A Systematic Approach Towards Efficient Private Matrix Multiplication

Jinbao Zhu and Songze Li, Member, IEEE

Abstract—We consider the problems of Private and Secure Matrix Multiplication (PSMM) and Fully Private Matrix Multiplication (FPMM), for which matrices privately selected by a master node are multiplied at distributed worker nodes without revealing the indices of the selected matrices, even when a certain number of workers collude with each other. We propose a novel systematic approach to solve PSMM and FPMM with colluding workers, which leverages solutions to a related Secure Matrix Multiplication (SMM) problem where the data (rather than the indices) of the multiplied matrices are kept private from colluding workers. Specifically, given an SMM strategy based on polynomial codes or Lagrange codes, one can exploit the special structure inspired by the matrix encoding function to design private coded queries for PSMM/FPMM, such that the algebraic structure of the computation result at each worker resembles that of the underlying SMM strategy. Adopting this systematic approach provides novel insights in private query designs for private matrix multiplication, substantially simplifying the processes of designing PSMM and FPMM strategies. Furthermore, the PSMM and FPMM strategies constructed following the proposed approach outperform the state-of-the-art strategies in one or more performance metrics including recovery threshold (minimal number of workers the master needs to wait for before correctly recovering the multiplication result), communication cost, and computation complexity, demonstrating a more flexible tradeoff in optimizing system efficiency.

Index Terms—Coded distributed computing, secure matrix multiplication, private and secure matrix multiplication, fully private matrix multiplication, polynomial codes, Lagrange codes.

I. INTRODUCTION

In the era of Big Data, performing computationally intensive tasks on a single machine is becoming infeasible due to limited processing power and storage space. As an efficient solution, distributed computing has emerged as a natural approach to overcome such limitations, by partitioning the large computing task into smaller sub-tasks, and outsourcing them to many distributed worker nodes. However, scaling the computation across distributed workers is also faced with efficiency challenges including additional communication overhead compared to centralized processing, and prolonged task execution time due to slow or delay-prone worker nodes, known as the straggler effect [1], [2]. Meanwhile, distributing sensitive raw data across worker nodes may raise serious security and privacy concerns. Therefore, designing computation and communication efficient strategies that are robust to straggler effect, while providing data privacy and security is of vital importance for distributed computing applications.

Matrix multiplication, as one of the key building blocks in various engineering applications like machine learning and big data analysis, is typically carried out in a distributed manner for practically sized input matrices [3]–[6]. In this paper we focus on improving the computation and communication efficiency of two distributed private matrix multiplication problems, over a distributed computing system consisting of a master node and N worker nodes. For the first Private and Secure Matrix Multiplication (PSMM) problem, as illustrated in Fig. 1, the master owns a confidential matrix A and all workers have access to a library $L^B$ of V public matrices $B^{(1)}, B^{(2)}, \ldots, B^{(V)}$. The goal of the master is to compute the product $AB^{(\theta)}$ for some $\theta \in \{1, 2, \ldots, V\}$ from the distributed system, while keeping the index $\theta$ private and the matrix A secure from any up to T colluding workers.

To do that, the master sends an encoded version of the matrix A to each worker, along with a query that instructs the worker to encode the library $L^B$ and compute a response for the master. To mitigate the influence of stragglers, the master...
Secure Matrix Multiplication (SMM) is another problem that is related to the interested PSMM and FPMM problems. In SMM, the master wishes to compute the product of two owned matrices \( A \) and \( B \) in the distributed system, without revealing anything about \( A \) and \( B \) to the workers. Computing strategies for the SMM problem, based on how the matrices \( A \) and \( B \) are securely encoded, can be categorized into SMM based on polynomial codes [7], [9]–[15] and SMM based on Lagrange codes [16], [17], where polynomial codes [18]–[21] and Lagrange codes [22] are constructed by leveraging the algebraic structure of polynomial functions and Lagrange interpolate polynomials, respectively. An essential component behind these coded strategies is to construct appropriate encoding functions of \( A \) and \( B \), such that the desired product \( AB \) can be recovered by interpolating a polynomial from worker responses. The state-of-the-art strategies for SMM based on polynomial codes and Lagrange codes are reflected in [11], [15], and [16], respectively. Having observed the similarities between PSMM/FPMM and SMM in requiring privacy-preserving matrix multiplication, and their key difference that whether or not queries for the workers are needed, we are interested in the question:

Would it be possible to construct an efficient PSMM/FPMM strategy, simply via designing private queries on top of an SMM strategy?

We answer the above question in the affirmative, and propose a novel systematic approach to construct efficient computation strategies for both PSMM and FPMM problems. Specifically, as illustrated in Fig. 3, we start with an SMM strategy (based on polynomial codes or Lagrange codes), and make use of the special structure inspired by the matrix encoding functions in the SMM strategy to create private queries that facilitate a form of interference alignment, separating the desired and interfering partitions of the matrices in the libraries, such that the response computed at each worker has identical structure as that of the SMM strategy. Consequently, as in the SMM strategy, the desired product can be recovered via polynomial interpolation from the responses, with the same recovery threshold. Our major contributions in this paper are

- Establishes a generic connection between the PSMM/FPMM and the SMM problem, which helps to significantly simplify the design process of a PSMM/FPMM strategy;
- Compared with state-of-the-art PSMM and FPMM strategies, the strategies constructed from the proposed approach introduce new private query designs, and achieve more flexible tradeoffs between recovery threshold, communication cost and computation complexity, further improving the system efficiency. See Section VII for detailed comparisons.

A. Related Work

Coded computing has recently emerged as a technique of utilizing information/coding theoretical tools to inject redundant data and computations into distributed computing systems, to mitigate communication and straggler bottlenecks, and provide security and privacy for various computation tasks (see, e.g., [5], [10], [11], [15], [16], [19], [20], [22]–[33]). Privately retrieving a message from a distributed storage system without revealing the index of the message has been studied extensively in the problem of Private Information Retrieval (PIR) [34]–[40] in recent years. With the focus on index privacy, the PSMM and FPMM problems can be viewed as secure matrix multiplication problems with additional PIR

![Fig. 3. Illustration of the proposed approach to design PSMM and FPMM solutions from a SMM solution and a compatible private query design.](image-url)
requirement on the indices of interested matrices within public libraries.

**Private and Secure Matrix Multiplication:** The problem of private matrix multiplication was first introduced in [41] without colluding constraint (i.e., $T = 1$) and security guarantee on $A$. The work [8] imposed the security constraint on $A$ to consider a non-colluding PSMM problem, and improved the recovery threshold of the strategy proposed in [41], through adopting the random query design in [35] to ensure privacy, and employing polynomial codes [18] to complete desired computation. Subsequently in [30], the authors presented another non-colluding PSMM strategy that combines MDS-coded PIR scheme [38] with polynomial codes, and show that the strategy outperforms [8] in terms of upload and download communication costs. However, the strategy in [30] provides no resistance to stragglers, and has a high computation complexity. Further in [42], for the asymptotic setting (i.e., the number of matrices $V \to \infty$), a better tradeoff between upload and download cost was achieved by exploiting the idea of PIR based on Cross Subspace Alignment (CSA) [43], at the expense of a higher computation complexity. The authors proposed in [7], [44] novel strategies for the non-colluding PSMM problem using the query design in [8], yielding a more flexible tradeoff between recovery threshold, communication cost and computation complexity. Lagrange codes [22] were also employed to create non-colluding PSMM strategy [16], with the help of the query design in [8] and bilinear complexity [45], [46]. A very recent work [47] presented a computation strategy based on polynomial codes for the $T$-colluding PSMM problem considered in this paper, but it requires excessive communication cost and computation complexity.

**Fully Private Matrix Multiplication:** Much less work has been done in the literature for FPMM problem. In [16], a non-colluding FPMM strategy was proposed based on Lagrange codes and bilinear complexity, by resorting to the query design in [8]. Later in [48], a $T$-colluding FPMM strategy was introduced by using the idea of CSA.

In general, the current works [7], [16], [44] have well addressed the problems of PSMM and FPMM without colluding constraint. It is valuable to note that, all these works construct their strategies using the query design in [8] to ensure privacy, i.e., there is no difference in the private queries sent to workers. While the PSMM and FPMM problems with colluding constraint have been studied in [47], [48], these strategies require either high recovery threshold or huge communication cost and computation complexity.

### B. Organization

The rest of this paper is organized as follows. In Section II, we formally formulate the problems of PSMM and FPMM. In Section III, we review the problem of SMM and its strategies based on polynomial codes and Lagrange codes. In Section IV, we summarize the main results of the paper. Sections V and VI present the proposed computation strategies for PSMM and FPMM, respectively, by exploiting the encoding structure of SMM strategies. Section VII gives comparison with other related work. Finally, the paper is concluded in Section VIII.

**Notation:** Let boldface and cursive capital letters represent matrices and sets, respectively, e.g., $A$ and $K$. For a finite set $K$, $|K|$ denotes its cardinality. Denote $\mathbb{Z}^+$ the set of positive integers. For any $m, n \in \mathbb{Z}^+$ such that $m < n$, $[n]$ and $[m : n]$ denote the sets $\{1, 2, \ldots, n\}$ and $\{m, m+1, \ldots, n\}$, respectively. Define $A_K$ as $\{A_{k_1}, \ldots, A_{k_m}\}$ for any index set $K = \{k_1, \ldots, k_m\} \subseteq [n]$.

### II. Problem Formulations

Consider a distributed computing system consisting of one master node and $N$ worker nodes, where each worker is connected to the master through an orthogonal communication link. The workers are honest-but-curious, which means that they will follow the prescribed protocol faithfully, yet may potentially collude to infer information about additional data inputs. We consider two private distributed matrix computation problems of private and secure matrix multiplication and fully private matrix multiplication. In the rest of this section, we describe the formulations of these two problems respectively.

#### A. Private and Secure Matrix Multiplication

For the problem of Private and Secure Matrix Multiplication (PSMM) depicted in Fig. 1, the master owns a confidential matrix $A$ of dimension $\lambda \times \omega$, and all workers have access to a library $\mathcal{L}^B$ of $V$ public matrices $B^{(1)}, B^{(2)}, \ldots, B^{(V)}$ with dimension $\lambda \times \gamma$, for some $\lambda, \omega, \gamma \in \mathbb{Z}^+$. All the entries of the matrices are over a finite field $\mathbb{F}_q$ for some prime power $q$.

The master privately selects an index $\theta \in [V]$ and wishes to compute the product $AB^{(\theta)}$ from the distributed system, while keeping its indexed index $A$ private and its own matrix $A$ secure from any colluding subset of up to $T$ out of the $N$ workers. To this end, the master employs a computation strategy of PSMM consisting of the following three phases:

- **Sharing:** To ensure security, the master locally generates a private randomness, denoted by $Z^A$, which is used to encode the matrix $A$ according to encoding functions $f = (f_1, \ldots, f_N)$, where $f_i$ is the encoding function for worker $i$. Denote the encoded version of matrix $A$ for worker $i$ by $\tilde{A}_i$, i.e.,

  $$\tilde{A}_i = f_i(A, Z^A), \quad \forall i \in [N].$$

To privately complete computation, the master also generates $N$ queries $Q^{(\theta)}_i$ based on the index $\theta$ and another locally generated private randomness $Z^\theta$. Then the encoded matrix $\tilde{A}_i$ and the query $Q^{(\theta)}_i$ are shared with worker $i \in [N]$.

- **Computation:** Upon receiving $Q^{(\theta)}_i$, worker $i$ first uses encoding function $h_i$ to encode the library $\mathcal{L}^B$ into $\tilde{B}_i$, i.e.,

  $$\tilde{B}_i = h_i(B^{(V)}, Q^{(\theta)}_i), \quad \forall i \in [N],$$

and then computes the response $Y_i^{(\theta)}$ and sends it back to the master, which is a deterministic function of the received $\tilde{A}_i$ and the encoded matrix $\tilde{B}_i$. 


• Reconstruction: For some design parameter $K \leq N$, the master only waits for the responses from the fastest $K$ workers, and recovers the desired product $\mathbf{AB}^{(\theta)}$ from their responses. This allows the computation strategy to tolerate any subset of up to $N - K$ stragglers.

A valid PSMM strategy must satisfy the following three requirements.

- **Privacy Constraint:** The strategies for computing any two distinct products $\mathbf{AB}^{(\theta)}$ and $\mathbf{AB}^{(\theta')}$ must be indistinguishable with respect to any $T$ colluding workers, i.e., for all $\theta \neq \theta' \in [V]$ and $T \subseteq [N], |T| = T$,

$$
\left( Q_T^{(\theta)}, \tilde{A}_T, \mathbf{B}^{(\theta)}(V), Y_T^{(\theta)} \right) \sim \left( Q_T^{(\theta')}, \tilde{A}_T, \mathbf{B}^{(\theta')}(V), Y_T^{(\theta')} \right),
$$

where $X \sim Y$ means that the random variables $X$ and $Y$ are identically distributed. Equivalently, the index $\theta$ of the desired product is hidden from all the information available to any $T$ colluding workers, i.e.,

$$
I(\theta; Q_T^{(\theta)}, \tilde{A}_T, \mathbf{B}^{(\theta)}(V), Y_T^{(\theta)}) = 0 .
$$

- **Security Constraint:** Any $T$ colluding workers must not learn any information about the confidential matrix $\mathbf{A}$, i.e., for all $T \subseteq [N], |T| = T$,

$$
I(\mathbf{A}; Q_T^{(\theta)}, \tilde{A}_T, \mathbf{B}^{(\theta)}(V), Y_T^{(\theta)}) = 0 .
$$

- **Correctness Constraint:** The desired product should be correctly reconstructed from the collection of responses of any fastest $K$ workers, i.e.,

$$
H(\mathbf{AB}^{(\theta)}|Y_K^{(\theta)}) = 0 , \quad \forall K \subseteq [N], |K| = K .
$$

The performance of a PSMM strategy is evaluated by the following key metrics:

1. The recovery threshold $K$, which is the minimum number of workers that the master needs to wait for in order to recover the desired product $\mathbf{AB}^{(\theta)}$.
2. The communication cost, which is comprised of the upload cost for matrix $\mathbf{A}$ and download cost from workers, defined as

$$
P_u \triangleq \frac{\sum_{i=1}^{N} H(\tilde{A}_i)}{\lambda_0}, \quad P_d \triangleq \max_{K : K \subseteq [N], |K| = K} \frac{H(Y_K^{(\theta)})}{\lambda_Y} .
$$

which are normalized with the number of symbols contained in the matrix $\mathbf{A}$ and the desired product $\mathbf{AB}^{(\theta)}$, respectively.

3. The computation complexity, which includes the complexities of encoding, worker computation and decoding. The encoding complexity $C_e$ at the master is defined as the number of arithmetic operations required to compute the encoding functions $f$. The complexity of worker computation $C_w$ is defined as the maximal number of arithmetic operations required to compute the response $Y_i^{(\theta)}$, over all worker $i \in [N]$. Finally, the decoding complexity $C_d$ at the master is defined as the maximal number of arithmetic operations required to decode the desired product $\mathbf{AB}^{(\theta)}$ from the responses of fastest workers in $K$, over all $K \subseteq [N]$ with $|K| = K$.

B. Fully Private Matrix Multiplication

We describe the problem of Fully Private Matrix Multiplication (FPMM) illustrated in Fig. 2. In contrast to the above PSMM problem where the master has a confidential matrix $\mathbf{A}$, in the FPMM problem, there is a library $\mathcal{L}^A$ of $U$ public matrices $\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(U)} \in \mathbb{F}_q^{K \times N}$ that are accessible to the $N$ workers, i.e., each worker has access to the two libraries $\mathcal{L}^A$ and $\mathcal{L}^B$. The master is interested in computing the product $\mathbf{A}^{(\theta)}\mathbf{B}^{(\theta_2)}$ utilizing the distributed computing system, while keeping the indices of the desired product $\theta_1$ and $\theta_2$ private from any $T$ colluding workers, for any $\theta_1 \in [U]$ and $\theta_2 \in [V]$.

To do so, similar to PSMM, a computation strategy for FPMM operates in the following three phases:

- **Sharing:** The master generates the queries $Q^{(\theta_1)}_i$ and $Q^{(\theta_2)}_i$ for the two libraries $\mathcal{L}^A$ and $\mathcal{L}^B$ according to the interested indices $\theta_1$ and $\theta_2$ and locally generated private randomness $Z^{(\theta_1)}$ and $Z^{(\theta_2)}$, respectively, and then shares $Q^{(\theta_1)}_i$ and $Q^{(\theta_2)}_i$ with worker $i \in [N]$.

- **Computation:** Upon receiving $Q^{(\theta_1)}_i$ and $Q^{(\theta_2)}_i$, worker $i$ first encodes the two libraries $\mathcal{L}^A$ and $\mathcal{L}^B$ using encoding functions $f_i$ and $h_i$, respectively. The encoded versions of $\mathcal{L}^A$ and $\mathcal{L}^B$ for worker $i \in [N]$ are given by

$$
\tilde{A}_i = f_i(\mathbf{A}^{(\theta_1)}, Q_i^{(\theta_1)}), \quad \tilde{B}_i = h_i(\mathbf{B}^{(\theta_2)}, Q_i^{(\theta_2)}).$$

Then the worker $i$ computes the response $Y_i^{(\theta_1, \theta_2)}$ and sends it back to the master, which is a function of the encoded matrices $\tilde{A}_i$ and $\tilde{B}_i$.

- **Reconstruction:** The master recovers the desired product $\mathbf{A}^{(\theta_1)}\mathbf{B}^{(\theta_2)}$ from the responses of any fastest $K$ workers. A valid computation strategy for FPMM must satisfy the following constraints.

- **Fully Privacy Constraint:** The desired indices $\theta_1$ and $\theta_2$ must be hidden from all the information available to any $T$ colluding workers, i.e., for all $T \subseteq [N], |T| = T$,

$$
I(\theta_1, \theta_2; Q_T^{(\theta_1)}, Q_T^{(\theta_2)}, \mathbf{A}^{(\theta_1)}(V), \mathbf{B}^{(\theta_2)}(V), Y_T^{(\theta_1, \theta_2)}) = 0 .
$$

- **Correctness Constraint:** With the responses of any fastest $K$ workers, the desired product must be recovered, i.e.,

$$
H(\mathbf{A}^{(\theta_1)}\mathbf{B}^{(\theta_2)}|Y_K^{(\theta_1, \theta_2)}) = 0 , \quad \forall K \subseteq [N], |K| = K .
$$

Similar to the PSMM problem, the performance of an FPMM strategy is evaluated by the following key quantities: 1) the recovery threshold $K$; 2) the normalized download cost $P_d$; and 3) the computation complexities consisting of generating response at each worker $C_w$ and decoding desired product at the master $C_d$.

For the above formulated PSMM and FPMM problems, our goal in this paper is to design efficient computation strategies that minimize the recovery threshold, the communication cost and the computation complexity. As the first step, we review...
the strategies proposed to solve a related distributed matrix multiplication problem, which serves as a building block of our approach to solve the private matrix multiplication problems.

III. PRELIMINARIES: SECURE MATRIX MULTIPLICATION

In this section, we summarize the state-of-the-art computation strategies for the Secure Matrix Multiplication (SMM) problem, which will be exploited to construct the strategies for PSMM and FPMM in the following sections. In the SMM problem, the master owns two confidential matrices $A \in \mathbb{F}_q^{m \times n}$ and $B \in \mathbb{F}_q^{n \times \lambda}$ and is interested in computing the product $C = AB$ in the distributed computing system, without revealing anything about $A$ and $B$ to any $T$ colluding workers.

We first introduce a lemma that will be used in security and privacy proofs.

**Lemma 1 (Generalized Secret Sharing [15], [49]):** For any parameters $L, T, k, r \in \mathbb{Z}^+$, let $W_1, \ldots, W_L \in \mathbb{F}_q^{L \times T}$ be $L$ secrets, and $Z_1, \ldots, Z_T$ be $T$ random matrices with the same dimensions as secrets whose entries are chosen independently and uniformly from $\mathbb{F}_q$. Let $\alpha_1, \ldots, \alpha_N$ be $N$ pairwise distinct elements from $\mathbb{F}_q$. Define a function of $x$ as

$$y(x) = W_1u_1(x) + \ldots + W_Lu_L(x) + Z_1\ell_1(x) + \ldots + Z_T\ell_T(x),$$

where $u_1(x), \ldots, u_L(x), \ell_1(x), \ldots, \ell_T(x) \in \mathbb{F}_q[x]$ are the determininistic functions of $x$. If the matrix

$$V = \begin{bmatrix} v_1(\alpha_1) & \cdots & v_T(\alpha_1) \\ \vdots & \ddots & \vdots \\ v_1(\alpha_T) & \cdots & v_T(\alpha_T) \end{bmatrix}_{T \times T}$$

is non-singular over $\mathbb{F}_q$ for any $T = \{i_1, \ldots, i_T\} \subseteq [N]$ with $|T| = T$, then the $T$ values $y(\alpha_{i_1}), \ldots, y(\alpha_{i_T})$ can not learn any information about the secrets $W_1, \ldots, W_L$, i.e.,

$$I(y(\alpha_{i_1}), \ldots, y(\alpha_{i_T}); W_1, \ldots, W_L) = 0.$$  

Let $m, p, n \in \mathbb{Z}^+$ be any partitioning parameters of data matrices such that $m|\lambda, p|\omega$ and $n|\gamma$. To efficiently exploit the computation power of distributed workers and establish the feasible tradeoff between system performance, the matrices $A$ and $B$ are partitioned into $m \times p$ and $p \times n$ equal-size sub-matrices, respectively, as shown below:

$$A = \begin{bmatrix} A_{1,1} & \cdots & A_{1,p} \\ \vdots & \ddots & \vdots \\ A_{m,1} & \cdots & A_{m,p} \end{bmatrix}, \quad B = \begin{bmatrix} B_{1,1} & \cdots & B_{1,n} \\ \vdots & \ddots & \vdots \\ B_{p,1} & \cdots & B_{p,n} \end{bmatrix},$$

where $A_{k,\ell} \in \mathbb{F}_q^{\frac{m}{\gamma} \times \frac{\lambda}{\omega}}$ for any $k \in [m], \ell \in [p]$ and $B_{\ell,j} \in \mathbb{F}_q^{\frac{\lambda}{\gamma} \times \frac{\omega}{\omega}}$ for any $\ell \in [p], j \in [n]$. Accordingly, the desired product $C = AB$ involves a total of $mn$ linear combinations of products of sub-matrices, i.e.,

$$C = AB = \begin{bmatrix} C_{1,1} & \cdots & C_{1,n} \\ \vdots & \ddots & \vdots \\ C_{m,1} & \cdots & C_{m,n} \end{bmatrix},$$

where $C_{k,j} = \sum_{\ell=1}^{p} A_{k,\ell}B_{\ell,j}$ for all $k \in [m], j \in [n]$.

To the best of our knowledge, the state-of-the-art strategies for SMM with arbitrary partitioning of matrices above can be divided into two categories in terms of coding techniques, i.e., SMM based on polynomial codes [11], [15] and SMM based on Lagrange codes [16]. The essential components behind these coded strategies lie in constructing the encoding functions of matrices $A$ and $B$, denoted by $f(x)$ and $h(x)$ respectively, such that the desired products of sub-matrices $C_{k,j}, k \in [m], j \in [n]$ can be recovered by interpolating the product polynomial $g(x) = f(x) \cdot h(x)$. Next, we present these two approaches and their performance.

A. Secure Matrix Multiplication Based on Polynomial Codes

Let $Z^A_1, \ldots, Z^A_T$ and $Z^B_1, \ldots, Z^B_T$ be $T$ random matrices distributed independently and uniformly on $\mathbb{F}_q^{\frac{m}{\gamma} \times \frac{\lambda}{\omega}}$ and $\mathbb{F}_q^{\frac{\lambda}{\gamma} \times \frac{\omega}{\omega}}$, respectively. In general, the goal of SMM strategies based on polynomial codes is to design a group of appropriate positive integers $(a_k, b_{\ell,j}, c_{i}, d_i : k \in [m], \ell \in [p], j \in [n], i \in [T])$ to construct the encoding functions of matrices $A$ and $B$ (5) as

$$f_p(x) = \sum_{k=1}^{m} \sum_{\ell=1}^{p} A_{k,\ell}x^{a_k+\ell} + \sum_{i=1}^{T} Z^A_i x^{c_i},$$

$$h_p(x) = \sum_{\ell=1}^{p} \sum_{j=1}^{n} B_{\ell,j}x^{b_{\ell,j}+i} + \sum_{i=1}^{T} Z^B_i x^{d_i},$$

such that the following criteria are satisfied:

C1 The product polynomial

$$g(x) = \sum_{r=0}^{\delta} g_r x^r = f_p(x) \cdot h_p(x)$$

contains all the desired sub-products $C_{k,j}, k \in [m], j \in [n]$ as coefficients, i.e., $C_{k,j} \in \{g_r : r \in [0 : \delta]\}$ for all $k \in [m], j \in [n]$, where $g_r$ is the coefficient of $x^r$ in $f_p(x) \cdot h_p(x)$, and $\delta$ is the degree of the polynomial $f_p(x) \cdot h_p(x)$, given by

$$\delta \triangleq \max \{a_k, b_{\ell,j}, c_i : k \in [m], \ell \in [p], i \in [T]\} + \max \{b_{\ell,j}, d_i : \ell \in [p], j \in [n], i \in [T]\}.$$  

C2 The following matrices

$$\left[\begin{array}{c} a_1^{(c_1)} a_2^{(c_1)} \cdots a_{T}^{(c_1)} \\ a_1^{(c_2)} a_2^{(c_2)} \cdots a_{T}^{(c_2)} \\ \vdots \ \vdots \ \vdots \\ a_1^{(c_T)} a_2^{(c_T)} \cdots a_{T}^{(c_T)} \end{array}\right]_{T \times T} = \left[\begin{array}{c} a_1^{(d_1)} a_2^{(d_1)} \cdots a_{T}^{(d_1)} \\ a_1^{(d_2)} a_2^{(d_2)} \cdots a_{T}^{(d_2)} \\ \vdots \ \vdots \ \vdots \\ a_1^{(d_T)} a_2^{(d_T)} \cdots a_{T}^{(d_T)} \end{array}\right]_{T \times T}$$

are non-singular over $\mathbb{F}_q$ for any $T = \{i_1, \ldots, i_T\} \subseteq [N]$ with $|T| = T$, where $a_1, a_2, \ldots, a_N$ are $N$ pairwise distinct non-zero elements from $\mathbb{F}_q$.

To complete the computation $C = AB$ (6), the master first shares the evaluations of $f_p(x)$ and $h_p(x)$ at $x = a_i$ with worker $i \in [N]$. Then the worker $i$ computes the product $f_p(a_i) \cdot h_p(a_i)$ and sends it back to the master on successful completion, which is equivalent to evaluating of the polynomial $g(x) = f_p(x) \cdot h_p(x)$ at $x = a_i$. Thus, the master can interpolate $g(x)$ from any $K = \delta + 1$ responses by using Lagrange interpolation rule, and then recovers all the
desired products of sub-matrices $C_{k,j}, k \in [m], j \in [n]$ from the coefficients of $g(x)$ by $C1$. The matrices $A$ and $B$ are secure against any $T$ colluding workers by C2 and Lemma 1. Therefore, the SMM strategy based on polynomial codes achieves the recovery threshold $K = \delta + 1$.

As far as we know, the state-of-the-art strategies for SMM based on polynomial codes are reflected in [11], [15], which are summarized in the following lemma.

Lemma 2: For any arbitrary partitioning of matrices $A$ and $B$ with parameters $m, p, n$, the state-of-the-art strategies for SMM based on polynomial codes achieve

- the recovery threshold $K = (m + 1)(np + T) - 1$ [15] by setting $a_{k,\ell} = (k - 1)(np + T) + \ell - 1, b_{\ell,j} = jp - \ell, c_{\ell} = (m - 1)(np + T) + np + t - 1, d_{\ell} = np + t - 1$ for all $k \in [m], \ell \in [p], j \in [n], t \in [T]$,
- the recovery threshold $K = (n + 1)(mp + T) - 1$ [15] by setting $a_{k,\ell} = (k - 1)p + \ell - 1, b_{\ell,j} = (j - 1)(mp + T) + p - \ell, c_{\ell} = mp + t - 1, d_{\ell} = (n - 1)(mp + T) + mp + t - 1$ for all $k \in [m], \ell \in [p], j \in [n], t \in [T]$,
- and the recovery threshold $K = 2mpn + 2T - 1$ [11] by setting $a_{k,\ell} = (k - 1)p + \ell - 1, b_{\ell,j} = jp - \ell, c_{\ell} = mpn + t - 1, d_{\ell} = mpn + t - 1$ for all $k \in [m], \ell \in [p], j \in [n], t \in [T]$.

That is, the current best SMM strategies based on polynomial codes achieve the recovery threshold $K = \min((m + 1)(np + T) - 1, (n + 1)(mp + T) - 1, 2mpn + 2T - 1)$.

B. Secure Matrix Multiplication Based on Lagrange Codes

Lagrange codes were originally introduced in [22] and is widely applied to solve batch processing problems for coded distributed computing [17], [19]. To solve SMM using Lagrange codes [16], [20], matrix multiplication is first converted into the problem of computing the element-wise product of two batches of sub-matrices, by employing the concept of bilinear complexity, and then Lagrange codes can directly operate on the problem of batch sub-matrix multiplication.

Definition 1 (Bilinear Complexity [45], [46]): Let $C = [C_{k,j}]_{k \in [m], j \in [n]}$ be the product of any matrices $A = [A_{k,j}]_{k \in [m], j \in [p]}$ and $B = [B_{\ell,j}]_{\ell \in [p], j \in [n]}$, where $C_{k,j} = \sum_{\ell = 1}^{p} A_{k,\ell} B_{\ell,j}$. The bilinear complexity, denoted by $R(m, p, n)$, is defined as the minimum number of active multiplications for the problem of multiplying the two matrices $A$ and $B$. Moreover, an upper bound construction with rank $R$ for bilinear complexity\footnote{The upper bound construction of bilinear complexity is known for many cases of parameters $m, p, n$, based on the recursive method in [45].} means that, there exists tensors $a \in \mathbb{F}_q^{R \times m \times p}, b \in \mathbb{F}_q^{R \times p \times n}, c \in \mathbb{F}_q^{R \times m \times n}$ satisfying

\[
\sum_{r = 1}^{R} c_{r,k,j} \left( \sum_{\ell = 1}^{m} \sum_{\ell' = 1}^{p} a_{r,k,\ell} A_{k,\ell'} c_{\ell'} + \sum_{\ell = 1}^{p} b_{r,\ell,j} B_{\ell,j} \right) = a_{r,k,j}, \quad b_{r,\ell,j} = b_{r,\ell,j}, \quad \forall k \in [m], j \in [n].
\]

It is straightforward to observe from Definition 1 that, bilinear complexity enables converting the matrix multiplication problem $C = AB$ in (6) into computing the element-wise products of two batches of sub-matrices of length $R$. Specifically, given any upper bound construction for bilinear complexity with tensors $a = (a_{r,k,\ell}), b = (b_{r,\ell,j}), c = (c_{r,k,j})$ and rank $R$, the desired products of sub-matrices $C_{k,j}$ can be recovered by

\[
\sum_{r = 1}^{R} c_{r,k,j} A_{r,\ell,j} B_{r,\ell,j} = C_{r,k,j}, \quad \forall k \in [m], j \in [n],
\]

if one obtains the element-wise product $\{A_{r,\ell,j}, B_{r,\ell,j}, R \in [R]\}$ of the two batches of sub-matrices $(A_1, A_2, \ldots, A_R)$ and $(B_1, B_2, \ldots, B_R)$, where

\[
c_{r,k,j} = \sum_{\ell = 1}^{m} a_{r,k,\ell} A_{r,\ell,j}, \quad \forall r \in [R],
\]

\[
b_{r,\ell,j} = \sum_{k = 1}^{n} b_{r,\ell,j} B_{r,\ell,j}, \quad \forall r \in [R].
\]

That is, bilinear complexity converts the matrix multiplication $C = AB$ into the problem of computing the element-wise product of two batches of sub-matrices $(A_1, A_2, \ldots, A_R)$ and $(B_1, B_2, \ldots, B_R)$. Then Lagrange codes are operated as follows.

Let $\{\beta_r, \alpha_i : r \in [R], i \in [N]\}$ be $R + T + N$ distinct elements from $\mathbb{F}_q$. Construct the encoding functions $f_L(x)$ and $h_L(x)$ of the two batches of sub-matrices as Lagrange interpolation polynomials of degree $R + T - 1$, such that

\[
f_L(\beta_r) = \left\{ \begin{array}{ll} A_r, & \forall r \in [R] \\ Z^A_r, & \forall r \in [R + 1 : R + T] \end{array} \right.,
\]

\[
h_L(\beta_r) = \left\{ \begin{array}{ll} B_r, & \forall r \in [R] \\ Z^B_r, & \forall r \in [R + 1 : R + T] \end{array} \right.,
\]

where $Z^A_{R+1}, \ldots, Z^A_{R+T}$ and $Z^B_{R+1}, \ldots, Z^B_{R+T}$ are random matrices over $\mathbb{F}_q$ with the same dimensions as $A_r$ and $B_r$, respectively. By Lagrange interpolation rule, the polynomials $f_L(x)$ and $h_L(x)$ are written as

\[
f_L(x) = \sum_{r = 1}^{R} A_r \prod_{j \in [R+T]\setminus\{r\}} \frac{x - \beta_j}{\beta_r - \beta_j} + \sum_{k = R + 1}^{R + T} Z^A_k \prod_{j \in [R+T]\setminus\{k\}} \frac{x - \beta_j}{\beta_k - \beta_j},
\]

\[
h_L(x) = \sum_{r = 1}^{R} B_r \prod_{j \in [R+T]\setminus\{r\}} \frac{x - \beta_j}{\beta_r - \beta_j} + \sum_{k = R + 1}^{R + T} Z^B_k \prod_{j \in [R+T]\setminus\{k\}} \frac{x - \beta_j}{\beta_k - \beta_j}.
\]

To complete computation, the master first shares the evaluations of $f_L(x)$ and $h_L(x)$ at point $x = \alpha_i$ with worker $i \in [N]$, who then responds with the product $f_L(\alpha_i) \cdot h_L(\alpha_i)$. Apparently, the master can interpolate the product polynomial $g(x) = f_L(x) \cdot h_L(x)$ from any $K = \deg(g(x)) + 1 = 2R + 2T - 1$ responses, and then evaluates $g(x)$ at $x = \beta_1, \beta_2, \ldots, \beta_R$ to obtain the element-wise product $\{A, B : r \in [R]\}$ by (13)-(14). It is straightforward to prove the security of $A$ and $B$ by Lemma 1. Hence, the SMM strategy based on Lagrange codes achieves the recovery threshold $K = 2R + 2T - 1$.  

TABLE I

| Partitioning Parameters | Best Known Bilinear Complexity $R$ [50] | Recovery Threshold for PSMM |
|-------------------------|----------------------------------------|----------------------------|
| $m = 2, p = 2, n = 2$   | 7                                      | $\min\{11 + 3T, 15 + 2T\}$ |
| $m = 3, p = 3, n = 3$   | 23                                     | $\min\{35 + 4T, 53 + 2T\}$ |
| $m = 5, p = 5, n = 5$   | 98                                     | $\min\{149 + 6T, 249 + 2T\}$ |

IV. MAIN RESULTS

We state our main results in this section. For brevity, we focus on the results of recovery threshold, and present the communication costs and computation complexities of proposed strategies for PSMM (resp. FPMM) in Section V-C (resp. Section VI-C).

Theorem 1: For the problem of private and secure matrix multiplication with $T$ colluding workers and the partitioning parameters $m, p, n$, give any positive integers $\{a_k, b_{\ell j}, c_{\ell}, d_t : k \in [m], \ell \in [p], j \in [n], t \in [T]\}$ satisfying C1-C2, there exists a computation strategy based on polynomial codes that achieves a recovery threshold of $K = \max\{a_k, b_{\ell j}, c_{\ell}, d_t : k \in [m], \ell \in [p], j \in [n], t \in [T]\} + \max\{b_{\ell j}, d_t : \ell \in [p], j \in [n], t \in [T]\} + 1$.

Theorem 1 is proved in Section V-A by presenting our proposed PSMM strategy based on polynomial codes. The confidential matrix $A$ and index $\theta$ are secured by polynomial codes such that the response computed at each worker resembles that of the SMM strategy in Section III-A.

From Lemma 2, we can directly obtain the following corollary of Theorem 1.

Corollary 1: The recovery threshold $K = \min\{(m+1)(np+T)−1, (n+1)(mp+T)−1, 2mpn+2T−1\}$ can be achieved by some PSMM strategy based on polynomial codes.

Remark 1: While the recovery threshold presented in Corollary 1 is obtained by adopting the degree parameters in Lemma 2 for the underlying polynomial codes, we can flexibly optimize system performance over degree parameters $\{a_k, b_{\ell j}, c_{\ell}, d_t : k \in [m], \ell \in [p], j \in [n], t \in [T]\}$ that satisfy conditions C1-C2. This remains to be an interesting future research problem.

Theorem 2: For the problem of private and secure matrix multiplication with $T$ colluding workers and the partitioning parameters $m, p, n$, there exists a computation strategy based on Lagrange codes that achieves a recovery threshold of $K = 2R + 2T − 1$, where $R$ denotes the rank of any construction for bilinear complexity of multiplying an $m$-by-$p$ matrix and a $p$-by-$n$ matrix.

Theorem 2 is proved in Section V-B by constructing a PSMM strategy based on Lagrange codes. The confidential matrix $A$ and index $\theta$ are secured by Lagrange codes such that the response computed at each worker resembles that of the SMM strategy in Section III-B.

Remark 2: As far as we know, the general upper bound construction $R$ for bilinear complexity remains open. Reference [50] lists the current best known upper construction of bilinear complexity for almost all possible partitioning parameters $m, p, n$ with $m \in [2 : 32], n \leq p \leq n \leq 32$. For some specific combinations of parameters $(m, p, n)$, we compare the recovery thresholds achieved by our proposed PSMM strategies in Table I. For each specific combination of $(m, p, n)$, which strategy achieves a smaller recovery threshold depends on the value of the security parameter $T$. For example, for the parameter case of $(m = 3, p = 3, n = 3)$, the PSMM strategy based on polynomial codes outperforms the one based on Lagrange codes in terms of recovery threshold when $T < 5$. A similar discussion holds for the following FPMM strategies based on polynomial codes and Lagrange codes. The detailed comparisons between the two proposed strategies for PSMM and FPMM problems are presented in Sections V-C and VI-C, respectively.

Theorem 3: For the problem of fully private matrix multiplication with $T$ colluding workers and the partitioning parameters $m, p, n$, give any positive integers $\{a_k, b_{\ell j}, c_{\ell}, d_t : k \in [m], \ell \in [p], j \in [n], t \in [T]\}$ satisfying C1-C2, there exists a computation strategy based on polynomial codes that achieves a recovery threshold of $K = \min\{a_k, b_{\ell j}, c_{\ell}, d_t : k \in [m], \ell \in [p], j \in [n], t \in [T]\} + \max\{b_{\ell j}, d_t : \ell \in [p], j \in [n], t \in [T]\} + 1$.

Theorem 3 is proved in Section VI-A by presenting a FPMM strategy based on polynomial codes. The confidential matrix indices $\theta_1$ and $\theta_2$ are secured by polynomial codes such that the response computed at each worker resembles that of the SMM strategy in Section III-A.

The following corollary is immediate from Lemma 2.

Corollary 2: The recovery threshold $K = \min\{(m+1)(np+T)−1, (n+1)(mp+T)−1, 2mpn+2T+1\}$ can be achieved by some FPMM strategy based on polynomial codes.

Theorem 4: For the problem of fully private matrix multiplication with $T$ colluding workers and the partitioning parameters $m, p, n$, there exists a computation strategy based on Lagrange codes that achieves a recovery threshold of $K = 2R + 2T − 1$, where $R$ denotes the rank of any construction for bilinear complexity of multiplying an $m$-by-$p$ matrix and a $p$-by-$n$ matrix.

Theorem 4 is proved in Section VI-B by presenting a FPMM strategy based on Lagrange codes. The confidential matrix indices $\theta_1$ and $\theta_2$ are secured by Lagrange codes such that the response computed at each worker resembles that of the SMM strategy in Section III-B.

Remark 3: We may further consider the presence of some adversarial workers of size $E$ who maliciously return arbitrarily erroneous responses to the master. In our proposed strategies for PSMM and FPMM, the responses of all the workers can be viewed as evaluations of a polynomial at distinct points, and accordingly the responses constitute a Reed-Solomon codeword. Thus, our proposed strategies can provide robustness against the $E$
adversarial workers by waiting for responses from 2E more workers.

V. COMPUTATION STRATEGIES FOR PRIVATE AND SECURE MATRIX MULTIPLICATION

In this section, we first present two PSMM strategies, which are constructed by exploiting the structure of SMM strategies based on polynomial codes and Lagrange codes, respectively. Then their security, privacy, communication cost and computation complexities are analyzed. This provides tools for Theorems 1 and 2.

To better attain the tradeoff with respect to system performance, similar to (5), the matrices $A$ and $B^{(i)}$ are divided into $m \times p$ and $p \times n$ equal-size sub-matrices, respectively, for any partitioning parameters $m, p, n$, i.e., for all $v \in [V]$,

$$
A = \begin{bmatrix}
A_{1,1} & \cdots & A_{1,p} \\
\vdots & \ddots & \vdots \\
A_{m,1} & \cdots & A_{m,n}
\end{bmatrix}, \quad B^{(v)} = \begin{bmatrix}
B^{(v)}_{1,1} & \cdots & B^{(v)}_{1,n} \\
\vdots & \ddots & \vdots \\
B^{(v)}_{p,1} & \cdots & B^{(v)}_{p,n}
\end{bmatrix},
$$

(17)

where $A_{k,\ell} \in \mathbb{F}_q^{\frac{m \times n}{p}}$ for any $k \in [m], \ell \in [p]$, and $B^{(v)}_{j,\ell} \in \mathbb{F}_q^{\frac{m \times n}{p}}$ for any $\ell \in [p], j \in [n]$. Then the desired product $C^{(v)} = AB^{(v)}$ is given by

$$
C^{(v)} = AB^{(v)} = \begin{bmatrix}
C^{(v)}_{1,1} & \cdots & C^{(v)}_{1,n} \\
\vdots & \ddots & \vdots \\
C^{(v)}_{m,1} & \cdots & C^{(v)}_{m,n}
\end{bmatrix} = \sum_{\ell=1}^{p} A_{k,\ell} B^{(v)}_{\ell,j}
$$

where $C^{(v)}_{k,j} = \sum_{\ell=1}^{p} A_{k,\ell} B^{(v)}_{\ell,j}$ for any $k \in [m], j \in [n]$.

A. PSMM Strategy Based on Polynomial Codes

We start with proving Theorem 1. We show that any SMM strategy on polynomial codes with recovery threshold $K = 17$ and the following assignment [15]:

$$
a_{1,1} = 0, \quad a_{1,2} = 1, \quad a_{2,1} = 6, \quad a_{2,2} = 7, \quad c_1 = 10, \quad c_2 = 11, \\
b_{1,1} = 1, \quad b_{1,2} = 3, \quad b_{2,1} = 0, \quad b_{2,2} = 2, \quad d_1 = 4, \quad d_2 = 5.
$$

Then the encoding functions of $A$ and $B^{(1)}$ are in the forms of

$$
f_A(x) = A_{1,1} + A_{1,2}x + A_{2,1}x^6 + A_{2,2}x^7 + Z_1^1 x^{10} + Z_2^1 x^{11},
$$

(20)

$$
h_B(x) = B^{(1)}_{2,1} + B^{(1)}_{1,1}x + B^{(1)}_{2,2}x^6 + B^{(1)}_{1,2}x^7 + Z_1^2 x^{14} + Z_2^2 x^{15},
$$

(21)

where $Z_1^1, Z_2^1, Z_1^2, Z_2^2$ are the matrices with corresponding dimensions and will be specified later. The computation $C^{(1)}$ can be completed by interpolating the polynomial $g(x)$:

$$
g_1 = A_{1,1}B^{(1)}_{1,1} + A_{1,2}B^{(1)}_{2,1} = C^{(1)}_{1,1},
$$

$$
g_2 = A_{1,1}B^{(1)}_{1,2} + A_{1,2}B^{(1)}_{2,2} = C^{(1)}_{1,2},
$$

$$
g_3 = A_{1,1}B^{(1)}_{1,1} + A_{1,2}B^{(1)}_{2,2} = C^{(1)}_{1,2},
$$

$$
g_7 = A_{2,1}B^{(1)}_{1,1} + A_{2,2}B^{(1)}_{2,1} = C^{(1)}_{2,1},
$$

$$
g_9 = A_{2,1}B^{(1)}_{1,2} + A_{2,2}B^{(1)}_{2,2} = C^{(1)}_{2,2}.
$$

In PSMM, to ensure security, let $Z_1^i, Z_2^i$ be independently and uniformly random matrices over $\mathbb{F}_q$. Then the master shares $\tilde{A}_i = f_{A}(\alpha_i)$ with worker $i$, where $\alpha_1, \alpha_2, \ldots, \alpha_m$ are pairwise distinct non-zero elements from $\mathbb{F}_q$. Along with $\tilde{A}_i$, for the partitioning sub-matrices $B^{(1)}_{\ell,j}$ and $B^{(2)}_{\ell,j}$, the master also shares with the queries $q^{(1)}_{\ell,j}(\alpha)$ and $q^{(2)}_{\ell,j}(\alpha)$, respectively, for all $\ell, j = 1, 2$, which are given by

$$
q^{(1)}_{\ell,j}(\alpha) = \alpha_i^{h_{\ell,j}} + z_{\ell,j}^{1,\alpha} + z_{\ell,j}^{2,\alpha},
$$

$$
q^{(2)}_{\ell,j}(\alpha) = z_{\ell,j}^{1,\alpha} + z_{\ell,j}^{2,\alpha},
$$

where $z_{\ell,j}^{1,\alpha}, z_{\ell,j}^{2,\alpha}$ are uniformly random noises in $\mathbb{F}_q$ that protect $T = 2$ colluding parties. Upon the queries, each worker $i$ encodes $B^{(1)}_{\ell,i}, B^{(2)}_{\ell,i}$ into

$$
\tilde{B}_i = B^{(1)}_{\ell,i}q^{(1)}_{\ell,i}(\alpha) + B^{(1)}_{1,2}q^{(1)}_{1,2}(\alpha) + B^{(1)}_{2,1}q^{(1)}_{2,1}(\alpha)
$$

$$
+ B^{(1)}_{2,2}q^{(1)}_{2,2}(\alpha) + B^{(1)}_{1,1}q^{(1)}_{1,1}(\alpha) + B^{(1)}_{1,2}q^{(1)}_{1,2}(\alpha) + B^{(1)}_{2,1}q^{(1)}_{2,1}(\alpha) + B^{(1)}_{2,2}q^{(1)}_{2,2}(\alpha)
$$

$$
+ B^{(2)}_{\ell,i}q^{(2)}_{\ell,i}(\alpha) + B^{(2)}_{1,2}q^{(2)}_{1,2}(\alpha) + B^{(2)}_{2,1}q^{(2)}_{2,1}(\alpha) + B^{(2)}_{2,2}q^{(2)}_{2,2}(\alpha)
$$

$$
= B^{(1)}_{\ell,i} + B^{(1)}_{1,1}\alpha_i + B^{(1)}_{2,2}\alpha_i^2 + B^{(1)}_{1,2}\alpha_i^3 + Z_1^{\alpha_i^4} + Z_2^{\alpha_i^5} = h_B(\alpha_i),
$$

where we set $Z_B^{(i)} = B^{(1)}_{1,2}q^{(1)}_{1,2}(\alpha) + B^{(1)}_{2,1}q^{(1)}_{2,1}(\alpha) + B^{(1)}_{2,2}q^{(1)}_{2,2}(\alpha) + B^{(2)}_{1,2}q^{(2)}_{1,2}(\alpha) + B^{(2)}_{2,2}q^{(2)}_{2,2}(\alpha) + B^{(2)}_{1,1}q^{(2)}_{1,1}(\alpha) + B^{(2)}_{1,2}q^{(2)}_{1,2}(\alpha) + B^{(2)}_{2,2}q^{(2)}_{2,2}(\alpha)$ for any $i = 1, 2$ that is constant for all workers and align interference to the dimension corresponding to $x^t$ if $t = 1$ or $x^{2t}$ if $t = 2$, similar to (21).

Each worker $i$ computes $\tilde{A}_i \tilde{B}_i$ as a response, which can be viewed as evaluating of $g(x) = f_{A}(x) \cdot h_B(x)$ at point $x = \alpha_i$. Hence, the master can interpolate the product $g(x)$ from any
$K = \deg(g(x)) + 1 = 17$ responses, and recovers the desired computation $AB^{(1)}$.

Next, we formally describe the general PSMM construction based on polynomial codes. Let the positive integers $[a_{\ell}, c_i, b_{\ell, t}, c_t, d_t : k \in [m], \ell \in [p], j \in [n], t \in [T]]$ satisfying C1-C2 be the parameters of the SMM strategy based on polynomial codes. To ensure the security of its own matrix $A$ (17), the master employs the encoding function defined in (7) to encode $A$ as

$$f_A(x) = \sum_{k=1}^{m} \sum_{\ell=1}^{p} A_{k, \ell} x^{a_{k, \ell}} + \sum_{i=1}^{T} Z_i^{A} x^i,$$  \hspace{1cm} (22)

where $Z_1^{A}, \ldots, Z_T^{A}$ are the random matrices over $\mathbb{F}_q$ with the same dimension as $A_{k, \ell}$. To keep the index $\theta$ private, the master generates $VTpn$ random noises $\{z^{(v)}_{\ell,j} : \ell \in [p], j \in [n], v \in [V], t \in [T]\}$ independently and uniformly from $\mathbb{F}_q$. Then based on the structure of the encoding function defined in (8), construct the query polynomial $q^{(v)}_{\ell,j}(x)$ for any $\ell \in [p], j \in [n], v \in [V]$ as

$$q^{(v)}_{\ell,j}(x) = \sum_{i=1}^{T} z^{(v)}_{\ell,j,i} x^i + \begin{cases} \delta^{(v)}_{\ell,j,1} x_1, & \text{if } v = \theta \\ 0, & \text{if } v \neq \theta \end{cases}.$$

(23)

Here the number of the query polynomials is deliberately designed to be equal to the number of the partitioning submatrices in the library $L^{B}$. In particular, each query polynomial $q^{(v)}_{\ell,j}(x)$ corresponds to the sub-matrix $B_{\ell,j}$ for any $\ell \in [p], j \in [n], v \in [V]$, and will be used as encoding coefficient to encode $B_{\ell,j}$, as shown in (26). When $v = \theta$, the sub-matrix $B_{\ell,j}$ is desired to be computed and the coefficient $x_1$ is used to encode $B_{\ell,j}$ in the same way as the encoding function in (8). When $v \neq \theta$, the coefficient 0 is used to eliminate interference from the undesired sub-matrix $B_{\ell,j}$. Moreover, the term $\sum_{i=1}^{T} z^{(v)}_{\ell,j,i} x^i$ is designed to possess robustness against $T$ colluding privacy, and to align interference from all the partitioning sub-matrices in $L^{B}$ (see (29)) since it has identical structure across all these partitioning submatrices.

Let $a_1, a_2, \ldots, a_N$ be the pairwise distinct non-zero elements in $\mathbb{F}_q$. The master shares the evaluations of $f_A(x)$ and $\{q^{(v)}_{\ell,j}(x) : \ell \in [p], j \in [n], v \in [V]\}$ at point $x = a_i$ with worker $i$, i.e.,

$$\tilde{A}_i = f_A(a_i), \hspace{1cm} Q^{(v)}_i = \{q^{(v)}_{\ell,j}(a_i) : \ell \in [p], j \in [n], v \in [V]\}.$$  \hspace{1cm} (24)

After receiving the query $Q^{(v)}_i$, worker $i$ computes the matrices $B^{(v)}_i$ by taking a linear combination of the elements in $Q^{(v)}_i$ and all the partitioning sub-matrices of $B^{(v)}_i$ (17), given by

$$B_i = \sum_{v=1}^{V} \sum_{\ell=1}^{p} \sum_{j=1}^{n} B^{(v)}_{\ell,j} \cdot q^{(v)}_{\ell,j}(a_i).$$

(25)

Denote the encoding function of $B^{(v)}_i$ by

$$h_{B}(x) = \sum_{v=1}^{V} \sum_{\ell=1}^{p} \sum_{j=1}^{n} B^{(v)}_{\ell,j} \cdot q^{(v)}_{\ell,j}(x).$$

(26)

where (a) follows by (23), and

$$Z^{B}_i = \sum_{v=1}^{V} \sum_{\ell=1}^{p} \sum_{j=1}^{n} B^{(v)}_{\ell,j} \cdot q^{(v)}_{\ell,j}(x), \hspace{1cm} \forall t \in [T]$$

(27)

which are identical for all workers and thus can be viewed as constant terms.

We can observe from (28)-(29) that, the encoding function $h_{B}(x)$ of $B^{(v)}_i$ efficiently separates the desired submatrices and the interference in the same structure as the encoding function in (8), where the desired sub-matrices $B^{(v)}_{\ell,j}\in[p],j\in[n]$ appear along the $pn$ dimensions corresponding to $\{x^{(v)}_{\ell,j}\}_{\ell\in[p],j\in[n]}$ and the interference from all the partitioning sub-matrices of $B^{(v)}_i$ is aligned along the $T$ dimensions corresponding to $\{x^{(v)}_{\ell,j}\}_{t\in[T]}$. Thus, we have the fact from the SMM strategy based on polynomial codes in Section III-A that, the desired computation $C^{(\theta)}_i = AB^{(\theta)}$ can be recovered from the product polynomial $g(x) = f_A(x) \cdot h_{B}(x)$.

Next, each worker $i$ computes the product

$$Y^{(\theta)}_i = \tilde{A}_i B_i$$

(28)

and send it back to the master, which is equivalent to evaluating of the polynomial $g(x) = f_A(x) \cdot h_{B}(x)$ at point $x = a_i$ by (24) and (26)-(27). Thus, the master can interpolate $g(x)$ from the responses of any $K = \deg(g(x)) + 1 = \max\{a_{\ell,c_i,c_t} : k \in [m], \ell \in [p], j \in [T]\} + \max\{b_{\ell,j}, d_t : \ell \in [p], j \in [n], t \in [T]\} + 1$ workers and then recovers $C^{(\theta)}_i = AB^{(\theta)}$.

### B. PSMM Strategy Based on Lagrange Codes

Now we state the strategy that proves Theorem 2. We will show that, given any upper bound construction $a = (a_{r,c_i,c_t}), b = (b_{r,c_i})$ with rank $R$ for bilinear complexity of multiplying an $m$-by-$p$ matrix and a $p$-by-$n$ matrix, the SMM strategy based on Lagrange codes can be exploited to construct a PSMM strategy with same recovery threshold. Let us start with an example to illustrate the idea.

**Example 2:** Consider the same parameters as Example 1, i.e., $V = m = p = n = T = 2$. The partitions of matrices $A, B^{(1)}, B^{(2)}$ and the desired computation $C^{(1)} = AB^{(1)}$ are shown in (18) and (19), respectively.

Firstly, we use Strassen’s construction [46] with bilinear complexity $R = 7$ to encode the sub-matrices of $A$ and $B^{(1)}, B^{(2)}$ as

$$A_1 = A_{1,1} + A_{2,2}, \hspace{1cm} B^{(1)}_1 = B^{(1)}_{1,1} + B^{(1)}_{2,2}, \hspace{1cm} B^{(2)}_1 = B^{(2)}_{1,1} + B^{(2)}_{2,2},$$

$$A_2 = A_{2,1} + A_{2,2}, \hspace{1cm} B^{(1)}_2 = B^{(1)}_{1,1} + B^{(1)}_{2,2}, \hspace{1cm} B^{(2)}_2 = B^{(2)}_{1,1},$$

$$A_3 = A_{1,1}, \hspace{1cm} B^{(1)}_3 = B^{(1)}_{1,2} - B^{(1)}_{2,2}, \hspace{1cm} B^{(2)}_3 = B^{(2)}_{1,2} - B^{(2)}_{2,2}.$$
that, the element-wise product \( \{ \beta \} \) by evaluating of the product polynomial

\[
\sum_{r=1}^{7} A_{r} \prod_{j \in [9]} x - \beta_{j}^{r}
\]

\[
\sum_{r=8}^{9} A_{r} \prod_{j \in [9]} x - \beta_{j}^{r}
\]

The desired computation \( C^{(1)} \) in (19) can be recovered from the element-wise product \( \{ A_{r} B_{r}^{(1)} \}_{r \in [7]} \) by

\[
C_{1,1} = A_{1} B_{1}^{(1)} + A_{2} B_{2}^{(1)} - A_{3} B_{3}^{(1)} + A_{7} B_{7}^{(1)},
\]

\[
C_{1,2} = A_{3} B_{3}^{(1)} + A_{5} B_{5}^{(1)},
\]

\[
C_{1,2} = A_{2} B_{2}^{(1)} + A_{4} B_{4}^{(1)},
\]

\[
C_{2,2} = A_{1} B_{1}^{(1)} - A_{3} B_{3}^{(1)} + A_{3} B_{3}^{(1)} + A_{6} B_{6}^{(1)}.
\]

Let \( \{ \beta, \alpha : r \in [9], i \in [N] \} \) be \( N+9 \) distinct elements from \( \mathbb{F}_{q} \.

Further, we know from SMM based on Lagrange codes that, the element-wise product \( \{ A_{r} B_{r}^{(1)} \}_{r \in [7]} \) can be obtained by evaluating the product polynomial \( g(x) = f_{A}(x) \cdot h_{B}(x) \) at points \( x = \beta_{1}, \ldots, \beta_{7} \), where

\[
f_{A}(x) = \sum_{r=1}^{7} A_{r} \prod_{j \in [9]} x - \beta_{j}^{r}
\]

\[
h_{B}(x) = \sum_{r=1}^{7} B_{r}^{(1)} \prod_{j \in [9]} x - \beta_{j}^{r}
\]

Here \( Z_{A}^{0}, Z_{B}^{0} \) and \( Z_{A}^{1}, Z_{B}^{1} \) are the matrices with corresponding dimensions and will be specified later.

In PSMM, let \( Z_{A}^{r}, Z_{B}^{r} \) be random matrices to be ensured security over \( \mathbb{F}_{q} \), and the master shares \( \tilde{A}_{i} = f_{A}(\alpha_{i}) \) with worker \( i \). Moreover, for any \( r \in [7] \), the master also shares the queries \( q^{(1)}(\alpha_{i}) \) and \( q^{(2)}(\alpha_{i}) \) for the sub-matrices \( B_{r}^{(1)} \) and \( B_{r}^{(2)} \), respectively, which are given by

\[
q^{(1)}(\alpha_{i}) = \prod_{j \in [9]} \frac{\alpha_{i} - \beta_{j}^{r}}{\beta_{j}^{r} - \beta_{j}}
\]

\[
q^{(2)}(\alpha_{i}) = \prod_{j \in [9]} \frac{\alpha_{i} - \beta_{j}^{r}}{\beta_{j}^{r} - \beta_{j}}
\]

where \( q_{r,8}^{(1)}, q_{r,9}^{(1)} \) and \( q_{r,8}^{(2)}, q_{r,9}^{(2)} \) are random noises from \( \mathbb{F}_{q} \). Then worker \( i \) encodes the library into

\[
\tilde{B}_{i} = \sum_{r=1}^{7} B_{r}^{(1)} q^{(1)}(\alpha_{i}) + \sum_{r=1}^{7} B_{r}^{(2)} q^{(2)}(\alpha_{i})
\]

\[
\sum_{r=1}^{7} B_{r}^{(1)} \prod_{j \in [9]} \frac{\alpha_{i} - \beta_{j}^{r}}{\beta_{j}^{r} - \beta_{j}}
\]

\[
\sum_{r=1}^{7} B_{r}^{(2)} \prod_{j \in [9]} \frac{\alpha_{i} - \beta_{j}^{r}}{\beta_{j}^{r} - \beta_{j}}
\]

where we set \( Z_{B}^{k} = \sum_{r=1}^{7} B_{r}^{(1)} q_{r,k}^{(1)} + \sum_{r=1}^{7} B_{r}^{(2)} q_{r,k}^{(2)} \) for any \( k = 8, 9 \). Next, worker \( i \) computes \( A_{i} \tilde{B}_{i} \), as a response, which is the evaluation of \( g(x) = f_{A}(x) \cdot h_{B}(x) \) at point \( x = \alpha_{i} \). Hence, the master can interpolate the product \( g(x) \) from any \( K = \deg(g(x)) + 1 = 17 \) responses, and then recovers \( C^{(1)} = A B^{(1)} \).

The general construction is described as follows. Similar to (11) and (12), the master and each worker converts the matrices \( A \) and \( B^{(v)} \) into a batch of sub-matrices of length \( R \), respectively, as shown below.

\[
A_{r} = \sum_{k=1}^{p} \sum_{l=1}^{m} a_{r,k,l} A_{k,l}, \quad \forall r \in [R].
\]

\[
B_{r}^{(v)} = \sum_{\ell=1}^{n} \sum_{j=1}^{v} b_{r,\ell,j} B_{\ell,j}^{(v)}, \quad \forall r \in [R], v \in [V].
\]

By (10), the master can recover the desired computation \( A_{r} B^{(v)} \) if it is able to obtain the element-wise product \( \{ A_{r} B^{(v)} \}_{r \in [R]} \) of the two batches of sub-matrices \( \{ A_{1}, \ldots, A_{R} \} \) and \( \{ B_{1}, \ldots, B_{R}^{(v)} \} \).

Let \( \{ \beta_{r}, \alpha_{i} : r \in [R + T], i \in [N] \} \) be \( R + T + N \) pairwise distinct elements from \( \mathbb{F}_{q} \). To complete the computation, the master employs the encoding function defined in (15) to encode the batch of sub-matrices \( \{ A_{1}, \ldots, A_{R} \} \) as

\[
f_{A}(x) = \sum_{r=1}^{R} A_{r} \prod_{j \in [R + T] \setminus [r]} x - \beta_{j}^{r}
\]

\[
+ \prod_{k=R+1}^{R+T} Z_{A}^{k} \prod_{j \in [R + T] \setminus [k]} x - \beta_{j}^{k}
\]

where \( Z_{A}^{0}, \ldots, Z_{A}^{R+T} \) are random matrices over \( \mathbb{F}_{q} \) with the same dimension as \( A_{r} \).

To keep the index \( \theta \) private, let \( \{ z_{r,k}^{(v)} : k \in [R + 1 : R + T], r \in [R], v \in [V] \} \) be \( TRV \) independently and uniformly random noises from \( \mathbb{F}_{q} \). Then, given any \( v \in [V], r \in [R] \), based on the structure of the encoding function defined in (16), the master constructs the query polynomial \( q_{r}^{(v)}(x) \) as

\[
q_{r}^{(v)}(x) = \sum_{k=R+1}^{R+T} z_{r,k}^{(v)} \prod_{j \in [R + T] \setminus [k]} x - \beta_{j}^{k}
\]

\[
+ \prod_{j \in [R + T] \setminus [r]} x - \beta_{j}^{r} \quad \text{if } v = \theta,
\]

\[
0 \quad \text{if } v \neq \theta.
\]

Then the master evaluates \( f_{A}(x) \) and \( \{ q_{r}^{(v)}(x) : r \in [R], v \in [V] \} \) at point \( x = \alpha_{i} \), and sends them to worker \( i \), i.e.,

\[
\tilde{A}_{i} = f_{A}(\alpha_{i}),
\]

\[
\tilde{Q}^{(v)}_{i} = \{ q_{r}^{(v)}(\alpha_{i}) : r \in [R], v \in [V] \}.
\]

Upon the received query, worker \( i \) encodes the matrices \( B_{r}^{(v)} \) into

\[
\tilde{B}_{i} = \sum_{v=1}^{V} \sum_{r=1}^{R} B_{r}^{(v)} \cdot q_{r}^{(v)}(\alpha_{i}).
\]
By (34), we can denote the encoding function \( h_\mathbf{B}(x) \) of matrices \( \mathbf{B}^{(v)} \) by

\[
h_\mathbf{B}(x) = \sum_{v=1}^{V} \sum_{r=1}^{R} \mathbf{B}_{r}^{(v)} \cdot q_r^{(v)}(x)
\]

\[
= \sum_{r=1}^{R} \mathbf{B}_{r}^{(0)} \cdot \prod_{j \in [R+T] \setminus \{ r \}} x - \beta_j \cdot \prod_{j \in [R+T] \setminus \{ r \}} \beta_r - \beta_j
\]

\[
+ \sum_{k=R+1}^{R+T} \mathbf{Z}_{k}^{\mathbf{B}} \cdot \prod_{j \in [R+T] \setminus \{ k \}} x - \beta_j \cdot \prod_{j \in [R+T] \setminus \{ k \}} \beta_k - \beta_j,
\]

where

\[
\mathbf{Z}_{k}^{\mathbf{B}} = \sum_{v=1}^{V} \sum_{r=1}^{R} \mathbf{B}_{r}^{(v)} \cdot z_r^{(v)}.
\]

Note from (38) that, the function \( h_\mathbf{B}(x) \) has the identical structure as \( h_L(x) \) in (16), which aligns the batch of desired sub-matrices \( \{ \mathbf{B}_{r}^{(0)} \}_{r \in [R]} \) along the \( R \) dimensions corresponding to \( \{ \prod_{j \in [R+T] \setminus \{ r \}} x - \beta_j \cdot \prod_{j \in [R+T] \setminus \{ r \}} \beta_r - \beta_j \} \) and the interference from the matrices \( \mathbf{B}^{(v)} \) is aligned along the \( T \) dimensions corresponding to \( \{ \prod_{j \in [R+T] \setminus \{ k \}} x - \beta_j \cdot \prod_{j \in [R+T] \setminus \{ k \}} \beta_k - \beta_j \} \).

Then worker \( i \) computes \( \mathbf{Y}^{(i)} = \mathbf{A}_i \mathbf{B}_i \), which is the evaluation of the product polynomial \( g(x) = f_\mathbf{A}(x) \cdot h_\mathbf{B}(x) \) at point \( x = a_i \). Thus, the master can interpolate the polynomial \( g(x) \) from the responses of any \( K = \deg(g(x)) + 1 = 2R + 2T - 1 \) workers, and then evaluates \( g(x) \) at points \( x = \beta_1, \ldots, \beta_R \) to obtain the desired element-wise product \( \{ \mathbf{A}_r \mathbf{B}_{r}^{(0)} : r \in [R] \} \).

C. Security, Privacy, Communication Cost and Computation Complexity for PSMM

In this subsection, we prove the security and privacy of the two PSMM strategies above, analyse their communication cost and computation complexities, and compare their performance.

1) Security and Privacy: In the proposed PSMM strategy based on polynomial codes, the encoding matrices \( \mathbf{A}_i \) (24) and the queries \( q_{r,\ell}(x) \) (25) sent to workers are generated by evaluating the encoding polynomial \( f_\mathbf{A}(x) \) of matrix \( \mathbf{A} \) and the query polynomials \( \{ q_{r,\ell}(x) \}_{\ell \in [p], j \in [n], v \in [V]} \) at distinct points, respectively. Here the encoding polynomial \( f_\mathbf{A}(x) \) (22) (resp. each of query polynomial \( q_{r,\ell}(x) \) (23)) is constructed by employing \( T \) independent and uniform random noises to mask the confidential matrix \( \mathbf{A} \) (resp. the interested index \( \theta \)), which ensures that the data sent to any \( T \) workers are secure (resp. private). The security and privacy of the PSMM strategy based on Lagrange codes follows from similar argument. Their formal proofs are presented in Appendix-A.

2) Communication Cost: In the two PSMM strategies, the master sends an encoding sub-matrix with the same dimension of \( \frac{\lambda}{m} \times \frac{\omega}{p} \) to each worker by (24) and (35), and downloads a matrix with the same dimension of \( \frac{\lambda}{m} \times \frac{\gamma}{n} \) from each of responsive workers. Thus, by (3), the two strategies achieve the same upload cost \( P_u = \frac{N \times \frac{\lambda}{m} \times \frac{\omega}{p}}{\lambda \times \sigma} = \frac{N}{mp} \) and the download cost

\[
P_d = \frac{K \times \frac{\lambda}{m} \times \frac{\omega}{p}}{\lambda \times \gamma} = \frac{K}{mn},
\]

where \( K = \max\{a_{k,\ell,c_i} : k \in [m], \ell \in [p], t \in [T] \} + \max\{b_{l,j,d_i} : \ell \in [p], j \in [n], t \in [T] \} + 1 \) and \( K = 2R + 2T - 1 \) for the PSMM strategies based on polynomial codes and Lagrange codes, respectively.

3) Computation Complexity: In the PSMM strategy based on polynomial codes, the encoding process for matrix \( \mathbf{A} \) can be viewed as evaluating a polynomial of degree less than \( N \) at \( N \) points for \( \frac{\lambda \omega}{mp} \) times by (22) and (24), and decoding requires interpolating a \( (K - 1) \)th degree polynomial for \( \frac{\gamma \omega}{mn} \) times, where \( K = \max\{a_{k,\ell,c_i} : k \in [m], \ell \in [p], t \in [T] \} + \max\{b_{l,j,d_i} : \ell \in [p], j \in [n], t \in [T] \} + 1 \). It is well known [51] that the evaluation of a \( K \)th degree polynomial at \( k + 1 \) arbitrary points can be done in \( O(k \log k)^2 \log \log k \) arithmetic operations, and consequently, its dual problem, interpolation of a \( (K - 1) \)th degree polynomial from \( k + 1 \) arbitrary points can be performed in the same arithmetic operations \( O(k \log k)^2 \log \log k \). Thus, encoding and decoding achieve the complexities \( O\left(\frac{\lambda \omega N (\log N)^2 \log \log N}{mp} \right) \) and \( O\left(\frac{\gamma \omega N (\log N)^2 \log \log N}{mn} \right) \), respectively. The complexity at each worker includes generating a linear combination of \( Vpn \) sub-matrices with dimension \( \frac{\omega \gamma}{p} \times \frac{\omega \gamma}{n} \), and multiplying two coded sub-matrices with sizes \( \frac{\lambda}{m} \times \frac{\omega}{p} \) and \( \frac{\omega}{p} \times \frac{\gamma}{n} \), which requires a complexity of \( O(V \omega \gamma + \frac{\lambda \omega \gamma}{mp}) \) at most.

In the PSMM strategy based on Lagrange codes, the encoding process for matrix \( \mathbf{A} \) includes generating a bath of sub-matrices by taking a linear combination of \( mp \) partitioning sub-matrices with dimension \( \frac{\lambda \omega}{mp} \) for \( R \) times (31), and encoding the batch of sub-matrices by evaluating a polynomial of degree \( R + T - 1 < N \) at \( N \) points for \( \frac{\lambda \omega}{mp} \) times by (33) and (35), which achieves the complexity

\[
O(R \lambda \omega + \frac{\lambda \omega N (\log N)^2 \log \log N}{mp}).
\]

The complexity at each worker consists of generating another bath of sub-matrices by taking a linear combination of \( pn \) partitioning sub-matrices with dimension \( \frac{\omega \gamma}{p} \times \frac{\omega \gamma}{n} \) for \( \frac{\gamma \omega}{mn} \) times (32), computing a linear combination of \( VR \) sub-matrices with dimension \( \frac{\omega \gamma}{p} \times \frac{\omega \gamma}{n} \) (37), and multiplying two coded sub-matrices with sizes \( \frac{\lambda}{m} \times \frac{\omega}{p} \) and \( \frac{\omega}{p} \times \frac{\gamma}{n} \), which requires a complexity of \( O(VR \omega \gamma + \frac{\lambda \omega \gamma}{mp}) \). Decoding requires interpolating of a \( (K - 1) \)th degree polynomial for \( \frac{\gamma \omega}{mn} \) times, then evaluating the polynomial at \( R < K \) points for \( \frac{\lambda \omega}{mn} \) times, and finally computing the linear combinations of \( R \) sub-matrices with dimension \( \frac{\gamma \omega}{mn} \) for \( mn \) times (10), which achieves the complexity

\[
O\left(\frac{\gamma \omega K (\log K)^2 \log \log K}{mn} + R \lambda \omega \right)
\]

\( K = 2R + 2T - 1 \).

4) Performance Comparison Between the Proposed PSMM Strategies: We summarize the performance of the two proposed PSMM strategies in Table II. Recall from Remark 2 that which one of the two PSMM strategies achieves a smaller recovery threshold depends on the value of the security parameter \( T \). We observe from Table II that, regardless of the value of \( T \), the PSMM strategy based on polynomial codes outperforms the strategy based on Lagrange codes in terms of encoding complexity and worker computation complexity, with the upload cost being identical. When \( K_F \leq K_L \) for some values of the parameter \( T \), the PSMM strategy based on polynomial codes achieves a smaller recovery threshold, download cost, and decoding complexity, and otherwise (i.e., when \( K_F > K_L \)) the one based on Lagrange codes achieves a smaller recovery
threshold and download cost. In the former case of $K_P \leq K_L$, the strategy based on polynomial codes is preferable to facilitate a faster execution of PSMM. However, in the latter case of $K_P > K_L$, one should choose which strategy to implement according to the system resources including the computation and communication capabilities of the master and the workers.

VI. COMPUTATION STRATEGIES FOR FULLY PRIVATE MATRIX MULTIPLICATION

In this section, we present the FPMM strategies based on polynomial codes and Lagrange codes for proving Theorems 3 and 4, respectively, and then analyse their privacy, communication cost and computation complexities.

As illustrated in Fig. 2, the goal of the master in FPMM is to compute the product $A^{(u)}B^{(v)}$ for any $\theta_1 \in [U]$ and $\theta_2 \in [V]$, while keeping the indices of the desired product $\theta_1$ and $\theta_2$ private from any $T$ colluding workers. Consider arbitrary partitioning parameters $m, p, n$, the matrices $A^{(u)}$ and $B^{(v)}$ are divided into $m \times p$ and $p \times n$ equal-size sub-matrices, respectively, i.e., for all $u \in [U], v \in [V]$,

\[
A^{(u)} = \begin{bmatrix}
A_{1,1}^{(u)} & \cdots & A_{1,p}^{(u)} \\
\vdots & \ddots & \vdots \\
A_{m,1}^{(u)} & \cdots & A_{m,p}^{(u)}
\end{bmatrix}, \quad B^{(v)} = \begin{bmatrix}
B_{1,1}^{(v)} & \cdots & B_{1,n}^{(v)} \\
\vdots & \ddots & \vdots \\
B_{p,1}^{(v)} & \cdots & B_{p,n}^{(v)}
\end{bmatrix},
\]

(39)

where $A_{k,\ell}^{(u)} \in \mathbb{F}_q^{\frac{m}{p} \times \frac{n}{p}}$ for any $k \in [m], \ell \in [p]$, and $B_{\ell,j}^{(v)} \in \mathbb{F}_q^{\frac{m}{p} \times \frac{n}{p}}$ for any $\ell \in [p], j \in [n]$. Thus, the desired product $C^{(\theta_1,\theta_2)} = A^{(\theta_1)}B^{(\theta_2)}$ is given by

\[
C^{(\theta_1,\theta_2)} = A^{(\theta_1)}B^{(\theta_2)} = \begin{bmatrix}
C_{1,1}^{(\theta_1,\theta_2)} & \cdots & C_{1,n}^{(\theta_1,\theta_2)} \\
\vdots & \ddots & \vdots \\
C_{m,1}^{(\theta_1,\theta_2)} & \cdots & C_{m,n}^{(\theta_1,\theta_2)}
\end{bmatrix}
\]

with $C_{k,\ell}^{(\theta_1,\theta_2)} = \sum_{\ell,j} A_{k,\ell}^{(\theta_1)}B_{\ell,j}^{(\theta_2)}$ for any $k \in [m], j \in [n]$.

A. FPMM Strategy Based on Polynomial Codes

We start with proving Theorem 3. Similar to PSMM strategy based on polynomial codes, we show that any SMM strategy based on polynomial codes can be exploited to construct the FPMM strategy with same recovery threshold.

Let the positive integers $\{k, l, b_k, c_l, d_l : k \in [m], \ell \in [p], j \in [n], t \in [T]\}$ satisfying C1-C2 be the parameters of the

### TABLE II

| Recovery Threshold          | PSMM based on Polynomial codes | PSMM based on Lagrange codes |
|-----------------------------|-------------------------------|-----------------------------|
| $K_P = \max(a_k, b_k, c_l, d_t : k \in [m], \ell \in [p], t \in [T])$ + $\max(b_k, d_t : \ell \in [p], j \in [n], t \in [T]) + 1$ | $K_L = 2R + 2T - 1$ |
| Upload and Download ($P_u, P_d$) | $(\frac{N}{mp} \times \frac{K_L}{EP})$ | $(\frac{N}{mp} \times \frac{K_L}{LP})$ |
| Encoding Complexity $C_A$ | $O(\lambda_P N \log N \log N)$ | $O(R \omega_P + \lambda_P N \log N \log N)$ |
| Worker Computation $C_w$ | $O(V \omega_P + \lambda_P N \log N)$ | $O(V R \omega_P + \lambda_P N \log N)$ |
| Decoding Complexity $C_d$ | $O(\lambda_P K_L \log K_L \log \log K_L)$ | $O(\lambda_P K_L \log K_L \log \log K_L + R \gamma)$ |

Here, $K_P$ can take the value of $\min((m+1)(np + T) - 1, (n+1)(mp + T) - 1, 2mpn + 2T - 1)$ by Corollary 1, and $R$ denotes any upper construction of bilinear complexity.
\[
\hat{B}_i = \sum_{v=1}^{V} \sum_{\ell=1}^{m} \sum_{j=1}^{p} B_{i,j}^{(v)} \cdot q_{i,j}^{(v)}(\alpha_i).
\]  

By (40)–(43), we have
\[
\sum_{a_i=1}^{m} \sum_{b_i=1}^{p} A_{a_i,b_i}(x) = \sum_{v=1}^{V} \sum_{\ell=1}^{m} \sum_{j=1}^{p} B_{i,j}^{(v)} \cdot q_{i,j}^{(v)}(x) = h_B(x),
\]

where we set
\[
Z_A^A = \sum_{a_i=1}^{m} \sum_{b_i=1}^{p} A_{a_i,b_i}(x) \quad \forall t \in [T],
\]
\[
Z_B^B = \sum_{v=1}^{V} \sum_{\ell=1}^{m} \sum_{j=1}^{p} B_{i,j}^{(v)} \cdot q_{i,j}^{(v)}(x) \quad \forall t \in [T],
\]

which are independent of workers and thus can be viewed as constant terms.

Then each worker \( i \in [N] \) computes the product \( Y_{i}^{(1,2)} = \hat{A}_i B_i \) as a response for the master, which is equivalent to evaluating the product polynomial \( g(x) = f_A(x) \cdot h_B(x) \) at point \( x = \alpha_i \) by (46)–(49). Thus, the master can interpolate the product \( g(x) \) from the responses of any \( K = \max[a_i, c_i, t_k : k \in [m], \ell \in [p], t \in [T]] + \max[b_{i,j}, d_{i,j} : \ell \in [p], j \in [n], t \in [T]] + 1 \) workers and then recovers the desired computation \( C^{(1,2)} = A^{(1)} B^{(1)} \) by interpolating the product polynomial \( g(x) = f_A(x) \cdot h_B(x) \).

To keep the indices \( \theta_1 \) and \( \theta_2 \) private, the master constructs the query polynomials \( \rho_r^{(u)}(x) \) and \( q_r^{(v)}(x) \) by exploiting the structure of the encoding functions in (52) and (53), respectively, for all \( r \in [R], u \in [U], v \in [V] \), as follows.

\[
\rho_r^{(u)}(x) = \sum_{k=R+1}^{R+T} \sum_{j \in [R+T]\{r\}} x - \beta_j \prod_{b \in [R+T]\{r\}} \frac{x - \beta_b}{\beta_b - \beta_j} + \sum_{k=R+1}^{R+T} Z_k^A \cdot \prod_{j \in [R+T]\{r\}} \frac{x - \beta_j}{\beta_k - \beta_j}
\]

where \( Z_k^A \) and \( Z_k^B \) are arbitrary matrices over \( \mathbb{F}_q \) with the same dimensions as \( A^{(1)} \) and \( B^{(1)} \) respectively, and will be explained later. We know from Section III-B that, one can recover the desired computation \( C^{(1,2)} = A^{(1)} B^{(1)} \) by interpolating the product polynomial \( g(x) = f_A(x) \cdot h_B(x) \).

B. FPMM Strategy Based on Lagrange Codes

We now present the FPMM strategy based on Lagrange codes for proving Theorem 4. Let \( a = (a_{r,k,\ell}), b = (b_{r,k,\ell}), c = (c_{r,k,\ell}) \) be an upper bound construction with rank \( R \) for bilinear complexity. Each worker first converts the matrices \( A^{(u)} \) and \( B^{(v)} \) (39) into a batch of sub-matrices of length \( R \), respectively:

\[
A_r^{(u)} = \sum_{k=1}^{m} a_{r,k,\ell} A_{k,\ell}^{(u)} \quad \forall r \in [R], u \in [U],
\]

\[
B_r^{(v)} = \sum_{j=1}^{p} b_{r,j} B_{j}^{(v)} \quad \forall r \in [R], v \in [V].
\]

Let \( \{b_{r,j}, a_{r,k,\ell} : r \in [R + T], i \in [N] \} \) be \( R + T + N \) distinct elements from \( \mathbb{F}_q \). Denote the polynomial functions of \( A^{(1)} \) and \( B^{(2)} \) by

\[
f_A(x) = \sum_{r=1}^{R} A_r^{(1)} \cdot \prod_{j \in [R+T]\{r\}} \frac{x - \beta_j}{\beta_r - \beta_j} + \sum_{k=R+1}^{R+T} Z_k^A \cdot \prod_{j \in [R+T]\{r\}} \frac{x - \beta_j}{\beta_k - \beta_j}.
\]

By (52)–(55), we have

\[
u \sum_{u=1}^{U} \sum_{r=1}^{R} A_r^{(u)} \cdot \rho_r^{(u)}(x) = \sum_{r=1}^{R+T} A_r^{(1)} \cdot \prod_{j \in [R+T]\{r\}} \frac{x - \beta_j}{\beta_r - \beta_j} + \sum_{k=R+1}^{R+T} Z_k^A \cdot \prod_{j \in [R+T]\{r\}} \frac{x - \beta_j}{\beta_k - \beta_j}.
\]
TABLE III

| Recovery Threshold | PPMM based on Polynomial codes | PPMM based on Lagrange codes |
|--------------------|-------------------------------|------------------------------|
| $K_P = \max\{a_{k,t}, c_{i} : k \in [m], t \in [P], t \in [T]\} + \max\{b_{t,j}, d_{t} : t \in [P], j \in [n], t \in [T]\} + 1 | K_L = 2R + 2T - 1 |

where we set

$$
Z_k^A = \sum_{u=1}^U \sum_{r=1}^R A_{t,k}^{(u)} \cdot z_{r,k}^{(u)}, \quad \forall k \in [R + 1 : R + T],
$$

$$
Z_k^B = \sum_{u=1}^U \sum_{r=1}^R B_{t,k}^{(u)} \cdot z_{r,k}^{(u)}, \quad \forall k \in [R + 1 : R + T],
$$

which are constant terms independently of workers. Then, worker $i$ computes $Y_i^{(\alpha, \beta)} = A_iB_i$, which is the evaluate of the product polynomial $g(x) = f_A(x) \cdot h_B(x)$ at point $x = \alpha$. Thus, the master can interpolate $g(x)$ from the responses of any $K = 2R + 2T - 1$ workers, and then obtains the desired product $C^{(\alpha, \beta)} = A^{(\alpha)}B^{(\beta)}$.

C. Privacy, Communication Cost and Computation Complexity for PPMM

For the two FPMM strategies above, their privacy, communication cost, computation complexities and comparisons follow from the similar discussion to the PSMM strategies in Section V-C. We briefly outline as follows.

1) Privacy: In the two PPMM strategies, the queries $Q_i^{(\alpha)}$ and $Q_i^{(\beta)}$ sent to workers are generated by evaluating the query polynomials $\{p^{(\alpha)}(v)\}_{v \in [\mathbb{V}]}$ and $\{q^{(\beta)}(v)\}_{v \in [\mathbb{V}]}$ at distinct points, where each of query polynomials is constructed by employing $T$ independent and uniform random noises to mask interested index, which ensures the privacy of the queries sent to any $T$ workers. Their formal proofs are given in Appendix-B.

2) Communication Cost: In the two strategies, the master downloads a matrix with the same dimension of $\frac{K}{m} \times \frac{K}{n}$ from each of responsive workers. Thus, the two strategies achieve the download cost $P_d = \frac{K \times \frac{K}{m} \times \frac{K}{n}}{m} = \frac{K^2}{mn}$, where $K = \max\{a_{k,t}, c_{i} : k \in [m], t \in [P], t \in [T]\} + \max\{b_{t,j}, d_{t} : t \in [P], j \in [n], t \in [T]\} + 1$ and $K = 2R + 2T - 1$ for the PPMM strategies based on polynomial codes and Lagrange codes, respectively.

3) Computation Complexity: In the PPMM strategy based on polynomial codes, the complexity at each worker includes encoding the matrices $A^{(U)}$ by taking a linear combination of $Ump$ sub-matrices with dimension $\frac{K^2}{mn}$ (46), encoding the matrices $B^{(V)}$ by taking a linear combination of $Vpn$ sub-matrices with dimension $\frac{K^2}{np}$ (47), and multiplying the two coded sub-matrices with sizes $\frac{K^2}{m} \times \frac{K^2}{n}$ and $\frac{K^2}{n} \times \frac{K^2}{n}$, which requires a complexity of $O(U\omega + V\omega + \frac{K^2}{mn})$ at most. Decoding requires interpolating a $(K - 1)$-th degree polynomial for $\frac{K^2}{mn}$ times, which achieves the complexity $O(\frac{K^2}{mn} \log K^2 \log \log K^2)$, where $K = \max\{a_{k,t}, c_{i} : k \in [m], t \in [P], t \in [T]\} + \max\{b_{t,j}, d_{t} : t \in [P], j \in [n], t \in [T]\} + 1$.

In the PPMM strategy based on Lagrange codes, the complexity at each worker consists of generating the two batches of sub-matrices $(A_{t,k}^{(\alpha)}), u \in [U]$ and $(B_{t,k}^{(\beta)}), v \in [V]$ by (50) and (51), encoding the two batches of sub-matrices into $A_i$ and $B_i$, by (58)-(59), multiplying the two coded sub-matrices $A_i$ and $B_i$, which achieves the complexity $O(UR\omega + V\omega + \frac{K^2}{mn})$. Decoding is identical to the PSMM strategy based on Lagrange codes, and achieves the complexity $O(\frac{K^2}{mn} \log K^2 \log \log K^2 + R\lambda)$, where $K = 2R + 2T - 1$.

4) Performance Comparison Between the Proposed PPMM Strategies: The performance of the two PPMM strategies are summarized in Table III. Following a discussion similar to Section V-C, it is straightforward to obtain from Table III that, the performance of the PPMM strategy based on polynomial codes strictly outperforms the one based on Lagrange codes when $K_P \leq K_L$ for some values of the parameter $T$, and otherwise the one based on Lagrange codes achieves a smaller recovery threshold and download cost, but still with a higher computation complexity at each worker.

VII. COMPARISON WITH RELATED WORKS

The most valuable aspect of this paper is that we propose a novel systematic approach to construct efficient computation strategies for private matrix multiplication problems. The key idea is to start with an SMM strategy (polynomial codes-based or Lagrange codes-based), and then carefully design queries at the master such that 1) the interested matrix indices are completely hidden from any $T$ colluding workers, and 2) the response computed from the query and the local data at each worker resembles the response computed in the SMM strategy. Strategies constructed following this approach directly inherit the correctness of matrix multiplication from the underlying SMM strategy, and the original problem is essentially reduced to the problem of designing private queries that are compatible with the chosen SMM strategy, which substantially simplifies the design process for private matrix multiplication strategies. To clearly see the innovations of this approach in
References [7], [8], [44] exploit Polynomial codes (see, e.g., [18], [20]) to solve the PSMM problem without colluding workers (i.e., $T = 1$). While both the prior works [7], [8], [44] and our first proposed PSMM strategy employ Polynomial codes to encode the confidential matrix $A$, the main difference lies in how the queries are designed, which accordingly leads to different worker responses and decoding operation. In [7], [8], [44], the queries are designed such that 1) all elements in a query sent to any individual worker have identical distribution, so no information about the interested index elements in a query sent to any individual worker have identical codes to encode the confidential matrix $A$, the main difference lies in how the queries are designed, which accordingly leads to different worker responses and decoding operation. In [7], [8], [44], the queries sent to worker $i$ is constructed as $Q_i^{(v)} = \{a_1, \ldots, a_{q-1}, s, a_{q+1}, \ldots, a_{v}\}$ for any $i \in [N]$, where $\{a_1, \ldots, a_{q-1}, s, a_{q+1}, \ldots, a_{v}\}$ and $\{a_1, \ldots, a_{q-1}, s, a_{q+1}, \ldots, a_{v}\}$ are selected uniformly i.i.d. from $\mathbb{F}_q$. It is not clear how one can generalize this query design to the colluding case considered in this paper, while maintaining the privacy requirement. In our proposed strategy, inspired by the specific structure of the polynomial codes utilized to encode the confidential matrix $B$ in the SMM problem, we design polynomially coded queries (see, e.g., (23)) that facilitate a form of interference alignment, separating the desired and interfering partitioning sub-matrices in the library $\mathcal{L}_B^k$, such that the local matrix $\tilde{B}$ obtained from the library and the received query has identical structure as the encoding of the matrix $B$ in SMM, for each worker $i$, as shown in (28). Consequently, the proposed PSMM strategies achieve the same recovery thresholds as the SMM strategies.

For the case of non-colluding workers with $T = 1$, the best known recovery threshold achieved by PSMM strategies based on polynomial codes is $mn+mp+n$ [44]. The recovery threshold and download cost of our PSMM strategy based on polynomial codes is superior to that when $m > n$ and $p > 1$ by Corollary 1, with the other performance of upload cost and computation complexity being identical.

The problem of PSMM with $T$ colluding workers was first studied in [47], for the extremely special case of $p = n = 1$, i.e., only matrix $A$ is horizontally divided into $m$ equal-size sub-matrices as $A_1, A_2, \ldots, A_m$. To complete desired computation, the PSMM strategy based on polynomial codes in [47] shares $V$ encoding versions $A_1^{(1)}, A_2^{(2)}, \ldots, A_m^{(V)}$ of matrix $A$ to each worker $i$ in a security-preserving manner, which are also used as a query to instruct the worker to compute the response $Y_i^{(v)} = A_i^{(v)}B^{(v)}$ for the master, where $A_i^{(v)}$ for any $v \in [V]$ is given by

$$A_i^{(v)} = \sum_{l=1}^{T} Z_i^{(v)}a_{T}^{l-1} + \sum_{k=1}^{m} A_k a_{l}^{T+k-1} \quad \text{if } v = \theta$$

$$A_i^{(v)} = 0 \quad \text{if } v \neq \theta.$$

Here $a_1, \ldots, a_N$ are distinct elements on $\mathbb{F}_q$ and $(Z_i^{(v)})_{i \in [T], v \in [V]}$ are random noise matrices. However, this approach requires uploading $V$ encoding versions of $A$ and pairwise multiplying these encoding sub-matrices with all public matrices in the library $\mathcal{L}^k$ for each worker, implying a significantly communication and computation overheads. In our PSMM strategy based on polynomial code, the master shares only one secure encoding version $A, (24)$ of $A$, and sends a private query to encode the library $\mathcal{L}_B^k$ into one encoding sub-matrix $\tilde{B}, (26)$. The response of each worker is completed by computing the product of the two encoding sub-matrices $Y_i^{(v)} = \tilde{A}_i \tilde{B}, (30)$. For this special case of $p = n = 1$, compared with the PSMM strategy in [47], our proposed strategy reduces upload cost, encoding complexity, and worker computation complexity by a factor of $O(V)$, at the expense of increased recovery threshold by a factor of 2, where $V$ is the number of matrices in the library $\mathcal{L}_B^k$ and is typically large in current big data era (see detailed comparison in Table IV).

The FPMM problem with $T$-colluding workers was previously investigated in [48] for the special case of $p = 1$, and a strategy with recovery threshold $(m+1)(n+T)+T-1$ is constructed based on the idea of Cross Subspace Alignment (CSA) introduced in [43]. In the strategy proposed in [48], the queries sent to worker $i$ are constructed as $Q_i^{(v)} = \{q_{i,1}^{(v)}(\alpha_i) : k \in [m], u \in [\ell]\}$ and $Q_i^{(\ell)} = \{q_j^{(\ell)}(\alpha_i) : j \in [n], v \in [\ell]\}$, where $\rho_k^{(\alpha)}(\alpha_i)$ and $q_j^{(\ell)}(\alpha_i)$ are given by

$$\rho_k^{(\alpha)}(\alpha_i) = \sum_{l=1}^{T} z_{k,l}^{(u)}a_{l}^{T-1} + \left\{ \begin{array}{ll} a_{l}^{(n+T)k}, & \text{if } u = \theta_1 \\ 0, & \text{if } u \neq \theta_1 \end{array} \right.$$  

$$\phi_j^{(\ell)}(\alpha_i) = \sum_{l=1}^{T} z_{j,l}^{(u)}a_{l}^{T-1} + \left\{ \begin{array}{ll} a_{l}^{(n+T)\ell}, & \text{if } v = \theta_2 \\ 0, & \text{if } v \neq \theta_2 \end{array} \right.$$

for some evaluation point $a_i$ and random noises $z_{k,l}^{(u)}, z_{j,l}^{(u)}$. In our FPMM strategy based on polynomial codes, similar to PSMM, with the goal of resembling the structure of encoding functions for the matrices $A$ and $B$ in SMM strategies, our FPMM strategies design corresponding polynomially coded queries (see, e.g., (42) and (43)) for the libraries $\mathcal{L}^A$ and $\mathcal{L}^B$, respectively. For this special case of $p = 1$, our strategy based on polynomial codes strictly outperforms that in [48] in terms of recovery threshold and download cost by Corollary 1, while the performance with respect to other measures are identical.

For this special cases of $T = 1$, our proposed strategy based on Lagrange codes achieves identical system

| TABLE IV | PERFORMANCE COMPARISON FOR PSMM STRATEGIES BASED ON POLYNOMIAL CODES, FOR THE CASE OF $p = n = 1$, AND $T > 1$ COLLUDING WORKERS |
|-----------------|-----------------|-----------------|-----------------|
| **Previous PSMM Strategy** [47] | **Our PSMM Strategy** |
| Recovery Threshold | $K = m + 1 + T$ | $K = m + 2T - 1$ |
| Upload Cost | $\frac{1}{2}$ | $\frac{1}{2}$ |
| Download Cost | $\frac{1}{2}$ | $\frac{1}{2}$ |
| Encoding Complexity | $O(V \log m \log \log n)$ | $O(V \log m \log \log n)$ |
| Worker Computation | $O(V \log m \log \log n)$ | $O(V \log m \log \log n)$ |
| Decoding Complexity | $O(V \log m \log \log n)$ | $O(V \log m \log \log n)$ |
performance as that in [16], for both problems of PSMM and FPMM. The query design in our proposed strategy differs significantly from the design in [16]. While the query design in [16] follows the ideas in [7], [8] for the PSMM problem, and is difficult to generalize to the case of colluding workers, following our systematic approach, we design Lagrange coded queries of the private indices \( \theta_1 \) and \( \theta_2 \), such that the computed response at each worker exhibits identical structure of the response computed using the SMM strategy based on Lagrange codes.

VIII. CONCLUSION

In this paper, we focused on designing efficient PSMM and FPMM strategies that minimize the recovery threshold, communication cost and complexity. We showed that, for current SMM strategies based on polynomial codes and Lagrange codes can be used to construct PSMM/FPMM strategies with same recovery threshold, by exploiting the structure inspired by the encoding functions of the SMM strategies to create private queries. This establishes a generic connection between PSMM/FPMM and SMM, and provides a novel systematic approach towards designing PSMM and FPMM strategies. The resulting strategies constructed from this approach improve one or more efficiency metrics including recovery threshold, communication cost and computation complexity, compared with the state of the art, achieving a more flexible tradeoff in optimizing system efficiency.

APPENDIX

In this Appendix, we prove the security and/or privacy of the proposed PSMM/FPMM strategies based on polynomial codes and Lagrange codes.

A. Proof of Security and Privacy for PSMM

We start with proving the security and privacy of the proposed PSMM strategy based on polynomial codes.

Security for PSMM Based on Polynomial Codes: Let \( \mathcal{T} = \{i_1, \ldots, i_T\} \subseteq [N] \) be any \( T \) indices of the \( N \) workers. Then,\[
\begin{align*}
&I(\mathbf{A}; \tilde{\mathcal{T}}, \tilde{\mathbf{B}}^{(VI)}, \tilde{\mathbf{Y}}^{(VI)}|\mathcal{T}) = I(\mathbf{A}; \tilde{\mathcal{T}}) + I(\mathbf{A}; \tilde{\mathbf{B}}^{(VI)}|\tilde{\mathcal{T}}) + I(\mathbf{A}; \tilde{\mathbf{Y}}^{(VI)}|\tilde{\mathcal{T}}, \tilde{\mathbf{B}}^{(VI)}) \\
&= I(\mathbf{A}; \tilde{\mathcal{T}}) + I(\mathbf{A}; \tilde{\mathbf{B}}^{(VI)}|\tilde{\mathcal{T}}) + I(\mathbf{A}; \tilde{\mathbf{Y}}^{(VI)}|\tilde{\mathcal{T}}, \tilde{\mathbf{B}}^{(VI)}) \\
&\overset{(a)}{=} I(\mathbf{A}; \tilde{\mathcal{T}}) + I(\mathbf{A}; \tilde{\mathbf{B}}^{(VI)}|\tilde{\mathcal{T}}) + I(\mathbf{A}; \tilde{\mathbf{Y}}^{(VI)}|\tilde{\mathcal{T}}, \tilde{\mathbf{B}}^{(VI)}) \\
&\overset{(b)}{=} I(\mathbf{A}; \tilde{\mathcal{T}}) + I(\mathbf{A}; \tilde{\mathbf{B}}^{(VI)}|\tilde{\mathcal{T}}) + I(\mathbf{A}; \tilde{\mathbf{Y}}^{(VI)}|\tilde{\mathcal{T}}, \tilde{\mathbf{B}}^{(VI)}) \\
&\overset{(c)}{=} 0,
\end{align*}
\]
where (a) is because the queries \( \tilde{\mathbf{Q}}^{(VI)} \) and the data matrices \( \tilde{\mathbf{B}}^{(VI)} \) are generated independently of the encoded matrix \( \tilde{\mathbf{A}}^{(VI)} \) and the matrix \( \mathbf{A} \) by (22)–(25), and the responses \( \tilde{\mathbf{Y}}^{(VI)} \) (30) are the deterministic function of \( \tilde{\mathbf{A}}^{(VI)}, \tilde{\mathbf{Q}}^{(VI)} \) and \( \tilde{\mathbf{B}}^{(VI)} \) by (25) and (26), such that 0 = \( I(\tilde{\mathbf{A}}^{(VI)}, \tilde{\mathbf{Q}}^{(VI)}, \tilde{\mathbf{B}}^{(VI)}) \geq I(\tilde{\mathbf{A}}^{(VI)}; \tilde{\mathbf{Q}}^{(VI)}, \tilde{\mathbf{B}}^{(VI)}|\tilde{\mathcal{T}}) \geq 0 \) and \( I(\tilde{\mathbf{A}}^{(VI)}; \tilde{\mathbf{Q}}^{(VI)}, \tilde{\mathbf{B}}^{(VI)}|\tilde{\mathcal{T}}) = 0 \); (b) follows by (17) and (24); (c) follows from (22), (9) and Lemma 1.

The security of our PSMM strategy based on polynomial codes follows from (2).

Privacy for PSMM Based on Polynomial Codes: By (23) and (25), the query elements \( q_{\ell,j}^{(v)}(\alpha_{i_1}), \ldots, q_{\ell,j}^{(v)}(\alpha_{i_T}) \) sent to the workers \( \mathcal{T} \) are protected by \( T \) random noises \( z_{\ell,j,1}^{(v)}, \ldots, z_{\ell,j,T}^{(v)} \) chosen independently and uniformly from \( \mathbb{F}_{q^v} \) for any \( \ell \in [p], j \in [n] \) and \( v \in [V] \), as shown below.

\[
\begin{bmatrix}
q_{\ell,j}^{(v)}(\alpha_{i_1}) \\
q_{\ell,j}^{(v)}(\alpha_{i_2}) \\
\vdots \\
q_{\ell,j}^{(v)}(\alpha_{i_T})
\end{bmatrix}
= h_{\ell,j}^{(v)}(\alpha_{i_T})
\begin{bmatrix}
h_{\ell,j}^{(v)}(\alpha_{i_1}) \\
h_{\ell,j}^{(v)}(\alpha_{i_2}) \\
\vdots \\
h_{\ell,j}^{(v)}(\alpha_{i_T})
\end{bmatrix}
+ \begin{bmatrix}
a_{\ell,1}^{(v)} & a_{\ell,2}^{(v)} & \cdots & a_{\ell,T}^{(v)} \\
a_{\ell,2}^{(v)} & a_{\ell,1}^{(v)} & \cdots & a_{\ell,T}^{(v)} \\
\vdots & \vdots & \ddots & \vdots \\
a_{\ell,T}^{(v)} & a_{\ell,T-1}^{(v)} & \cdots & a_{\ell,1}^{(v)}
\end{bmatrix}
\begin{bmatrix}
\alpha_{i_1} \\
\alpha_{i_2} \\
\vdots \\
\alpha_{i_T}
\end{bmatrix}
\Rightarrow F_{\ell,j}^{(v)}
\Rightarrow z_{\ell,j}^{(v)}
\]

where

\[
h_{\ell,j}^{(v)}(x) = \begin{cases}
x + j, & \text{if } v = \theta \\ 0, & \text{if } v \neq \theta. \end{cases}
\]

Recall from (9) that \( F_{\ell,j} \) is invertible, whose inverse matrix is denoted by \( (F_{\ell,j})^{-1} \). Then,

\[
\begin{align*}
&I(\theta; \tilde{\mathbf{Q}}^{(VI)}|\tilde{\mathcal{T}}) \overset{(a)}{=} I(\theta; q_{\ell,j}^{(v)}(\alpha_{i_1}) : i \in \mathcal{T}) \overset{(b)}{=} I(\theta; h_{\ell,j}^{(v)} + F_{\ell,j} \cdot z_{\ell,j}^{(v)}) \overset{(c)}{=} I(\theta; h_{\ell,j}^{(v)} + F_{\ell,j} \cdot z_{\ell,j}^{(v)}) \overset{(d)}{=} H\left(\left((F_{\ell,j})^{-1} \cdot h_{\ell,j}^{(v)} + z_{\ell,j}^{(v)})_{\ell \in [p], j \in [n], v \in [V]} \right| \theta\right) \overset{(e)}{=} 0,
\end{align*}
\]
where (a) follows by (25); (b) holds because \( (F_{\ell,j})^{-1} \cdot h_{\ell,j}^{(v)} \) are constant numbers by (61) when \( \theta \) is given, and \( z_{\ell,j}^{(v)} \) are generated independently of \( \theta \), and \( \ell \in [p], j \in [n], v \in [V] \), such that \( H\left((F_{\ell,j})^{-1} \cdot h_{\ell,j}^{(v)} + z_{\ell,j}^{(v)})_{\ell \in [p], j \in [n], v \in [V]} \right| \theta\right) = H\left((z_{\ell,j}^{(v)})_{\ell \in [p], j \in [n], v \in [V]} \right| \theta\right) \). (c) is due to the fact that all the noises in \( (z_{\ell,j}^{(v)})_{\ell \in [p], j \in [n], v \in [V]} \) are i.i.d. uniformly distributed over \( \mathbb{F}_{q^v} \), and are generated independently of \( \{(F_{\ell,j})^{-1} \cdot h_{\ell,j}^{(v)})_{\ell \in [p], j \in [n], v \in [V]} \right| \theta\right) \) and \( \{(z_{\ell,j}^{(v)})_{\ell \in [p], j \in [n], v \in [V]} \right| \theta\right) \) are identically and uniformly distributed over \( \mathbb{F}_{q^{Vp}} \), i.e., \( H\left((F_{\ell,j})^{-1} \cdot h_{\ell,j}^{(v)} + z_{\ell,j}^{(v)})_{\ell \in [p], j \in [n], v \in [V]} \right| \theta\right) = H\left((z_{\ell,j}^{(v)})_{\ell \in [p], j \in [n], v \in [V]} \right| \theta\right) = VPm.\]


Further, we have

\[ I(\theta; Q_T^{(v)}, \tilde{A}_T, B^{(V)}, Y_T^{(v)}) = I(\theta; Q_T^{(v)}) + I(\theta; \tilde{A}_T, B^{(V)}|Q_T^{(v)}) + I(\theta; Y_T^{(v)}|Q_T^{(v)}, \tilde{A}_T, B^{(V)}) \]

(69)

\[ \equiv (a) I(\theta; Q_T^{(v)}) \]

(70)

\[ \equiv (b) 0, \]

(71)

where \((a)\) is similar to (60) and \((b)\) follows by (68).

Thus, the privacy of our PSMM strategy based on polynomial codes follows by (1).

We next turn to prove the security and privacy of the PSMM strategy based on Lagrange codes. Before that, a useful lemma is provided.

**Lemma 3 (Generalized Cauchy Matrix [52]):** Let \(\alpha_1, \ldots, \alpha_T\) and \(\beta_1, \ldots, \beta_T\) be pairwise distinct elements from \(F_q\), and \(v_1, \ldots, v_T\) be \(N\) nonzero elements from \(F_q\). Denote by \(f_k(x)\) a polynomial of degree \(T - 1\)

\[ f_k(x) = \prod_{j=1}^{T-1} \frac{x - \beta_j}{\beta_k - \beta_j}, \quad \forall \ k \in [T]. \]

Then the following generalized Cauchy matrix is invertible over \(F_q\):

\[
\begin{bmatrix}
    v_1f_1(\alpha_1) & v_2f_2(\alpha_1) & \cdots & v_Tf_T(\alpha_1) \\
    v_1f_1(\alpha_2) & v_2f_2(\alpha_2) & \cdots & v_Tf_T(\alpha_2) \\
    \vdots & \vdots & \ddots & \vdots \\
    v_1f_1(\alpha_T) & v_2f_2(\alpha_T) & \cdots & v_Tf_T(\alpha_T)
\end{bmatrix}_{T \times T}
\]

Security for PSMM Based on Lagrange Codes: For any subset \(T = \{i_1, i_2, \ldots, i_T\} \subseteq [N]\) of size \(T\),

\[ I(A; \tilde{A}_T) \leq I\left(\{A_{k, \ell}\}_{k \in [m], \ell \in [p]}; f_{\mathcal{A}}(\alpha_i), i \in T\right) \leq 0, \]

(60)

where \((a)\) follows from (17) and (35); \((b)\) follows by (31), (33) and Lemmas 1 and 3.

Similar to (60), it is straightforward to prove

\[ I(A; Q_T^{(v)}, \tilde{A}_T, B^{(V)}|Q_T^{(v)}) = I(A; A_T) = 0. \]

Thus, the security of our PSMM strategy based on Lagrange codes follows from (2).

**Privacy for PSMM Based on Lagrange Codes:** By (34) and (36), the query elements \(q_r^{(v)}(\alpha_{i_1}), \ldots, q_r^{(v)}(\alpha_{i_T})\) sent to the workers \(T\) are protected by the \(T\) random noises \(\epsilon_{r,1}, \ldots, \epsilon_{r,T}\), for any \(r \in [R], v \in [V]\), as follows.

\[
\begin{bmatrix}
    q_r^{(v)}(\alpha_{i_1}) \\ 
    \vdots \\ 
    q_r^{(v)}(\alpha_{i_T})
\end{bmatrix}^T = \begin{bmatrix}
    h_r^{(v)}(\alpha_{i_1}) \\ 
    \vdots \\ 
    h_r^{(v)}(\alpha_{i_T})
\end{bmatrix} + \begin{bmatrix}
    \epsilon_{r,1}^{(v)}(\alpha_{i_1}) \\ 
    \vdots \\ 
    \epsilon_{r,T}^{(v)}(\alpha_{i_T})
\end{bmatrix}
\]

(72)

and

\[
q_v(x) = \prod_{j \in [R]\setminus[r]} \frac{x - \beta_j}{\beta_v - \beta_j}, \quad \forall k \in [T].
\]

(73)

By **Lemma 3** again, \(F_r^{(v)}\) is invertible for any subset \(T\). Thus, similar to (62)–(72), it is easy to prove

\[ I(\theta; Q_T^{(v)}, \tilde{A}_T, B^{(V)}, Y_T^{(v)}) = I(\theta; Q_T^{(v)}) = I(\theta; \{q_v^{(v)}(\alpha_i) : i \in T\}) \leq 0 \]

for the PSMM strategy based on Lagrange codes, i.e., our PSMM strategy based on Lagrange codes is private.

**B. Proof of Privacy for FPMM**

We next prove the privacy for our two FPMM strategies based on polynomial codes and Lagrange codes.

For any \(T = \{i_1, i_2, \ldots, i_T\} \subseteq [N]\) of size \(|T| = T\), the FPMM strategy based on polynomial codes satisfies

\[ I(\theta_1, \theta_2; Q_T^{(v)}, Q_T^{(v)}, A^{(U)}, B^{(V)}, Y_T^{(v),\theta_2}) = I(\theta_1, \theta_2; Q_T^{(v)}, Q_T^{(v)}, A^{(U)}, B^{(V)}) + I(\theta_1, \theta_2; Y_T^{(v),\theta_2}|Q_T^{(v)}, Q_T^{(v)}, A^{(U)}, B^{(V)}) \]

(61)

\[ \equiv I(\theta_1, \theta_2; Q_T^{(v)}, Q_T^{(v)}) \]

(62)

\[ \equiv (a) I(\theta_1, \theta_2; \{q_r^{(v)}(\alpha_i), q_v^{(v)}(\alpha_i) : i \in T\})_{k \in [m], \ell \in [p], j \in [n], u \in [U], v \in [V]} \]

(63)

\[ \equiv 0, \]

(64)

where \((a)\) follows by the similar argument to (60); \((b)\) is due to (44) and (45); \((c)\) holds because the query elements \(q_r^{(v)}(\alpha_i) : i \in T\) and \(q_v^{(v)}(\alpha_i) : i \in T\) sent to workers \(T\) are protected by random noises \(\epsilon_{r,1}^{(v)}(\alpha_i)\) and \(\epsilon_{r,1}^{(v)}(\alpha_i)\) by (42) and (43) respectively, for all \(k \in [m], \ell \in [p], j \in [n], u \in [U], v \in [V]\), thus \(I(\theta_1, \theta_2; \{q_r^{(v)}(\alpha_i), q_v^{(v)}(\alpha_i) : i \in T\})_{k \in [m], \ell \in [p], j \in [n], u \in [U], v \in [V]} = 0\) follows similar to (62)–(68).

Similarly, by (54)–(57) and **Lemma 3**, it is easy to prove that the FPMM strategy based on Lagrange codes satisfies

\[ I(\theta_1, \theta_2; Q_T^{(v)}, Q_T^{(v)}, A^{(U)}, B^{(V)}, Y_T^{(v),\theta_2}) = I(\theta_1, \theta_2; Q_T^{(v)}, Q_T^{(v)}) \]

(65)

\[ = I(\theta_1, \theta_2; \{q_r^{(v)}(\alpha_i), q_v^{(v)}(\alpha_i) : i \in T\})_{k \in [m], \ell \in [p], j \in [n], u \in [U], v \in [V]} = 0. \]

So the privacy of our two FPMM strategies follows by (4).

**References**

[1] J. Dean and L. A. Barroso, “The tail at scale,” Commun. ACM, vol. 56, no. 2, pp. 74–80, 2013.

[2] N. J. Yadawdikar, B. Hariharan, J. E. Gonzalez, and R. Katz, “Multi-task learning for straggler avoiding predictive job scheduling,” J. Mach. Learn. Res., vol. 17, no. 1, pp. 3692–3728, 2016.

[3] K.-H. Huang and J. A. Abraham, “Algorithm-based fault tolerance for matrix operations,” IEEE Trans. Comput., vol. C-33, no. 6, pp. 518–528, Jun. 1984.
