Learning-Augmented Perfectly Secure Collaborative Matrix Multiplication

Zixuan He*, Mohammad Reza Deylam Salehi, Derya Malak*, Photios A. Stavrou*
*Communication Systems Department, EURECOM, Sophia-Antipolis, France
{zixuan.he, reza.deylam-salehi, derya.malak, fotios.stavrou}@eurecom.fr

***Abstract*—This paper presents a perfectly secure matrix multiplication (PSMM) protocol for multiparty computation (MPC) of $\mathrm{A}^{\top}\mathrm{B}$ over finite fields. The proposed scheme guarantees correctness and information-theoretic privacy against threshold-bounded, semi-honest colluding agents, under explicit local storage constraints. Our scheme encodes submatrices as evaluations of sparse masking polynomials and combines coefficient alignment with Beaver-style randomness to ensure perfect secrecy. We demonstrate that any colluding set of parties below the security threshold observes uniformly random shares, and that the recovery threshold is optimal, matching existing information-theoretic limits. Building on this framework, we introduce a learning-augmented extension that integrates tensor-decomposition-based local block multiplication, capturing both classical and learned low-rank methods. We demonstrate that the proposed learning-based PSMM preserves privacy and recovery guarantees for MPC, while providing scalable computational efficiency gains (up to 80%) as the matrix dimensions grow.**

I. Introduction

Matrix multiplication (MM) is a fundamental primitive in modern networked and distributed systems that underlies state estimation and control (e.g., Kalman filtering [1]–[3], linear quadratic regulator [4]–[6]), large-scale optimization [7], [8], and learning pipelines [9]. In networked control systems (NCSs) [10]–[12], such computations increasingly involve multi-agent architectures [13] and shared computation infrastructures where multiple parties may contribute data, models, or computational resources while maintaining confidentiality requirements, which therefore makes security and privacy central concerns since semi-trusted agents may expose proprietary or safety-critical information. This motivates the investigation of exploiting *information-theoretically secure* distributed MM protocols that (i) preserve perfect privacy against collusions, (ii) obey strict storage constraints, and (iii) achieve near-optimal recovery thresholds with minimal communication.

Classical secure multiparty computation (MPC) protocols such as the schemes of Shamir [14], Blakley [15], and their extension to the BGW protocol [16], provide information-theoretic protection against collusion up to a threshold through polynomial-encoded functions. Subsequent triple-based and SPDZ-style systems [17]–[19] improve the online phase but still retain high communication overhead for large-scale matrix operations. When directly applied to MM with per-worker storage constraints, these approaches typically require a larger number of agents to tolerate threshold collusion, resulting in excessive resource redundancy and processing latency. Recent advances in coded computing [20]–[25] have shown that polynomial encodings can drastically reduce recovery thresholds in distributed linear algebra. In particular, *polynomial sharing* [26] achieves tight worker bounds for computing matrix polynomials under storage and resiliency constraints. However, to the best of our knowledge, while existing frameworks may provide a unified coded MM primitive that guarantees *perfect privacy*, *bounded storage*, for distributed computation, they do *not* incorporate intelligent or learning-based mechanisms such as low-rank decomposition or learned computation, to enhance local computational efficiency.

In this paper, we develop a perfectly secure matrix multiplication (PSMM) framework for multiparty computation, and establish guarantees for the correctness of reconstruction and information-theoretic privacy against collusion. Beyond security, we introduce, for the first time within this framework, a *learning-augmented PSMM* (LA-PSMM) scheme that integrates data-driven low-rank structure into local computations to improve computational efficiency. Notably, this learning-based integration is shown to preserve the original information-theoretic privacy and recovery guarantees. The main contributions of this paper are listed as follows.

- We propose a finite-field multiparty protocol that achieves perfect privacy against threshold-bounded semi-honest collusions, while explicitly accounting for local storage constraints. The scheme encodes shared inputs as evaluations of sparse masking polynomials, where selected coefficients align with partitioned matrix blocks, and all remaining terms are masked using random coefficients inspired by Beaver's triple construction (see eqs. (1), (2)).
- We establish the correctness and privacy guarantees of the proposed protocol. In particular, we show that any colluding set of agents below the prescribed threshold observes polynomial evaluations that are uniformly random and statistically independent of the underlying secrets, thereby ensuring perfect information-theoretic privacy (see Lemma 1). Moreover, we prove that the number of agents required for exact recovery of all target blocks is sufficient and matches the optimum in coded computing [26] (see Theorem 1, Proposition 2).
- We introduce a novel LA-PSMM framework that improves local computational efficiency without sacrificing security. In LA-PSMM, each agent replaces dense local block multiplication with a reduced-complexity tensor-decomposition approach. This framework couples modern learning-based decompositions (e.g., AlphaTensor [27]) that are generalizations of classical approaches such as

Strassen's [28] and Laderman's [29] algorithms. By exploiting learning-based decomposition, we enhance them through the integration of security and privacy, thereby enabling additional gains that benefit sparsity in local computation. We show that this operation preserves the polynomial support of local computations and thus does not affect the global recovery threshold, ensuring that privacy and correctness guarantees remain intact. Finally, numerical simulations validate LA-PSMM by demonstrating significant reductions in agent-side computational complexity compared with standard PSMM.

Notation: We denote the set of real numbers by $\mathbb{R}$. Different natural sets are defined as $\mathbb{N} \triangleq \{1, 2, \ldots\}$, $\mathbb{N}_0 \triangleq \{0, 1, \ldots\}$, and $\mathbb{N}_j^N \triangleq \{j, \ldots, N\}$, $j \leq N$, $N \in \mathbb{N}$. We denote a finite field by $\mathbb{F}$ with $|\mathbb{F}|$ sufficiently large. Matrices are denoted by upright uppercase letters (e.g., A, B and $\mathrm{A}^\top, \mathrm{B}^\top$ the transposes). For $k, m \in \mathbb{N}$ ($k < m$), we define the k horizontal partition of a matrix $\mathrm{A} \in \mathbb{F}^{m\times m}$ by $\mathrm{A} = [\mathrm{A}_1 \cdots \mathrm{A}_i \cdots \mathrm{A}_k]$, where $\mathrm{A}_i \in \mathbb{F}^{m\times(m/k)}$, $i \in \mathbb{N}_1^k$. We denote $\mathbf{0}$ a zero block of suitable size. For a matrix $\mathrm{A} \in \mathbb{F}^{m\times n}$, $\mathrm{vec}(\mathrm{A}) \in \mathbb{F}^{m\cdot n}$ denotes the column-wise vectorization of A. The operator $\mathrm{mat}(\cdot)$ reshapes a vector in $\mathbb{F}^{m\cdot n}$ into a matrix in $\mathbb{F}^{m\times n}$.

II. Preliminaries

We consider the multiparty computation setup as illustrated in Fig. 1, where an NCS is comprised of a source plant, one controller (co-located with the actuator), and N distributed computation agents, with $N \in \mathbb{N}$. The agents are assumed to be connected by authenticated, private point-to-point channels with the controller and with the source plant, and to be semi-honest (passive adversary), with up to $t-1$ agents potentially colluding. The controller only learns the final computation outputs from the agents to apply its control law.

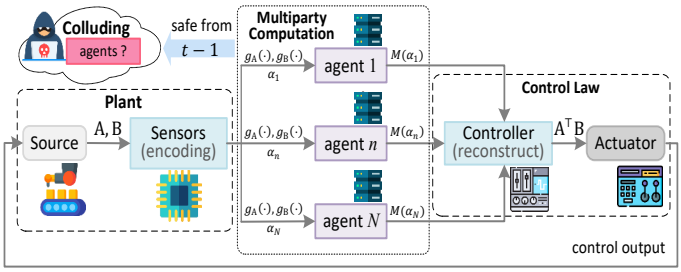

Fig. 1: An NCS architecture incorporating a multiparty computation setup.

Operation (Protocol): **(i)** *Sharing*. The source plant generates two secret information matrices $\mathrm{A}, \mathrm{B} \in \mathbb{F}^{m\times m}$ for some finite field $\mathbb{F}$ and dimension $m \in \mathbb{N}$, which need to be computed for multiplication aimed to be known by the controller. For example, these matrices $\mathrm{A}, \mathrm{B} \in \mathbb{F}^{m\times m}$ can be interpreted as correlation coefficient matrices arising from a multivariate, time-invariant linear state-space model. Due to the security concern, we use linear (additive) secret sharing, that is, Shamir sharing [14], and Beaver's Triple [17]. The secrets A, B will be partitioned into k column blocks respectively

$$\mathrm{A} = [\mathrm{A}_1 \cdots \mathrm{A}_j \cdots \mathrm{A}_k],\ \mathrm{B} = [\mathrm{B}_1 \cdots \mathrm{B}_j \cdots \mathrm{B}_k],$$

where $k \in \mathbb{N}, k < m$, and $\mathrm{A}_j, \mathrm{B}_j \in \mathbb{F}^{m\times(m/k)}$, $j \in \mathbb{N}_1^k$. Each partitioned sub-block of A, B is then encoded as the shares by leveraging a *polynomial sharing* scheme similar to [26]. This means that we encode the sub-blocks as coefficients of carefully designed sparse polynomials and distribute their evaluations as shares, which ensures that upon multiplying the encoded evaluations, the desired sub-block multiplications align in target coefficient positions, while random coefficients occupy all other positions to mask the secrets. Specifically, the sensor picks independent random masks so that each agent $n \in \mathbb{N}_1^N$ receives evaluations of two polynomials

$$g_{\mathrm{A}}(x) = \sum_{j=1}^{k} \mathrm{A}_j x^{j-1} + \sum_{\ell=1}^{t-1} \mathrm{R}_\ell^{(\mathrm{A})}\, x^{k^2+\ell-1}, \tag{1}$$

$$g_{\mathrm{B}}(x) = \sum_{j=1}^{k} \mathrm{B}_j x^{k(j-1)} + \sum_{\ell=1}^{t-1} \mathrm{R}_l^{(\mathrm{B})}\, x^{k^2+\ell-1}, \tag{2}$$

at distinct public broadcasting points $\alpha_1, \ldots, \alpha_N \in \mathbb{F}$ to substitute x, where $\mathrm{R}_\ell^{(\mathrm{A})}, \mathrm{R}_\ell^{(\mathrm{B})}$ are i.i.d. uniform mask blocks[1] in $\mathbb{F}^{m\times(m/k)}$, and $g_{\mathrm{A}}, g_{\mathrm{B}} : \mathbb{F}^{m\times m} \to \mathbb{F}^{m\times m}$ denote polynomial functions. Hence, each agent $n \in \mathbb{N}_1^N$ stores $g_{\mathrm{A}}(\alpha_n), g_{\mathrm{B}}(\alpha_n)$ that fit the storage budget, which is predefined based on physical constraints. This placement mirrors the *sparse* exponent pattern while ensuring coefficient alignment upon multiplication [26].

(ii) *Local computation and occasional communication*. Each agent $n \in \mathbb{N}_1^N$ processes a local multiplication computation with the received α_n of the form

$$M(\alpha_n) = g_{\mathrm{A}}(\alpha_n)^\top g_{\mathrm{B}}(\alpha_n).$$

According to the design, by writing the multiplication computation in the summation form $M(x) = \sum_\nu \mathrm{M}_\nu x^\nu$, $\nu \in \mathbb{N}_0$, the coefficient M_ν for any $i, j \in \mathbb{N}_1^k$ satisfies

$$\mathrm{M}_{i-1+k(j-1)} = \mathrm{A}_i^\top \mathrm{B}_j\,.$$

Therefore, all remaining coefficients include at least one random mask, which ensures privacy at the controller so that no exact information about A or B is carried. This coefficient-placement argument is standard in polynomial sharing [26].

(iii) *Reconstruction of the result at the controller*. Each agent linearly recombines its available evaluation $M(\alpha_n)$ into polynomial shares of the target blocks, and transmits compact shares to the controller through the private channel. The controller then interpolates the needed coefficients to recover the target matrix. In particular, one multiplication requires only N evaluations sufficient to interpolate all nonzero coefficients of $M(x)$. The degree and sparsity details are provided in the next section.

Objective: The security goal of the considered system is to ensure *correctness*, preserve *privacy* against any $t-1$ colluding agents, and guarantee *perfect information-theoretic privacy*, meaning that no additional information about the secrets is leaked to the controller beyond the agents' outputs.

[1]We view $\mathrm{R}_\ell^{(\mathrm{A})}, \mathrm{R}_\ell^{(\mathrm{B})}$ as $[a]$ and $[b]$ the components of a matrix-valued Beaver triple $([a], [b], [c] = [a] \times [b])$.

III. Main Results

We present our main results in this section. We first give a lemma to show that privacy is maintained for the agent shares.

Lemma 1 (Masking lemma)**.** *For any $p, q \in \mathbb{N}$, let $g(x) = \sum_{\ell=1}^{t-1} \mathrm{M}_\ell x^{\ell-1}$ with i.i.d. uniform coefficients M_ℓ over $\mathbb{F}^{p\times q}$. For any distinct $\beta_1, \ldots, \beta_{t-1} \in \mathbb{F}$, the tuple $\big(g(\beta_1), \ldots, g(\beta_{t-1})\big)$ is uniformly distributed over $(\mathbb{F}^{p\times q})^{t-1}$ and independent of any other variables.*

Proof. By Lagrange interpolation, evaluation at $t-1$ distinct points is a bijection between coefficients and evaluations, and uniform coefficients imply uniform evaluations [14]. Hence, no information on other variables leaks through these masked evaluations. □

We next show the correctness maintained in the secret sharing in (1) and (2) during the agent's local computation.

Proposition 1 (Correctness of triple-based multiplication)**.** *For any $a, b, c \in \mathbb{N}$, let $\mathrm{A} \in \mathbb{F}^{a\times b}$ and $\mathrm{B} \in \mathbb{F}^{b\times c}$ be two secret matrices, and let $[\mathrm{A}], [\mathrm{B}]$ denote any two of their linear secret shares among the shared parties. Assume access to a Beaver triple $(\mathrm{R}_1, \mathrm{R}_2, \mathrm{R}_3)$ related to $[\mathrm{A}], [\mathrm{B}]$ with random matrices $\mathrm{R}_1 \in \mathbb{F}^{a\times b}$, $\mathrm{R}_2 \in \mathbb{F}^{b\times c}$ and $\mathrm{R}_3 = \mathrm{R}_1\mathrm{R}_2 \in \mathbb{F}^{a\times c}$. Define the opened (i.e., publicly reconstructed) matrices*

$$\mathrm{D} = \mathrm{A} - \mathrm{R}_1, \ \textit{and} \ \ \mathrm{E} = \mathrm{B} - \mathrm{R}_2$$

with the same respective dimensions. Then the recombined value C satisfies

$$\mathrm{C} = \mathrm{R}_3 + \mathrm{D}\mathrm{R}_2 + \mathrm{R}_1\mathrm{E} + \mathrm{D}\mathrm{E} = \mathrm{A}\mathrm{B} \in \mathbb{F}^{a\times c}.$$

Moreover, each term in C can be computed from local shares and the publicly opened D, E without revealing A or B.

Proof. See [30, Appendix A]. □

The following result shows the information-theoretic privacy of the considered secret sharing scheme.

Theorem 1 (Agent bound for secure MM)**.** *Let $\mathrm{A}, \mathrm{B} \in \mathbb{F}^{m\times m}$ be k partitioned respectively as*

$$\mathrm{A} = \big[\mathrm{A}_1 \cdots \mathrm{A}_i \cdots \mathrm{A}_k\big], \ \mathrm{B} = \big[\mathrm{B}_1 \cdots \mathrm{B}_j \cdots \mathrm{B}_k\big], \tag{3}$$

where $\mathrm{A}_i, \mathrm{B}_j \in \mathbb{F}^{m\times(m/k)}$, $i, j \in \mathbb{N}^k$. Assume each agent stores at most a $1/k$ fraction of each input (one $m \times (m/k)$ block per matrix), and up to $t-1$ agents may collude. There exists distinct encoding points $\alpha_1, \ldots, \alpha_N \in \mathbb{F}$ and encoders

$$g_{\mathrm{A}}(x) = \sum_{i=1}^{k} \mathrm{A}_i \, x^{i-1} + \sum_{\ell=1}^{t-1} \mathrm{R}_\ell^{(\mathrm{A})} \, x^{k^2+\ell-1}, \tag{4}$$

$$g_{\mathrm{B}}(x) = \sum_{j=1}^{k} \mathrm{B}_j \, x^{k(j-1)} + \sum_{\ell=1}^{t-1} \mathrm{R}_\ell^{(\mathrm{B})} \, x^{k^2+\ell-1}, \tag{5}$$

with i.i.d. random masks $\mathrm{R}_\ell^{(\mathrm{A})}, \mathrm{R}_\ell^{(\mathrm{B})} \in \mathbb{F}^{m\times(m/k)}$, such that the protocol that sends evaluations $g_{\mathrm{A}}(\alpha_n), g_{\mathrm{B}}(\alpha_n)$ to agent $n \in \mathbb{N}_1^N$ and aggregates $M(\alpha_n) = g_{\mathrm{A}}(\alpha_n)^\top g_{\mathrm{B}}(\alpha_n)$ to securely and perfectly reconstruct $\mathrm{A}^\top\mathrm{B}$ using N agents, where

$$N \le \min\left\{2k^2 + 2t - 3, \ k^2 + kt + t - 2\right\}. \tag{6}$$

Proof. See [30, Appendix B]. □

Remark 1 (Positioning relative to polynomial sharing)**.** *The agent threshold (bound) in Theorem 1 provides evaluations sufficient to interpolate all nonzero coefficients of $M(x) = g_{\mathrm{A}}(x)^\top g_{\mathrm{B}}(x)$ and hence obtain $\mathrm{A}^\top\mathrm{B}$ from every recovered multiplication $\mathrm{A}_i^\top\mathrm{B}_j$, $i, j \in \mathbb{N}^k$, and it coincides with the optimal recovery bounds known for general polynomial sharing schemes [26]. However, our result is obtained under a more restrictive and structured model: we focus exclusively on MM, enforce a zero-gap alignment of all useful coefficients, and require closure under composition for subsequent multiplications. Achieving the same threshold under these additional constraints is nontrivial and essential for MPC-compatible and tensor-augmented secure computation.*

Remark 2 (Improvement via polynomial sharing)**.** *The bound in Theorem 1 strictly improves over naive job-splitting + BGW [31], which requires $k^2(2t-1)$ agents for one multiplication. Our scheme keeps N independent of the number of chained multiplications when results are immediately reshared (closure under polynomial sharing), as observed in [26].*

We next highlight the importance of our approach from the state-of-the-art through the following proposition.

Proposition 2 (Advantage under reduced degrees-of-freedom)**.** *Consider the PSMM encoding of Theorem 1 with storage fraction $1/k$ and privacy against $t-1$ colluding agents. Let the target block multiplications be*

$$\mathrm{Z}_{i,j} \triangleq \mathrm{A}_i^\top\mathrm{B}_j \in \mathbb{F}^{(m/k)\times(m/k)}, \ i, j \in \mathbb{N}_1^k.$$

Assume the input class satisfies a known structural constraint such that the collection

$$\mathcal{Z} \triangleq \{\mathrm{Z}_{i,j} : \ i, j \in \mathbb{N}_1^k\}$$

admits at most s degrees-of-freedom (DOF), in the following sense:

***DOF assumption**: There exist known linear maps $\mathcal{L}_1, \ldots, \mathcal{L}_s$, for $s \in \mathbb{N}$, and known coefficient tensors $\{\Gamma_\ell^{(i,j)} : l \in \mathbb{N}_1^s\}$ such that for all admissible inputs (A, B),*

$$\mathrm{Z}_{i,j} \ = \ \sum_{\ell=1}^{s} \Gamma_\ell^{(i,j)} \, \mathcal{L}_\ell(\mathrm{A}, \mathrm{B}), \ \forall\, i, j \in \mathbb{N}_1^k.$$

Equivalently, $\{\mathrm{Z}_{i,j}\}$ lies in a known s-dimensional linear subspace of $\big(\mathbb{F}^{(m/k)\times(m/k)}\big)^{k^2}$. Then the number of agent evaluations required to recover $\mathrm{A}^\top\mathrm{B}$ (and hence all needed $\mathrm{Z}_{i,j}$) with perfect privacy can be reduced from

$$N^\star(k, t) = \min\{\, 2k^2 + 2t - 3, \ k^2 + kt + t - 2 \,\},$$

to

$$N_{\textit{struct}}(k, t, s) \ \le \ \min\{\, 2s + 2t - 3, \ s + ks + t - 2 \,\},$$

by (i) keeping the same PSMM masking/tail degree for t-privacy and (ii) decoding only the s effective DOF rather than all k^2 block coefficients.

In particular, for any fixed N, the tolerable collusion level satisfies

$$t_{struct} \geq t_{worst}, \quad \forall s < k^2,$$

i.e., reduced DOF strictly increases collusion tolerance (or equivalently reduces the required number of agents).

Proof. See [30, Appendix D]. □

IV. Learning-Augmented Perfectly Secure Matrix Multiplication (LA-PSMM) Design

This section extends our PSMM protocol by incorporating a *learning-based optimization layer* inspired by AlphaTensor [27]. The goal is to jointly achieve algorithmic efficiency in the local multiplications via reinforcement learning (RL) while maintaining perfect information-theoretic security under the proposed PSMM framework. The resulting protocol, termed **Learning-Augmented PSMM (LA-PSMM)**, unifies our secret sharing and coded-computation guarantees with a data-driven discovery of efficient multiplication schemes.

A. Motivation and Overview

While PSMM ensures privacy and optimal agent thresholds, each agent is still required to perform a dense local MM $M(\alpha_n) = g_{\mathrm{A}}(\alpha_n)^\top g_{\mathrm{B}}(\alpha_n)$, whose arithmetic complexity scales as $\mathcal{O}\big((\frac{m}{k})^3\big)$. Recent advances, such as AlphaTensor [27], demonstrate that deep RL can identify low-rank decompositions of the MM tensor, thereby reducing the number of scalar multiplications.

The LA-PSMM design equips each agent with an RL-discovered *tensor decomposition pattern* that replaces the naive dense multiplication with a sequence of bilinear forms, while maintaining perfect secrecy through our masking and coded sharing mechanism (cf. Proposition 3).

B. RL-Enhanced Block Multiplication

Let each agent receive its encoded inputs $g_{\mathrm{A}}(\alpha_n), g_{\mathrm{B}}(\alpha_n) \in \mathbb{F}^{m\times(m/k)}$. AlphaTensor represents MM as a 3-way tensor that maps the bilinear form vectorized inputs to the output via

$$\mathrm{vec}(\mathrm{C}) = \sum_{r=1}^{T} u_r^\top \mathrm{vec}(\mathrm{A}) v_r^\top \mathrm{vec}(\mathrm{B}) w_r,$$

where $T \in \mathbb{N}$ is the tensor rank (number of scalar multiplications), and (u_r, v_r, w_r) are the low-rank decompositions RL agent searches for to minimize T.

We adapt this formulation to our finite-field setting. Each agent holds masked inputs and executes only the local operations corresponding to its learned decomposition pattern $\{u_r, v_r, w_r\}_{r=1}^{T_l}$ with a learned rank T_l. Formally, for agent n we define

$$M(\alpha_n) \triangleq \sum_{r=1}^{T_l} \Big(\langle u_r, \mathrm{vec}(g_{\mathrm{A}}(\alpha_n))\rangle \cdot \langle v_r, \mathrm{vec}(g_{\mathrm{B}}(\alpha_n))\rangle\Big) \cdot \mathrm{mat}(w_r),$$

where $\mathrm{mat}(\cdot)$ reshapes the vector w_r back into a $(m/k) \times (m/k)$ matrix block. Each agent thus performs fewer scalar multiplications, replacing the standard $(m/k)^3$ cost with $T_l \ll (m/k)^3$ while keeping the same algebraic structure required by PSMM.

C. Algorithmic Integration

The LA-PSMM protocol preserves all privacy guarantees of PSMM with the only modification in the *local multiplication* step as a learned low-rank decomposition instead of a dense matrix product. We present the procedure below.

Proposition 3 (Agent bound under tensorized local multiplications)**.** *Let $\mathrm{A}, \mathrm{B} \in \mathbb{F}^{m\times m}$ be k partitioned respectively as* (3) *with $\mathrm{A}_i, \mathrm{B}_j \in \mathbb{F}^{m\times(m/k)}$, $i, j \in \mathbb{N}^k$. Consider the encoded polynomials* (4)-(5) *as in Theorem 1, and let $\alpha_1, \ldots, \alpha_N \in \mathbb{F}$ be distinct public evaluation points. Suppose that each agent $n \in \mathbb{N}_1^N$ does not form the dense product $g_{\mathrm{A}}(\alpha_n)^\top g_{\mathrm{B}}(\alpha_n)$ directly, but instead evaluates the same bilinear map via a rank-T tensor decomposition, i.e.*

$$M(\alpha_n) = \sum_{r=1}^{T} \big\langle u_r, \mathrm{vec}(g_{\mathrm{A}}(\alpha_n))\big\rangle \big\langle v_r, \mathrm{vec}(g_{\mathrm{B}}(\alpha_n))\big\rangle \mathrm{vec}(w_r), \quad (7)$$

for some fixed triplets (u_r, v_r, w_r) over $\mathbb{F}$ that realize the usual MM on $(m/k) \times (m/k)$ blocks (e.g., Strassen or an AlphaTensor-empowered decomposition). Then, the resulting global product polynomial $M(x) \triangleq g_{\mathrm{A}}(x)^\top g_{\mathrm{B}}(x)$ has the same set of nonzero exponents as in Theorem 1, and all desired block multiplications $\mathrm{A}_i^\top \mathrm{B}_j$ appear at exponents $i-1+k(j-1)$, while every other exponent contains at least one random mask. Consequently, the number of agents N required to interpolate $M(x)$ (entrywise) is unchanged and still satisfies (6)*.*

Proof. The proof follows the structure of Theorem 1, but with the role of the local bilinear realization being explicit. □

The implementation of Proposition 3 is illustrated in Algorithm 1, and we provide a lemma to show that with the learning integration, the privacy-preserving property is still maintained.

Lemma 2 (Operator choice does not affect privacy or recovery threshold)**.** *For any available operator $M \in \mathcal{M}$ whose local computation implements the exact bilinear map $M(\alpha) = g_{\mathrm{A}}(\alpha)^\top g_{\mathrm{B}}(\alpha)$ for all $\alpha \in \mathbb{F}$, the LA-PSMM protocol in Algorithm 1 achieves the same (i) perfect privacy against any $t-1$ colluding agents and (ii) agent threshold $N^\star(k,t)$ as Theorem 1.*

Proof. Consider LA-PSMM with the matrices $\mathrm{A}, \mathrm{B} \in \mathbb{F}^{m\times m}$ and any colluding agents set $\mathcal{T} \subseteq \mathbb{N}_1^N$ with $|\mathcal{T}| \leq t-1$. Agent n receives $\big(g_{\mathrm{A}}(\alpha_n), g_{\mathrm{B}}(\alpha_n)\big)$ and then calculate the output $\hat{M}(\alpha_n)$. By assumption on the operator $M \in \mathcal{M}$, $\forall \alpha \in \mathbb{F}$, we have $\hat{M}(\alpha) = M(\alpha) = g_A(\alpha)^\top g_B(\alpha)$, thus, $\hat{M}(\alpha_n)$ is a deterministic function of the shares $\big(g_{\mathrm{A}}(\alpha_n), g_{\mathrm{B}}(\alpha_n)\big)$.

From privacy perspective: The joint distributed shares $\{g_{\mathrm{A}}(\alpha_{n'}), g_{\mathrm{B}}(\alpha_{n'}) : n' \in \mathcal{T}\}$ are independent of A, B by the same masking arguments as in Theorem 1 (the random masks provide perfect $t-1$ privacy). Since $\{\hat{M}(\alpha_{n'}) : n' \in \mathcal{T}\}$ is obtained by deterministic post-processing of these shares, it cannot reveal additional information, hence, LA-PSMM is still perfectly private against any $t-1$ colluding agents.

From recovery threshold perspective: The controller receives $\{\hat{M}(\alpha_n) : n \in \mathbb{N}_1^N\} = \{g_{\mathrm{A}}(\alpha_n)^\top g_{\mathrm{B}}(\alpha_n) : n \in \mathbb{N}_1^N\}$,

Algorithm 1 Learning-Augmented Perfectly Secure Matrix Multiplication (LA-PSMM)

Input: Matrices $\mathrm{A}, \mathrm{B} \in \mathbb{F}^{m\times m}$, parameters (k, t, N), learned decomposition $\{(u_r, v_r, w_r)\}_{r=1}^{T_l}$

Partition A, B into k column blocks as in (3)
Perform polynomial encodings $g_{\mathrm{A}}, g_{\mathrm{B}}$ as in (4), (5) with random masks $\mathrm{R}_\ell^{\mathrm{A}}, \mathrm{R}_\ell^{\mathrm{B}}$
Choose distinct public points $\alpha_1, \ldots, \alpha_N \in \mathbb{F}$
Send to each agent n the evaluations $g_{\mathrm{A}}(\alpha_n), g_{\mathrm{B}}(\alpha_n)$
for $n = 1 : N$ **do**
    Agent n receives $g_{\mathrm{A}}(\alpha_n), g_{\mathrm{B}}(\alpha_n)$
    Initialize $M(\alpha_n) \leftarrow \mathbf{0}^{(m/k)\times(m/k)}$
    for $r = 1 : T_l$ **do**
        $a_r \leftarrow \langle u_r, \mathrm{vec}(g_{\mathrm{A}}(\alpha_n))\rangle$
        $b_r \leftarrow \langle v_r, \mathrm{vec}(g_{\mathrm{B}}(\alpha_n))\rangle$
        $M(\alpha_n) \leftarrow M(\alpha_n) + (a_r \cdot b_r)\,\mathrm{mat}(w_r)$
    end for
    Agent n sends $M(\alpha_n)$ to the controller
end for
Controller collects $\{M(\alpha_n) : n \in \mathbb{N}_1^N\}$
Controller solves the blockwise Vandermonde system induced by $\{\alpha_n\}$ to interpolate $M(x) = g_{\mathrm{A}}(x)^\top g_{\mathrm{B}}(x)$
Extract coefficients $M_{i-1+k(j-1)} = \mathrm{A}_i^\top \mathrm{B}_j$, $\forall i, j \in \mathbb{N}_1^k$
Stack $\{\mathrm{A}_i^\top \mathrm{B}_j\}$ to form $\mathrm{A}^\top \mathrm{B}$

Output: Multiplication $\mathrm{A}^\top \mathrm{B}$ is reconstructed at the controller

i.e., the same evaluation values of $M(x) = g_{\mathrm{A}}(x)^\top g_{\mathrm{B}}(x)$ as Theorem 1. Therefore, the same interpolation succeeds from the same number of agents, namely $N^\star(k, t)$, which still preserves the reconstruction. This completes the proof. □

V. Numerical Results

In this section, we provide numerical simulations to support our theoretical framework.

We show that LA-PSMM has lower computational complexity than PSMM. In PSMM, each agent performs a coded dense multiplication between an $(s \times m)$ and an $(m \times s)$ matrix with computational complexity $\mathcal{O}(m^3/k^2)$ per agent and $\mathcal{O}(Nm^3/k^2)$ total. In LA-PSMM, the dense product is replaced by a learned low-rank expansion of length T_l, where each agent evaluates two linear forms on vectors of length $ms = m^2/k$ and accumulates T_l rank-1 terms into an $s\times s$ output with computational complexity $\mathcal{O}(T_l m^2/k)$ per agent and $\mathcal{O}(NT_l m^2/k)$ total, yielding a local computational reduction whenever an improvement when $T_l \ll m/k$. Encoding remains $\mathcal{O}(N(k+t)m^2/k)$, and decoding via entrywise interpolation is $\mathcal{O}((m/k)^2 N^2)$ with a naive Vandermonde solver (or near-linear in N per entry with fast interpolation).

Fig. 2 depicts total computational complexity across agents for $k = 8$, $t = 4$, and $N = 98$. LA-PSMM consistently requires less computation than PSMM, and the gap grows with dimension m (up to 80%), indicating that learning-augmented local multiplication increasingly amortizes cost as problem size increases. Fig. 3 further shows the computation gain, defined as the ratio of PSMM to LA-PSMM local cost, which rises monotonically with m, reflecting increasing benefit from the learning-based tensor decomposition. Overall, LA-PSMM delivers substantial, scalable savings in computation while preserving PSMM's security guarantees.

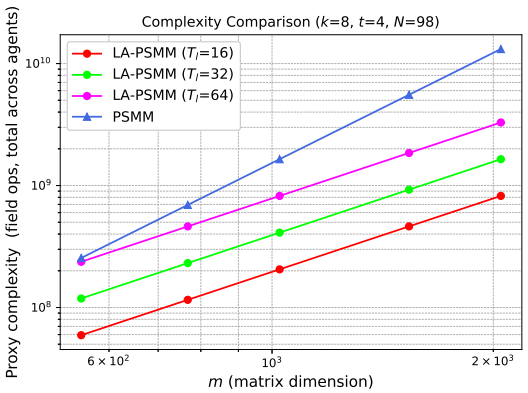

Fig. 2: The agent computation complexity comparison of PSMM vs. LA-PSMM with different learned rank T_l.

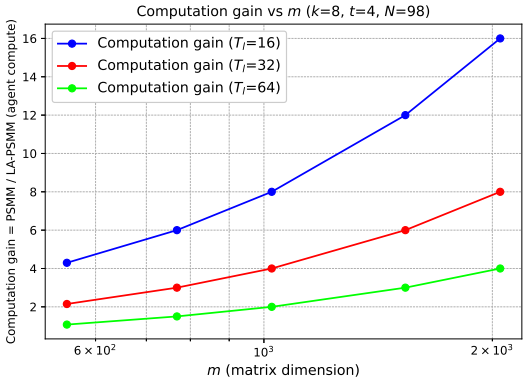

Fig. 3: The agent computation gain of LA-PSMM with different learned rank T_l.

VI. Conclusion

We introduced a PSMM protocol for MPC that combines sparse masking polynomial encoding with careful coefficient alignment and Beaver-style randomness to ensure correctness and information-theoretic privacy against threshold-bounded collusions under explicit local storage constraints. We further proposed an LA-PSMM extension that utilizes tensor-decomposition-based local block multiplication to exploit the learned low-rank structure, which preserves privacy and recovery guarantees while improving computational efficiency.

VII. Acknowledgement

This work was supported in part by the Huawei France–EURECOM Chair on Future Wireless Networks. D. Malak was also partially supported by the ERC Starting Grant SENSIBILITÉ (101077361) and the French National Research Agency (ANR-22-PEFT-0010), France 2030.

REFERENCES

[1] D. Simon, *Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches*. Wiley-Interscience, 2006.

[2] P. A. Stavrou, J. Østergaard, and C. D. Charalambous, "Zero-delay rate distortion via filtering for vector-valued Gaussian sources," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 5, pp. 841–856, Oct 2018.

[3] P. A. Stavrou and M. Skoglund, "Indirect NRDF for partially observable Gauss-Markov processes with MSE distortion: Characterizations and optimal solutions," *IEEE Trans. Autom. Control*, vol. 69, no. 9, pp. 5867 – 5882, 2024.

[4] D. P. Bertsekas, *Dynamic programming and optimal control*. Belmont, MA., USA: Athena Scientific, 2005, vol. 4.

[5] P. A. Stavrou and M. Skoglund, "LQG control and linear policies for noisy communication links with synchronized side information at the decoder," *Automatica*, vol. 123, p. 109306, 2021.

[6] P. A. Stavrou, M. Skoglund, and T. Tanaka, "Sequential source coding for stochastic systems subject to finite rate constraints," *IEEE Trans. Autom. Control*, vol. 67, no. 8, pp. 3822–3835, 2022.

[7] I. Loshchilov, T. Glasmachers, and H.-G. Beyer, "Large scale black-box optimization by limited-memory matrix adaptation," *IEEE Trans. on Evolution. Computation*, vol. 23, no. 2, pp. 353–358, Jul. 2018.

[8] M. Smelyanskiy, S. Skedzielewski, and C. Dulong, "Parallel computing for large-scale optimization problems: Challenges and solutions." *Intel Technology Journal*, vol. 9, no. 2, May 2005.

[9] O. Ordentlich and Y. Polyanskiy, "Optimal quantization for matrix multiplication," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*. IEEE, Jun. 2025, pp. 1–6.

[10] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, "A survey of recent results in networked control systems," *Proc. IEEE*, vol. 95, no. 1, pp. 138–162, Jan 2007.

[11] S. Schlor, M. Hertneck, S. Wildhagen, and F. Allgöwer, "Multi-party computation enables secure polynomial control based solely on secret-sharing," in *Proc. IEEE Conf. Decision Control*, Austin, TX, USA, Feb. 2021, pp. 4882–4887.

[12] M. S. Darup, A. B. Alexandru, D. E. Quevedo, and G. J. Pappas, "Encrypted control for networked systems: An illustrative introduction and current challenges," *IEEE Control Syst. Mag.*, vol. 41, no. 3, pp. 58–78, 2021.

[13] S. Yüksel and T. Başar, *Stochastic networked control systems: stabilization and optimization under information constraints*. Birkhäuser New York, NY, USA, 2014.

[14] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[15] G. R. Blakley, "Safeguarding cryptographic keys," in *International Workshop on Managing Requirements Knowledge*, New York, NY, USA, Jun. 1979, pp. 313–313.

[16] A. Wigderson, M. Or, and S. Goldwasser, "Completeness theorems for noncryptographic fault-tolerant distributed computations," in *Proceedings of the 20th Annual Symposium on the Theory of Computing (STOC'88)*, Chicago, Illinois, USA, 1988, pp. 1–10.

[17] D. Beaver, "Efficient multiparty protocols using circuit randomization," in *Annual international cryptology conference*. Springer, 1991, pp. 420–432.

[18] M. Keller, "MP-SPDZ: A versatile framework for multi-party computation," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, New York, NY, USA, Oct. 2020, pp. 1575–1590.

[19] Y. Yang *et al.*, "Maliciously secure circuit private set intersection via SPDZ-compatible oblivious PRF," *Proceedings on Privacy Enhancing Technologies*, 2025.

[20] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," *IEEE Trans. Inf. Theory*, vol. 64, no. 3, pp. 1514–1529, 2017.

[21] W.-T. Chang and R. Tandon, "On the capacity of secure distributed matrix multiplication," in *Proc. IEEE Global Commun. Conf.* IEEE, 2018, pp. 1–6.

[22] Q. Yu, S. Li, N. Raviv, S. M. M. Kalan, M. Soltanolkotabi, and S. A. Avestimehr, "Lagrange coded computing: Optimal design for resiliency, security, and privacy," in *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 2019, pp. 1215–1225.

[23] Z. Jia and S. A. Jafar, "On the capacity of secure distributed batch matrix multiplication," *IEEE Trans. Inf. Theory*, vol. 67, no. 11, pp. 7420–7437, 2021.

[24] S. Dutta *et al.*, "On the Optimal Recovery Threshold of Coded Matrix Multiplication," *IEEE Trans. Inf. Theory*, vol. 66, no. 1, pp. 278–301, 2020.

[25] R. Cartor, R. G. D'Oliveira, S. E. Rouayheb, D. Heinlein, D. Karpuk, and A. Sprintson, "Secure distributed matrix multiplication with precomputation," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*. IEEE, 2024, pp. 2568–2573.

[26] H. Akbari-Nodehi and M. A. Maddah-Ali, "Secure coded multi-party computation for massive matrix operations," *IEEE Trans. Inf. Theory*, vol. 67, no. 4, pp. 2379–2398, 2021.

[27] A. Fawzi *et al.*, "Discovering faster matrix multiplication algorithms with reinforcement learning," *Nature*, vol. 610, no. 7930, pp. 47–53, Oct. 2022.

[28] V. Strassen, "Gaussian elimination is not optimal," *Numerische mathematik*, vol. 13, no. 4, pp. 354–356, 1969.

[29] J. D. Laderman, "A noncommutative algorithm for multiplying 3*3 matrices using 23 multiplications," *Bulletin of the American Mathematical Society*, 1976.

[30] Z. He, M. R. Deylam Salehi, D. Malak, and P. A. Stavrou, "Learning-augmented perfectly secure collaborative matrix multiplication," *arXiv*, 2026, extended version with proofs. [Online]. Available: http://arxiv.org/abs/2601.09916

[31] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation," in *Providing sound foundations for cryptography: on the work of Shafi Goldwasser and Silvio Micali*, 2019, pp. 351–371.