summations do not increase the bound of the relative error. The relative error increases by β_* after rotation and increases by β^* after multiplication. So the relative error of each summand in line 10 is bounded by $\beta_{\mathbf{A}} + \beta_{\mathbf{B}} + \beta^* + (\log n) \cdot \beta_*$. □

Proof of Lemma 6.

Proof. From Lemma 5 the message of $\mathsf{ct}_{\mathbf{A}_j}$ is bounded by $\delta^{2^j} 2^p/n$ which implies that the message of $\mathsf{ct}_{\mathbf{V}_r}$ is bounded by

$$2^{p-t} \prod_{j=0}^{r-1} (1 + \delta^{2^j}/n) < \frac{2^{p-t}}{(1-\delta)^{1/n}} < n^{1/n} 2^{p-t}$$

The relative error β_j of $\mathtt{ct}_{\mathbf{A}_j}$ is bounded by $\beta_j \leq 2^j(\beta + \beta^* + (\log n)\beta_*)$, which implies that the relative error β'_j of $\mathtt{ct}_{\mathbf{A}_j} + i$ is bounded by

$$\beta'_j \leq \beta_j / \Big(1 + \frac{n}{\delta^{2^j}}\Big)$$

Using induction on j, we can show that a relative error β''_j of $\mathtt{ct}_{\mathbf{V}_j}$ is bounded by

$$\beta''_j \leq \Big(\sum_{k=0}^{j-1} \frac{2^k \delta^{2^k}}{n + \delta^{2^k}}\Big) \cdot (\beta + \beta^* + (\log n)\beta_*) + (j-1) \cdot (\beta^* + (\log n)\beta_*) \leq$$
$$\frac{1}{n} \sum_{k=0}^{j-1} (2^k \delta^{2^k}) \cdot (\beta + \beta^* + (\log n)\beta_*) + (j-1) \cdot (\beta^* + (\log n)\beta_*) \leq$$
$$\frac{2}{n(1-\delta)} \cdot (\beta + \beta^* + (\log n)\beta_*) + (j-1) \cdot (\beta^* + (\log n)\beta_*) \leq$$
$$2\beta + (j+1) \cdot (\beta^* + (\log n)\beta_*)$$

Finally after last rescaling we get the required result □