Distributed and Private Coded Matrix Computation with Flexible Communication Load

Malihe Aliasgari[*], **Osvaldo Simeone** [†], and **Jörg Kliewer**[*]
[*]Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, U.S.A.
[†]King's College London, Department of Informatics, London, U.K.

***Abstract*—Tensor operations, such as matrix multiplication, are central to large-scale machine learning applications. These operations can be carried out on a distributed computing platform with a master server at the user side and multiple workers in the cloud operating in parallel. For distributed platforms, it has been recently shown that coding over the input data matrices can reduce the computational delay, yielding a trade-off between recovery threshold and communication load. In this work, we impose an additional security constraint on the data matrices and assume that workers can collude to eavesdrop on the content of these data matrices. Specifically, we introduce a novel class of secure codes, referred to as secure generalized PolyDot codes, that generalizes previously published non-secure versions of these codes for matrix multiplication. These codes extend the state-of-the-art by allowing a flexible trade-off between recovery threshold and communication load for a fixed maximum number of colluding workers.**

***Index Terms*—Coded distributed computation, distributed learning, secret sharing, information theoretic security.**

I. Introduction

At the core of many signal processing and machine learning applications are tensor operations such as matrix multiplications [1]. In the presence of practically sized data sets, such operations are typically carried out using distributed computing platforms with a master server and multiple workers that can operate in parallel over distinct parts of the data set. The master server plays the role of the parameter server, distributing data to the workers and periodically reconciling their internal state. Typically, workers are commercial off-the-shelf servers that are characterized by possible temporary failures and delays. While current distributed computing platforms conventionally handle straggling servers by means of replication of computing tasks, recent work has shown that encoding the input data can help reduce the computation latency, which depends on the number of tolerated stragglers by orders of magnitude, e.g., [2]. More generally, coding is able to control the trade-off between computational delay and communication load between workers and master server [3]–[6]. Furthermore, stochastic coding can help keeping both input and output data secure from eavesdropping and colluding workers (see, e.g., [7]–[10]). This paper contributes to this line of work by investigating the trade-off between computational delay and communication load as a function of the privacy level.

As illustrated in Fig. 1, we focus on the basic problem of computing the matrix multiplication $\mathbf{C} = \mathbf{AB}$ in a distributed computing system of P workers that can process each only a fraction $1/m$ and $1/n$ of matrices $\mathbf{A}$ and $\mathbf{B}$, respectively. Three performance criteria are of interest: (*i*) the recovery threshold P_R, that is, the number of workers that need to complete their task before the master server can recover the product $\mathbf{C}$; (*ii*) the communication load C_L between workers and master server; and (*iii*) the maximum number P_C of colluding servers that ensures perfect secrecy for both data matrices $\mathbf{A}$ and $\mathbf{B}$. In order to put our contribution in perspective, we briefly review next prior related work.

Consider first solutions that provide no security guarantees, i.e., $P_C = 0$. As a direct extension of [3], a first approach is to use product codes that apply separate MDS codes to encode the two matrices [11]. The recovery threshold of this scheme is improved by [4] which introduces so called polynomial codes. This construction is proved to be optimal under the assumption that minimal communication is allowed between workers and master server. In [12] so called MatDot codes are introduced, resulting in a lower recovery threshold at the expense of a larger communication load. The construction in [13] bridges the gap between polynomial and MatDot codes and presents so called PolyDot codes, yielding a trade-off between recovery threshold and communication load. An extension of this scheme in [14], [15], termed Generalized PolyDot (GPD) codes improves on the recovery threshold of PolyDot codes.

Much less work has been done in the literature if security constraints are factored in, i.e., if $P_C \neq 0$. In [7] Lagrange coding is presented which achieves the minimum recovery threshold for multilinear functions by generalizing MatDot codes. In [8]–[10] a reduction of the communication load is addressed by extending polynomial codes. While these works focus on either minimizing recovery threshold or communication load, the *trade-off* between these two fundamental quantities has not been addressed in the open literature to the best of our knowledge. In this paper, we intend to fill this void and present a novel class of secure computation codes, referred to as secure GPD (SGPD) codes, that generalize GPD codes at all communication load levels, yielding a new achievable trade-off between recovery threshold and communication load as a function of the desired privacy level.

II. System Model

As illustrated in Fig. 1, we consider a distributed computing system with a master server and P workers. The master server is interested in computing securely the matrix product $\mathbf{C} = \mathbf{AB}$ of two data matrices $\mathbf{A}$ and $\mathbf{B}$ with dimensions

This work was supported in part by the European Research Council (ERC) under the European Union Horizon 2020 research and innovative programme (grant agreement No 725731) and by U.S. NSF grants CNS-1526547, CCF-1525629.

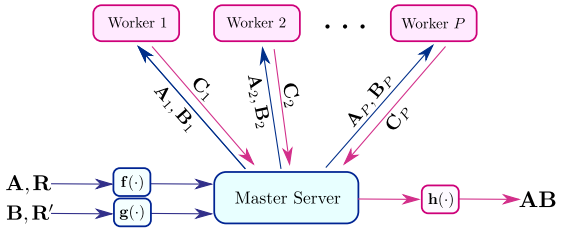

Fig. 1: The master server encodes the input matrices $\mathbf{A}$ and $\mathbf{B}$ and random matrices $\mathbf{R}$ and $\mathbf{R}'$, resp., to define the computational tasks of the workers. The workers may fail or straggle, and they are honest but curious, with colluding subsets of workers of size of at most P_C. The master server must be able to decode the product $\mathbf{AB}$ from the output of a subset of P_R workers, which defines the recovery threshold.

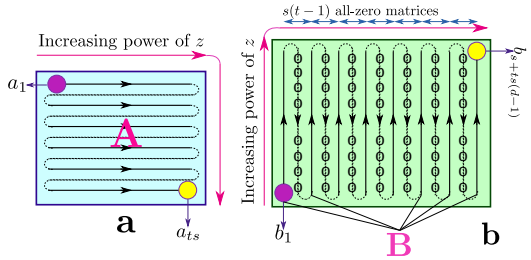

Fig. 2: Construction of the time sequences $\mathbf{a}$ and $\mathbf{b}$ used to define a GPD code. The zero dashed line in $\mathbf{b}$ indicates all-zero block sequences. Each solid arrows in $\mathbf{a}$ and $\mathbf{b}$ shows a distinct row of $\mathbf{A}$ and a column of $\mathbf{B}$, respectively.

$T \times S$ and $S \times D$, respectively. The matrices have entries from a sufficient large finite field $\mathbb{F}$, with $|\mathbb{F}| > P$. Both matrices $\mathbf{A}$ and $\mathbf{B}$ contain confidential data. The P workers receive information on matrices $\mathbf{A}$ and $\mathbf{B}$ from the master; they process this information and respond to the master, which finally recovers the product $\mathbf{AB}$ with minimal computational effort. Each worker can receive and process only TS/m and SD/n symbols, resp., for some integers m and n. We assume that the workers are honest but curious, and impose the secrecy constraint that, even if up to $P_C < P$ workers collude, the workers cannot obtain any information about both matrices $\mathbf{A}$ and $\mathbf{B}$ based on the data received from the master server.

To keep the data secure and to leverage possible computational redundancy at the workers (namely, if $P/m > 1$ and/or $P/n > 1$), the master server sends encoded versions of the input matrices to the workers. Due to communication and storage constraints the encoded matrices $\mathbf{A}_p = \mathbf{f}_p(\mathbf{A}, \mathbf{R})$ with $\mathbf{f}_p : \mathbb{F}^{TS/m} \times \mathbb{F}^{TS/m} \to \mathbb{F}^{TS/m}$ and $\mathbf{B}_p = \mathbf{g}_p(\mathbf{B}, \mathbf{R}')$ with $\mathbf{g}_p : \mathbb{F}^{SD/n} \times \mathbb{F}^{SD/n} \to \mathbb{F}^{SD/n}$, to be sent to each p-th worker, $p = 1, \ldots, P$, have TS/m and SD/n entries, resp., for some encoding functions $\mathbf{f}_p(\cdot)$ and $\mathbf{g}_p(\cdot)$. The security constraint imposes the condition

$$I(\mathbf{A}_{\mathcal{P}}, \mathbf{B}_{\mathcal{P}}; \mathbf{A}, \mathbf{B}) = 0, \tag{1}$$

where $\mathbf{A}_{\mathcal{P}} = \{\boldsymbol{A}_p\}_{p \in \mathcal{P}}$ and $\mathbf{B}_{\mathcal{P}} = \{\boldsymbol{B}_p\}_{p \in \mathcal{P}}$ for all subsets of $\mathcal{P} \subset [1, P]$ of P_C workers, where the random matrices $\mathbf{R}$ and $\mathbf{R}'$ serve as random keys in order to meet the security constraint (1).

Each worker p computes the product $\mathbf{C}_p = \mathbf{A}_p \mathbf{B}_p$ of the encoded sub-matrices $\mathbf{A}_p$ and $\mathbf{B}_p$. The master server collects a subset of $P_R \leq P$ outputs from the workers as defined by the subset $\mathbf{C}_{\mathcal{P}_R} = \{\mathbf{C}_p\}_{p \in \mathcal{P}_R}$ with $|\mathcal{P}_R| = P_R$. It then applies a decoding function $\mathbf{h}\left(\mathbf{C}_{\mathcal{P}_R}\right)$, $\mathbf{h} : \underbrace{\mathbb{F}^{TD/td} \times \cdots \times \mathbb{F}^{TD/td}}_{P_R \text{ times}} \to \mathbb{F}^{TD}$, where correct decoding translates into the condition

$$H(\mathbf{AB} | \mathbf{C}_{\mathcal{P}_R}) = 0. \tag{2}$$

For given storage parameters m and n, the performance of a coding and decoding scheme is measured by the triple (P_C, P_R, C_L), where P_C is the maximum number of colluding workers; P_R is the recovery threshold, i.e., the minimum number of workers whose outputs are used by the master to recover the product $\mathbf{AB}$, and C_L is the communication load defined as $C_L = \sum_{p \in \mathcal{P}_R} |\mathbf{C}_p|$. Here, $|\mathbf{C}_p|$ is the dimension of the product matrix $\mathbf{C}_p$ computed by worker p. Note that (2) requires the inequality $\min\{P_R/m, P_R/n\} \geq 1$ or $P_R \geq P_{R,\min} \triangleq \max\{m, n\}$, which is hence a lower bound for the minimum recovery threshold. Furthermore, the communication load is lower bounded by $C_L \geq C_{L,\min} \triangleq TD$, which is the size of the product $\mathbf{C} = \mathbf{AB}$.

A. Generalized PolyDot Code without Security Constraint

In this subsection, we review GPD codes proposed [12] and subsequently improved in [14], [15]. This construction achieves the best currently known trade-off between recovery threshold P_R and communication load C_L for $P_C = 0$. The equivalent entangled polynomial codes of [15] have the same properties in terms of (P_R, P_C). The GPD codes for $P_C = 0$ also achieve the optimal recovery threshold among all linear coding strategies in the cases of $t = 1$ or $d = 1$, they also minimize the recovery threshold for the minimum communication load $C_{L,\min}$ [4], [15].

The GPD code splits the data matrices $\mathbf{A}$ and $\mathbf{B}$ both horizontally and vertically as

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{1,1} & \ldots & \mathbf{A}_{1,s} \\ \vdots & \ddots & \vdots \\ \mathbf{A}_{t,1} & \ldots & \mathbf{A}_{t,s} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{B}_{1,1} & \ldots & \mathbf{B}_{1,d} \\ \vdots & \ddots & \vdots \\ \mathbf{B}_{s,1} & \ldots & \mathbf{B}_{s,d} \end{bmatrix}. \tag{3}$$

The parameters s, t, and d can be set arbitrarily under the constraints $m = ts$ and $n = sd$. Note that polynomial codes use $s = 1$, while MatDot codes have $t = d = 1$ [13]. All the sub-matrices $\mathbf{A}_{ij}$ and $\mathbf{B}_{kl}$ have dimensions $T/t \times S/s$ and $S/s \times D/d$, respectively. The GPD code computes each block (i, j) of the product $\mathbf{C} = \mathbf{AB}$, namely $\mathbf{C}_{i,j} = \sum_{k=1}^{s} \mathbf{A}_{i,k} \mathbf{B}_{k,j}$, for $i = 1, \ldots, t$ and $j = 1, \ldots, d$, in a distributed fashion. This is done by means of polynomial encoding and polynomial interpolation. As we review next, the computation of block $\mathbf{C}_{i,j}$ can be interpreted as the evaluation of the middle sample of the convolution $\mathbf{c}_{i,j} = \mathbf{a}_i * \mathbf{b}_j$ between the block sequences $\mathbf{a}_i = [\mathbf{A}_{i,1}, \ldots, \mathbf{A}_{i,s}]$ and $\mathbf{b}_j = [\mathbf{B}_{s,j}, \ldots, \mathbf{B}_{1,j}]$. In fact, the s-th sample of the block sequence $\mathbf{c}_{i,j}$ equals $\mathbf{C}_{i,j}$, i.e., $[\mathbf{c}_{i,j}]_s = \mathbf{C}_{i,j}$. The computation is carried out distributively in the frequency domain by using z-transforms with different workers being assigned distinct samples in the frequency domain.

To elaborate, define the block sequence $\mathbf{a}$ obtained by concatenating the block sequences $\mathbf{a}_i$ as $\mathbf{a} = \{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_t\}$. Pictorially, a sequence $\mathbf{a}$ is obtained from the matrix $\mathbf{A}$ by

reading the blocks in the left-to-right top-to-bottom order, as seen in Fig. 2. We also introduce the longer time block sequence $\mathbf{b}$ as $\mathbf{b} = \{\mathbf{b}_1, \mathbf{0}, \mathbf{b}_2, \mathbf{0}, \ldots, \mathbf{b}_d\}$ with $\mathbf{0}$ being a block sequence of $s(t^*-1)$ all-zero block matrices with dimensions $S/s \times D/d$. The sequence $\mathbf{b}$ can be obtained from matrix $\mathbf{B}$ by following the bottom-to-top left-to-right order shown in Fig. 2 and by adding the all-zero block sequences between any two columns of $\mathbf{B}$.

In the frequency domain, the z-transforms of sequences $\mathbf{a}$ and $\mathbf{b}$ are obtained as

$$\mathbf{F}_{\mathbf{a}}(z) = \sum_{r=0}^{ts-1} [\mathbf{a}]_{r+1} z^r = \sum_{i=1}^{t} \sum_{j=1}^{s} \mathbf{A}_{i,j} z^{s(i-1)+j-1}, \tag{4}$$

$$\mathbf{F}_{\mathbf{b}}(z) = \sum_{r=0}^{s-1+ts(d-1)} [\mathbf{b}]_{r+1} z^r = \sum_{k=1}^{s} \sum_{l=1}^{d} \mathbf{B}_{k,l} z^{s-k+ts(l-1)}, \tag{5}$$

respectively. The master server evaluates the polynomials $\mathbf{F}_{\mathbf{a}}(z)$ and $\mathbf{F}_{\mathbf{b}}(z)$ in P non-zero distinct points $z_1, \ldots z_P \in \mathbb{F}$ and sends the corresponding linearly encoded matrices $\mathbf{A}_p = \mathbf{F}_{\mathbf{a}}(z_p)$ and $\mathbf{B}_p = \mathbf{F}_{\mathbf{b}}(z_p)$ to worker p. The encoding functions are hence given by the polynomial evaluations (4) and (5), for $z_1, \ldots, z_p$. Worker p computes the multiplication $\mathbf{F}_{\mathbf{a}}(z_p)\mathbf{F}_{\mathbf{b}}(z_p)$ and sends it to the master server. The master server computes the inverse z-transform for the received products $\{\mathbf{A}_p\mathbf{B}_p\}_{p\in\mathcal{P}_R} = \{\mathbf{F}_{\mathbf{a}}(z_p)\mathbf{F}_{\mathbf{b}}(z_p)\}_{p\in\mathcal{P}_R}$, obtaining the linear convolution $\mathbf{a} * \mathbf{b}$. From $\mathbf{a} * \mathbf{b}$, we can see that the master server is able to compute all the desired blocks $\mathbf{C}_{i,j}$ by reading the middle samples of the convolutions $\mathbf{c}_{i,j} = \mathbf{a}_i * \mathbf{b}_j$ from samples of the sequence $\mathbf{c} = \mathbf{a} * \mathbf{b}$ in the order $[\mathbf{c}]_{s-1} = \mathbf{C}_{1,1}, [\mathbf{c}]_{2s-1} = \mathbf{C}_{2,1}, \ldots, [\mathbf{c}]_{ts-1} = \mathbf{C}_{t,1}, [\mathbf{c}]_{s-1+t^*s} = \mathbf{C}_{1,2}, \ldots, [\mathbf{c}]_{ts-1+t^*s} = \mathbf{C}_{t,2}, \ldots$, and so on. Note that, in particular, the zero block subsequences added to sequence $\mathbf{b}$ ensure that no interference from the other convolutions $\mathbf{c}_{i',j'}$ affect the middle (s-th) sample of a convolution $\mathbf{c}_{i,j}$ with $i' \neq i$ and $j' \neq j$. To carry out the inverse transform, the master server needs to collect as many values $\mathbf{F}_{\mathbf{a}}(z_p)\mathbf{F}_{\mathbf{b}}(z_p)$ as there are samples of the sequence $\mathbf{a} * \mathbf{b}$, yielding the recovery threshold

$$P_R = tsd + s - 1. \tag{6}$$

Equivalently, in terms of the underlying polynomial interpretation, the master server needs to collect a number of evaluations of the polynomial $\mathbf{F}_{\mathbf{a}}(z)\mathbf{F}_{\mathbf{b}}(z)$, being equal to the degree of $\mathbf{F}_a(z)\mathbf{F}_b(z)$ plus one. This computation is of complexity order $\mathcal{O}(TDP_R \log^2(P_R))$ [13]. Furthermore, the communication load is given as

$$C_L = P_R \frac{TD}{td}, \tag{7}$$

where $TD/(td)$ is the size of each matrix $\mathbf{F}_{\mathbf{a}}(z)\mathbf{F}_{\mathbf{b}}(z)$.

III. Secure PolyDot Codes

In this section, we propose a novel extension of the GPD code that is able to ensure the secrecy constraint for any $P_C < P$. We also derive the corresponding achievable set of triples (P_C, P_R, C_L). As we will see, the projection of this set onto the plane defined by the condition $P_C = 0$ includes the set of pairs (P_R, C_L) in (6) and (7) obtained by the GPD code [14]. The proposed SGPD code augments matrices $\mathbf{A}$ and $\mathbf{B}$ by adding P_C random block matrices to $\mathbf{A}$ and $\mathbf{B}$, in a manner similar to prior works (see, e.g., [7], [9], [10]), yielding augmented matrices $\mathbf{A}^*$ and $\mathbf{B}^*$. However, as we will see, a direct application of the GPD construction to these matrices is suboptimal.

Therefore, we propose a novel way to construct sequences $\mathbf{a}^*$ and $\mathbf{b}^*$ from matrices $\mathbf{A}^*$ and $\mathbf{B}^*$ that enables the definition of a more efficient code by means of the z-transform approach discussed in the previous section. Based on the discussion in the previous section, this goal can be realized by decreasing the length of sequence $\mathbf{c}^* = \mathbf{a}^* * \mathbf{b}^*$, which can in turn be ensured by reducing the length of the sequence $\mathbf{b}^*$ for a given length of sequence $\mathbf{a}^*$. We accomplish this objective by *(i)* adaptively appending rows *or* columns with random elements to matrix $\mathbf{A}$ and correspondingly columns *or* rows to $\mathbf{B}$, which can reduce the recovery threshold; and *(ii)* modifying the zero padding procedure (see Fig. 2) for the construction of sequence $\mathbf{b}^*$. In order to account for *(i)*, we separately consider the two cases $s < t$ and $s \geq t$.

A. Secure Generalized PolyDot Code: The $s < t$ Case

When $s < t$, we augment input matrices $\mathbf{A}$ and $\mathbf{B}$ by adding

$$\Delta_{P_C} \triangleq \left\lceil \frac{P_C}{s} \right\rceil, \tag{8}$$

random row and column blocks to matrices $\mathbf{A}$ and $\mathbf{B}$, respectively. Accordingly, the $t^* \times s$ augmented block matrix $\mathbf{A}^*$ with $t^* = t + \Delta_{P_C}$, is obtained as

$$\mathbf{A}^* = \begin{bmatrix} \mathbf{A} \\ \mathbf{R} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{1,1} & \ldots & \mathbf{A}_{1,s} \\ \vdots & \ddots & \vdots \\ \mathbf{A}_{t,1} & \ldots & \mathbf{A}_{t,s} \\ \mathbf{R}_{1,1} & \ldots & \mathbf{R}_{1,s} \\ \vdots & \ddots & \vdots \\ \mathbf{R}_{\Delta_{P_C},1} & \ldots & \mathbf{R}_{\Delta_{P_C},s} \end{bmatrix}, \tag{9}$$

while the $s \times d^*$ augmented matrix $\mathbf{B}^* = [\mathbf{B}\ \mathbf{R}']$ with $d^* = d + \Delta_{P_C}$ is obtained as

$$\mathbf{B}^* = \begin{bmatrix} \mathbf{B}_{1,1} & \ldots & \mathbf{B}_{1,d} & \mathbf{R}'_{s,1} & \ldots & \mathbf{R}'_{s,\Delta_{P_C}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{B}_{s,1} & \ldots & \mathbf{B}_{s,d} & \mathbf{R}'_{1,1} & \ldots & \mathbf{R}'_{1,\Delta_{P_C}} \end{bmatrix}. \tag{10}$$

In (9) and (10), if s divides P_C, all block matrices $\mathbf{R}_{ij} \in \mathbb{F}^{\frac{T}{t}\times\frac{S}{s}}$ and $\mathbf{R}'_{ij} \in \mathbb{F}^{\frac{S}{s}\times\frac{D}{d}}$ are generated with i.i.d. uniform random elements in $\mathbb{F}$. Otherwise, if $\Delta_{P_C} - P_C/s > 0$, the last $s\Delta_{P_C} - P_C$ matrices in (9) with right-to-left ordering in the last row of $\mathbf{R}_{ij}$ and in (10) with top-to-bottom ordering in the last column of $\mathbf{R}'_{ij}$, resp., are all-zero block matrices.

In the SGPD construction the block sequence $\mathbf{a}^*$ is defined in the same way as in the conventional GPD, yielding $\mathbf{a}^* = \{\mathbf{a}_1, \ldots, \mathbf{a}_t, \mathbf{r}_1, \ldots, \mathbf{r}_{\Delta_{P_C}}\}$, where $\mathbf{r}_i$ is the i-th row of the block matrix $\mathbf{R}$, $i = 1, \ldots, \Delta_{P_C}$. We also define the time block sequence $\mathbf{b}^* = \{\mathbf{b}, \mathbf{r}'\}$ as $\mathbf{b}^* = \{\mathbf{b}_1, \mathbf{0}, \mathbf{b}_2, \mathbf{0}, \ldots, \mathbf{b}_d, \mathbf{0}, \mathbf{r}'_1, \mathbf{r}'_2, \ldots, \mathbf{r}'_{\Delta_{P_C}}\}$, where $\mathbf{0}$ is block

sequences of $s(t^*-1)$ all-zero block matrices, resp., with dimensions $S/s \times D/d$, while $\mathbf{r}'_j$ is the j-th column of the random matrix $\mathbf{R}'$. The key novel idea of this construction is that no zero matrices are introduced between columns of matrix $\mathbf{R}'$. As shown in Theorem 1 below, this construction allows the master server to recover all the desired submatrices $\mathbf{C}_{i,j}$ for $i = 1, \ldots, t$ and $j = 1, \ldots, d$ from the middle samples of the convolutions $\mathbf{c}_{i,j} = \mathbf{a}_i * \mathbf{b}_j$ (see Fig. 3 for an illustration).

Theorem 1. *For a given security level $P_C < P$, the proposed SGPD code achieves the recovery threshold P_R*

$$\begin{cases} tsd + s - 1, & \text{if } P_C = 0, \\ t^*s(d+1) + s\Delta_{P_C} - 1, & \text{if } P_C \geq 1 \text{ and } \Delta_{P_C} = \frac{P_C}{s}, \\ t^*s(d+1) - s\Delta_{P_C} + 2P_C - 1, & \text{if } P_C \geq 1 \text{ and } \Delta_{P_C} > \frac{P_C}{s}, \end{cases} \tag{11}$$

and the communication load in (7)*, where $t^* = t + \Delta_{P_C}$ and $d^* = d + \Delta_{P_C}$ for any integer values $t, s,$ and d such that $s < t$, $m = ts$, and $n = sd$.*

The proof is given in Appendix A.

Remark 1. *For $P_C \geq 1$ a direct application of the GPD construction in Fig. 2 yields a larger recovery threshold compared to* (11) *as*

$$P_R = \begin{cases} t^*sd^* + s - 1, & \text{if } \Delta_{P_C} = \frac{P_C}{s}, \\ dst^* + s - 1 - 2(s\Delta_{P_C} - P_C), & \text{if } \Delta_{P_C} > \frac{P_C}{s}. \end{cases}$$

B. Secure Generalized PolyDot Code: The $s \geq t$ Case

When $s \geq t$, we augment input matrices $\mathbf{A}$ and $\mathbf{B}$ by adding

$$\Delta'_{P_C} \triangleq \left\lceil \frac{P_C}{\min\{t, d\}} \right\rceil. \tag{12}$$

column and row blocks to matrices $\mathbf{A}$ and $\mathbf{B}$. This can be seen to yield a smaller recovery threshold. Accordingly, the $t \times s^*$ augmented block matrix $\mathbf{A}^* = [\mathbf{A}\ \mathbf{R}]$ with $s^* = s + \Delta'_{P_C}$, is obtained as

$$\mathbf{A}^* = \begin{bmatrix} \mathbf{A}_{1,1} & \ldots & \mathbf{A}_{1,s} & \mathbf{R}_{1,1} & \ldots & \mathbf{R}_{1,\Delta'_{P_C}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{t,1} & \ldots & \mathbf{A}_{t,s} & \mathbf{R}_{t,1} & \ldots & \mathbf{R}_{t,\Delta'_{P_C}} \end{bmatrix}, \tag{13}$$

while the $s^* \times d$ augmented block matrix $\mathbf{B}^*$ is defined as

$$\mathbf{B}^* = \begin{bmatrix} \mathbf{R}' \\ \mathbf{B} \end{bmatrix} = \begin{bmatrix} \mathbf{R}'_{\Delta'_{P_C},1} & \ldots & \mathbf{R}'_{\Delta'_{P_C},d} \\ \vdots & \ddots & \vdots \\ \mathbf{R}'_{1,1} & \ldots & \mathbf{R}'_{1,d} \\ \mathbf{B}_{1,1} & \ldots & \mathbf{B}_{1,d} \\ \vdots & \ddots & \vdots \\ \mathbf{B}_{s,1} & \ldots & \mathbf{B}_{s,d} \end{bmatrix}. \tag{14}$$

As for (13) and (14), if $\Delta'_{P_C} - P_C/\min\{t,d\} > 0$, the last $s\Delta'_{P_C} - P_C$ block matrices in (13) with bottom-to-top right-to-left ordering in $\mathbf{R}$ and in (14) with right-to-left top-to-bottom ordering in $\mathbf{R}'$, resp., are all-zero block matrices. The construction of sequences $\mathbf{a}^*$ and $\mathbf{b}^*$ is analogous to the GPD in non-secure case. In particular, the time block sequence $\mathbf{a}^*$ is $\mathbf{a}^* = \{\mathbf{a}_1, \mathbf{r}_1, \mathbf{a}_2, \mathbf{r}_2, \ldots, \mathbf{a}_t, \mathbf{r}_t\}$, whereas the block sequence

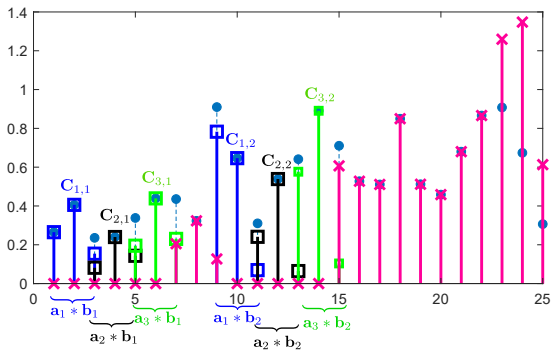

Fig. 3: Illustration of correct recovery threshold for $t = 3, s = 2, d = 2$, and $P_C = 2$. Dashed blue stems with filled markers represent the overall convolution $\mathbf{c}^*$. Individual convolutions $\mathbf{c}_{i,j}$ are shown in different colors with square markers. Contributions from one or both random matrices are shown as red crosses. The desired submatrices $\mathbf{C}_{i,j}$ are seen to equal the corresponding samples from the sequence $\mathbf{c}^*$, associated with the center points of the individual convolutions.

$\mathbf{b}^*$ is defined as $\mathbf{b}^* = \{\mathbf{b}_1, \mathbf{r}'_1, \mathbf{0}, \mathbf{b}_2, \mathbf{r}'_2, \mathbf{0}, \ldots, \mathbf{b}_d, \mathbf{r}'_d\}$, Here, $\mathbf{0}$ is a block sequence of $(t-1)s^*$ all-zero block matrices with dimensions $S/s \times D/d$.

Theorem 2. *For a given security level $P_C < P$, the proposed SGPD code achieves the recovery threshold*

$$P_R = \begin{cases} s^*(t^2+1) - 3, & \text{if } \Delta'_{P_C} > \frac{P_C}{s} \text{ and } t = d \\ tds^* + s^* - 1, & \text{otherwise}, \end{cases} \tag{15}$$

and the communication load in (7)*, where $s^* = s + \Delta'_{P_C}$ for any integer values $t, s,$ and d such that $s \geq t$, $m = ts$, and $n = sd$.*

The proof is omitted due to space reasons.

Example 1. *We now provide some numerical results of the proposed SGPD construction. We set $P = 3000$ and $m = n = 36$. The trade-off between communication load C_L and recovery threshold P_R for both non-secure conventional GPD codes ($P_C = 0$) and SGPD codes with $P_C = 11$ and $P_C = 29$ is illustrated in Fig. 4. The figure quantifies the loss in terms of achievable pairs (P_R, C_L) that is caused by the security constraint.*

APPENDIX A
PROOF OF THEOREM 1

The z-transforms of $\mathbf{a}^*$ and $\mathbf{b}^*$ are given as

$$\mathbf{F}_{\mathbf{a}^*}(z) = \underbrace{\sum_{i=1}^{t}\sum_{j=1}^{s} \mathbf{A}^*_{i,j} z^{s(i-1)+(j-1)}}_{\triangleq\ \mathbf{F}_1(z)} + \underbrace{\sum_{i=t+1}^{t^*}\sum_{j=1}^{s} \mathbf{A}^*_{i,j} z^{s(i-1)+j-1}}_{\triangleq\ \mathbf{F}_2(z)}, \tag{16}$$

$$\mathbf{F}_{\mathbf{b}^*}(z) = \underbrace{\sum_{k=1}^{s}\sum_{l=1}^{d} \mathbf{B}^*_{k,l} z^{s-k+t^*s(l-1)}}_{\triangleq\ \mathbf{F}_3(z)} + \underbrace{\sum_{k=1}^{s}\sum_{l=d+1}^{d^*} \mathbf{B}^*_{k,l} z^{t^*sd+s(l-d)-k}}_{\triangleq\ \mathbf{F}_4(z)}. \tag{17}$$

To reconstruct all blocks $\mathbf{C}_{i,j}$ of matrix $\mathbf{C} = \mathbf{AB}$, the master server carries out polynomial interpolation on the polynomial $\mathbf{F}_{\mathbf{a}^*}(z)\mathbf{F}_{\mathbf{b}^*}(z)$, or equivalently computes the inverse z-transform, upon receiving a number of multiplication results

equal to at least the length of the sequence $\mathbf{c}^* = \mathbf{a}^* * \mathbf{b}^*$. As we outline next, the (i,l) block $\mathbf{C}_{i,l} = \sum_{r=1}^{s} \mathbf{A}_{i,r}\mathbf{B}_{r,l}$, for all $i = 1,\dots,t$ and $l = 1,\dots,d$, of matrix $\mathbf{C} = \mathbf{AB}$ can be seen equal to the $(si-1+(l-1)t^*s)$-th sample of the convolution $\mathbf{c}^* = \mathbf{a}^* * \mathbf{b}^*$ (see Fig. 3).

Note that, by the properties of GPD codes, matrix $\mathbf{C}_{i,l}$ is the coefficient of the monomial $z^{si-1+(l-1)t^*s}$ in $\mathbf{F}_1(z)\mathbf{F}_3(z)$. We now need to show that no other contribution to this term arises from the products $\mathbf{F}_1(z)\mathbf{F}_4(z)$, $\mathbf{F}_2(z)\mathbf{F}_3(z)$, and $\mathbf{F}_2(z)\mathbf{F}_4(z)$. The terms in the product $\mathbf{F}_1(z)\mathbf{F}_4(z)$ have exponents $(t^*sd+s(i-1)+s(l-d)-1)$, for $i = 1,\dots,t$ and $l = d+1,\dots,d^*$, which do not include the desired values $(si-1+(l-1)t^*s)$ for $i = 1,\dots,t$ and $l = 1,\dots,d$. A similar discussion applies to the product $\mathbf{F}_2(z)\mathbf{F}_3(z)$, whose exponents are $(s(i+t^*l-t^*)-1)$, for $i = t+1,\dots,t^*$ and $l = 1,\dots,d$, and $\mathbf{F}_2(z)\mathbf{F}_4(z)$, whose exponents are $(t^*sd+s(i-1)+s(l-d)-1)$, for $i = t+1,\dots,t^*$ and $l = d+1,\dots,d^*$.

In order to recover the convolution $\mathbf{c}^*$, the master server needs to collect a number of values of the product $\mathbf{F}_\mathbf{a}(z)\mathbf{F}_\mathbf{b}(z)$ equal to the length of the sequence $\mathbf{c}^*$, which can be computed as the degree $\deg(\mathbf{F}_\mathbf{a}(z)\mathbf{F}_\mathbf{b}(z))+1$, with $\deg(\mathbf{F}_\mathbf{a}(z)\mathbf{F}_\mathbf{b}(z))$

$$\begin{cases} t^*s(d+1)+s\Delta_{P_C}-1, & \text{if } \Delta_{P_C} = \frac{P_C}{s}, \\ dst^* - s\Delta_{P_C} + 2P_C + t - 2, & \text{if } \Delta_{P_C} > \frac{P_C}{s}, \end{cases} \tag{18}$$

which for $P_C \geq 1$ implies the recovery threshold P_R in (11). The communication load C_L in (7) follows from the fact that there are $TD/(td)$ entries in $\mathbf{F}_{\mathbf{a}^*}(z_p)\mathbf{F}_{\mathbf{b}^*}(z_p)$, $p \in [1, P_R]$.

The security constraint (1) can be proved in a manner similar to [8] by the following steps:

$$\begin{aligned} I(\mathbf{A},\mathbf{B};\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P}) &= H(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P}) - H(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P}|\mathbf{A},\mathbf{B}) \\ &\overset{(a)}{=} H(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P}) - H(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P}|\mathbf{A},\mathbf{B}) \\ &\quad + H(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P}|\mathbf{A},\mathbf{B},\mathbf{R}_1,\dots,\mathbf{R}_{P_C},\mathbf{R}'_1,\dots,\mathbf{R}'_{P_C}) \\ &\overset{(b)}{=} H(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P}) - H(\mathbf{R}_1,\dots,\mathbf{R}_{P_C},\mathbf{R}'_1,\dots,\mathbf{R}'_{P_C}) \\ &\overset{(c)}{\leq} \sum_{p=1}^{P_C} H(\mathbf{A}_p) + \sum_{p=1}^{P_C} H(\mathbf{B}_p) - P_C\frac{TS}{m}\log|\mathbb{F}| - P_C\frac{SD}{n}\log|\mathbb{F}| \\ &= P_C\frac{TS}{m}\log|\mathbb{F}| + P_C\frac{SD}{n}\log|\mathbb{F}| - P_C\frac{TS}{m}\log|\mathbb{F}| \\ &\quad - P_C\frac{SD}{n}\log|\mathbb{F}| = 0 \end{aligned}$$

where (a) follows from the definition of encoding functions, since $\mathbf{A}_\mathcal{P}$ is a deterministic function of $\mathbf{A}$ and $\mathbf{R}_p$ and $\mathbf{B}_\mathcal{P}$ is a deterministic function of $\mathbf{B}$ and $\mathbf{R}'_p$, resp., for all $p = 1,\dots,P_C$; (b) follows from (16) and (17), since from P_R polynomial evaluations $\mathbf{A}_\mathcal{P}$ and $\mathbf{B}_\mathcal{P}$ in (16) and (17) we can recover $2P_C$ unknowns when the coefficients $\mathbf{A}_{i,j}$ and $\mathbf{B}_{k,l}$ are known, given that we have $P_R \geq 2P_C$; (c) follows since $\mathbf{R}_p$ and $\mathbf{R}'_p$ are independent uniformly distributed entries and by upper bounding the joint entropy.

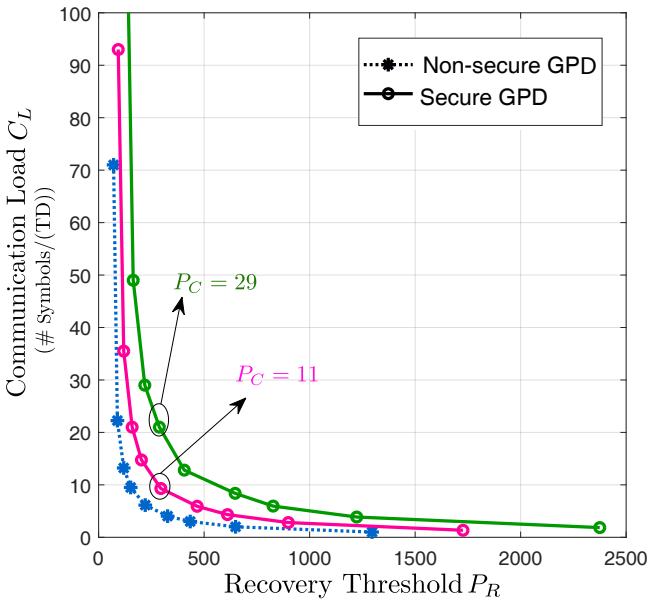

Fig. 4: Communication load C_L versus recovery threshold P_R for both GPD and SGPD codes when $P = 3000$ and $m = n = 36$.

REFERENCES

[1] M. Janzamin, H. Sedghi, and A. Anandkumar, "Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods," *arXiv preprint, arXiv:1506.08473*, 2015.

[2] G. Joshi, E. Soljanin, and G. Wornell, "Efficient redundancy techniques for latency reduction in cloud systems," *ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS)*, vol. 2, no. 2, pp. 12:1–12:30, Apr. 2017.

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

[4] Q. Yu, M. Maddah-Ali, and S. Avestimehr, "Polynomial codes: an optimal design for high-dimensional coded matrix multiplication," in *Proc. Advances in Neural Inform. Processing Systems*, Dec. 2017, pp. 4403–4413.

[5] S. Li, M. A. Maddah-Ali, Q. Yu, and A. S. Avestimehr, "A fundamental tradeoff between computation and communication in distributed computing," *IEEE Trans. on Inform. Theory*, vol. 64, no. 1, pp. 109–128, Sep. 2017.

[6] M. Aliasgari, J. Kliewer, and O. Simeone, "Coded computation against processing delays for virtualized cloud-based channel decoding," *IEEE Trans. on Commun.*, vol. 67, no. 1, pp. 28–38, Jan. 2019.

[7] Q. Yu, N. Raviv, J. So, and A. S. Avestimehr, "Lagrange coded computing: Optimal design for resiliency, security and privacy," *arXiv preprint arXiv:1806.00939*, 2018.

[8] W.-T. Chang and R. Tandon, "On the capacity of secure distributed matrix multiplication," *arXiv preprint, arXiv:1806.00469*, 2018.

[9] J. Kakar, S. Ebadifar, and A. Sezgin, "Rate-efficiency and straggler-robustness through partition in distributed two-sided secure matrix computation," *arXiv preprint, arXiv:1810.13006*, 2018.

[10] R. G. D'Oliveira, S. E. Rouayheb, and D. Karpuk, "GASP codes for secure distributed matrix multiplication," *arXiv preprint, arXiv:1812.09962*, 2018.

[11] K. Lee, C. Suh, and K. Ramchandran, "High-dimensional coded matrix multiplication," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, Jun. 2017, pp. 2418–2422.

[12] M. Fahim, H. Jeong, F. Haddadpour, S. Dutta, V. Cadambe, and P. Grover, "On the optimal recovery threshold of coded matrix multiplication," in *Proc. Communication, Control, and Computing (Allerton)*, Oct. 2017, pp. 1264–1270.

[13] S. Dutta, M. Fahim, F. Haddadpour, H. Jeong, V. Cadambe, and P. Grover, "On the optimal recovery threshold of coded matrix multiplication," *arXiv preprint, arXiv:1801.10292*, 2018.

[14] S. Dutta, Z. Bai, H. Jeong, T. M. Low, and P. Grover, "A unified coded deep neural network training strategy based on generalized polydot codes for matrix multiplication," *arXiv preprint arXiv:1811.10751*, 2018.

[15] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Straggler mitigation in distributed matrix multiplication: Fundamental limits and optimal coding," *arXiv preprint, arXiv:1801.07487*, 2018.