Analog Privacy-Preserving Coded Computing

Mahdi Soleymani*, Hessam Mahdavifar*, and A. Salman Avestimehr†
*Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109, USA
†Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90089 USA
Emails: mahdy@umich.edu, hessam@umich.edu, avestimehr@ee.usc.edu

***Abstract*—The state-of-the-art approaches to privacy-preserving coded computing rely on quantizing the data into a finite field, so that Shamir's secret sharing can be employed. Such coded computing solutions, however, are not properly scalable with the size of dataset, mainly due to computation overflows. To address such a critical issue, we propose a novel extension of certain coded computing schemes to the analog domain. This includes distributed polynomial evaluation and Lagrange coded computing (LCC) that are widely used in the literature. All the operations in the proposed protocols are done over the infinite fields of $\mathbb{R}/\mathbb{C}$ but for practical implementations floating-point numbers are used. We characterize the *privacy* of data in our proposed protocols, against any subset of colluding servers up to a certain size, in terms of the distinguishing security (DS) and the mutual information security (MIS) metrics. Also, the *accuracy* of outcome is characterized in a practical setting assuming operations are performed using floating-point numbers. Consequently, fundamental trade-offs between the accuracy of the outcome and their privacy level are observed in the analog domain and are numerically evaluated. Moreover, we implement analog LCC (ALCC) to perform matrix-matrix multiplication over a batch of matrices. It is observed that ALCC is superior compared to LCC, implemented using fixed-point numbers, assuming both schemes use an equal number of bits to represent data symbols.**

I. Introduction

There has been a growing interest in recent years towards performing computational tasks across networks of computational servers by utilizing their computational power in a parallel fashion [1]–[5]. Computations over massive datasets need to be carried out at an unprecedented scale that entails solutions scalable with the size of datasets associated with a wide range of problems including machine learning [6], optimization [7], etc. A well-established network architecture to perform such tasks in a distributed fashion consists of a *master* node together with a set of servers having communication links only with the master node [3], [4]. In such systems, a dataset is dispersed among the servers across the network to perform a certain computational task over the dataset. The master node then aggregates the results in order to recover the desired outcome.

Dispersing data across a network gives rise to several fundamental challenges in practice. One of the major concerns in such systems is to keep the data *private* as the computational tasks often involve sensitive data such as patients recordings, financial transactions, etc [8]–[10]. The servers are often assumed to be *honest-but-curious*, i.e., they do not deviate from the protocol but may accumulate the shares of data they receive and try to deduce information about the data. In such settings, the challenge is to utilize the computational power of the nodes while ensuring that *almost* no information about the dataset is revealed to them. Furthermore, this restriction is often extended to preserving the privacy against any subset of colluding nodes up to a certain size.

Several security metrics are considered in different contexts to measure privacy/security of data. This includes semantic security (SS) and distinguishing security (DS) in the cryptography literature [11], mutual information security (MIS) in communication settings [12], differential security in machine learning [13], etc. From the information-theoretic perspective, the *perfect* privacy condition in a distributed computation setting is that *no information* is leaked about the dataset to any of the servers/subsets of colluding servers up to a certain size. To this end, Shamir's seminal secret sharing scheme is the main building block in protocols providing *perfect* privacy in these settings [14]. In such protocols, the data symbols are always assumed to be elements of a finite field $\mathbb{F}_p$ leading to perfect privacy guarantees. However, this often comes at the expense of substantial accuracy losses due to fixed-point representation of the data and computation overflows. Especially, this becomes a major barrier in scalability of such protocols with respect to the dataset size.

In this paper, we provide a framework to construct the counterpart of Shamir's secret sharing scheme in the analog domain. This framework is then used to construct a privacy-preserving distributed polynomial evaluation protocol over real/complex datasets. Moreover, we further elaborate these ideas to extend the privacy-preserving LCC scheme to the analog domain and refer to it as analog LCC (ALCC). It is assumed that all the servers are honest-but-curious. All the operations in the proposed scheme are done over the infinite fields of $\mathbb{R}/\mathbb{C}$ but for practical implementations floating-point numbers are used. The proposed ALCC protocol enables *privately* evaluating a polynomial function over a batch of real/complex-valued dataset in parallel. We characterize the performance of the proposed schemes in terms of the *accuracy* of their outcome, when operations are performed using standard floating-point numbers, and the *privacy* of data in terms of the DS and MIS metrics when any subset of servers up to a certain size can collude. Furthermore, a fundamental trade-off between the accuracy of the outcome of the introduced protocols in the analog domain and their privacy level is observed and is numerically evaluated, in terms of various parameters of the schemes, when implemented using the floating-point numbers.

For ALCC in particular, we illustrate that the choice of certain parameters of Lagrange monomials leads to another new trade-off between accuracy and privacy which is specific to ALCC and does not have a counterpart in either our proposed polynomial evaluation protocol or LCC with fixed-point implementation over finite fields [4]. Hence, one has to carefully pick these parameters besides picking parameters of noise terms added in the protocol in order to avoid unnecessarily compromising accuracy/privacy in practice. Also, experiments are shown in which ALCC is implemented to perform matrix-matrix multiplication over a batch of matrices. The results indicate the superiority of the proposed

This material is based upon work supported by Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001117C0053 and FA8750-19-2-1005, ARO award W911NF1810400, NSF grants CCF–1763348, CCF–1909771, CCF–1941633, CCF-1703575 and CCF-1763673, ONR Award No. N00014-16-1-2189, and a gift from Intel/Avast/Borsetta via the PrivateAI institute. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

ALCC compared to the state-of-the-art LCC implemented using fixed-point numbers assuming both schemes use an equal number of bits to represent each data symbol.

II. Privacy-Preserving Polynomial evaluation

Consider a setup with N computation servers/parties indexed by $1, 2, \ldots, N$. Given a data symbol s, also referred to as a *secret*, a D-degree polynomial function of s denoted by $f(s) \stackrel{\text{def}}{=} \sum_{i=0}^{D} f_i s^i$ needs to be computed by utilizing the computational power of the parties, while the secret remains *private* assuming up to t parties can collude. The secret s is an instance of a continuous random variable Z taking values in $[-r, r]$. No further assumption is made on the probability distribution of Z.

In the considered protocol, given the secret s the polynomial $p(x)$ is constructed as follows: $p(x) \stackrel{\text{def}}{=} s + \sum_{j=1}^{t} n_j x^j$, where n_j's are i.i.d., drawn from a zero-mean circular symmetric complex Gaussian distribution with standard deviation $\frac{\sigma_n}{\sqrt{t}}$, denoted by $\mathcal{N}(0, \frac{\sigma_n^2}{t})$, where t is the maximum number of colluding parties. For evaluating the precision of the protocol in practice, the distribution of n_i's will be truncated, i.e., it is assumed that they are drawn from a truncated Gaussian distribution with a maximum absolute value, denoted by m, for $m \in \mathbb{R}^+$. The shares of the computation parties consist of the evaluation of $p(x)$ over certain complex-valued evaluation points $\omega_1, \ldots, \omega_N$, i.e., $y_i = p(\omega_i)$ is given to server i, for $i \in [N]$, where $[N] = \{1, 2, \ldots, N\}$. These equations are written in the matrix form as $\boldsymbol{y}_{N\times 1} = \boldsymbol{A}_{N\times(t+1)}\boldsymbol{x}_{(t+1)\times 1}$, where $\boldsymbol{x} = (s, n_1, \ldots, n_t)^{\mathrm{T}}$, $\boldsymbol{y} = (y_1, \ldots, y_N)^{\mathrm{T}}$, and $\boldsymbol{A}$ is the Vandemonde matrix associated with ω_i's. Then server i computes $z_i = f(y_i)$ and returns the result to the master node. The master then aims to recover $f(s)$. Let

$$[\boldsymbol{B}]_{ij} = \omega_i^{j-1}, \tag{1}$$

for $\boldsymbol{B}_{N\times(\tilde{D}+1)}$ with $\tilde{D} = Dt$. Then the system of equations

$$\boldsymbol{z}_{N\times 1} = \boldsymbol{B}_{N\times(\tilde{D}+1)}\boldsymbol{a}_{1\times(\tilde{D}+1)}, \tag{2}$$

where $\boldsymbol{z} = (z_1, \ldots, z_n)$ and $\boldsymbol{a}$ represents coefficients of $f(p(x))$, can be solved for $\boldsymbol{a}$ in order to recover $f(s) = f(p(0))$. Note that $N \geqslant \tilde{D} + 1$ is the necessary and sufficient condition on the number of parties in order to guarantee a successful interpolation of $f\big(p(x)\big)$, which is of degree $\tilde{D}$. Equivalently, it is the necessary and sufficient condition for recovery of $\boldsymbol{a}$ in (2). Throughout the rest of this section, it is assumed that $N = \tilde{D} + 1$, implying that all shares y_i's are needed to be returned to the master node for a successful recovery of the computation. Note that the master node does not need to compute the entire $\boldsymbol{a}$ in (2) and is only interested in recovering $f(s)$, the first entry of $\boldsymbol{a}$. Let $\tilde{\boldsymbol{b}}$ denote the first row of $\boldsymbol{B}^{-1}$, which is well-defined due to $\boldsymbol{B}$ being a Vandermonde matrix. Then the master node only needs to compute $\tilde{\boldsymbol{b}}\boldsymbol{z}$ to recover $f(s)$. Since ω_i's are fixed, $\tilde{\boldsymbol{b}}$ is computed once and is used every time the protocol is run.

Next, accuracy of the computation outcome of the proposed polynomial evaluation protocol is characterized in terms of other parameters of the protocol. In theory, if all the computations are done over the complex numbers with infinite precision, then $f(s)$ is computed accurately. In practice, data is represented using a finite number of bits, either as fixed point or floating point. Floating-point representation consists of a fixed-precision part and an exponent part specifying how the fixed-precision part is scaled. Let v denote the number of precision bits in the floating-point representation, i.e., the v most significant bits are kept in the fixed-precision part. Let also q denote the number of bits used to represent the power of the exponent part in the floating-point representation.

In general, in a system of linear equations $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}$, where $\boldsymbol{x}$ is the vector of unknown variables, the perturbation in the solution caused by the perturbation in $\boldsymbol{y}$ is characterized as follows. Let $\hat{\boldsymbol{y}}$ denote a noisy version of $\boldsymbol{y}$, where the noise can be caused by round-off errors, truncation, etc. Let also $\hat{\boldsymbol{x}}$ denote the solution to the considered linear system when $\boldsymbol{y}$ is replaced by $\hat{\boldsymbol{y}}$. Let $\Delta\boldsymbol{x} \stackrel{\text{def}}{=} \hat{\boldsymbol{x}} - \boldsymbol{x}$ and $\Delta\boldsymbol{y} \stackrel{\text{def}}{=} \hat{\boldsymbol{y}} - \boldsymbol{y}$ denote the perturbation, also referred to as error, in $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively. Then the relative perturbations of $\boldsymbol{x}$ is bounded in terms of that of $\boldsymbol{y}$ as follows [15]:

$$\frac{\|\Delta\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leqslant \kappa_{\boldsymbol{A}} \frac{\|\Delta\boldsymbol{y}\|}{\|\boldsymbol{y}\|}, \tag{3}$$

where $\kappa_{\boldsymbol{A}}$ is the condition number of $\boldsymbol{A}$ and $\|\cdot\|$ denotes the l^2-norm. As mentioned earlier, the Gaussian distribution of n_i's is truncated in practice. This together with (3) are used to provide a deterministic (non-probabilistic) guarantee on the accuracy of the computation result expressed in the following theorem. It is assumed that the computations at the master and servers do not impose any errors other than precision loss due to finite representation of the results.

Theorem 1: Let $\Delta f(s)$ denote the perturbation of $f(s)$ in the protocol discussed in this section. Let $mt + r \geqslant 1$ and $c \stackrel{\text{def}}{=} \sum_{i=0}^{D} |f_i|$. Then,

$$\Delta f(s) \leqslant c\sqrt{N}(mt + r)^D \frac{\kappa_{\boldsymbol{B}}}{\lambda_{\min}} 2^{-(v+1)}, \tag{4}$$

where t is the maximum number of colluding parties, m is the truncation parameter of the Gaussian distribution, $\kappa_{\boldsymbol{B}}$ and λ_{min} respectively are the condition number and minimum singular value of $\boldsymbol{B}$ given in (1), v is the number of precision bits, and r is the bound on the absolute value of the secret. In particular, by setting $\omega_i = \exp\left(\frac{2\pi j i}{N}\right)$ for $i \in [N]$ and $j^2 = -1$, we have

$$\Delta f(s) \leqslant c\sqrt{N}(mt + r)^D 2^{-(v+1)}. \tag{5}$$

The proofs of results in this section are omitted due to space constraints and can be found in [16].

Next, we ensure the privacy of data against any subset of t colluding computational parties. To this end, an upper bound on the amount of information revealed about the data/secret to the colluding parties is derived. Let $T = \{i_1, \ldots, i_t\}$ denote the set of indices for the colluding parties. Then the MIS metric is defined as the mutual information between S and all shares Y_i's for $i \in T$, in the worst case, i.e.,

$$\eta_c \stackrel{\text{def}}{=} \max_{T} I(S; Y_{i_1}, \ldots, Y_{i_t}). \tag{6}$$

DS metric, denoted by η_s, is another metric for security which is defined using the *total variation* (TV) distance metric $D_{\text{TV}}(.,.)$. In general, for any two probability measures P_1 and P_2 on a σ-algebra $\mathcal{F}$, $D_{\text{TV}}(P_1, P_2)$ is defined as $\sup_{A\in\mathcal{F}} |P_1(A) - P_2(A)|$.

$\log_{10}(\sigma_n)$	5	11	18
$\log_{10}(\Delta f(s))$	-9.80	-4.80	0.196
$\log_{10}(\eta_s)$	-2.36	-7.35	-12.4

TABLE I: Demonstration of the trade-off between DS security metric and accuracy. The upper bound on η_s in (9) is calculated versus the upper bound on $\Delta f(s)$ obtained in Theorem 1 at $\sigma_n = 10^5, 10^{10}, 10^{18}$. Other parameters are $c = 1$, $t = 1$, $D = 1$, $\alpha = 10$, $r = 255$ and $v = 52$.

In particular, it is defined in our protocol as follows:

$$\eta_s \stackrel{\text{def}}{=} \max_i \sup_{s_1,s_2\in\mathbb{D}_S} D_{\text{TV}}(P_{Y_{i_1},\cdots,Y_{i_t}|S=s_1}, P_{Y_{i_1},\cdots,Y_{i_t}|S=s_2}), \tag{7}$$

where $\mathbb{D}_S$ is the support of S. It is known that the MIS and DS metrics can be directly related to each other over the space of discrete/continuous random variables [17], [18]. In particular, it is shown that:

$$\eta_s \leqslant \sqrt{2\eta_c}. \tag{8}$$

In the following theorem η_c is upper bounded using the known results on the capacity of a single-input multiple-output (SIMO) channel under equal power constraints [19] . Consequently, it implies an upper bound on η_s as well due to (8).

Theorem 2: For the polynomial evaluation scheme proposed in this section we have

$$\eta_c \leqslant \log_2(1+\frac{r^2t^2}{\sigma_n^2}), \text{ and, } \eta_s \leqslant \sqrt{2\log_2(1+t^2\frac{r^2}{\sigma_n^2})}. \tag{9}$$

Remark 1: Comparing the results of Theorem 1 and Theorem 2 indicates that increasing the variance of the noise σ_n improves the privacy but at the same time reduces the accuracy of the computations. This demonstrates a fundamental trade-off between the privacy and the accuracy of the proposed protocol in the analog domain.

The results of Theorem 2 are derived by assuming the additive noise terms n_j's are drawn from a Gaussian distribution. While this assumption is valid in theory, such terms need to be truncated in practice as they can not be arbitrarily large. Furthermore, as shown earlier in this section, in order to provide guarantees on the accuracy of the computations, n_j's need to be bounded, i.e., $|n_j| \leqslant m$ for some $m \in \mathbb{R}^+$. In the following theorem, we extend the results on bounding the DS security metric in the proposed protocol to the case where n_j's are drawn from a truncated Gaussian probability distribution. To simplify the computation, it is assumed that the truncation threshold is $m = \alpha\frac{\sigma_n}{\sqrt{t}}$, for some $\alpha \in \mathbb{R}^+$.

Theorem 3: The DS metric for the case where n_j's are drawn from a truncated Gaussian distribution with truncation level $\alpha\frac{\sigma_n}{\sqrt{t}}$ satisfies the following inequality:

$$\eta_s' \leqslant \frac{1}{\rho}\eta_s + \frac{1}{\rho}(2\exp(-\frac{1}{2}(\alpha - \frac{2r\sqrt{t}}{\sigma_n})^2))^t,$$

where $\rho = (1-2\exp(-\frac{\alpha^2}{2}))^t$.

Theorem 3 implies that picking, for instance, $\alpha = 10$ with $t = 10$, and already having a very small $\frac{r}{\sigma_n}$ is sufficient to obtain almost the same bound on the DS metric as in the case where the noise terms are not truncated. Hence, truncation of the noise terms does not compromise the privacy of data in the protocol as long as α is picked sufficiently large.

III. Analog Lagrange Coded Computing

In this section, we extend the privacy-preserving LCC scheme to the analog domain and refer to it as analog LCC (ALCC). We also analyze the accuracy and privacy of ALCC using the framework developed in Section II.

Consider a dataset $\boldsymbol{X} = (\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k)$ with $\boldsymbol{X}_i \in \mathbb{R}^{m\times n}$ for all $i \in [k]$, where $[k]$ denotes $\{1, 2, \ldots, k\}$. Each entry of $\boldsymbol{X}_i$'s is an independent realization of the random variable S. We consider the problem of evaluating a polynomial $f : \mathbb{R}^{m\times n} \to \mathbb{R}^{u\times h}$ over the dataset $\boldsymbol{X}$ while keeping the *privacy* of $\boldsymbol{X}$. More specifically, we say $f(\cdot)$ is a D-degree polynomial function if all entries of the output matrix are multivariate polynomial functions of the entries of the input with total degree at most D.

The distributed computing setup is the same as that of polynomial evaluation scheme, described in Section II. The goal of the master node in ALCC is to compute $f(\boldsymbol{X}_i)$ for all $i \in [k]$, where f is a degree-D polynomial, using the computational power of the parties. This is done in such a way that the dataset is kept *private* from the parties assuming up to t of them can collude. Note that this setup is similar to the one considered for LCC in [4] with the main difference that in [4] the dataset and all the computations are assumed to be over a finite field.

Next we discuss the encoding process in ALCC. Let $\boldsymbol{N}_1, \ldots, \boldsymbol{N}_t$ denote random matrices with i.i.d. entries drawn from $\mathcal{N}(0, \frac{\sigma_n^2}{t})$. Let γ and ω denote the N-th and the $(k+t)$-th root of unity, respectively. In other words, $\gamma = \exp(\frac{2\pi j}{N})$ and $\omega = \exp(\frac{2\pi j}{k+t})$, where $j^2 = -1$. In ALCC, the Lagrange polynomial is constructed as

$$u(z) = \sum_{j=1}^{k} \boldsymbol{X}_j l_j(z) + \sum_{j=k+1}^{k+t} \boldsymbol{N}_{j-k} l_j(z) \tag{10}$$

where $l_j(z) \stackrel{\text{def}}{=} \prod_{l\in[k+t]\setminus j} \frac{z-\beta_l}{\beta_j-\beta_l}$ for all $j \in [k+t]$, are Lagrange monomials. Furthermore, the parameters β_j's are picked to be equally spaced on the circle of radius β centered around 0 in the complex plane, for some $\beta \in \mathbb{R}$, i.e.,

$$\beta_j = \beta\omega^{j-1}. \tag{11}$$

The shares of encoded dataset to be distributed to the servers consist of the evaluation of $u(z)$ over the N-th roots of unity in the complex plane, i.e., $\boldsymbol{Y}_i = u(\alpha_i)$, where

$$\alpha_i = \gamma^{i-1}, \tag{12}$$

is sent to node i, for $i \in [N]$.

Next, we discuss the decoding step during which the master node recovers the desired outcome by collecting and processing the results returned by a sufficient number of servers. The i-th server computes $f(\boldsymbol{Y}_i)$ and returns the result back to the master node. The master node then recovers $f(\boldsymbol{X}_i)$, for $i \in [k]$, in two steps. In the first step, it recovers the polynomial $f(u(z))$ by using the results returned from at least $(k+t-1)D+1$ servers. Note that this is the minimum number of returned evaluations needed to guarantee a successful interpolation of $f(u(z))$ since $f(u(z))$ has degree $(k+t-1)D$. For ease of notation, let $\tilde{D} = (k+t-1)D$. In the second step, to recover $f(\boldsymbol{X}_i)$'s, the master node computes $f(\beta_j)$ for $j \in [k]$.

The ALCC protocol can also take into account the issue regarding stragglers same as how it is done in LCC [4]. Let the maximum number of stragglers be denoted by $\bar{s}$. Hence, the number of computational parties is assumed to be $N = \tilde{D} + \bar{s} + 1$.

The decoder's task is to interpolate the polynomial $f(u(z))$ followed by evaluating it over α_i's, for $i \in [N]$. Let $f(u(z)) = \sum_{i=0}^{\tilde{D}} \boldsymbol{V}_i z^i$, with $\boldsymbol{V}_i \in \mathbb{R}^{u \times h}$. Let $A = \{i_1, \ldots, i_{\tilde{D}+1}\}$ denote the indices of the *non-straggler* users. The interpolation step at the decoder is equivalent to inverting a $(\tilde{D}+1) \times (\tilde{D}+1)$ Vandermonde matrix $\boldsymbol{C}$ where

$$[\boldsymbol{C}]_{kl} = \gamma^{(l-1)i_k}. \tag{13}$$

Let c denote the maximum absolute value of the coefficients of $f_{ij}(\cdot)$ for all $i \in [u]$ and $j \in [h]$, $c \stackrel{\text{def}}{=} \max_{i,j} c_{ij}$, and $\lambda_{\min}$ denote the minimum singular value of $\boldsymbol{C}$, defined in (13).

Theorem 4: The absolute error on the entries of $f(\boldsymbol{X}_j)$, for $j \in [k]$, in the outcome of ALCC is bounded as follows:

$$\Delta f_{gl}^j \leqslant \overline{\beta} \frac{c(mne)^D}{\lambda_{\min}} \sqrt{\tilde{D}+1} (kr + t\theta\sigma_n)^D \kappa_{\boldsymbol{C}} 2^{-v} (1 + O(\frac{1}{\sigma_n})), \tag{14}$$

where $\overline{\beta} = \frac{\beta^{\tilde{D}+2}-1}{\beta^2-1}$, $\boldsymbol{C}$ is defined in (13), and v is the number of precision bits in the floating-point representation.

The proofs of results in this section are omitted due to space constraints and can be found in [20].

Theorem 4 provides an upper bound on the accuracy of the outcome of ALCC with floating-point implementation for a general polynomial function $f(\cdot)$. However, $f(\cdot)$ often has a certain structure in practice that can be leveraged to strengthen the result of Theorem 4. More specifically, we say that $f(\cdot)$ is a *matrix polynomial function*, or simply a *matrix polynomial*, if it can be expressed by matrix addition, multiplication, and transposition as well as addition and multiplication by a constant matrix/vector/scalar. For instance, $f(\boldsymbol{X}) = \boldsymbol{a}\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$, for some vector $\boldsymbol{a}$, is such a matrix polynomial function. The following corollary provides a stronger accuracy bound on the outcome of ALCC with matrix polynomial as its underlying function.

Corollary 5: Let $f(\cdot)$ be a matrix polynomial function. Then, the absolute error on the entries of $f(\boldsymbol{X}_j)$, for $j \in [k]$, in the outcome of ALCC is bounded as follows:

$$\Delta f_{gl}^j \leqslant C(kr + t\theta\sigma_n)^D \kappa_{\boldsymbol{C}} 2^{-v} (1 + O(\frac{1}{\sigma_n})), \tag{15}$$

where $C \stackrel{\text{def}}{=} \overline{\beta} \frac{c \max(m,n)^D}{\lambda_{\min}} \sqrt{\tilde{D}+1}$.

***Remark** 2:* Note that when no stragglers are assumed, i.e., $\bar{s} = 0$, picking α_i's in the proposed ALCC protocol according to (12) implies that the matrix $\boldsymbol{C}$, defined in (13), is a unitary matrix. Hence, we have $\kappa_{\boldsymbol{C}} = 1$ which is the minimum possible for the condition number $\kappa_{\boldsymbol{C}}$. For the case of ALCC with stragglers, i.e., $\bar{s} > 0$, one can utilize the upper bond $\kappa_{\boldsymbol{C}} \leqslant O(N^{\bar{s}+6})$ [21, Theorem 1], This together with (14) leads to an upper bound on the accuracy of ALCC scheme with $\bar{s}$ stragglers.

Next, we analyze the privacy level of data in ALCC by considering MIS and DS security metrics, same as in Section II. For $j \in [k]$ and $i \in [t]$, let X_j and N_i denote the (g,l) element of the matrices $\boldsymbol{X}_j$ and $\boldsymbol{N}_i$, respectively, for some fixed $g \in [m]$ and $l \in [n]$. Let also Y_i denote the corresponding entry of $\boldsymbol{Y}_i$, for $i \in [N]$. For the sake of clarity, g and l are fixed throughout this section. However, the analysis does not depend on the specific choice of g and l. Recall that $T = \{i_1, \cdots, i_t\}$ denotes the set of indices for the colluding parties. Let X, N, and Y_T denote $(X_1, \cdots, X_k)^{\mathrm{T}}$, $(N_1, \cdots, N_t)^{\mathrm{T}}$, and $(Y_{i_1}, \cdots, Y_{i_t})^{\mathrm{T}}$, respectively, where $(\cdot)^{\mathrm{T}}$ is the transpose operation.

The following equation relates the encoded symbols received by the colluding set of parties T to the dataset symbols and the added noise symbols:

$$Y_T = \boldsymbol{L}_T X + \tilde{\boldsymbol{L}}_T N, \tag{16}$$

where $\boldsymbol{L}_T$ and $\tilde{\boldsymbol{L}}_T$ are $t \times k$ and $t \times t$ complex-valued matrices, respectively, such that

$$[\boldsymbol{L}_T]_{jr} = l_r(\alpha_{i_j}), \text{ and } [\tilde{\boldsymbol{L}}_T]_{jr} = l_{k+r}(\alpha_{i_j}). \tag{17}$$

The equation (16) resembles the input-output relation of a MIMO channel with k transmit and t receive antennas. Then, off-the-shelf results on the capacity of MIMO channel with equal-power allocation constraint and correlated noise along with the certain choice of β_j's are leveraged to provide an upper bound on η_c in ALCC in the following theorem.

Theorem 6: In the proposed ALCC, the MIS metric η_c, defined in (6), is upper bounded as follows:

$$\eta_c \leqslant \max_T \log_2 |\boldsymbol{I}_t + \frac{r^2 t}{\sigma_n^2} \tilde{\boldsymbol{\Sigma}}_T^{-1} \boldsymbol{\Sigma}_T|, \tag{18}$$

where $\tilde{\boldsymbol{\Sigma}}_T \stackrel{\text{def}}{=} \tilde{\boldsymbol{L}}_T \tilde{\boldsymbol{L}}_T^H$ and $\Sigma_T \stackrel{\text{def}}{=} \boldsymbol{L}_T \boldsymbol{L}_T^H$. Also, $\boldsymbol{L}_T$ and $\tilde{\boldsymbol{L}}_T$ are specified in (17). In particular, for $r = o(\sigma_n)$ we have

$$\eta_c \leqslant \frac{1}{\ln(2)} \max_T \mathrm{tr}(\tilde{\boldsymbol{\Sigma}}_T^{-1} \boldsymbol{\Sigma}_T) \frac{r^2 t}{\sigma_n^2} + o(\frac{r^2}{\sigma_n^2}). \tag{19}$$

Combining the result of Theorem 6 together with the relation between η_c and η_s, provided in (8), yields the following upper bound on the DS metric η_s:

$$\eta_s \leqslant \sqrt{2 \max_T \log_2 |\boldsymbol{I}_t + \frac{r^2 t}{\sigma_n^2} \tilde{\boldsymbol{\Sigma}}_T^{-1} \boldsymbol{\Sigma}_T|}. \tag{20}$$

***Remark** 3:* Note that both (18) and (20) imply that increasing the standard deviation of the noise, i.e., σ_n, while other parameters of ALCC are fixed, improves bounds on the privacy level. However, this improvement comes at the expense of degrading the accuracy of the outcome, according to Theorem 4. This exhibits a fundamental trade-off between the accuracy and privacy of ALCC similar to the polynomial evaluation scheme in Section II.

Finally, the DS metric is characterized for ALCC with truncated noise terms.

Theorem 7: The DS metric, defined in (7), for the case where the entries of $\boldsymbol{N}_i$'s in (10) are drawn from a truncated complex Gaussian distribution with truncation level $\theta \frac{\sigma_n}{\sqrt{t}}$ satisfies the following inequality:

$$\eta_s' \leqslant \frac{1}{w} \eta_s + \frac{1}{w} (2 \exp(-\frac{1}{2}(\theta - \frac{\overline{d}_{\text{mean}} \sqrt{t}}{\sigma_n})^2))^t,$$

where $w = (1 - 2\exp(-\frac{\theta^2}{2}))^t$ and $\overline{d}_{\text{mean}} \stackrel{\text{def}}{=} \frac{kr}{k+t} \frac{(\frac{1}{\beta})^{k+t}-1}{(\frac{1}{\beta})-1}$.

A numerical evaluation of the bound provided in Theorem 7 implies that, for instance, having $\theta = 10$ with $t = 10$, together with a very small $\frac{r}{\sigma_n}$, which is often the case in practice, we get $\eta_s' \approx \eta_s$. In other words, the privacy of dataset is not compromised by truncating the noise terms as long as θ is large enough, e.g., $\theta = 10$.

IV. Numerical Results and Experiments

In this section we numerically evaluate the bounds provided on the accuracy and privacy of ALCC in Section III. Furthermore, we demonstrate the performance of ALCC when applied to a certain computational task through experiments.

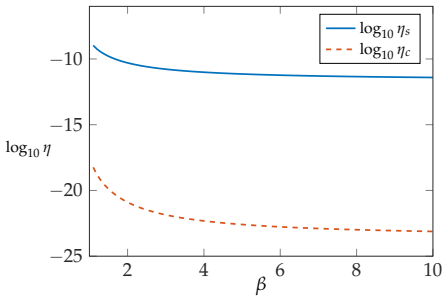

Fig. 1: Upper bounds on η_s and η_c for $N = 15, k = 4, t = 4, \sigma_n = 10^{23}, r = 10^{10}$.

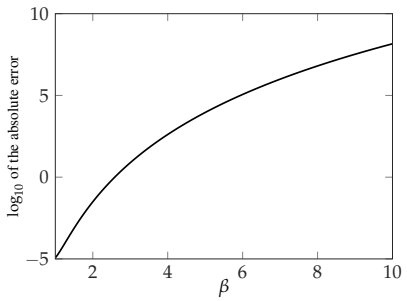

Fig. 2: Upper bound on the accuracy of ALCC versus β for $D = 2, k = 4, t = 4, s = 0, c = 1, m = n = 1000, r = 10^{10}, \theta = 3, \sigma_n = 10^{23}$ and $v = 200$.

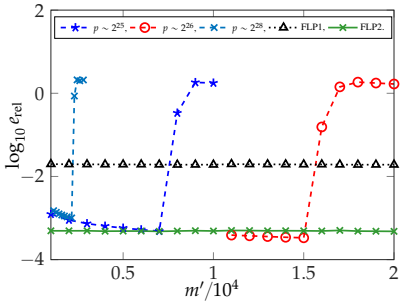

Fig. 3: Comparison of the relative error in the outcome between ALCC and LCC. For LCC, different values of p are considered ($p \sim 2^{25}$, $p \sim 2^{26}$, $p \sim 2^{28}$). For ALCC, $\beta = 2$ and $\beta = 1.5$ are considered for FLP1 and FLP2, respectively. Also, in both protocols we have $k = 5, t = 3, s = 0, N = 15, \sigma_n = 10^6$, and $n = 100$.

The provided upper bounds on the MIS security metric, specified in (18), as well as the DS metric, characterized in (20), for a certain set of parameters and the results are shown in Figure 1. Both η_s and η_c are plotted versus β. The plot indicates that increasing β, in general, leads to enhancement in the privacy of the ALCC protocol. However, the provided upper bound on the accuracy of the outcome of ALCC, provided in (14) and (15), implies that the precision loss would also grow by increasing β. The upper bound on the absolute error in ALCC with general underlying matrix polynomial function is plotted versus β in Figure 2 for a certain set of parameters. Note that the terms $o(\frac{1}{\sigma_n})$ are discarded in the plot in Figure 2 since $\frac{1}{\sigma_n} = 10^{-23}$ is negligible. This together with the plot in Figure 1 demonstrates a new fundamental trade-off between the accuracy and the privacy of the ALCC protocol which is specific to ALCC and is controlled by the choice of β. It can be also observed from Figure 1 and Figure 2 that a reasonable value for β, e.g., $\beta = 1.5$, can be picked for which the upper bounds on η_s and η_c are reasonably low, e.g., $\sim 10^{-10}$ and $\sim 10^{-20}$, respectively, while the upper bound on the error in the outcome is reasonable for practical purposes, e.g., $\sim 10^{-3}$.

Next, we demonstrate the performance of ALCC when applied to a certain computational task through experiments. Our experiments indicate that the precision of ALCC outcome closely follows that of a *centralized* computation, that is when the computations are done directly at a central node without any encoding and decoding. In particular, it is shown that the accuracy of ALCC is *scalable* with dataset size, i.e., the precision of the results remains *almost* the same for a wide range of sizes of the dataset. Moreover, the performance of LCC [4] employing fixed-point representation applied to the same computational task is demonstrated. It is shown that the error in the outcome of LCC experiences a sharp increase due to overflow errors as the dataset size passes a certain threshold.

We consider the task of performing a certain matrix-matrix multiplication. For the sake of clarity, we consider computing $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ where $\boldsymbol{X} \in \mathbb{R}^{m' \times n}$ is a *tall* real-valued matrix, i.e., $m' \gg n$. Such computation is one of the main building blocks in various learning algorithms including training a linear regression model [4], or a logistic regression model [16], [22], etc., where $\boldsymbol{X}$ represents a dataset consisting of m' samples in an n-dimensional feature space. The matrix $\boldsymbol{X}$ can be represented as a batch of matrices $\boldsymbol{X} = (\boldsymbol{X}_1^{\mathrm{T}}, \cdots, \boldsymbol{X}_k^{\mathrm{T}})^{\mathrm{T}}$, where $\boldsymbol{X}_i \in \mathbb{R}^{m \times n}$ with $m' = k \times m$. Then we have $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X} = \sum_{i=1}^{k} \boldsymbol{X}_i^{\mathrm{T}}\boldsymbol{X}_i$. Hence, the task of computing $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ is reduced to evaluating a degree-2 polynomial over a batch of matrices, consisting of $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k$, for which ALCC can be utilized to provide speed up by leveraging the computational power of distributed servers in parallel.

Let $\boldsymbol{Y}$ denote the result of computing $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ in a centralized fashion employing floating-point operations. Let also $\boldsymbol{Y}'$ denote the result of a distributed computing protocol, e.g., ALCC. In order to measure the accuracy loss of the outcome in the distributed protocol compared to the centralized one, we consider the following notion of *relative error*: $e_{\text{rel}} \stackrel{\text{def}}{=} \frac{\|\boldsymbol{Y}' - \boldsymbol{Y}\|}{\|\boldsymbol{Y}\|}$. In a sense, e_{rel} measures how much the outcome precision is proportionally compromised by utilizing a distributed protocol while providing privacy/speed up. The entries of the dataset $\boldsymbol{X}$ in our experiments are drawn independently from a zero-mean Gaussian distribution with variance 1. We use 64 bits for both the fixed-point and the floating-point numbers to implement both the LCC and the ALCC protocols in our experiments, respectively.

In Figure 3, the relative error is plotted for both LCC and ALCC versus the parameter m', that is proportional to the size of the dataset. For LCC, this is plotted for a few different choices for the size of the underlying finite field p. Also, for ALCC, e_{rel} is plotted for two values of β. It can be observed in Figure 3 that for all the scenarios considered for LCC, there exists a certain threshold for m' after which the computation results become very unreliable due to a very high e_{rel}. As discussed earlier, this significant precision loss is mostly due to overflow errors that are inherent to the fixed-point implementation employed by LCC. As expected, the sharp increase in e_{rel} occurs at a larger value for m' when a larger p is picked. However, the choice of p is limited by the number of bits available for representing fixed-point numbers. Furthermore, the advantage of ALCC compared to LCC is evident in Figure 3 by observing that the relative error in the outcome of ALCC with floating-point implementation is *almost* constant for the considered range of sizes of the dataset. This motivates employing ALCC in certain problems involving very large datasets.

REFERENCES

[1] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 439–450.

[2] V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, and N. Taft, "Privacy-preserving ridge regression on hundreds of millions of records," in *2013 IEEE Symposium on Security and Privacy*. IEEE, 2013, pp. 334–348.

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

[4] 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*, 2019, pp. 1215–1225.

[5] S. Li, S. Avestimehr *et al.*, "Coded computing," *Foundations and Trends® in Communications and Information Theory*, vol. 17, no. 1, pp. 1–148, 2020.

[6] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 265–283.

[7] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proceedings of the 3rd international symposium on Information processing in sensor networks*, 2004, pp. 20–27.

[8] R. Wright and Z. Yang, "Privacy-preserving bayesian network structure computation on distributed heterogeneous data," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004, pp. 713–718.

[9] M. Kantarcioglu and C. Clifton, "Privacy-preserving distributed mining of association rules on horizontally partitioned data," *IEEE transactions on knowledge and data engineering*, vol. 16, no. 9, pp. 1026–1037, 2004.

[10] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu, "Tools for privacy preserving distributed data mining," *ACM Sigkdd Explorations Newsletter*, vol. 4, no. 2, pp. 28–34, 2002.

[11] S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of computer and system sciences*, vol. 28, no. 2, pp. 270–299, 1984.

[12] C. E. Shannon, "Communication theory of secrecy systems," *Bell system technical journal*, vol. 28, no. 4, pp. 656–715, 1949.

[13] I. Dinur and K. Nissim, "Revealing information while preserving privacy," in *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 2003, pp. 202–210.

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

[15] J. W. Demmel, *Applied numerical linear algebra*. Siam, 1997, vol. 56.

[16] M. Soleymani, H. Mahdavifar, and A. S. Avestimehr, "Privacy-preserving distributed learning in the analog domain," *arXiv preprint arXiv:2007.08803*, 2020.

[17] M. Bellare, S. Tessaro, and A. Vardy, "A cryptographic treatment of the wiretap channel," *Advances in Cryptology – CRYPTO*, 2012.

[18] C. Ling, L. Luzzi, J.-C. Belfiore, and D. Stehlé, "Semantically secure lattice codes for the gaussian wiretap channel," *IEEE Transactions on Information Theory*, vol. 60, no. 10, pp. 6399–6416, 2014.

[19] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. Cambridge university press, 2005.

[20] M. Soleymani, H. Mahdavifar, and A. S. Avestimehr, "Analog lagrange coded computing," *arXiv preprint arXiv:2008.08565*, 2020.

[21] A. Ramamoorthy and L. Tang, "Numerically stable coded matrix computations via circulant and rotation matrix embeddings," *arXiv preprint arXiv:1910.06515*, 2019.

[22] J. So, B. Guler, A. S. Avestimehr, and P. Mohassel, "CodedPrivateML: A fast and privacy-preserving framework for distributed machine learning," *arXiv preprint arXiv:1902.00641*, 2019.