PRIVATE COMPUTATION FOR MACHINE LEARNING APPLICATIONS

A Thesis

by

SEYEDEH NAHID ESMATI

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Alexander Sprintson
Committee Members,	Srinivas Shakkottai
	Anxiao Jiang
Head of Department,	Miroslav Begovic

May 2021

Major Subject: Computer Engineering

ABSTRACT

In many practical settings, a user needs to perform computations—for example, using machine learning or cloud/edge computing algorithms—on the data stored in a storage system that consists of one or multiple remote servers. The direct data access may, however, result in exposing the identity of the data items used for the computation process to the server(s), and this can violate the privacy of the user. In recent years, several privacy-preserving schemes with information-theoretic privacy guarantees were proposed for some special cases of the private computation paradigm. Examples of such scenarios are Private Information Retrieval (PIR) and Private Linear Computation (PLC). Inspired by these works, in this thesis we introduce the problem of single-server *Private Linear Transformation (PLT)*, which generalizes the PIR and PLC problems. In the PLT problem, there is a user that wishes to compute multiple linear combinations of a subset of items belonging to a dataset stored on a single remote server. The goal of the user is to minimize the total amount of information being downloaded from the server while keeping the identities of items required for the computation private. This problem is motivated by several applications such as linear transformation for dimensionality reduction in machine learning.

In this thesis, we make a significant progress towards characterizing the fundamental performance limits of the single-server PLT problem for two information-theoretic privacy conditions, called *joint privacy* and *individual privacy*, which have been recently considered for PIR and PLC. We prove converse bounds using a mix of linear-algebraic and information-theoretic arguments that are tailored for the single-server case, and are different from the commonly-used techniques in multi-server PIR and PLC. We design optimal privacy-preserving schemes by leveraging ideas from the literature on coding theory, such as *Maximum Distance Separable (MDS) codes*, and *interference alignment*. In addition, we briefly discuss the PLT settings in which the user has a *prior side information* about the dataset, and propose novel PLT schemes to show the advantages of side information in reducing the download cost for each type of privacy considered.

DEDICATION

To Mom and Dad

and

To my sister

This work is a sign of my love to you!

ACKNOWLEDGMENTS

I would like to express my deepest appreciation to my advisor, Professor Alexander Sprintson, for his constant support and patience throughout my study. Without his invaluable expertise and insightful suggestions I would not have been able to complete this thesis. I would also like to thank Professor Srinivas Shakkottai and Professor Andrew Jiang for kindly accepting to be on my thesis committee.

I wish to extend my sincere thanks to Professors Krishna Narayanan, Natarajan Sivakumar, Jean-Francois Chamberland, Tie Liu, Serap Savari, Chao Tian, Dileep Kalathil, Catherine Yan, and Rostislav Grigorchuk for their excellent teaching style, which contributed to my development as a graduate student.

Thanks should also go to my friends for their kindness and support during the time we spent together.

I am forever thankful to my husband for his incredible support and understanding. His endless love and belief in me have always helped me through stressful times. I am also extremely grateful to my beloved parents, my lovely sister, and my wonderful niece, Arnika, for their love, caring, and emotional support throughout my life. I cannot adequately thank them for encouraging me to pursue my dreams.

NOMENCLATURE

ML	Machine Learning
PIR	Private Information Retrieval
PLC	Private Linear Computation
PLT	Private Linear Transformation
MDS	Maximum Distance Separable
GRS	Generalized Reed-Solomon
JPLT	Private Linear Transformation with Joint Privacy
IPLT	Private Linear Transformation with Individual Privacy
SI	Side Information
USI	Uncoded Side Information
CSI	Coded Side Information
ILP	Integer Linear Programming

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

1. INTRODUCTION

1.1 Paradigm of Private Computation

Nowadays, many different types of data are stored in remote data centers (servers). These datasets are constantly being queried by users for different purposes, and serve as input for a broad range of machine learning and cloud/edge computing algorithms. The direct access to data can however expose the identity of data items a user is interested in. This, in turn, may expose the user's algorithms, preferences, and objectives. For example, consider a learning algorithm that requires only a small subset of features of data samples in the dataset, and needs to compute multiple linear combinations of these features as new combined features in the projected space. By monitoring the specific indices of the features accessed by such an algorithm, a curious server may gain insight on the inner working of the algorithm. Accordingly, one key requirement in the design of such systems is to hide the identity of the data items—that are accessed and provided as an input to the user's algorithm—from the server(s). The research on this type of privacy, which we refer to as the *user privacy*, has been very limited. In contrast, the notion of *data privacy*, which is to protect the content of the data from the user (see e.g., [1–4]), was extensively studied, producing many algorithms, such as those based on secure multi-party computation [5, 6], differentially private computation [7, 8], and functional encryption schemes [9, 10]. We refer to the paradigm of computing functions of a dataset while maintaining the user privacy as *private computation*. In this problem, the goal is to design a system such that the user can privately compute their target function(s) of the dataset while the total amount of communication between the user and the server(s) (i.e., the sum of lengths of each query (upload) and each answer (download)) is minimized.

1.2 Related Work

The research on protecting user privacy in private computation was initiated in the work by Chor *et al.* [11] on *Private Information Retrieval (PIR)*. In PIR, the goal of the user is to retrieve a subset of data items with minimum possible communication cost, while hiding the identities of the

required data items from the remote server(s). Early on, it was shown that when there is only one server holding the dataset or multiple servers that can arbitrarily collude, a user that is interested in retrieving even one data item must download the entire dataset so as to achieve *information-theoretic privacy* [12]. In contrast, when multiple copies of the dataset are stored across multiple servers with limited collusion capability, it was shown that information-theoretic privacy can be achieved much more efficiently [13].

More recently, a novel and more realistic PIR model for information-theoretic privacy was proposed in [13]. This model focuses on large data items, and as a result, the goal in this model is to minimize the download cost only, as the upload cost becomes negligible in comparison. This model was also extended for *Private Linear Computation (PLC)* in [14,15]. In PLC, the goal of the user is to compute *one* linear combination of a subset of data items without revealing the identities of these items. In all these works, it is assumed that the dataset is stored across multiple servers and these servers have limited capability for collusion. These assumptions, however, may prevent the use of the existing PIR and PLC schemes in various practical scenarios. For example, many databases are stored on a single server or multiple servers that belong to the same entity, and hence can collude arbitrarily [16, 17].

Aside from information-theoretic privacy is *cryptographic privacy*, which assumes that the computational power of the server(s) is limited (see, e.g., [18, 19]). Unfortunately, the existing schemes for cryptographic privacy were not adopted in many practical settings, mainly because they do not fully capture the constraints and requirements imposed by real-world computing systems. For instance, such schemes do not scale well for large data items [20], and require computationally expensive homomorphic encryption [21]. The focus of this research is on information-theoretic privacy as it enables the strongest possible protection against curious server(s), and does not rely on any limiting physical or computational assumptions.

Starting from the work in [13–15], there have been several breakthrough developments in the design of privacy-preserving PIR and PLC protocols, see e.g., [13–15, 22–32]. Notwithstanding, there remain many important settings that have not been explored in detail.

1. The PIR problem was originally studied under the single-demand setting where the user wants to privately retrieve *one* item from the database, see e.g., [13, 33–36]. Later, the multi-demand PIR setting was also considered in [24, 29], where the goal is to privately retrieve *multiple* items simultaneously. The PLC problem was only studied under the single-demand setting, where the user wishes to privately compute *one* linear combination of the database items, see e.g., [14, 15, 17, 30, 31]. That said, several machine learning and cloud computing algorithms such as distributed gradient descent involve multiple simultaneous function computations on the same subset of dataset.

2. The classical PIR and PLC schemes do not consider any prior side information about the database at the user. However, in many practical scenarios, the user may possess some side information about the database in advance. For instance, the user may initially know some of the items in the database [16, 37] or some linear combinations of them [17, 38], where the identities of these items are initially unknown to the server(s). Recently, it was shown that both PIR and PLC can be performed much more efficiently (in terms of the download cost) when leveraging the user's side information (see, e.g., [16, 17]).

3. In different applications, the user may require different types of privacy. For instance, the user may only want *individual privacy*, i.e., to hide the identity of every item in their demand individually [17,39,40], or they may want *joint privacy*, i.e., they want to leak no information about the correlation between the identities of the items in their demand [17, 24, 29, 41, 42]. In addition, when the user has some side information about the database, they may also wish to hide the identities of their side information items [41, 43–46]. Most of the existing work on PIR focuses on joint privacy. Despite the fact that joint privacy is a stronger notion of privacy, individual privacy may still suffice in several practical scenarios [39, 40]. In most of the existing work on PLC, the privacy requirement is to hide the coding coefficients in the demanded linear combination [14, 15, 30, 31]. Although this type of privacy is stronger than both joint and individual privacy, but it may not be necessary in some applications [17, 37].

1.3 Motivation

Inspired by the PIR model of [13] and the PLC model of [14, 15] and motivated by their limitations listed above, in this thesis we introduce the problem of single-server *Private Linear Transformation (PLT)*. The PLT problem includes a single server that stores a dataset consisting of K messages; and a user that wants to compute L linear combinations of a set of D messages. The goal of the user is to perform the computation privately so that the identities of the messages required for the computation are protected (to some degree) from the server, while minimizing the total amount of information being downloaded from the server. The PLT problem generalizes the PIR and PLC problems. In particular, PLT reduces to PIR or PLC when $L = D$ or $L = 1$, respectively.

The PLT problem appears in several practical scenarios such as linear transformation for dimensionality reduction in Machine Learning (ML), see, e.g., [47]. Consider a dataset with N data samples, each with K attributes. Consider a user that wishes to implement an ML algorithm on a subset of D selected attributes, while protecting the privacy of the selected attributes. When D is large, the D-dimensional feature space is typically mapped onto a new subspace of lower dimension, say, L, and the ML algorithm operates on the new L-dimensional subspace instead. A commonly-used technique for dimensionality reduction is *linear transformation*, where an $L \times D$ matrix is multiplied by the $D \times N$ data submatrix (the submatrix of the $K \times N$ data matrix restricted to the D selected attributes). Thinking of the rows of the $K \times N$ data matrix as the K messages, the labels of the D selected attributes as the identities of the D messages in the support set of the required linear combinations, and the $L \times D$ matrix used for transformation as the coefficient matrix of the required linear combinations, this scenario matches the setup of the PLT problem.

A natural approach for PLT is to privately retrieve the items required for the computation using a PIR scheme, and then compute the required linear combinations locally. As was shown in [16, 38–42, 45, 48], in the single-server setting, the user can retrieve a single or multiple data items privately with a much lower download cost than the trivial scheme of downloading the entire dataset, by leveraging a prior side information about the dataset. However, when there is no side information, a PIR-based PLT approach is extremely expensive since all items in the dataset must

be downloaded in order to achieve information-theoretic privacy [12].

Another approach for PLT is to privately compute the required linear combinations separately via applying a PLC scheme multiple times. In [17, 37], it was shown that single-server PLC can be performed more efficiently than single-server PIR in terms of the download cost, regardless of whether the user has any side information or not. This suggests that a PLC-based PLT scheme can outperform a PIR-based PLT scheme; however, a PLC-based approach may still lead to an unnecessary overhead due to the excessive redundancy in the information being downloaded. This implies the need for novel PLT schemes with optimal download rate.

1.4 Contributions

In this thesis, we develop techniques for single-server PLT that enable a significant reduction in the communication and computational overhead. We will achieve this goal through (i) considering two recently-introduced notions of information-theoretic privacy, namely, joint privacy and individual privacy, that provide a satisfactory degree of privacy in several applications, and (ii) novel use of different types of a prior side information about the dataset available at the user.

We focus on the setting in which the coefficient matrix of the required linear combinations is a Maximum Distance Separable (MDS) matrix, i.e., it generates an MDS code. Recall that an $L \times D$ matrix is said to be MDS if every $L \times L$ submatrix of this matrix is invertible. The MDS coefficient matrices are motivated by the application of *random linear transformation* for dimensionality reduction (see, e.g., [49]), where a random $L \times D$ matrix is used for transformation. In particular, a simple application of Schwartz-Zippel lemma shows that a matrix whose entries are randomly chosen from a sufficiently large field is MDS with high probability.

First, we consider the problem of *PLT with Joint Privacy*, which we refer to as *JPLT* for short. The joint privacy requirement implies that, from the perspective of the server, any D-subset of messages is equally likely to be the support set of the required linear combinations. We characterize the capacity of JPLT with MDS coefficient matrices, where the capacity is defined as the supremum of download rates over all JPLT schemes. We prove the converse by using a mix of linear-algebraic and information-theoretic arguments, relying on a necessary condition for JPLT

schemes. These arguments are specifically tailored to the single-server setting and are fundamentally different from the commonly-used arguments in the multi-server PIR and PLC settings. We propose an achievability scheme, termed *Specialized MDS Code protocol*, which leverages the idea of extending an MDS code. In addition, we briefly discuss the settings in which the user has a prior side information about the messages in the dataset, and the settings in which the coefficient matrix of the required linear combinations is not MDS; and present lower and/or upper bounds on the capacity of these extensions of the JPLT setting.

Next, we consider the problem of *PLT with Individual Privacy*, referred to as *IPLT* for short. The individual privacy requirement implies that, from the server's perspective, every message is equally likely to belong to the D-subset of messages that constitute the support set of the required linear combinations. For IPLT with MDS coefficient matrices, we establish lower and upper bounds on the capacity—defined as the supremum of download rates over all IPLT schemes. In addition, we show that our bounds are tight under certain divisibility conditions, settling the capacity of IPLT for such cases. To prove the upper bound on the capacity, we use information-theoretic arguments based on a necessary condition for IPLT schemes, and formulate the problem as an integer linear programming (ILP) problem. Solving this ILP, we obtain the capacity upper bound. The lower bound on the capacity is proven by a novel achievability scheme, termed *Generalized Partition-and-Code with Partial Interference Alignment protocol*. In addition, we present lower bounds on the capacity of the IPLT settings with a prior side information and non-MDS coefficient matrices.

1.5 Organization of the Thesis

The remainder of the thesis is organized as follows. In Chapter 2, we formulate the problems of single-server PLT with joint and individual privacy along with several extensions of the PLT problem. In Chapter 3, we present our main results for PLT with joint privacy (JPLT), and provide the converse proof and an achievability scheme. In Chapter 4, we present our main results for PLT with individual privacy (IPLT), the proof of converse, and the proposed achievability scheme. Finally, in Chapter 5, we conclude the thesis, and provide a list of future work.

2. PROBLEM FORMULATION

Throughout, we denote random variables and their realizations by bold-face symbols (e.g., $\mathbf{X}, \mathbf{W}$) and regular symbols (e.g., X, W), respectively. Also, we denote by $H(\cdot)$ and $H(\cdot|\cdot)$ the (Shannon) entropy and conditional entropy functions, respectively.

2.1 Private Linear Transformation (PLT) Problem

Let $\mathbb{F}_p$ be a finite field of order p, and let $\mathbb{F}_q$ be an extension field of $\mathbb{F}_p$. Let K, D, L be positive integers such that $L \leq D \leq K$, and let $\mathcal{K}$ denote the set of integers $\{1, ..., K\}$. Let $\mathcal{W}$ be the set of all D-subsets W of $\mathcal{K}$, and let $\mathcal{V}$ be the set of all $L \times D$ matrices V (with entries from $\mathbb{F}_p$) that are *Maximum Distance Separable (MDS)*, i.e., V generates a $[D, L]$ MDS code of length D and dimension L. Note that an $L \times D$ matrix V is MDS if every $L \times L$ submatrix of V is invertible.

2.1.1 Model and Assumptions

Consider a server that stores K messages $X_1, \ldots, X_K$, where $X_i \in \mathbb{F}_q$ for $i \in \mathcal{K}$. Let $\mathrm{X} \triangleq [X_1, \ldots, X_K]^{\mathsf{T}}$. For every $\mathrm{S} \subset \mathcal{K}$, we denote by X_{S} the vector X restricted to its components indexed by S. We assume that $\mathbf{X}_1, \ldots, \mathbf{X}_K$ are independently and uniformly distributed over $\mathbb{F}_q$. That is, $H(\mathbf{X}_i) = \theta \triangleq \log_2 q$ for all $i \in \mathcal{K}$, and more generally, $H(\mathbf{X}_{\mathrm{S}}) = |\mathrm{S}|\theta$ for every $\mathrm{S} \subset \mathcal{K}$, where $|\mathrm{S}|$ denotes the size of the set S. Note that $H(\mathbf{X}) = K\theta$.

Consider a user that wants to compute the vector $\mathrm{Z}^{[\mathrm{W},\mathrm{V}]} \triangleq \mathrm{V}\mathrm{X}_{\mathrm{W}}$, where $\mathrm{W} \in \mathcal{W}$ and $\mathrm{V} \in \mathcal{V}$. That is, $\mathrm{Z}^{[\mathrm{W},\mathrm{V}]}$ contains L components $\mathrm{v}_1^{\mathsf{T}}\mathrm{X}_{\mathrm{W}}, \ldots, \mathrm{v}_L^{\mathsf{T}}\mathrm{X}_{\mathrm{W}}$, where $\mathrm{v}_l^{\mathsf{T}}$ is the lth row of V. Note that $H(\mathbf{Z}^{[\mathrm{W},\mathrm{V}]}) = H(\mathrm{v}_1^{\mathsf{T}}\mathbf{X}_{\mathrm{W}}, \ldots, \mathrm{v}_L^{\mathsf{T}}\mathbf{X}_{\mathrm{W}}) = L\theta$ because $\mathrm{v}_1^{\mathsf{T}}, \ldots, \mathrm{v}_L^{\mathsf{T}}$ are linearly independent. We refer to $\mathrm{Z}^{[\mathrm{W},\mathrm{V}]}$ as the *demand*, W as the *support index set of the demand*, V as the *coefficient matrix of the demand*, D as the *support size of the demand*, and L as the *dimension of the demand*.

In this work, we assume that (i) $\mathbf{W}$, $\mathbf{V}$, and $\mathbf{X}$ are independent random variables; (ii) $\mathbf{W}$ is uniformly distributed over all $\mathrm{W} \in \mathcal{W}$; (iii) $\mathbf{V}$ is uniformly distributed over all $\mathrm{V} \in \mathcal{V}$; and (iv) the parameters D and L, and the joint distribution of $\mathbf{W}$ and $\mathbf{V}$ are initially known by the server, whereas the server does not initially know the realizations W and V.

2.1.2 Privacy and Recoverability Conditions

Given W and V, the user generates a query $\mathrm{Q}^{[\mathrm{W},\mathrm{V}]}$, simply denoted by Q, and sends it to the server. The query Q is a function of W, V, and potentially a random key R (independent of W, V, and X) that is generated by the user and is initially unknown to the server. That is, $H(\mathbf{Q}|\mathbf{W},\mathbf{V},\mathbf{R}) = 0$, where $\mathbf{Q}^{[\mathbf{W},\mathbf{V}]}$ is denoted by $\mathbf{Q}$.

The query Q must satisfy one of the following privacy conditions:

(i) *Joint privacy condition:* Given the query Q, every D-subset of message indices must be equally likely to be the demand's support index set, i.e., for every $\mathrm{W}^* \in \mathcal{W}$, it must hold that

$$\Pr(\mathbf{W} = \mathrm{W}^*|\mathbf{Q} = \mathrm{Q}) = \Pr(\mathbf{W} = \mathrm{W}^*).$$

(ii) *Individual privacy condition:* Given the query Q, every message index must be equally likely to be in the demand's support index set, i.e., for every $i \in \mathcal{K}$, it must hold that

$$\Pr(i \in \mathbf{W}|\mathbf{Q} = \mathrm{Q}) = \Pr(i \in \mathbf{W}).$$

The joint privacy condition was previously considered for PIR and PLC (see, e.g., [17,29,41]), and the individual privacy condition was recently introduced in [39] and [17] for single-server PIR and PLC. Joint privacy is a stronger notion than individual privacy—the former implies the latter, but not vice versa. In particular, in the case of joint privacy, the query must not leak any information to the server about the correlation between the indices of the messages that constitute the support set of the demand; whereas, in the case of individual privacy, upon receiving the query, the server may gain some information about this correlation. Note that, for joint or individual privacy, it is not required that the query protects the privacy of the demand's coefficient matrix from the server.

The joint and individual privacy conditions are inspired by real-world scenarios. For example, preserving the privacy of the selected attributes in the application of random linear transformation for dimensionality reduction in Machine Learning (ML) prevents the server from learning the inner working of the user's ML algorithm. In particular, it may be required that, from the server's perspective, every D-subset of attributes is equally likely to be the user's desired attributes, or it may suffice that every individual attribute is equally likely to be one of the user's desired attributes.

Upon receiving the query Q, the server generates an answer $\mathrm{A}^{[\mathrm{W},\mathrm{V}]}$, simply denoted by A, and sends it back to the user. The answer A is a deterministic function of Q and X. That is, $H(\mathbf{A}|\mathbf{Q},\mathbf{X}) = 0$, where $\mathbf{A}^{[\mathbf{W},\mathbf{V}]}$ is denoted by $\mathbf{A}$.

The answer A, the query Q, and the realizations W, V must collectively enable the user to retrieve the demand $\mathrm{Z}^{[\mathrm{W},\mathrm{V}]}$, i.e., $H(\mathbf{Z}|\mathbf{A},\mathbf{Q},\mathbf{W},\mathbf{V}) = 0$, where $\mathbf{Z}^{[\mathbf{W},\mathbf{V}]}$ is denoted by $\mathbf{Z}$. We refer to this condition as the *recoverability condition*.

2.1.3 Rate of PLT Protocols and Capacity of PLT

For each type of privacy, the problem is to design a protocol for generating a query $\mathrm{Q}^{[\mathrm{W},\mathrm{V}]}$ and the corresponding answer $\mathrm{A}^{[\mathrm{W},\mathrm{V}]}$ for any given (W, V) such that both the privacy and recoverability conditions are satisfied. We refer to this problem as single-server *PLT with Joint Privacy* (or *JPLT* for short) or single-server *PLT with Individual Privacy* (or *IPLT* for short) when joint or individual privacy is required, respectively. Also, we refer to a PLT protocol that satisfies the joint or individual privacy condition as a *JPLT protocol* or an *IPLT protocol*, respectively.

Following the convention in the PIR and PLC literature, we measure the efficiency of a JPLT protocol or an IPLT protocol by its *rate*—defined as the ratio of the entropy of the demand (i.e., $H(\mathbf{Z}) = L\theta$) to the entropy of the answer (i.e., $H(\mathbf{A})$). We define the *capacity* of the JPLT or IPLT setting as the supremum of rates over all JPLT or IPLT protocols, respectively.

In this work, our goal is to establish (preferably matching) lower and upper bounds (in terms of the parameters K, D, and L) on the capacity of the JPLT and IPLT settings.

2.2 Extensions of the PLT Problem

Both the JPLT and IPLT settings can be extended to the settings in which the user has a prior side information about the messages stored at the server, or the settings in which the coefficient matrix of the demand is not an MDS matrix. In this work, we briefly outline lower and/or upper bounds on the capacity of these extended settings. However, a comprehensive study of these settings is beyond the focus of this work, and left for future work.

2.2.1 PLT with Side Information

Two types of Side Information (SI) were previously studied for PIR and PLC: *Uncoded SI (USI)* (see, e.g., [16]), and *Coded SI (CSI)* (see, e.g., [38]). In the case of USI, the user initially knows a randomly chosen subset of M messages (different from the D messages constituting the support set of the required linear combinations). In the case of CSI, instead of M uncoded messages, the user initially knows L randomly generated MDS coded combinations of M messages. In both cases of USI and CSI, the identities of these M messages are initially unknown to the server. Recent work on PIR and PLC shows the advantages of these types of SI in increasing the capacity (see, e.g., [17, 41]). These results motivate the study of PLT in the presence of SI.

For PLT with USI or CSI, the joint and individual privacy conditions are defined as follows. For joint privacy, the identities of all messages in the support set of the demand and those in the support set of the side information need to be protected. That is, given the query, every two disjoint subsets of messages, one of size D and the other of size M, must be equally likely to be the support set of the demand and the support set of the side information, respectively. For individual privacy, the identity of every message in the support sets of demand and side information must be protected. That is, given the query, every message must be equally likely to belong to the demand's support set, and every message must be equally likely to belong to the side information's support set.

2.2.2 PLT with Non-MDS Coefficient Matrices

Both the JPLT and IPLT settings (with or without SI) can also be extended to the settings in which the coefficient matrix of the demand is randomly chosen from the ensemble of all $L \times D$ matrices that have full row rank, i.e., all $L \times D$ matrices with L linearly independent rows. Note that the MDS assumption for the coefficient matrix is particularly useful for scenarios where the size of the operating field is sufficiently large (e.g. the field of real numbers, which is the case in many real-world applications such as machine learning and data science). However, in some other scenarios, a finite field of a relatively small size may be required. In such cases, the coefficient matrix of the demand is likely to be non-MDS, particularly when the matrix size is relatively large.

3. PRIVATE LINEAR TRANSFORMATION WITH JOINT PRIVACY (JPLT)

In this chapter, we first present a necessary condition for JPLT protocols in Section 3.1. Then, we summarize our main results for the JPLT setting in Section 3.2. The proof of converse is given in Section 3.3, and the achievability scheme is presented in Section 3.4. An illustrative example of the proposed achievability scheme is provided in Section 3.5. We conclude this chapter by briefly outlining our results for the extensions of the JPLT setting in Section 3.6.

3.1 A Necessary Condition for JPLT Protocols

The following lemma states a necessary (yet not always sufficient) condition for any JPLT protocol. This result follows from the joint privacy and recoverability conditions.

Lemma 1. *Given any JPLT protocol, for any* $\mathrm{W}^* \in \mathcal{W}$*, there must exist* $\mathrm{V}^* \in \mathcal{V}$*, such that*

$$H(\mathbf{Z}^{[\mathrm{W}^*,\mathrm{V}^*]}|\mathbf{A},\mathbf{Q}) = 0.$$

Proof. The proof is by the way of contradiction. Consider an arbitrary $\mathrm{W}^* \in \mathcal{W}$. Suppose that there does not exist any $\mathrm{V}^* \in \mathcal{V}$ such that $H(\mathbf{Z}^{[\mathrm{W}^*,\mathrm{V}^*]}|\mathbf{A},\mathbf{Q}) = 0$. This implies that $\mathrm{W}^* \in \mathcal{W}$ is not the support index set of the demand (otherwise, if $\mathrm{W} = \mathrm{W}^*$, the recoverability condition is not satisfied). This obviously violates the joint privacy condition, because given the query, every D-subset of message indices must be equally likely to be the demand's support index set. □

When considering *linear* JPLT schemes, i.e., the schemes in which the answer consists of only *linear* combinations of the messages, the necessary condition provided by Lemma 1 can be interpreted in the language of coding theory as follows. The coefficient matrix of the linear combinations corresponding to the answer must generate a (linear) code of length K that, when punctured at any $K - D$ coordinates, contains L codewords that are MDS, i.e., they generate a $[D, L]$ MDS code. Recall that puncturing a (linear) code at a coordinate is performed by deleting the column pertaining to that coordinate from the generator matrix of the code. A code satisfying

this condition is, however, not guaranteed to yield a JPLT scheme. A sufficient (but not necessary) condition is that the codes resulting from puncturing at any $K - D$ coordinates contain the *same number of groups* of L MDS codewords. Thus, designing a linear JPLT scheme with *maximum rate* reduces to constructing such a linear code with *minimum dimension*. This sufficient condition is, however, more combinatorial in nature, and the necessary condition provided by Lemma 1 proves more useful when deriving an information-theoretic converse bound.

3.2 Main Results for the JPLT Setting

Theorem 1. *For the JPLT setting with K messages, demand's support size D, and demand's dimension L, the capacity is given by*

$$\frac{L}{K - D + L}.$$

The proof of converse is based on information theoretic arguments relying mostly on the necessary condition for JPLT protocols—provided by Lemma 1. The converse bound naturally serves as an upper bound on the rate of any JPLT protocol. We prove the achievability by designing a linear JPLT protocol, termed the *Specialized MDS Code protocol*, that achieves the converse bound. This protocol generalizes those in [41] and [37] for single-server PIR and PLC (without SI) with joint privacy, and is based on the idea of extending the MDS code generated by the coefficient matrix of the demand. In particular, when the coefficient matrix of the demand generates a Generalized Reed-Solomon (GRS) code, we give an explicit construction of a GRS code that contains a specific collection of codewords—specified by the demand's support index set and coefficient matrix.

Remark 1. The result of Theorem 1 shows that, when there is only a single server and there is no prior side information available at the user, JPLT can be performed more efficiently than using either of the following two approaches: (i) retrieving the messages required for computation using a multi-message PIR scheme [41] and computing the required linear combinations locally, or (ii) computing each of the required linear combinations separately via applying a PLC scheme [37]. More specifically, the optimal rate for the approach (i) or (ii) is L/K or $1/(K-D+1)$, respectively, whereas an optimal JPLT scheme achieves the rate $L/(K - D + L)$. Fig. 3.1 depicts the download

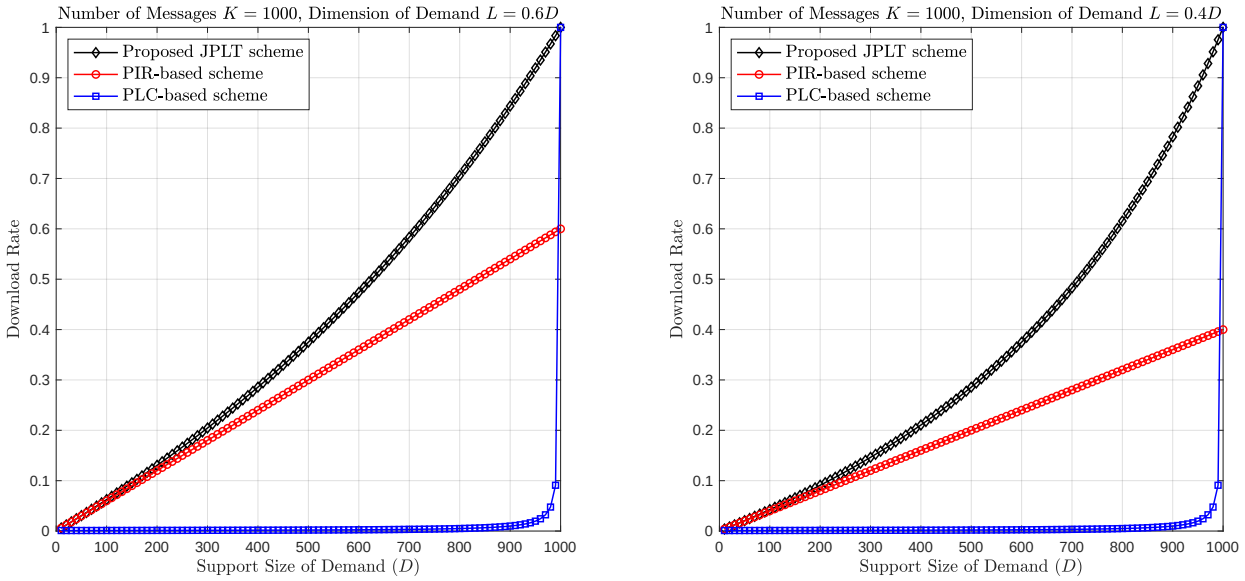

Figure 3.1: The download rate of the proposed scheme and the PIR-based and PLC-based schemes.

rate of the proposed JPLT scheme, the PIR-based scheme, and the PLC-based scheme, for different values of $D \in \{10, 20, \ldots, 1000\}$, where $K = 1000$, and $L/D = 0.6$ (left plot) or $L/D = 0.4$ (right plot). As can be seen, for a fixed ratio L/D (i.e., a fixed reduction factor in the application of linear transformation for dimensionality reduction), the advantage of the proposed scheme over the PIR-based scheme is more pronounced as D increases. For instance, for $L/D = 0.4$, the rate of the proposed scheme is about 15% and 30% more than that of the PIR-based scheme for $D = 250$ and $D = 500$, respectively. Comparing the proposed scheme and the PLC-based scheme, one can also observe that when the ratio L/D is fixed, the gap between the rate of the proposed scheme and that of the PLC-based scheme increases as D increases up to a threshold very close to K; and beyond this threshold, the gap decreases rapidly as D increases up to K. In addition, a comparison of the left and right plots in Fig. 3.1 shows that for a fixed value of D, the smaller is the ratio L/D (i.e., the larger is the reduction factor), the more is the advantage of the proposed scheme over the best of the other two schemes. For instance, for $D = 250$, the rate of the proposed scheme is about 10% and 15% more than that of the PIR-based scheme for $L/D = 0.6$ and $L/D = 0.4$, respectively.

Remark 2. In [37], it was shown that the rate $1/(K-D+1)$ is achievable for PLC (without SI) with joint privacy, but no converse result was presented. The result of Theorem 1 for $L=1$ proves the optimality of this rate. For $L=D$, the JPLT problem reduces to PIR (without SI) when joint privacy is required, and as was shown in [41], an optimal solution for this case is to download the entire dataset. This is consistent with the result of Theorem 1 for $L=D$.

3.3 Proof of Converse

Lemma 2. *The rate of any JPLT protocol for K messages, demand's support size D, and demand's dimension L, is upper bounded by $L/(K-D+L)$.*

Proof. Consider an arbitrary JPLT protocol that generates the query-answer pair $(\mathrm{Q}^{[\mathrm{W},\mathrm{V}]}, \mathrm{A}^{[\mathrm{W},\mathrm{V}]})$ for any given (W,V). To prove the rate upper bound, we need to show that $H(\mathbf{A}) \geq (K-D+L)\theta$, where $\mathbf{A}$ denotes $\mathbf{A}^{[\mathbf{W},\mathbf{V}]}$, and θ is the entropy of a message.

For the ease of notation, we define $T \triangleq K-D+1$. For every $1 \leq i \leq T$, let $\mathrm{W}_i \triangleq \{i, i+1, \ldots, i+D-1\}$. Note that $\mathrm{W}_1, \ldots, \mathrm{W}_T \in \mathcal{W}$. By Lemma 1, for any $1 \leq i \leq T$, there exists $\mathrm{V}_i \in \mathcal{V}$ such that $H(\mathbf{Z}_i|\mathbf{A},\mathbf{Q}) = 0$, where $\mathbf{Z}_i \triangleq \mathbf{Z}^{[\mathrm{W}_i,\mathrm{V}_i]}$. (Note that V_i is an MDS matrix.) This readily implies that $H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T|\mathbf{A},\mathbf{Q}) = 0$ since

$$H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T|\mathbf{A},\mathbf{Q}) \leq \sum_{i=1}^{T} H(\mathbf{Z}_i|\mathbf{A},\mathbf{Q}) = 0.$$

Thus, we can write

$$\begin{aligned} H(\mathbf{A}) &\geq H(\mathbf{A}|\mathbf{Q}) + H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T|\mathbf{Q},\mathbf{A}) && (3.1)\\ &= H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T|\mathbf{Q}) + H(\mathbf{A}|\mathbf{Q},\mathbf{Z}_1, \ldots, \mathbf{Z}_T) && (3.2)\\ &\geq H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T), && (3.3) \end{aligned}$$

where (3.1) holds because $H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T|\mathbf{A},\mathbf{Q}) = 0$, as shown earlier; (3.2) follows from the chain rule of conditional entropy; and (3.3) holds because (i) $\mathbf{Z}_i$'s are independent from $\mathbf{Q}$, noting that $\mathbf{Z}_i$'s only depend on $\mathbf{X}$, and $\mathbf{Q}$ is independent of $\mathbf{X}$, and (ii) $H(\mathbf{A}|\mathbf{Q},\mathbf{Z}_1, \ldots, \mathbf{Z}_T) \geq 0$.

To lower bound $H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T)$, we proceed as follows. By the chain rule of entropy, we have

$$H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T) = H(\mathbf{Z}_1) + \sum_{1<i\leq T} H(\mathbf{Z}_i|\mathbf{Z}_1, \ldots, \mathbf{Z}_{i-1}).$$

Let $\mathbf{Z}_{i,1}, \ldots, \mathbf{Z}_{i,L}$ be the L components of the vector $\mathbf{Z}_i$, i.e., $\mathbf{Z}_{i,l} \triangleq \mathrm{v}_{i,l}^{\mathsf{T}}\mathbf{X}_{\mathrm{W}_i}$, where $\mathrm{v}_{i,l}^{\mathsf{T}}$ is the lth row of V_i. Note that $\mathbf{Z}_i$ consists of L components $\mathbf{Z}_{i,1}, \ldots, \mathbf{Z}_{i,L}$, and these components are independent because their corresponding coefficient vectors $\mathrm{v}_{i,1}, \ldots, \mathrm{v}_{i,L}$ are linearly independent. Moreover, $\mathbf{Z}_{i,1}, \ldots, \mathbf{Z}_{i,L}$ are uniform over $\mathbb{F}_q$, i.e., $H(\mathbf{Z}_{i,l}) = \theta$ for $l \in \{1, \ldots, L\}$. Thus, $H(\mathbf{Z}_i) = H(\mathbf{Z}_{i,1}, \ldots, \mathbf{Z}_{i,L}) = L\theta$, particularly, $H(\mathbf{Z}_1) = L\theta$.

It should be clear that $\mathbf{X}_{i-D+1}$ belongs to the support set of $\mathbf{Z}_{i,l}$ for some $l \in \{1, \ldots, L\}$. Otherwise, V_i contains an all-zero column, which is a contradiction. Moreover, $\mathbf{X}_{i-D+1}$ does not belong to the support set of any of the components of $\mathbf{Z}_j$ for any $j < i$ (by construction). This implies that $\mathbf{Z}_i$ contains at least one component, namely, $\mathbf{Z}_{i,l}$, that cannot be written as a linear combination of the components in $\mathbf{Z}_1, \ldots, \mathbf{Z}_{i-1}$. Thus, $\mathbf{Z}_{i,l}$ is independent of $\mathbf{Z}_1, \ldots, \mathbf{Z}_{i-1}$. This further implies that $H(\mathbf{Z}_i|\mathbf{Z}_1, \ldots, \mathbf{Z}_{i-1}) \geq H(\mathbf{Z}_{i,l}) = \theta$, and consequently,

$$\sum_{1<i\leq T} H(\mathbf{Z}_i|\mathbf{Z}_1, \ldots, \mathbf{Z}_{i-1}) \geq (T-1)\theta.$$

Thus, we have

$$H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T) \geq L\theta + (T-1)\theta = (K-D+L)\theta. \tag{3.4}$$

Combining (3.3) and (3.4), we get $H(\mathbf{A}) \geq (K-D+L)\theta$, as was to be shown. □

3.4 Achievability Scheme

In this section, we propose a linear JPLT protocol, termed *Specialized MDS Code Protocol*, that achieves the rate $L/(K-D+L)$. An illustrative example of this protocol is given in Section 3.5. The proposed protocol consists of three steps as follows.

Step 1: Given the demand support index set W and the demand coefficient matrix $\mathrm{V} = [\mathrm{v}_1, \ldots, \mathrm{v}_l]^{\mathsf{T}}$, the user constructs a query $\mathrm{Q}^{[\mathrm{W},\mathrm{V}]}$ in the form of a matrix G, such that the user's

query, i.e., the matrix G, and the server's corresponding answer $\mathrm{A}^{[\mathrm{W},\mathrm{V}]}$, i.e., the vector GX, satisfy the joint privacy and recoverability conditions.

To satisfy the joint privacy condition, it is required that, for any index set $\mathrm{W}^* \in \mathcal{W}$, the code generated by the matrix G contains L codewords whose support index sets are some subsets of W^*, and the coordinates of these codewords (indexed by W^*) form an MDS matrix $\mathrm{V}^* \in \mathcal{V}$. By the properties of MDS codes [50], it is easy to verify that the generator matrix of any $[K, K-D+L]$ MDS code satisfies this requirement. Note, however, that not any such matrix is guaranteed to satisfy the recoverability condition. Indeed, for satisfying the recoverability condition, it is required that G, as a generator matrix, generates a code that contains L codewords with the support W, and the coordinates of these codewords (indexed by W) must conform to the coefficient matrix V. To construct a matrix G that satisfies these requirements, the user proceeds as follows.

First, the user constructs the parity-check matrix Λ of the $[D, L]$ MDS code generated by the matrix V. Since V is an MDS matrix, then Λ generates a $[D, D-L]$ MDS code (i.e., the dual of the MDS code generated by V).

The user then constructs a $(D-L) \times K$ matrix H that satisfies the following two conditions:

(i) The submatrix of H restricted to columns indexed by W (and all rows) is Λ, and

(ii) The matrix H is MDS.

Since Λ is an MDS matrix, constructing H reduces to extending the $[D, D-L]$ MDS code generated by Λ to a $[K, D-L]$ MDS code. Recall that extending a code is performed by adding new columns to the generator matrix of the code. Next, the user constructs a $(K-D+L) \times K$ matrix G that generates the MDS code defined by the parity-check matrix H. Note that H generates a $[K, D-L]$ MDS code, and hence, H is the parity-check matrix of a $[K, K-D+L]$ MDS code. The user sends the matrix G as the query $\mathrm{Q}^{[\mathrm{W},\mathrm{V}]}$ to the server.

In the following, we describe how to explicitly construct the matrix G when the coefficient matrix V generates a GRS code, i.e., the entry (i, j) of V is given by $\mathrm{V}_{i,j} \triangleq \nu_j \omega_j^{i-1}$, where $\nu_1, \ldots, \nu_D$ are D elements from $\mathbb{F}_p \setminus \{0\}$, and $\omega_1, \ldots, \omega_D$ are D distinct elements from $\mathbb{F}_p$. The

parameters $\nu_1, \ldots, \nu_D$ and $\omega_1, \ldots, \omega_D$ are the multipliers and the evaluation points of the GRS code generated by V, respectively. Since the dual of a GRS code is a GRS code [50], the parity-check matrix Λ of the GRS code generated by V is a $(D-L) \times D$ matrix whose entry (i,j) is given by $\Lambda_{i,j} \triangleq \lambda_j \omega_j^{i-1}$, where

$$\lambda_j \triangleq \nu_j^{-1} \prod_{k \in \{1,\ldots,D\} \setminus \{j\}} (\omega_j - \omega_k)^{-1}.$$

Note that $\lambda_1, \ldots, \lambda_D$ are nonzero. Extending the $(D-L) \times D$ matrix Λ to a $(D-L) \times K$ matrix H—satisfying the conditions (i) and (ii) specified earlier—is performed as follows.

Let $\mathrm{W} = \{i_1, \ldots, i_D\}$ and $\mathcal{K} \setminus \mathrm{W} = \{i_{D+1}, \ldots, i_K\}$, and let π be a permutation on $\mathcal{K}$ such that $\pi(j) = i_j$. Let $\lambda_{D+1}, \ldots, \lambda_K$ be $K-D$ elements chosen randomly (with replacement) from $\mathbb{F}_p \setminus \{0\}$, and let $\omega_{D+1}, \ldots, \omega_K$ be $K-D$ elements chosen randomly (without replacement) from $\mathbb{F}_p \setminus \{\omega_1, \ldots, \omega_D\}$. For every $j \in \{1, \ldots, D\}$, let the $\pi(j)$th column of H be the jth column of Λ, and for every $j \in \mathcal{K} \setminus \{1, \ldots, D\}$, let the $\pi(j)$th column of H be $[\lambda_j, \lambda_j \omega_j, \ldots, \lambda_j \omega_j^{D-L-1}]^{\mathsf{T}}$.

Since H is the parity-check matrix of a $[K, K-D+L]$ GRS code, the generator matrix of this code, denoted by G, can be simply constructed by taking the $\pi(j)$th column of G to be $[\alpha_j, \alpha_j \omega_j, \ldots, \alpha_j \omega_j^{K-D+L-1}]^{\mathsf{T}}$, where

$$\alpha_j \triangleq \lambda_j^{-1} \prod_{k \in \mathcal{K} \setminus \{j\}} (\omega_j - \omega_k)^{-1}.$$

Note that the parameters $\{\alpha_j\}_{j \in \mathcal{K}}$ and $\{\omega_j\}_{j \in \mathcal{K}}$ are the multipliers and the evaluation points of the GRS code generated by the matrix G, respectively.

Step 2: Given the query $\mathrm{Q}^{[\mathrm{W},\mathrm{V}]}$, i.e., the matrix G, the server computes $\mathrm{y} \triangleq \mathrm{GX}$, and sends the vector y back to the user as the answer $\mathrm{A}^{[\mathrm{W},\mathrm{V}]}$. In particular, when V generates a GRS code, the ith entry of the vector $\mathrm{y} = [y_1, \ldots, y_{K-D+L}]^{\mathsf{T}}$ is given by

$$y_i = \sum_{j \in \mathcal{K}} \alpha_j \omega_j^{i-1} X_j.$$

Step 3: Upon receiving the answer $\mathrm{A}^{[\mathrm{W},\mathrm{V}]}$, i.e., the vector $\mathrm{y} = \mathrm{GX}$, the user constructs a matrix $[\tilde{\mathrm{G}}, \tilde{\mathrm{y}}]$ by performing row operations on the augmented matrix $[\mathrm{G}, \mathrm{y}]$, so as to zero out the submatrix formed by the first L rows and the columns indexed by $\mathcal{K} \setminus \mathrm{W}$. Since the submatrix of $[\tilde{\mathrm{G}}, \tilde{\mathrm{y}}]$ formed by the first L rows and the columns indexed by W (or $\mathcal{K} \setminus \mathrm{W}$) is the matrix V (or an all-zero matrix), the lth component of the demand vector $\mathrm{Z}^{[\mathrm{W},\mathrm{V}]}$, i.e., $\mathrm{v}_l^{\mathsf{T}}\mathrm{X}_{\mathrm{W}}$, can be recovered as the lth entry of the vector $\tilde{\mathrm{y}}$.

In the case that V generates a GRS code, $\mathrm{Z}^{[\mathrm{W},\mathrm{V}]}$ can be recovered from the vector y as follows. First, the user constructs L polynomials $f_1(x), \ldots, f_L(x)$, where

$$f_l(x) \triangleq x^{l-1} \prod_{j \in \mathcal{K} \setminus \{1,\ldots,D\}} (x - \omega_j).$$

Let $\mathrm{c}_l \triangleq [c_{l,1}, \ldots, c_{l,K-D+L}]^{\mathsf{T}}$, where $c_{l,i}$ is the coefficient of the monomial x^{i-1} in the expansion of $f_l(x)$. The user then recovers $\mathrm{v}_l^{\mathsf{T}}\mathrm{X}_{\mathrm{W}}$ for $1 \leq l \leq L$ by computing $\mathrm{c}_l^{\mathsf{T}}\mathrm{y}$.

Lemma 3. *The Specialized MDS Code protocol is a linear JPLT protocol, and achieves the rate* $L/(K - D + L)$.

Proof. Since the answer $\mathrm{y} = \mathrm{GX}$ is a vector of length $K - D + L$, and the entries of this vector are linearly independent coded combinations of the messages $\mathbf{X}_1, \ldots, \mathbf{X}_K$ (noting that the matrix G has full row rank), the entropy of the answer is given by $(K - D + L)\theta$, where θ is the entropy of a message. Thus, the rate of this protocol is $L/(K - D + L)$.

Since G generates a $[K, K - D + L]$ MDS code with minimum distance $D - L + 1$, it is easy to verify that the joint privacy condition is satisfied. By the properties of MDS codes [50], an $[n, k]$ MDS code (with minimum distance $n - k + 1$) satisfies the following combinatorial condition: for any $n - k + 1 \leq d \leq n$ and any d-subset $\mathcal{I} \subseteq \{1, \ldots, n\}$, the code space contains a unique $(d - n + k)$-dimensional subspace on the coordinates indexed by $\mathcal{I}$, and any basis of this subspace forms an MDS matrix. This implies that the row space of the matrix G contains a unique L-dimensional subspace on every D-subset of coordinates, and each of these subspaces is equally likely to be the subspace spanned by the user's demand, from the server's perspective. Thus, given

the query (i.e., the matrix G), every D-subset of message indices is equally likely to be the support index set of the demand.

The recoverability follows readily from the construction. Let $\mathrm{U} \triangleq [\mathrm{u}_1, \ldots, \mathrm{u}_L]^{\mathsf{T}}$, where u_l is a column-vector of length K such that the vector u_l restricted to its components indexed by W is the vector v_l, and the rest of the components of the vector u_l are all zero. Note that V is the submatrix of U formed by columns indexed by W. We need to show that the rows of U are L codewords of the code generated by G. Since H is the parity-check matrix of the code generated by G, this is equivalent to showing that UH^{T} is an all-zero matrix. Firstly, the submatrix of UH^{T} restricted to columns indexed by W is given by $\mathrm{V}\Lambda^{\mathsf{T}}$, and $\mathrm{V}\Lambda^{\mathsf{T}}$ is an all-zero matrix because Λ is the parity-check matrix of the code generated by V. Secondly, the submatrix of UH^{T} formed by the columns indexed by $\mathcal{K} \setminus \mathrm{W}$ is an all-zero matrix because the submatrix of U restricted to these columns is an all-zero matrix. Thus, UH^{T} is an all-zero matrix, as was to be shown. □

3.5 An Example of the Proposed JPLT Protocol

Consider a scenario where the server has $K = 10$ messages $\mathrm{X}_1, \ldots, \mathrm{X}_{10} \in \mathbb{F}_{11}$, and the user wants to compute $L = 2$ linear combinations of $D = 5$ messages X_2, X_4, X_5, X_7, X_8, say,

$$Z_1 = X_2 + 3X_4 + 2X_5 + X_7 + 6X_8,$$
$$Z_2 = 3X_2 + 10X_4 + 7X_5 + 4X_7 + 8X_8.$$

Note that for this example, the demand's support index set is given by $\mathrm{W} = \{2, 4, 5, 7, 8\}$, and the demand's coefficient matrix is given by

$$\mathrm{V} = \begin{bmatrix} 1 & 3 & 2 & 1 & 6 \\ 3 & 10 & 7 & 4 & 8 \end{bmatrix}.$$

It is easy to verify that V generates a $[5, 2]$ GRS code with the nonzero multipliers $\{\nu_1, \ldots, \nu_5\} = \{1, 3, 2, 1, 6\}$ and the evaluation points $\{\omega_1, \ldots, \omega_5\} = \{3, 7, 9, 4, 5\}$. Thus, the user can obtain

the parity-check matrix Λ of this code as

$$\Lambda = \begin{bmatrix} 3 & 10 & 8 & 8 & 7 \\ 9 & 4 & 6 & 10 & 2 \\ 5 & 6 & 10 & 7 & 10 \end{bmatrix}.$$

Note that Λ generates a $[5, 3]$ MDS code with the nonzero multipliers $\{\lambda_1, \ldots, \lambda_5\} = \{3, 10, 8, 8, 7\}$ and the evaluation points $\{\omega_1, \ldots, \omega_5\} = \{3, 7, 9, 4, 5\}$.

Next, the user extends the 3×5 matrix Λ to a 3×10 matrix H that satisfies the conditions (i) and (ii) specified in the step 1 of the protocol. Suppose the user randomly chooses 6 additional nonzero multipliers $\{\lambda_6, \ldots, \lambda_{10}\} = \{3, 5, 1, 1, 4\}$ (from $\mathbb{F}_{11} \setminus \{0\}$) and 6 additional evaluation points $\{\omega_6, \ldots, \omega_{10}\} = \{6, 1, 10, 2, 8\}$ (from $\mathbb{F}_{11} \setminus \{\omega_1, \ldots, \omega_5\}$). Followed by constructing a permutation π as described in the step 1 of the protocol, say, $\{\pi(1), \ldots, \pi(10)\} = \{2, 4, 5, 7, 8, 1, 3, 6, 9, 10\}$, the user constructs the matrix H as

$$\mathrm{H} = \begin{bmatrix} 3 & \mathbf{3} & 5 & \mathbf{10} & \mathbf{8} & 1 & \mathbf{8} & \mathbf{7} & 1 & 4 \\ 7 & \mathbf{9} & 5 & \mathbf{4} & \mathbf{6} & 10 & \mathbf{10} & \mathbf{2} & 2 & 10 \\ 9 & \mathbf{5} & 5 & \mathbf{6} & \mathbf{10} & 1 & \mathbf{7} & \mathbf{10} & 4 & 3 \end{bmatrix},$$

where the columns indexed by $\pi(1), \pi(2), \pi(3), \pi(4), \pi(5)$ (i.e., $2, 4, 5, 7, 8$) correspond to the columns $1, 2, 3, 4, 5$ of Λ, respectively, and the columns indexed by $\pi(6), \pi(7), \pi(8), \pi(9), \pi(10)$ (i.e., $1, 3, 6, 9, 10$) correspond to the columns of the generator matrix of a $[5, 3]$ GRS code with the nonzero multipliers $\{\lambda_6, \ldots, \lambda_{10}\}$ and the evaluation points $\{\omega_6, \ldots, \omega_{10}\}$. That is, for every $i \in \{6, \ldots, 10\}$, the $\pi(i)$th column of H is given by $[\lambda_i, \lambda_i\omega_i, \lambda_i\omega_i^2]^{\mathsf{T}}$.

Since H generates a $[10, 3]$ GRS code with the nonzero multipliers $\{\lambda_1, \ldots, \lambda_{10}\}$ and the evaluation points $\{\omega_1, \ldots, \omega_{10}\}$, H can be thought of as the parity-check matrix of a $[10, 7]$ GRS code with the nonzero multipliers $\{\alpha_1, \ldots, \alpha_{10}\} = \{10, 7, 4, 5, 4, 9, 2, 1, 4, 4\}$ and the evaluation points $\{\omega_1, \ldots, \omega_{10}\} = \{6, 3, 1, 7, 9, 10, 4, 5, 2, 8\}$. (The process of computing α_i's is explained in the

step 1 of the protocol.) The user then obtains the generator matrix G of this code,

$$\mathrm{G} = \begin{bmatrix} 9 & 10 & 2 & 7 & 4 & 1 & 5 & 4 & 4 & 4 \\ 10 & 8 & 2 & 5 & 3 & 10 & 9 & 9 & 8 & 10 \\ 5 & 2 & 2 & 2 & 5 & 1 & 3 & 1 & 5 & 3 \\ 8 & 6 & 2 & 3 & 1 & 10 & 1 & 5 & 10 & 2 \\ 4 & 7 & 2 & 10 & 9 & 1 & 4 & 3 & 9 & 5 \\ 2 & 10 & 2 & 4 & 4 & 10 & 5 & 4 & 7 & 7 \\ 1 & 8 & 2 & 6 & 3 & 1 & 9 & 9 & 3 & 1 \end{bmatrix}.$$

Then, the user sends the matrix G as the query to the server.

Upon receiving the query, i.e., the matrix G, the server computes the vector $\mathrm{y} = \mathrm{GX}$, and sends it back to the user. The user then constructs two polynomials

$$\begin{aligned} f_1(x) &= (x-\omega_6)(x-\omega_7)(x-\omega_8)(x-\omega_9)(x-\omega_{10}) \\ &= (x-6)(x-1)(x-10)(x-2)(x-8), \end{aligned}$$

and $f_2(x) = xf_1(x)$ (for more details, see the step 3 of the protocol). It is easy to verify that the coefficient vectors of the polynomials $f_1(x)$ and $f_2(x)$ are given by $\mathrm{c}_1 = [8,1,8,9,6,1,0]^{\mathsf{T}}$ and $\mathrm{c}_2 = [0,8,1,8,9,6,1]^{\mathsf{T}}$, respectively. Then, the user recovers their demand, i.e., Z_1 and Z_2, by computing

$$\begin{aligned} Z_1 &= \mathrm{c}_1^{\mathsf{T}}\mathrm{y} = X_2 + 3X_4 + 2X_5 + X_7 + 6X_8, \\ Z_2 &= \mathrm{c}_2^{\mathsf{T}}\mathrm{y} = 3X_2 + 10X_4 + 7X_5 + 4X_7 + 8X_8. \end{aligned}$$

Note that for this example, the rate of the proposed protocol is $L/(K-D+L) = 2/7$, whereas the rate of a PIR-based scheme or a PLC-based scheme is $L/K = 2/10$ or $1/(K-D) = 1/5$, respectively.

3.6 Main Results for Extensions of the JPLT Setting

3.6.1 JPLT with Side Information

The result of Theorem 1 can be extended to JPLT with Side Information (SI). First, consider the case of Uncoded SI (USI), where the user initially knows a random subset of M messages (different from those indexed by W) and the identities of these messages are initially unknown to the server. Using similar techniques as in this work, we can show that the capacity of JPLT with USI is given by $L/(K-D-M+L)$, when the identities of the messages in the support set of the demand and those in the support set of the side information need to be protected jointly. In addition, under the same privacy condition, we can show that the capacity of JPLT with Coded SI (CSI) does not change, provided that the coefficient matrix of the side information and that of the demand form an MDS matrix when concatenated horizontally. Recall that in the case of CSI the user initially knows L randomly generated MDS coded combinations of M randomly chosen messages, and the identities of these M messages are not initially known by the server. This result is interesting, particularly when $L < M$, because it shows that for achieving joint privacy, only L ($< M$) MDS coded combinations of M messages is as efficient as M uncoded messages as SI.

To prove the converse it suffices to show that the rate of any JPLT-USI protocol is upper bounded by $L/(K-D-M+L)$. This upper bound naturally serves as an upper bound on the rate of any JPLT-CSI protocol, noting that any coded SI on M given messages can be obtained from an uncoded SI consisting of these M messages.

Lemma 4. *The rate of any JPLT-USI protocol for K messages, demand's support size D, demand's dimension L, and the side information's support size M, is upper bounded by $L/(K-D-M+L)$.*

The proof of Lemma 4 relies on a necessary condition for any JPLT-USI protocol stated in the following lemma. (The proof is similar to that of Lemma 1, and hence omitted to avoid repetition.)

Lemma 5. *Given any JPLT-USI protocol, for any $\mathrm{W}^* \in \mathcal{W}$ and any M-subset $\mathrm{S}^* \subseteq \mathcal{K} \setminus \mathrm{W}^*$, there must exist $\mathrm{V}^* \in \mathcal{V}$, such that*

$$H(\mathbf{Z}^{[\mathrm{W}^*,\mathrm{V}^*]}|\mathbf{A},\mathbf{Q},\mathbf{X}_{\mathrm{S}^*}) = 0.$$

Proof of Lemma 4. Consider an arbitrary JPLT-USI protocol. We need to show that $H(\mathbf{A}) \geq (K-D-M+L)\theta$. For simplifying the notation, let $T \triangleq K-D-M+1$, and let $\mathrm{W}_i \triangleq \{i+M, i+M+1, \ldots, i+D+M-1\}$ for any $1 \leq i \leq T$. Also, let $\mathrm{S} \triangleq \{1, \ldots, M\}$. Note that S and W_i (for $1 \leq i \leq T$) are disjoint. By the result of Lemma 5, for any $1 \leq i \leq T$, there exists $\mathrm{V}_i \in \mathcal{V}$ such that $H(\mathbf{Z}_i|\mathbf{A}, \mathbf{Q}, \mathbf{X}_\mathrm{S}) = 0$, where $\mathbf{Z}_i \triangleq \mathbf{Z}^{[\mathrm{W}_i, \mathrm{V}_i]}$. This readily implies that $H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T|\mathbf{A}, \mathbf{Q}, \mathbf{X}_\mathrm{S}) = 0$. Similar to the proof of Lemma 2, we can show that

$$
\begin{aligned}
H(\mathbf{A}) &\geq H(\mathbf{A}|\mathbf{Q}, \mathbf{X}_\mathrm{S}) + H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T|\mathbf{Q}, \mathbf{A}, \mathbf{X}_\mathrm{S}) && (3.5)\\
&= H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T|\mathbf{Q}, \mathbf{X}_\mathrm{S}) + H(\mathbf{A}|\mathbf{Q}, \mathbf{X}_\mathrm{S}, \mathbf{Z}_1, \ldots, \mathbf{Z}_T) && (3.6)\\
&\geq H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T), && (3.7)
\end{aligned}
$$

where (3.5) holds because $H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T|\mathbf{A}, \mathbf{Q}, \mathbf{X}_\mathrm{S}) = 0$; (3.6) follows from the chain rule of conditional entropy; and (3.7) holds because (i) $\mathbf{Z}_i$'s are independent from $(\mathbf{Q}, \mathbf{X}_\mathrm{S})$, noting that Q does not depend on the content of the messages, and $\mathbf{X}_\mathrm{S}$ is independent from the messages $\mathbf{X}_{M+1}, \ldots, \mathbf{X}_K$ that constitute the support sets of $\mathbf{Z}_1, \ldots, \mathbf{Z}_T$ since S and W_i are disjoint for all $1 \leq i \leq T$, and (ii) $H(\mathbf{A}|\mathbf{Q}, \mathbf{X}_\mathrm{S}, \mathbf{Z}_1, \ldots, \mathbf{Z}_T) \geq 0$. The rest of the proof is the same as that in the proof of Lemma 2 by showing that $H(\mathbf{Z}_1, \ldots, \mathbf{Z}_T) \geq L\theta + (T-1)\theta = (K-D-M+L)\theta$. □

The proof of achievability is based on a linear JPLT-CSI protocol, termed *Modified Specialized MDS Code protocol*, that achieves the rate $L/(K-D-M+L)$, provided that the $L \times (D+M)$ matrix formed by horizontally concatenating the demand's coefficient matrix and the side information's coefficient matrix, referred to as the *demand-side coefficient matrix*, is an MDS matrix. The Modified Specialized MDS Code protocol is similar to the Specialized MDS Code protocol for JPLT without SI—presented in Section 3.4, except where the demand's support index set W is replaced by the union of the support set of the demand and the support set of the side information, the demand's coefficient matrix V is replaced by the demand-side coefficient matrix, and D is replaced by $D+M$. Note that the Modified Specialized MDS Code protocol can also serve as an JPLT-USI protocol, since the user can always construct a coded SI from an uncoded SI.

Lemma 6. *The Modified Specialized MDS Code protocol is a linear JPLT-CSI protocol, and achieves the rate $L/(K-D-M+L)$, provided that the demand-side coefficient matrix is MDS.*

Proof. The proof follows from the exact same lines as those in the proof of Lemma 3, and hence omitted to avoid repetition. □

3.6.2 JPLT with Non-MDS Coefficient Matrices

The result of Theorem 1 can also be extended to the JPLT setting in which the coefficient matrix of the demand, V, is not MDS. In particular, when the matrix V is randomly chosen from the set of all $L \times D$ matrices that have full row rank and nonzero columns, using the same proof technique as the one in Section 3.3 for the case of MDS matrices, we can show that the rate of any JPLT protocol is upper bound by $L/(K-D+L)$. However, the achievability of this rate upper bound remains unknown in general. On the other hand, when the matrix V is randomly chosen from the set of all $L \times D$ matrices that have full row rank—regardless of containing any all-zero columns or not, using a JPLT protocol similar to the Specialized MDS Code protocol for the case of MDS matrices, we can achieve the rate $L/(K-D+L)$. This rate, however, may not be optimal, and the converse is still open. In the following, we present a sketch of the achievability scheme.

First, the user constructs an $L \times K$ matrix U such that the columns of U indexed by the demand's support index set W are the columns of the matrix V, and the rest of the columns of U are all zero. Next, the user constructs a $(K-D+L) \times K$ matrix $\tilde{\mathrm{G}}$ by vertically concatenating the $L \times K$ matrix U and a randomly generated $(K-D) \times K$ MDS matrix M, i.e., $\tilde{\mathrm{G}} = [\mathrm{U}^{\mathsf{T}}, \mathrm{M}^{\mathsf{T}}]^{\mathsf{T}}$. The user then constructs a $(K-D+L) \times K$ matrix G by multiplying $\tilde{\mathrm{G}}$ by a randomly generated $(K-D+L) \times (K-D+L)$ invertible matrix R, i.e., $\mathrm{G} = \mathrm{R}\tilde{\mathrm{G}}$, and sends the matrix G as the query to the server. Note that the matrix G, unlike the case of the Specialized MDS Code protocol, does not necessarily generate an MDS code. The server computes $\mathrm{y} = \mathrm{GX}$, and sends y back to the user as the answer. Given y, the user first computes $\tilde{\mathrm{y}} = [\tilde{y}_1, \ldots, \tilde{y}_{K-D+L}]^{\mathsf{T}} = \mathrm{R}^{-1}\mathrm{y}$, and then recovers the demand $\mathrm{VX_W} = [\tilde{y}_1, \ldots, \tilde{y}_L]^{\mathsf{T}}$. Using similar arguments as those in Section 3.4, it can be shown that this protocol satisfies both the joint privacy and recoverability conditions.

4. PRIVATE LINEAR TRANSFORMATION WITH INDIVIDUAL PRIVACY (IPLT)

In this chapter, we begin by presenting a necessary condition for IPLT protocols in Section 4.1. Section 4.2 summarizes our main results for the IPLT setting. The proof of converse and the achievability scheme are presented in Section 4.3 and Section 4.4, respectively. To further illustrate the proposed achievability scheme, two examples are provided in Section 4.5. Lastly, in Section 4.6, we briefly outline our results for the extensions of the IPLT setting.

4.1 A Necessary Condition for IPLT Protocols

The individual privacy and recoverability conditions yield a necessary (but not sufficient) condition for any IPLT protocol, stated in the following lemma.

Lemma 7. *Given any IPLT protocol, for any $i \in \mathcal{K}$, there must exist $\mathrm{W}^* \in \mathcal{W}$ with $i \in \mathrm{W}^*$, and $\mathrm{V}^* \in \mathcal{V}$, such that*

$$H(\mathbf{Z}^{[\mathrm{W}^*,\mathrm{V}^*]}|\mathbf{A},\mathbf{Q}) = 0.$$

Proof. The proof is by the way of contradiction, and relies on the joint privacy and recoverability conditions. Consider an arbitrary $i \in \mathcal{K}$. Suppose that for none of $\mathrm{W}^* \in \mathcal{W}$ with $i \in \mathrm{W}^*$ there exists $\mathrm{V}^* \in \mathcal{V}$ such that $H(\mathbf{Z}^{[\mathrm{W}^*,\mathrm{V}^*]}|\mathbf{A},\mathbf{Q}) = 0$. This implies that X_i is not one of the messages in the support set of the demand (i.e., $i \notin \mathrm{W}$); otherwise, if $i \in \mathrm{W}$, the recoverability condition is not satisfied. This, however, violates the individual privacy condition, because given the query, every message must be equally likely to be one of the messages in the demand's support set. □

The result of Lemma 7 establishes a connection between *linear codes* with a certain constraint and *linear schemes* for IPLT, i.e., any scheme in which the server's answer to the user's query consists of only *linear* combinations of the messages. In particular, the matrix of combination coefficients—pertaining to the linear combinations in the answer, must be the generator matrix of a linear code of length K that satisfies the following condition: for any coordinate i, there must exist $K - D$ coordinates different from i such that the code resulting from puncturing at these

$K-D$ coordinates contains L codewords that are MDS. Note, however, that this condition is only necessary and not sufficient. A sufficient (yet not necessary) condition is that, for *every coordinate* i, the punctured codes resulting from puncturing at any $K-D$ other coordinates (different from i) collectively contain the *same number of groups* of L codewords that are MDS. Maximizing the rate of a linear IPLT scheme is then equivalent to minimizing the dimension of a linear code that satisfies this sufficient condition. Despite the fact that this sufficient condition is stronger than the necessary condition provided by Lemma 7, the former is more combinatorial, whereas the latter is more information-theoretic and hence more useful in the converse proof.

4.2 Main Results for the IPLT Setting

Theorem 2. *For the IPLT setting with K messages, demand's support size D, and demand's dimension L, the capacity is lower and upper bounded by*

$$\left(\left\lfloor\frac{K}{D}\right\rfloor+\min\left\{\frac{R}{S},\frac{R}{L}\right\}\right)^{-1} \quad \textit{and} \quad \left(\left\lfloor\frac{K}{D}\right\rfloor+\min\left\{1,\frac{R}{L}\right\}\right)^{-1},$$

respectively, where $R \triangleq K \pmod D$ and $S \triangleq \gcd(D+R,R)$. Moreover, the lower and upper bounds match when $R \leq L$ or $R \mid D$.

To prove the converse bound, we use the necessary condition for IPLT protocols provided by Lemma 7 along with information-theoretic arguments, and formulate the problem as an integer linear programming (ILP) problem. Solving this ILP, we obtain the upper bound on the capacity. The lower bound on the capacity is proven by constructing an IPLT protocol, called *Generalized Partition-and-Code with Partial Interference Alignment*. This protocol is a generalization of the protocols previously proposed in [39] and [17] for single-server PIR and PLC (without SI) with individual privacy. The main ingredients of the proposed protocol are as follows: (i) constructing a properly designed family of subsets of messages, where some subsets are possibly overlapping, and (ii) designing a number of linear combinations for each subset, where the linear combinations pertaining to the overlapping subsets are partially aligned.

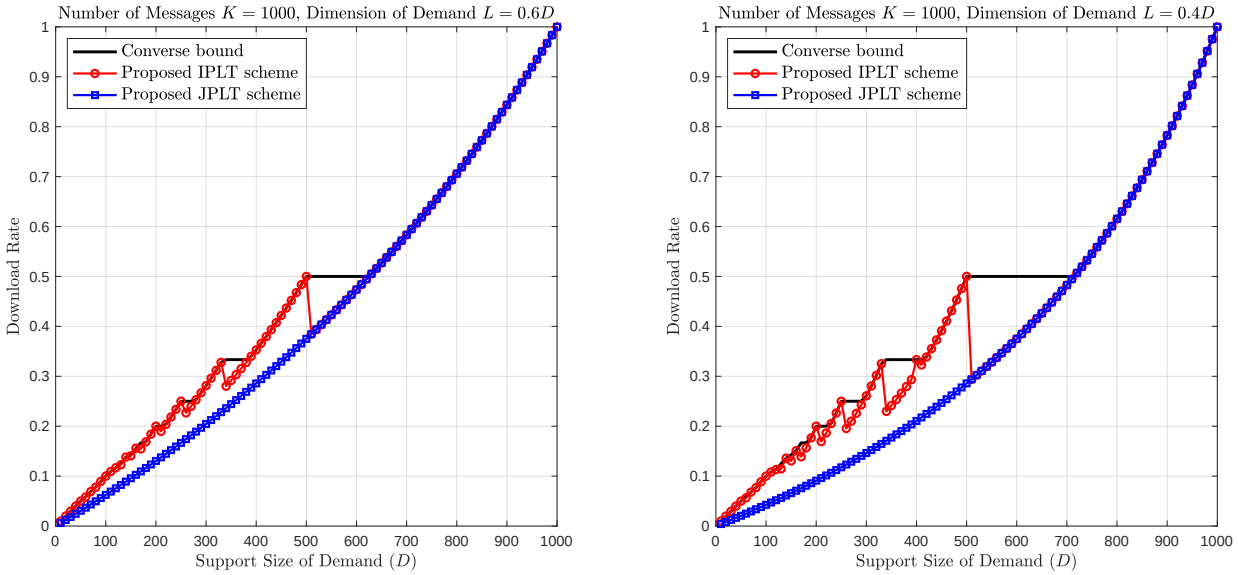

Figure 4.1: The download rate of the proposed IPLT and JPLT schemes.

Remark 3. As shown in [17], the capacity of PLC with individual privacy, which is a special case of IPLT for $L = 1$, is given by $\lceil \frac{K}{D+M} \rceil^{-1}$, where the user initially knows M (> 0) uncoded messages or one linear combination of M messages as side information. The capacity of this setting was, however, left open for $M = 0$. Theorem 2 provides a lower bound $(\lfloor \frac{K}{D} \rfloor + \min\{\frac{R}{S}, R\})^{-1}$ and an upper bound $(\lfloor \frac{K}{D} \rfloor + \min\{1, R\})^{-1}$ on the capacity of this setting. Interestingly, these bounds are matching when $R = 0$ or $R \mid D$, settling the capacity of PLC (without SI) with individual privacy for these cases. For $L = D$, IPLT reduces to PIR (without SI) with individual privacy, and an optimal scheme in this case is to download the entire dataset [39]. This is consistent with the result of Theorem 2 for $L = D$.

Remark 4. Naturally, the JPLT scheme proposed in Section 3.4 is also applicable as an IPLT scheme. This comes from the fact that joint privacy is a stronger notion that implies individual privacy. In Fig. 4.1, we compare the performance of the proposed IPLT and JPLT schemes for different values of $D \in \{10, 20, \ldots, 1000\}$, where $K = 1000$, and $L/D = 0.6$ (left plot) or $L/D = 0.4$ (right plot). One can observe that, when the ratio L/D is fixed, for sufficiently small values of D, the download rate of the proposed IPLT scheme is higher than that of the proposed

JPLT scheme; whereas, for values of D larger than a threshold, both schemes achieve the same rate. This implies that for sufficiently large D, achieving individual privacy is as costly as achieving joint privacy. In addition, for some values of D, the rate achieved by the proposed IPLT scheme matches the converse bound. This, in turn, confirms the optimality of the proposed IPLT scheme for such values of D. By comparing the left and right plots in Fig. 4.1, it can also be seen that for a sufficiently small value of D, the smaller is the ratio L/D, the better is the performance of the proposed IPLT scheme as compared to that of the proposed JPLT scheme. For instance, for $D = 250$, the rate of the proposed IPLT scheme is about 33% and 53% more than that of the proposed JPLT scheme for $L/D = 0.6$ and $L/D = 0.4$, respectively.

4.3 Proof of Converse

Lemma 8. *The rate of any IPLT protocol for K messages, demand's support size D, and demand's dimension L, is upper bounded by $(\lfloor \frac{K}{D} \rfloor + \min\{1, \frac{R}{L}\})^{-1}$, where $R \triangleq K \pmod D$.*

Proof. Consider an arbitrary IPLT protocol that generates the query-answer pair $(\mathrm{Q}^{[\mathrm{W},\mathrm{V}]}, \mathrm{A}^{[\mathrm{W},\mathrm{V}]})$ for any given (W, V). To prove the rate upper bound in the lemma, we need to show that

$$H(\mathbf{A}) \geq \left(L \left\lfloor \frac{K}{D} \right\rfloor + \min\{L, R\} \right) \theta.$$

Recall that $\mathbf{A}$ denotes $\mathbf{A}^{[\mathbf{W},\mathbf{V}]}$, and θ is the entropy of a message. Consider an arbitrary message index $k_1 \in \mathcal{K}$. By the result of Lemma 7, there exist $\mathrm{W}_1 \in \mathcal{W}$ with $k_1 \in \mathrm{W}_1$, and $\mathrm{V}_1 \in \mathcal{V}$ such that $H(\mathbf{Z}_1|\mathbf{A}, \mathbf{Q}) = 0$, where $\mathbf{Z}_1 \triangleq \mathbf{Z}^{[\mathrm{W}_1,\mathrm{V}_1]}$. By the same arguments as in the proof of Lemma 2 (see Section 3.3), we have

$$\begin{aligned} H(\mathbf{A}) &\geq H(\mathbf{A}|\mathbf{Q}) + H(\mathbf{Z}_1|\mathbf{A}, \mathbf{Q}) \\ &= H(\mathbf{Z}_1|\mathbf{Q}) + H(\mathbf{A}|\mathbf{Q}, \mathbf{Z}_1) \\ &= H(\mathbf{Z}_1) + H(\mathbf{A}|\mathbf{Q}, \mathbf{Z}_1) \end{aligned} \tag{4.1}$$

To further lower bound $H(\mathbf{A}|\mathbf{Q}, \mathbf{Z}_1)$, we proceed as follows. Take an arbitrary message index

$k_2 \notin \mathrm{W}_1$. Again, by Lemma 7, there exist $\mathrm{W}_2 \in \mathcal{W}$ with $k_2 \in \mathrm{W}_2$, and $\mathrm{V}_2 \in \mathcal{V}$ such that $H(\mathbf{Z}_2|\mathbf{A},\mathbf{Q}) = 0$, where $\mathbf{Z}_2 \triangleq \mathbf{Z}^{[\mathrm{W}_2,\mathrm{V}_2]}$. Using a similar technique as in (4.1), it follows that $H(\mathbf{A}|\mathbf{Q},\mathbf{Z}_1) \geq H(\mathbf{Z}_2|\mathbf{Q},\mathbf{Z}_1) + H(\mathbf{A}|\mathbf{Q},\mathbf{Z}_1,\mathbf{Z}_2)$, and consequently,

$$H(\mathbf{A}|\mathbf{Q},\mathbf{Z}_1) \geq H(\mathbf{Z}_2|\mathbf{Z}_1) + H(\mathbf{A}|\mathbf{Q},\mathbf{Z}_2,\mathbf{Z}_1). \tag{4.2}$$

Combining (4.1) and (4.2), we get

$$H(\mathbf{A}) \geq H(\mathbf{Z}_1) + H(\mathbf{Z}_2|\mathbf{Z}_1) + H(\mathbf{A}|\mathbf{Q},\mathbf{Z}_2,\mathbf{Z}_1). \tag{4.3}$$

We repeat this lower-bounding process multiple rounds until there is no message index left to take. Let n be the total number of rounds, and let $k_1,\ldots,k_n$ be the message indices chosen over the rounds. For every $i \in \{1,\ldots,n\}$, let $\mathrm{W}_i \in \mathcal{W}$ with $k_i \in \mathrm{W}_i$ and $k_i \notin \cup_{1\leq j<i}\mathrm{W}_j$, and $\mathrm{V}_i \in \mathcal{V}$, be such that $H(\mathbf{Z}_i|\mathbf{A},\mathbf{Q}) = 0$, where $\mathbf{Z}_i \triangleq \mathbf{Z}^{[\mathrm{W}_i,\mathrm{V}_i]}$. (For any $i \in \{1,\ldots,n\}$, the existence of W_i and V_i is guaranteed by the result of Lemma 7.) Note that $\cup_{1\leq i\leq n}\mathrm{W}_i = \mathcal{K}$. Similarly as before, we can show that

$$H(\mathbf{A}) \geq \sum_{1\leq i\leq n} H(\mathbf{Z}_i|\mathbf{Z}_{i-1},\ldots,\mathbf{Z}_1) + H(\mathbf{A}|\mathbf{Q},\mathbf{Z}_n,\ldots,\mathbf{Z}_1) \geq \sum_{1\leq i\leq n} H(\mathbf{Z}_i|\mathbf{Z}_{i-1},\ldots,\mathbf{Z}_1). \tag{4.4}$$

Let $\mathbf{Z}_{i,1},\ldots,\mathbf{Z}_{i,L}$ be the components of $\mathbf{Z}_i$, where $\mathbf{Z}_{i,l} \triangleq \mathrm{v}_{i,l}^{\mathsf{T}}\mathbf{X}_{\mathrm{W}_i}$, and $\mathrm{v}_{i,l}^{\mathsf{T}}$ is the lth row of V_i. Next, we show that

$$H(\mathbf{Z}_i|\mathbf{Z}_{i-1},\ldots,\mathbf{Z}_1) \geq \min\{N_i, L\}\theta, \tag{4.5}$$

where $N_i \triangleq |\mathrm{W}_i \setminus \cup_{1\leq j<i}\mathrm{W}_j|$ is the number of message indices that belong to W_i, but not $\cup_{1\leq j<i}\mathrm{W}_j$. (Note that $N_1 = |\mathrm{W}_1| = D$.) This is equivalent to showing that $\mathbf{Z}_i$ contains $M_i \triangleq \min\{N_i, L\}$ components that are independent of the components of $\mathbf{Z}_1,\ldots,\mathbf{Z}_{i-1}$. Note that the components of $\mathbf{Z}_1,\ldots,\mathbf{Z}_i$ are linear combinations of the messages $\mathbf{X}_1,\ldots,\mathbf{X}_K$. Let $\mathrm{u}_{i,l}$ be a column-vector of length K such that the vector $\mathrm{u}_{i,l}$ restricted to its components indexed by W_i is the vector $\mathrm{v}_{i,l}$, and the rest of the components of the vector $\mathrm{u}_{i,l}$ are all zero, and let $\mathrm{U}_i \triangleq [\mathrm{u}_{i,1},\ldots,\mathrm{u}_{i,L}]^{\mathsf{T}}$. Thus, we

need to show that the matrix U_i contains M_i rows that are linearly independent of the rows of the matrices $\mathrm{U}_1, \ldots, \mathrm{U}_{i-1}$. Note that the rows of the matrix U_i are linearly independent, because U_i contains V_i as a submatrix, and V_i is invertible.

Let S_i be an $L \times N_i$ submatrix of U_i formed by columns indexed by $\mathrm{W}_i \setminus \cup_{1 \leq j < i} \mathrm{W}_j$. Note that S_i is a submatrix of V_i, and every $L \times L$ submatrix of V_i is invertible. We consider two cases: (i) $N_i \leq L$, and (ii) $N_i > L$. In the case (i), the N_i columns of S_i are linearly independent. Otherwise, any $L \times L$ submatrix of V_i that contains S_i cannot be invertible, and hence a contradiction. In the case (ii), any L columns of S_i are linearly independent. Otherwise, S_i (and consequently, V_i) contains an $L \times L$ submatrix that is not invertible, which is a contradiction. By these arguments, the rank of S_i is $M_i = \min\{L, N_i\}$, and hence, S_i contains M_i linearly independent rows.

Without loss of generality, assume that the first M_i rows of S_i are linearly independent. Moreover, the submatrix of $[\mathrm{U}_1; \ldots; \mathrm{U}_{i-1}]$ restricted to its columns indexed by $\mathrm{W}_i \setminus \cup_{1 \leq j < i} \mathrm{W}_j$ (and all its rows) is an all-zero matrix, where $[\mathrm{U}_1; \ldots; \mathrm{U}_{i-1}]$ is a matrix formed by stacking $\mathrm{U}_1, \ldots, \mathrm{U}_{i-1}$ vertically. By these arguments, it follows that the first M_i rows of U_i are linearly independent of the rows of $[\mathrm{U}_1; \ldots; \mathrm{U}_{i-1}]$. This completes the proof of (4.5).

Combining (4.4) and (4.5), we have

$$H(\mathbf{A}) \geq \sum_{1 \leq i \leq n} \min\{L, N_i\}\theta \tag{4.6}$$

Recall that $N_i = |\mathrm{W}_i \setminus \cup_{1 \leq j < i} \mathrm{W}_j|$. Note that $1 \leq N_i \leq D$ since $\mathrm{W}_i \setminus \cup_{1 \leq j < i} \mathrm{W}_j$ is a subset of W_i, and the message index k_i belongs to $\mathrm{W}_i \setminus \cup_{1 \leq j < i} \mathrm{W}_j$. Moreover, $\sum_{i=1}^{n} N_i = K$ since W_1, $\mathrm{W}_2 \setminus \mathrm{W}_1$, $\ldots$, $\mathrm{W}_n \setminus \cup_{1 \leq j < n} \mathrm{W}_j$ form a partition of $\mathcal{K}$, and $|\mathrm{W}_1| = N_1 = D$, $|\mathrm{W}_2 \setminus \mathrm{W}_1| = N_2$, $\ldots$, $|\mathrm{W}_n \setminus \cup_{1 \leq j < n} \mathrm{W}_j| = N_n$. To obtain a converse bound, we need to minimize $\sum_{1 \leq i \leq n} \min\{L, N_i\}$, subject to the constraints (i) $N_1 = D$, and $1 \leq N_i \leq D$ for any $1 < i \leq n$, and (ii) $\sum_{1 \leq i \leq n} N_i = K$. To this end, we reformulate this optimization problem as follows. For every $j \in \{1, \ldots, D\}$, let $T_j \triangleq \sum_{1 \leq i \leq n} \mathbb{1}_{\{N_i = j\}}$ be the number of rounds i such that $N_i = j$. Using this notation, the objective function $\sum_{1 \leq i \leq n} \min\{L, N_i\}$ can be rewritten as $\sum_{1 \leq j \leq D} T_j \min\{L, j\}$, or equiv-

alently, $\sum_{1\le j\le L} T_j j + \sum_{L<j\le D} T_j L$; the constraint (i) reduces to $T_j \in \mathbb{N}_0 \triangleq \{0, 1, \dots\}$ for every $1 \le j < D$, and $T_D \in \mathbb{N} \triangleq \{1, 2, \dots\}$; and the constraint (ii) reduces to $\sum_{1\le j\le D} T_j j = K$. Thus, we need to solve the following integer linear programming (ILP) problem:

$$\begin{aligned} \text{minimize} \quad & \sum_{1\le j\le L} T_j j + \sum_{L<j\le D} T_j L \\ \text{subject to} \quad & \sum_{1\le j\le D} T_j j = K \\ & T_j \in \mathbb{N}_0, \quad \forall 1 \le j < D \\ & T_D \in \mathbb{N}. \end{aligned} \tag{4.7}$$

Solving this ILP using the Gomory's cutting-plane algorithm [51], it can be seen that an optimal solution is given by $T_D = \lfloor \frac{K}{D} \rfloor$, $T_R = 1$, and $T_j = 0$ for all $j \notin \{R, D\}$, where $R \triangleq K \pmod D$, and the optimal value is given by $L\lfloor \frac{K}{D} \rfloor + \min\{L, R\}$. This implies that

$$\sum_{1\le i\le n} \min\{L, N_i\} \ge L \left\lfloor \frac{K}{D} \right\rfloor + \min\{L, R\}. \tag{4.8}$$

Combining (4.6) and (4.8), we get $H(\mathbf{A}) \ge (L\lfloor \frac{K}{D} \rfloor + \min\{L, R\})\theta$, as was to be shown. □

4.4 Achievability Scheme

This section presents an IPLT protocol, called *Generalized Partition-and-Code with Partial Interference Alignment*, that achieves the rate $(\lfloor \frac{K}{D} \rfloor + \min\{\frac{R}{S}, \frac{R}{L}\})^{-1}$, where $R \triangleq K \pmod D$ and $S \triangleq \gcd(D + R, R)$. Two examples of the proposed IPLT protocol are given in Section 4.5.

In the description of the protocol, we denote by $\tilde{\mathrm{W}}$ a sequence of length D (instead of a set of size D) that the user initially constructs by randomly permuting the elements in the demand's support index set W, and denote by $\tilde{\mathrm{V}}$ an $L \times D$ matrix that the user initially constructs by applying the same permutation on the columns of the demand's coefficient matrix V.

We consider two different cases: (i) $L \le S$, and (ii) $L > S$. In each case, the protocol consists of three steps as follows.

Step 1: The user constructs a matrix G and a permutation π, and sends them as the query $\mathrm{Q}^{[\mathrm{W},\mathrm{V}]}$ to the server. In the following, we describe the construction of the matrix G and the permutation π for the cases (i) and (ii) separately.

Case (i):

Recall that in this case, $L \leq S$. Let $n \triangleq \lfloor \frac{K}{D} \rfloor - 1$, $m \triangleq \frac{R}{S} + 1$, and $t \triangleq \frac{D}{S} - 1$. The user constructs an $L(n+m) \times K$ matrix G,

$$\mathrm{G} = \begin{bmatrix} \mathrm{G}_1 & 0 & \dots & 0 & 0 \\ 0 & \mathrm{G}_2 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & \mathrm{G}_n & 0 \\ 0 & 0 & \dots & 0 & \mathrm{G}_{n+1} \end{bmatrix} \tag{4.9}$$

where $\mathrm{G}_1, \dots \mathrm{G}_n$ are $L \times D$ matrices, and G_{n+1} is an $Lm \times (D+R)$ matrix. The matrices (blocks) $\mathrm{G}_1, \dots, \mathrm{G}_n, \mathrm{G}_{n+1}$ are constructed as follows.

The user randomly selects one of the blocks $\mathrm{G}_1, \dots, \mathrm{G}_{n+1}$, where each of the blocks $\mathrm{G}_1, \dots, \mathrm{G}_n$ is selected with probability $\frac{D}{K}$, and the block G_{n+1} is selected with probability $\frac{D+R}{K}$. Let i^* be the index of the selected block. Depending on the choice of i^*, the description of the protocol is different. In the following, we consider the cases of $1 \leq i^* \leq n$ and $i^* = n+1$ separately.

First, consider the case of $1 \leq i^* \leq n$. In this case, the user takes G_{i^*} to be the matrix $\tilde{\mathrm{V}}$, i.e., $\mathrm{G}_{i^*} = \tilde{\mathrm{V}}$. For any $i \in \{1, \dots, n\} \setminus \{i^*\}$, the user takes G_i to be a randomly generated MDS matrix of size $L \times D$. The construction of G_{n+1} is as follows. First, the user randomly generates an MDS matrix C of size $L \times (D+R)$, and partitions the columns of C into $t+m$ column-blocks each of size $L \times S$, i.e., $\mathrm{C} = [\mathrm{C}_1, \dots, \mathrm{C}_{t+m}]$. Then, the user constructs $\mathrm{G}_{n+1} = [\mathrm{B}_1, \mathrm{B}_2]$,

$$\mathrm{B}_1 \triangleq \begin{bmatrix} \alpha_1 \omega_{1,1} \mathrm{C}_1 & \dots & \alpha_t \omega_{1,t} \mathrm{C}_t \\ \vdots & \vdots & \vdots \\ \alpha_1 \omega_{m,1} \mathrm{C}_1 & \dots & \alpha_t \omega_{m,t} \mathrm{C}_t \end{bmatrix}, \quad \mathrm{B}_2 \triangleq \begin{bmatrix} \alpha_{t+1} \mathrm{C}_{t+1} & & \\ & \ddots & \\ & & \alpha_{t+m} \mathrm{C}_{t+m} \end{bmatrix},$$

where $\alpha_1, \ldots, \alpha_{t+m}$ are $t+m$ randomly chosen elements from $\mathbb{F}_p \setminus \{0\}$, and $\omega_{i,j} \triangleq (x_i - y_j)^{-1}$ for $1 \leq i \leq m$ and $1 \leq j \leq t$, where $x_1, \ldots, x_m$ and $y_1, \ldots, y_t$ are $t+m$ distinct elements chosen at random from $\mathbb{F}_p$. Note that $\omega_{i,j}$ is the entry (i,j) of an $m \times t$ Cauchy matrix.

Now, consider the case of $i^* = n+1$. For any $i \in \{1, \ldots, n\}$, the user takes G_i to be a randomly generated MDS matrix of size $L \times D$. The user then constructs G_{n+1} with a structure similar to that in the previous case, but for a different choice of matrices $\mathrm{C}_1, \ldots, \mathrm{C}_{t+m}$ and parameters $\alpha_1, \ldots, \alpha_{t+m}$, as specified below.

First, the user partitions the columns of $\tilde{\mathrm{V}}$ into $t+1$ column-blocks each of size $L \times S$, i.e., $\tilde{\mathrm{V}} = [\tilde{\mathrm{V}}_1, \ldots, \tilde{\mathrm{V}}_{t+1}]$. The user then randomly chooses $t+1$ indices from $\{1, \ldots, t+m\}$, say, $i_1, \ldots, i_{t+1}$, and for any $1 \leq j \leq t+1$, takes $\mathrm{C}_{i_j} = \tilde{\mathrm{V}}_j$. Next, the user randomly generates the rest of C_i's such that $\mathrm{C} = [\mathrm{C}_1, \ldots, \mathrm{C}_{t+m}]$ is an MDS matrix. The choice of α_i's is described below.

Hereafter, we refer to the submatrix of G_{n+1} formed by the ith L rows as the ith row-block of G_{n+1}. Note that G_{n+1} has m row-blocks. Let s be the number of column-block indices i_j for $j \in \{1, \ldots, t+m\}$ such that $i_j > t$. Note that $\mathrm{C}_{i_1}, \ldots, \mathrm{C}_{i_{t-s+1}}$ belong to the matrix B_1, and $\mathrm{C}_{i_{t-s+2}}, \ldots, \mathrm{C}_{i_{t+1}}$ belong to the matrix B_2. Let $\mathcal{I} \triangleq \{i_1, \ldots, i_{t+1}\}$ be the index set of those column-blocks of C that correspond to the column-blocks of $\tilde{\mathrm{V}}$. Let $\mathcal{I}_1 \triangleq \{i_1, \ldots, i_{t-s+1}\}$, and let $\mathcal{I}_2 \triangleq \mathcal{I} \setminus \mathcal{I}_1$. Note that for any $i \in \mathcal{I}_1$, C_i appears in all row-blocks of G_{n+1}, and for any $i \in \mathcal{I}_2$, C_i appears only in the $(i-t)$th row-block of G_{n+1}.

The parameters α_i's are to be chosen such that, by performing row-block operations on G_{n+1}, the user can construct an $L \times (D+R)$ matrix with $t+m$ column-blocks each of size $L \times S$ that satisfies the following two conditions: (a) the blocks indexed by $\{1, \ldots, t+m\} \setminus \mathcal{I}$ are all zero, and (b) the blocks indexed by $\mathcal{I} = \{i_1, \ldots, i_{t+1}\}$ are $\mathrm{C}_{i_1}, \ldots, \mathrm{C}_{i_{t+1}}$. For simplifying the notation, let $\{j_1, \ldots, j_{s-1}\} \triangleq \{1, \ldots, t\} \setminus \mathcal{I}$, and let $\{k_1, \ldots, k_s\} \triangleq \mathcal{I}_2 = \{i_{t-s+2}, \ldots, i_{t+1}\}$.

To perform row-block operations, for every $i \in \mathcal{I}_2 = \{k_1, \ldots, k_s\}$, the user multiplies the $(i-t)$th row-block of G_{n+1} by a nonzero coefficient c_i. Let $\mathrm{c} \triangleq [c_{k_1}, \ldots, c_{k_s}]^{\mathsf{T}}$. Upon choosing $\alpha_{j_1}, \ldots, \alpha_{j_{s-1}}$ randomly from $\mathbb{F}_p \setminus \{0\}$, it follows that the condition (a) is satisfied so long as $\mathrm{M}_1\mathrm{c}$

is an all-zero vector, where

$$\mathrm{M}_1 \triangleq \begin{bmatrix} \omega_{k_1-t,j_1} & \omega_{k_2-t,j_1} & \dots & \omega_{k_s-t,j_1} \\ \omega_{k_1-t,j_2} & \omega_{k_2-t,j_2} & \dots & \omega_{k_s-t,j_2} \\ \vdots & \vdots & \vdots & \vdots \\ \omega_{k_1-t,j_{s-1}} & \omega_{k_2-t,j_{s-1}} & \dots & \omega_{k_s-t,j_{s-1}} \end{bmatrix}.$$

Since M_1 is a Cauchy matrix by the choice of $\omega_{i,j}$'s, the submatrix of M_1 formed by columns indexed by $\{2, \dots, s\}$ is invertible [50]. Thus, for any arbitrary $c_{k_1} \neq 0$, there is a unique solution for the vector c. Note also that all the components of c are nonzero because M_1 is a super-regular matrix, i.e., every square submatrix of M_1 is invertible (by the properties of Cauchy matrices).

Given the vector c, the condition (b) is satisfied so long as $\alpha_{k_1} = 1/c_{k_1}, \dots, \alpha_{k_s} = 1/c_{k_s}$, and $\alpha_{i_1}, \dots, \alpha_{i_{t-s+1}}$ are chosen such that $\mathrm{M}_2\mathrm{c}$ is an all-one vector, where

$$\mathrm{M}_2 \triangleq \begin{bmatrix} \alpha_{i_1}\omega_{k_1-t,i_1} & \dots & \alpha_{i_1}\omega_{k_s-t,i_1} \\ \alpha_{i_2}\omega_{k_1-t,i_2} & \dots & \alpha_{i_2}\omega_{k_s-t,i_2} \\ \vdots & \vdots & \vdots \\ \alpha_{i_{t-s+1}}\omega_{k_1-t,i_{t-s+1}} & \dots & \alpha_{i_{t-s+1}}\omega_{k_s-t,i_{t-s+1}} \end{bmatrix}.$$

Solving for $\alpha_{i_1}, \dots, \alpha_{i_{t-s+1}}$, it follows that

$$\alpha_{i_j} \triangleq \left(\sum_{1 \leq l \leq s} c_{k_l}\omega_{k_l-t,i_j} \right)^{-1}$$

for $1 \leq j \leq t-s+1$. Note that $\alpha_{i_1}, \dots, \alpha_{i_{t-s+1}}$ are nonzero. Note also that $\sum_{1 \leq l \leq s} c_{k_l}\omega_{k_l-t,i_j}$ is nonzero because the jth row of M_2 is linearly independent of the rows of M_1.

Lastly, for any $i \in \{1, \dots, t+m\} \setminus \{i_1, \dots, i_{t+1}, j_1, \dots, j_{s-1}\}$, the user chooses α_i randomly from $\mathbb{F}_p \setminus \{0\}$. This completes the construction of the matrix G.

Next, the user constructs a permutation π as follows. Let $\tilde{\mathrm{W}} = \{l_1, \dots, l_D\}$, and let $\mathcal{K} \setminus \mathrm{W} = \{l_{D+1}, \dots, l_K\}$. First, consider the case of $1 \leq i^* \leq n$. In this case, the user constructs a per-

mutation π such that: for every $1 \leq j \leq D$, $\pi(l_j) = (i^* - 1)D + j$; and for every $D < j \leq K$, $\pi(l_j)$ is a randomly chosen element from $\mathcal{K} \setminus \{\pi(l_k)\}_{1 \leq k < j}$. Next, consider the case of $i^* = n + 1$. Recall that $i_1, \ldots, i_{t+1}$ are the indices of those column-blocks of C that correspond to the column-blocks of $\tilde{\mathrm{V}}$. Thus, the user constructs a permutation π such that: for every $1 \leq k \leq t + 1$ and $(k-1)S+1 \leq j \leq kS$, $\pi(l_j) = nD + (i_k - 1)S + f_j$, where $f_j = j \pmod S$ if $S \nmid j$, and $f_j = S$ if $S \mid j$; and for every $D < j \leq K$, $\pi(l_j)$ is a randomly chosen element from $\mathcal{K} \setminus \{\pi(l_k)\}_{1 \leq k < j}$.

Case (ii):

Recall that in this case, $L > S$. Let $n \triangleq \lfloor \frac{K}{D} \rfloor - 1$, and $m \triangleq \frac{R}{L} + 1$. The user constructs an $L(n + m) \times K$ matrix G with a structure similar to (4.9), where $\mathrm{G}_1, \ldots, \mathrm{G}_n$ are constructed similarly as in the previous case, but the construction of G_{n+1} is different. In the following, we will only explain how to construct G_{n+1}.

For the case of $1 \leq i^* \leq n$, the user randomly generates an $[D + R, L + R]$ MDS code, and takes G_{n+1} to be the generator matrix of this code. For the case of $i^* = n+1$, the user constructs a $[D + R, L + R]$ MDS code using the same technique as in the step 1 of the Specialized MDS Code protocol—described in Section 3.4, except where K is replaced by $D + R$, and W is replaced by a randomly chosen D-subset of $\{1, \ldots, D+R\}$, say, $\{h_1, \ldots, h_D\}$. The user then uses the generator matrix of the constructed MDS code as G_{n+1}.

Next, the user constructs a permutation π. For the case of $1 \leq i^* \leq n$, the permutation π is generated exactly the same as in the case (i), whereas the construction of the permutation π for the case of $i^* = n + 1$ is different from that in the case (i). Similarly as before, let $\tilde{\mathrm{W}} = \{l_1, \ldots, l_D\}$, and let $\mathcal{K} \setminus \mathrm{W} = \{l_{D+1}, \ldots, l_K\}$. For the case of $i^* = n + 1$, the user constructs a permutation π such that: for every $1 \leq j \leq D$, $\pi(l_j) = nD + h_j$; and for every $D < j \leq K$, $\pi(l_j)$ is a randomly chosen element from $\mathcal{K} \setminus \{\pi(l_k)\}_{1 \leq k < j}$.

Step 2: Given the query $\mathrm{Q}^{[\mathrm{W},\mathrm{V}]}$, i.e., the matrix G and the permutation π, the server first constructs the vector $\tilde{\mathrm{X}} \triangleq \pi(\mathrm{X})$ by permuting the components of the vector X according to the permutation π, i.e., $\tilde{\mathrm{X}}_{\pi(l)} \triangleq \mathrm{X}_l$ for $l \in \mathcal{K}$. Then, the server computes the vector $\mathrm{y} \triangleq \mathrm{G}\tilde{\mathrm{X}}$, and sends y back to the user as the answer $\mathrm{A}^{[\mathrm{W},\mathrm{V}]}$.

Step 3: Upon receiving the answer $\mathrm{A}^{[\mathrm{W},\mathrm{V}]}$, i.e., the vector y, the user recovers the demand vector $\mathrm{Z}^{[\mathrm{W},\mathrm{V}]}$ as follows. For every $1 \leq i \leq n$, let y_i be the vector y restricted to its components indexed by $\{(i-1)L+1, \ldots, iL\}$, and let y_{n+1} be the vector y restricted to its components indexed by $\{nL+1, \ldots, nL+mL\}$. For the case of $1 \leq i^* \leq n$, the demand $\mathrm{Z}^{[\mathrm{W},\mathrm{V}]}$ can be recovered from the vector y_{i^*} for both cases (i) and (ii). For the case of $i^* = n+1$, the demand $\mathrm{Z}^{[\mathrm{W},\mathrm{V}]}$ can be recovered by performing proper row-block or row operations on the augmented matrix $[\mathrm{G}_{n+1}, \mathrm{y}_{n+1}]$ for the case (i) or (ii), respectively.

Lemma 9. *The Generalized Partition-and-Code with Partial Interference Alignment protocol is a linear IPLT protocol, and achieves the rate* $(\lfloor \frac{K}{D} \rfloor + \min\{\frac{R}{S}, \frac{R}{L}\})^{-1}$*, where* $R \triangleq K \pmod{D}$ *and* $S \triangleq \gcd(D+R, R)$.

Proof. To avoid repetition, we only present the proof for the case (i) in the following. Using similar arguments, the results can be proven for the case (ii).

In the case (i), it is easy to see that the rate of the protocol is $L\theta/(L(n+m)\theta) = (n+m)^{-1} = (\lfloor \frac{K}{D} \rfloor + \frac{R}{S})^{-1}$. This is because the matrix G has $L(n+m)$ rows, and the vector $\mathrm{y} = \mathrm{G}\tilde{\mathrm{X}}$ contains $L(n+m)$ independently and uniformly distributed components, each with entropy θ. From the construction, it should also be obvious that the recoverability condition is satisfied.

Next, we show that the individual privacy condition is satisfied. Let $\tilde{\mathrm{X}} \triangleq [X_{i_1}, \ldots, X_{i_K}]^{\mathsf{T}}$. For every $1 \leq j \leq n$, let $\mathcal{I}_j$ be the set of jth group of D elements in $\{i_1, \ldots, i_{nD}\}$, and for every $1 \leq j \leq t+m$, let $\mathcal{I}_{n+j}$ be the set of jth group of S elements in $\{i_{nD+1}, \ldots, i_K\}$. For any positive integers a, b such that $b \leq a$, we denote by $C_{a,b}$ the binomial coefficient $\binom{a}{b}$. For every $1 \leq j \leq n$, let $\mathrm{W}_j \triangleq \mathcal{I}_j$, and for every $1 \leq j \leq r \triangleq C_{t+m,t+1}$, let $\mathrm{W}_{n+j} = \cup_{k \in \mathcal{J}_j} \mathcal{I}_k$, where $\mathcal{J}_1, \ldots, \mathcal{J}_r$ are all $(t+1)$-subsets of $\{n+1, \ldots, n+t+m\}$. Note that, given the user's query, $\mathrm{W}_1, \ldots, \mathrm{W}_n, \mathrm{W}_{n+1}, \ldots, \mathrm{W}_{n+t+m}$ are the only possible demand's support index sets from the server's perspective. Let $\mathrm{Q} \triangleq \{\mathrm{G}, \pi\}$ be the user's query. To prove that the individual privacy condition is satisfied, we need to show that $\Pr(i \in \mathbf{W} | \mathbf{Q} = \mathrm{Q}) = \Pr(i \in \mathbf{W})$ for every $i \in \mathcal{K}$, or equivalently, $\Pr(i \in \mathbf{W} | \mathbf{Q} = \mathrm{Q})$ is the same for all $i \in \mathcal{K}$. Fix an arbitrary $i \in \mathcal{K}$. In the following, we consider two different cases: (i) $\pi(i) \leq nD$, and (ii) $\pi(i) > nD$.

First, consider the case (i). In this case, there exists a unique j (for any $1 \leq j \leq n$) such that $i \in \mathrm{W}_j$. Thus, $\Pr(i \in \mathbf{W} | \mathbf{Q} = \mathrm{Q}) = \Pr(\mathbf{W} = \mathrm{W}_j | \mathbf{Q} = \mathrm{Q})$. By applying Bayes' rule, we have

$$\begin{aligned}\Pr(\mathbf{W} = \mathrm{W}_j | \mathbf{Q} = \mathrm{Q}) &= \frac{\Pr(\mathbf{Q} = \mathrm{Q} | \mathbf{W} = \mathrm{W}_j)}{\Pr(\mathbf{Q} = \mathrm{Q})} \Pr(\mathbf{W} = \mathrm{W}_j) \\ &= \frac{\Pr(\mathbf{Q} = \mathrm{Q} | \mathbf{W} = \mathrm{W}_j)}{\Pr(\mathbf{Q} = \mathrm{Q})} \times \frac{1}{C_{K,D}}. \end{aligned} \tag{4.10}$$

The structure of G—the size and the position of the blocks $\mathrm{G}_1, \ldots, \mathrm{G}_{n+1}$—does not depend on (W, π), and the matrix V and all other MDS matrices used in the construction of G are generated independently from (W, π). This implies that $\mathbf{G}$ is independent of $(\mathbf{W}, \boldsymbol{\pi})$. Thus, $\Pr(\mathbf{Q} = \mathrm{Q}) = \Pr(\mathbf{G} = \mathrm{G}, \boldsymbol{\pi} = \pi) = \Pr(\mathbf{G} = \mathrm{G}) \Pr(\boldsymbol{\pi} = \pi)$, and

$$\begin{aligned}\frac{\Pr(\mathbf{Q} = \mathrm{Q} | \mathbf{W} = \mathrm{W}_j)}{\Pr(\mathbf{Q} = \mathrm{Q})} &= \frac{\Pr(\mathbf{G} = \mathrm{G}, \boldsymbol{\pi} = \pi | \mathbf{W} = \mathrm{W}_j)}{\Pr(\mathbf{G} = \mathrm{G}) \Pr(\boldsymbol{\pi} = \pi)} \\ &= \frac{\Pr(\mathbf{G} = \mathrm{G}) \Pr(\boldsymbol{\pi} = \pi | \mathbf{W} = \mathrm{W}_j)}{\Pr(\mathbf{G} = \mathrm{G}) \Pr(\boldsymbol{\pi} = \pi)} \\ &= \frac{\Pr(\boldsymbol{\pi} = \pi | \mathbf{W} = \mathrm{W}_j)}{\Pr(\boldsymbol{\pi} = \pi)}. \end{aligned} \tag{4.11}$$

Given $\mathbf{W} = \mathrm{W}_j$, the conditional probability of the event of $\boldsymbol{\pi} = \pi$ is equal to the joint probability of the two events $\boldsymbol{\pi}(\mathbf{W}) = \pi(\mathrm{W}_j)$ and $\boldsymbol{\pi}(\mathcal{K} \setminus \mathbf{W}) = \pi(\mathcal{K} \setminus \mathrm{W}_j)$. Note that $\Pr(\boldsymbol{\pi}(\mathbf{W}) = \pi(\mathrm{W}_j)) = \Pr(\boldsymbol{i^*} = j) \times \frac{1}{D!} = \frac{D}{K} \times \frac{1}{D!}$, where i^* is the index of the block selected by the user in the step 1 of the protocol, and $\Pr(\boldsymbol{\pi}(\mathcal{K} \setminus \mathbf{W}) = \pi(\mathcal{K} \setminus \mathrm{W}_j)) = \frac{1}{(K-D)!}$ by the construction of the permutation π in the step 1 of the protocol. Thus,

$$\Pr(\boldsymbol{\pi} = \pi | \mathbf{W} = \mathrm{W}_j) = \frac{D}{K} \times \frac{1}{D!} \times \frac{1}{(K-D)!}. \tag{4.12}$$

Combining (4.10)-(4.12), we have

$$\Pr(i \in \mathbf{W} | \mathbf{Q} = \mathrm{Q}) = \frac{1}{\Pr(\boldsymbol{\pi} = \pi)} \times \frac{D}{K} \times \frac{1}{K!}. \tag{4.13}$$

Now, consider the case (ii). In this case, there exist $s \triangleq C_{t+m-1,t}$ distinct indices $j_1, \ldots, j_s$ $(1 \leq j_1, \ldots, j_s \leq r)$ such that $i \in \mathrm{W}_{n+j_1}, \ldots, i \in \mathrm{W}_{n+j_s}$. Similarly as in the case (i), we have

$$\begin{aligned}
\Pr(i \in \mathbf{W} | \mathbf{Q} = \mathrm{Q}) &= \sum_{1 \leq k \leq s} \Pr(\mathbf{W} = \mathrm{W}_{n+j_k} | \mathbf{Q} = \mathrm{Q}) \\
&= \sum_{1 \leq k \leq s} \frac{\Pr(\mathbf{Q} = \mathrm{Q} | \mathbf{W} = \mathrm{W}_{n+j_k})}{\Pr(\mathbf{Q} = \mathrm{Q})} \Pr(\mathbf{W} = \mathrm{W}_{n+j_k}) \\
&= \sum_{1 \leq k \leq s} \frac{\Pr(\mathbf{G} = \mathrm{G}) \Pr(\boldsymbol{\pi} = \pi | \mathbf{W} = \mathrm{W}_{n+j_k})}{\Pr(\mathbf{G} = \mathrm{G}) \Pr(\boldsymbol{\pi} = \pi)} \times \frac{1}{C_{K,D}} \\
&= \frac{1}{\Pr(\boldsymbol{\pi} = \pi)} \sum_{1 \leq k \leq s} \Pr(\boldsymbol{\pi} = \pi | \mathbf{W} = \mathrm{W}_{n+j_k}) \times \frac{1}{C_{K,D}} \\
&= \frac{1}{\Pr(\boldsymbol{\pi} = \pi)} \sum_{1 \leq k \leq s} \left(\frac{D+R}{K} \times \frac{1}{r} \times \frac{1}{D!} \times \frac{1}{(K-D)!} \right) \frac{1}{C_{K,D}} \\
&= \frac{1}{\Pr(\boldsymbol{\pi} = \pi)} \times s \left(\frac{D+R}{K} \times \frac{1}{r} \times \frac{1}{D!} \times \frac{1}{(K-D)!} \right) \frac{1}{C_{K,D}} \\
&= \frac{1}{\Pr(\boldsymbol{\pi} = \pi)} \times \frac{s}{r} \times \frac{D+R}{K} \times \frac{1}{K!} \\
&= \frac{1}{\Pr(\boldsymbol{\pi} = \pi)} \times \frac{D}{D+R} \times \frac{D+R}{K} \times \frac{1}{K!} \\
&= \frac{1}{\Pr(\boldsymbol{\pi} = \pi)} \times \frac{D}{K} \times \frac{1}{K!}. \qquad (4.14)
\end{aligned}$$

Comparing (4.13) and (4.14), one can see that $\Pr(i \in \mathbf{W} | \mathbf{Q} = \mathrm{Q})$ is the same for all $i \in \mathcal{K}$. □

4.5 Examples of the Proposed IPLT Protocol

In this section, we provide two illustrative examples of the proposed protocol. Example 1 corresponds to a scenario with $L \leq S$, and Example 2 corresponds to a scenario with $L > S$.

Example 1. Consider a scenario where the server has $K = 24$ messages, $X_1, \ldots, X_{24} \in \mathbb{F}_{17}$, and the user wishes to compute $L = 2$ linear combinations of $D = 9$ messages X_2, X_4, X_5, X_7, X_8, X_{10}, X_{11}, X_{18}, X_{23}, say,

$$Z_1 = 2X_2 + 15X_4 + 3X_5 + 6X_7 + X_8 + 4X_{10} + 11X_{11} + 13X_{18} + 9X_{23},$$

$$Z_2 = 6X_2 + 9X_4 + 4X_5 + 3X_7 + 11X_8 + 15X_{10} + 13X_{11} + 8X_{18} + X_{23}.$$

For this example, the demand's support index set is given by $\mathrm{W} = \{2, 4, 5, 7, 8, 10, 11, 18, 23\}$, and the demand's coefficient matrix is given by

$$\mathrm{V} = \begin{bmatrix} 2 & 15 & 3 & 6 & 1 & 4 & 11 & 13 & 9 \\ 6 & 9 & 4 & 3 & 11 & 15 & 13 & 8 & 1 \end{bmatrix}.$$

It is easy to verify that the matrix V is MDS, i.e., every 2×2 submatrix of V is invertible (over $\mathbb{F}_{17}$).

Let $\tilde{\mathrm{W}}$ be a sequence of length 9 that the user initially constructs by randomly permuting the elements in W, for example, $\tilde{\mathrm{W}} = \{10, 4, 8, 11, 7, 23, 18, 2, 5\}$, and let $\tilde{\mathrm{V}}$ be a 2×9 matrix that the user initially constructs by applying the same permutation on the columns of the matrix V, i.e.,

$$\tilde{\mathrm{V}} = \begin{bmatrix} 4 & 15 & 1 & 11 & 6 & 9 & 13 & 2 & 3 \\ 15 & 9 & 11 & 13 & 3 & 1 & 8 & 6 & 4 \end{bmatrix}.$$

Note that $\mathrm{V}\mathrm{X}_{\mathrm{W}} = \tilde{\mathrm{V}}\mathrm{X}_{\tilde{\mathrm{W}}}$ by the construction of $\tilde{\mathrm{W}}$ and $\tilde{\mathrm{V}}$.

For this example, $R = K \pmod{D} = 6$, $S = \gcd(D + R, R) = 3$, $n = \lfloor \frac{K}{D} \rfloor - 1 = 1$, $m = \frac{R}{S} + 1 = 3$, and $t = \frac{D}{S} - 1 = 2$. Note that $L = 2 < S = 3$.

The query of the user consists of an 8×24 matrix G and a permutation π on $\{1, \ldots, 24\}$. The matrix G is constructed using two matrices (blocks) G_1 and G_2 of size 2×9 and 6×15, respectively,

$$\mathrm{G} = \begin{bmatrix} \mathrm{G}_1 & 0_{2\times 15} \\ 0_{6\times 9} & \mathrm{G}_2 \end{bmatrix}, \tag{4.15}$$

where the construction of the blocks G_1 and G_2 is described below.

The user randomly selects one of the blocks $\mathrm{G}_1, \mathrm{G}_2$, where the probability of selecting the block G_1 is $\frac{9}{24}$, and the probability of selecting the block G_2 is $\frac{15}{24}$. Depending on G_1 or G_2 being selected, the construction of each of these blocks is different. In this example, suppose the user selects G_2. In this case, the user takes G_1 to be a randomly generated MDS matrix of size 2×9,

say,

$$\mathrm{G}_1 = \begin{bmatrix} 3 & 14 & 11 & 8 & 4 & 10 & 5 & 5 & 6 \\ 12 & 16 & 3 & 4 & 6 & 3 & 7 & 15 & 4 \end{bmatrix}. \tag{4.16}$$

To construct G_2, the user first constructs a 2×15 matrix $\mathrm{C} = [\mathrm{C}_1, \mathrm{C}_2, \mathrm{C}_3, \mathrm{C}_4, \mathrm{C}_5]$, where the column-blocks $\mathrm{C}_1, \ldots, \mathrm{C}_5$, each of size 2×3, are constructed as follows. The user partitions the columns of $\tilde{\mathrm{V}}$ into three column-blocks $\tilde{\mathrm{V}}_1, \tilde{\mathrm{V}}_2, \tilde{\mathrm{V}}_3$, each of size 2×3, i.e.,

$$\tilde{\mathrm{V}}_1 = \begin{bmatrix} 4 & 15 & 1 \\ 15 & 9 & 11 \end{bmatrix}, \qquad \tilde{\mathrm{V}}_2 = \begin{bmatrix} 11 & 6 & 9 \\ 13 & 3 & 1 \end{bmatrix}, \qquad \tilde{\mathrm{V}}_3 = \begin{bmatrix} 13 & 2 & 3 \\ 8 & 6 & 4 \end{bmatrix}.$$

The user then randomly chooses three indices i_1, i_2, i_3 from $\{1, 2, 3, 4, 5\}$, say, $i_1 = 1$, $i_2 = 3$, and $i_3 = 5$, and takes $\mathrm{C}_{i_1} = \mathrm{C}_1 = \tilde{\mathrm{V}}_1, \mathrm{C}_{i_2} = \mathrm{C}_3 = \tilde{\mathrm{V}}_2$, and $\mathrm{C}_{i_3} = \mathrm{C}_5 = \tilde{\mathrm{V}}_3$. . Next, the user takes the remaining column-blocks of C, i.e., C_2 and C_4, to be randomly generated matrices of size 2×3 such that $\mathrm{C} = [\mathrm{C}_1, \mathrm{C}_2, \mathrm{C}_3, \mathrm{C}_4, \mathrm{C}_5]$ is an MDS matrix. For this example, suppose the user takes C_2 and C_4 as

$$\mathrm{C}_2 = \begin{bmatrix} 1 & 4 & 7 \\ 5 & 7 & 6 \end{bmatrix}, \qquad \mathrm{C}_4 = \begin{bmatrix} 6 & 3 & 12 \\ 9 & 3 & 15 \end{bmatrix}.$$

Thus, the matrix C is given by

$$\mathrm{C} = \begin{bmatrix} \tilde{\mathrm{V}}_1 & \mathrm{C}_2 & \tilde{\mathrm{V}}_2 & \mathrm{C}_4 & \tilde{\mathrm{V}}_3 \end{bmatrix} = \begin{bmatrix} 4 & 15 & 1 & 1 & 4 & 7 & 11 & 6 & 9 & 6 & 3 & 12 & 13 & 2 & 3 \\ 15 & 9 & 11 & 5 & 7 & 6 & 13 & 3 & 1 & 9 & 3 & 15 & 8 & 6 & 4 \end{bmatrix}.$$

It is easy to verify that the matrix C is MDS. The user then randomly chooses $t + m = 5$ distinct elements x_1, x_2, x_3, y_1, y_2 from $\mathbb{F}_{17}$, say, $x_1 = 1, x_2 = 5, x_3 = 7, y_1 = 11, y_2 = 16$, and constructs a 3×2 Cauchy matrix whose entry (i, j) is given by $\omega_{i,j} \triangleq (x_i - y_j)^{-1}$, i.e.,

$$\begin{bmatrix} \omega_{1,1} & \omega_{1,2} \\ \omega_{2,1} & \omega_{2,2} \\ \omega_{3,1} & \omega_{3,2} \end{bmatrix} = \begin{bmatrix} 5 & 9 \\ 14 & 3 \\ 4 & 15 \end{bmatrix}.$$

Next, the user constructs the matrix G_2 as

$$\mathrm{G}_2 = \begin{bmatrix} \alpha_1\omega_{1,1}\mathrm{C}_1 & \alpha_2\omega_{1,2}\mathrm{C}_2 & \alpha_3\mathrm{C}_3 & 0_{2\times 3} & 0_{2\times 3} \\ \alpha_1\omega_{2,1}\mathrm{C}_1 & \alpha_2\omega_{2,2}\mathrm{C}_2 & 0_{2\times 3} & \alpha_4\mathrm{C}_4 & 0_{2\times 3} \\ \alpha_1\omega_{3,1}\mathrm{C}_1 & \alpha_2\omega_{3,2}\mathrm{C}_2 & 0_{2\times 3} & 0_{2\times 3} & \alpha_5\mathrm{C}_5 \end{bmatrix}$$
$$= \begin{bmatrix} 5\alpha_1\mathrm{C}_1 & 9\alpha_2\mathrm{C}_2 & \alpha_3\mathrm{C}_3 & 0_{2\times 3} & 0_{2\times 3} \\ 14\alpha_1\mathrm{C}_1 & 3\alpha_2\mathrm{C}_2 & 0_{2\times 3} & \alpha_4\mathrm{C}_4 & 0_{2\times 3} \\ 4\alpha_1\mathrm{C}_1 & 15\alpha_2\mathrm{C}_2 & 0_{2\times 3} & 0_{2\times 3} & \alpha_5\mathrm{C}_5 \end{bmatrix},$$

where the (scalar) parameters $\alpha_1, \ldots, \alpha_5$ are chosen such that by performing row-block operations on G_2, the user can obtain the matrix $[\mathrm{C}_1, 0_{2\times 3}, \mathrm{C}_3, 0_{2\times 3}, \mathrm{C}_5]$. Note that the second and fourth column-blocks of G_2, i.e., the column-blocks that contain scalar multiples of C_2 and C_4, do not contain any column-block of $\tilde{\mathrm{V}}$, and hence must be eliminated by row-block operations. Thus, the user randomly chooses the parameters α_2 and α_4 (corresponding to the second and fourth column-blocks of G_2) from $\mathbb{F}_{17} \setminus \{0\}$, say $\alpha_2 = 2$ and $\alpha_4 = 10$. The parameters α_1, α_3, and α_5 are chosen as follows. To perform row-block operations on G_2, suppose that the user multiplies the first and third row-blocks of G_2 by scalars c_3 and c_5, respectively, and constructs the matrix

$$c_3 \begin{bmatrix} 5\alpha_1\mathrm{C}_1 & 9\alpha_2\mathrm{C}_2 & \alpha_3\mathrm{C}_3 & 0_{2\times 3} & 0_{2\times 3} \end{bmatrix} + c_5 \begin{bmatrix} 4\alpha_1\mathrm{C}_1 & 15\alpha_2\mathrm{C}_2 & 0_{2\times 3} & 0_{2\times 3} & \alpha_5\mathrm{C}_5 \end{bmatrix}$$
$$= \begin{bmatrix} (5c_3 + 4c_5)\alpha_1\mathrm{C}_1 & (9c_3 + 15c_5)\alpha_2\mathrm{C}_2 & c_3\alpha_3\mathrm{C}_3 & 0_{2\times 3} & c_5\alpha_5\mathrm{C}_5 \end{bmatrix}.$$

Thus, the user can recover the matrix $[\mathrm{C}_1, 0_{2\times 3}, \mathrm{C}_3, 0_{2\times 3}, \mathrm{C}_5]$ by performing row-block operations on the matrix G_2 so long as $(5c_3 + 4c_5)\alpha_1 = 1$, $9c_3 + 15c_5 = 0$, $c_3\alpha_3 = 1$, and $c_5\alpha_5 = 1$. Note that the choice of $\omega_{i,j}$'s to be entries of a Cauchy matrix guarantees that this system of equations has a nonzero solution for all parameters $c_3, c_5, \alpha_1, \alpha_3, \alpha_5$, and the solution is unique for any arbitrary (but fixed) value of $c_3 \neq 0$. Choosing c_3 to be an arbitrary element in $\mathbb{F}_{17} \setminus \{0\}$, say, $c_3 = 1$, the user then needs to take $c_5 = -\frac{9}{15}c_3 = 13$. Given $c_3 = 1$ and $c_5 = 13$, the user then finds

$\alpha_1 = \frac{1}{5c_3+4c_5} = 3$, $\alpha_3 = \frac{1}{c_3} = 1$, and $\alpha_5 = \frac{1}{c_5} = 4$. The user then constructs G_2 as

$$\mathrm{G}_2 = \begin{bmatrix} 15\mathrm{C}_1 & \mathrm{C}_2 & \mathrm{C}_3 & 0_{2\times 3} & 0_{2\times 3} \\ 8\mathrm{C}_1 & 6\mathrm{C}_2 & 0_{2\times 3} & 10\mathrm{C}_4 & 0_{2\times 3} \\ 12\mathrm{C}_1 & 13\mathrm{C}_2 & 0_{2\times 3} & 0_{2\times 3} & 4\mathrm{C}_5 \end{bmatrix}. \tag{4.17}$$

Combining G_1 and G_2 given by (4.16) and (4.17), the user then constructs the matrix G as in (4.15).

Next, the user constructs a permutation π on $\{1, \ldots, 24\}$. Note that the columns 13, 14, 15, 16, 17, 18, 22, 23, 24 of the matrix G are constructed based on the columns $1, \ldots, 9$ of the matrix $\tilde{\mathrm{V}}$, respectively, and the columns $1, \ldots, 9$ of $\tilde{\mathrm{V}}$ correspond to the messages X_{10}, X_4, X_8, X_{11}, X_7, X_{23}, X_{18}, X_2, X_5, respectively. Thus, the user constructs the permutation π such that $\{\pi(10), \pi(4), \pi(8), \pi(11), \pi(7), \pi(23), \pi(18), \pi(2), \pi(5)\} = \{10, 11, 12, 16, 17, 18, 22, 23, 24\}$. For $i \in \{1, \ldots, 24\} \setminus \{2, 4, 5, 7, 8, 10, 11, 18, 23\}$, the user then randomly chooses $\pi(i)$ (subject to the constraint that π forms a valid permutation on $\{1, \ldots, 24\}$). For this example, suppose that the user takes $\{\pi(1), \pi(3), \pi(6), \pi(9), \pi(12), \ldots, \pi(17), \pi(19), \pi(20), \pi(21), \pi(21), \pi(22), \pi(24)\} = \{13, 20, 9, 19, 21, 14, 4, 8, 15, 1, 7, 3, 6, 2, 5\}$

Then, the user sends the matrix G and the permutation π to the server as the query. Upon receiving the user's query, the server first permutes the components of the vector $\mathrm{X} = [X_1, \ldots, X_{24}]^{\mathsf{T}}$ according to the permutation π to obtain the vector $\tilde{\mathrm{X}} = \pi(\mathrm{X})$, i.e., $\tilde{X}_{\pi(i)} = X_i$ for $i \in \{1, \ldots, 24\}$. For this example, the vector $\tilde{\mathrm{X}}$ is given by

$$\begin{aligned} \tilde{\mathrm{X}} = [&X_{17}, X_{22}, X_{20}, X_{14}, X_{24}, X_{21}, X_{19}, X_{15}, X_6, X_{10}, X_4, \\ &X_8, X_1, X_{13}, X_{16}, X_{11}, X_7, X_{23}, X_9, X_3, X_{12}, X_{18}, X_2, X_5]^{\mathsf{T}}. \end{aligned}$$

Then the server computes $\mathrm{y} = \mathrm{G}\tilde{\mathrm{X}}$, and sends the vector y back to the user as the answer. Let $\mathrm{T}_1 = \{1, \ldots, 9\}$, $\mathrm{T}_2 = \{10, 11, 12\}$, $\mathrm{T}_3 = \{13, 14, 15\}$, $\mathrm{T}_4 = \{16, 17, 18\}$, $\mathrm{T}_5 = \{19, 20, 21\}$, and $\mathrm{T}_6 = \{22, 23, 24\}$. For any $\mathrm{T} \subset \{1, \ldots, 24\}$, we denote by $\tilde{\mathrm{X}}_{\mathrm{T}}$ the vector $\tilde{\mathrm{X}}$ restricted to its components indexed by T. Note that $[\tilde{\mathrm{X}}_{\mathrm{T}_2}^{\mathsf{T}}, \tilde{\mathrm{X}}_{\mathrm{T}_4}^{\mathsf{T}}, \tilde{\mathrm{X}}_{\mathrm{T}_6}^{\mathsf{T}}]^{\mathsf{T}} = \mathrm{X}_{\tilde{\mathrm{W}}}$, and $\mathrm{y} = [\mathrm{y}_1^{\mathsf{T}}, \mathrm{y}_2^{\mathsf{T}}]^{\mathsf{T}}$, where $\mathrm{y}_1 \triangleq \mathrm{G}_1\tilde{\mathrm{X}}_{\mathrm{T}_1}$, and $\mathrm{y}_2 \triangleq \mathrm{G}_2[\tilde{\mathrm{X}}_{\mathrm{T}_2}^{\mathsf{T}}, \tilde{\mathrm{X}}_{\mathrm{T}_3}^{\mathsf{T}}, \tilde{\mathrm{X}}_{\mathrm{T}_4}^{\mathsf{T}}, \tilde{\mathrm{X}}_{\mathrm{T}_5}^{\mathsf{T}}, \tilde{\mathrm{X}}_{\mathrm{T}_6}^{\mathsf{T}}]^{\mathsf{T}}$. Let I be the identity matrix of size 2×2.

Then, the user recovers $[Z_1, Z_2]^{\mathsf{T}} = \mathrm{V}\mathrm{X}_{\mathrm{W}} = \tilde{\mathrm{V}}\mathrm{X}_{\tilde{\mathrm{W}}}$ by computing

$$\begin{bmatrix} c_3\mathrm{I} & 0_{2\times 2} & c_5\mathrm{I} \end{bmatrix} \mathrm{y}_2$$

$$= \begin{bmatrix} c_3\mathrm{I} & 0_{2\times 2} & c_5\mathrm{I} \end{bmatrix} \mathrm{G}_2 \begin{bmatrix} \tilde{\mathrm{X}}_{\mathrm{T}_2} \\ \tilde{\mathrm{X}}_{\mathrm{T}_3} \\ \tilde{\mathrm{X}}_{\mathrm{T}_4} \\ \tilde{\mathrm{X}}_{\mathrm{T}_5} \\ \tilde{\mathrm{X}}_{\mathrm{T}_6} \end{bmatrix} = \begin{bmatrix} c_3\mathrm{I} & 0_{2\times 2} & c_5\mathrm{I} \end{bmatrix} \begin{bmatrix} 15\mathrm{C}_1 & \mathrm{C}_2 & \mathrm{C}_3 & 0_{2\times 3} & 0_{2\times 3} \\ 8\mathrm{C}_1 & 6\mathrm{C}_2 & 0_{2\times 3} & 10\mathrm{C}_4 & 0_{2\times 3} \\ 12\mathrm{C}_1 & 13\mathrm{C}_2 & 0_{2\times 3} & 0_{2\times 3} & 4\mathrm{C}_5 \end{bmatrix} \begin{bmatrix} \tilde{\mathrm{X}}_{\mathrm{T}_2} \\ \tilde{\mathrm{X}}_{\mathrm{T}_3} \\ \tilde{\mathrm{X}}_{\mathrm{T}_4} \\ \tilde{\mathrm{X}}_{\mathrm{T}_5} \\ \tilde{\mathrm{X}}_{\mathrm{T}_6} \end{bmatrix}$$

$$= \begin{bmatrix} (15c_3 + 12c_5)\mathrm{C}_1 & (c_3 + 13c_5)\mathrm{C}_2 & c_3\mathrm{C}_3 & 0_{2\times 3} & 4c_5\mathrm{C}_5 \end{bmatrix} \begin{bmatrix} \tilde{\mathrm{X}}_{\mathrm{T}_2} \\ \tilde{\mathrm{X}}_{\mathrm{T}_3} \\ \tilde{\mathrm{X}}_{\mathrm{T}_4} \\ \tilde{\mathrm{X}}_{\mathrm{T}_5} \\ \tilde{\mathrm{X}}_{\mathrm{T}_6} \end{bmatrix}$$

$$= \begin{bmatrix} \mathrm{C}_1 & 0_{2\times 3} & \mathrm{C}_3 & 0_{2\times 3} & \mathrm{C}_5 \end{bmatrix} \begin{bmatrix} \tilde{\mathrm{X}}_{\mathrm{T}_2} \\ \tilde{\mathrm{X}}_{\mathrm{T}_3} \\ \tilde{\mathrm{X}}_{\mathrm{T}_4} \\ \tilde{\mathrm{X}}_{\mathrm{T}_5} \\ \tilde{\mathrm{X}}_{\mathrm{T}_6} \end{bmatrix} = \begin{bmatrix} \tilde{\mathrm{V}}_1 & 0 & \tilde{\mathrm{V}}_2 & 0 & \tilde{\mathrm{V}}_3 \end{bmatrix} \begin{bmatrix} \tilde{\mathrm{X}}_{\mathrm{T}_2} \\ \tilde{\mathrm{X}}_{\mathrm{T}_3} \\ \tilde{\mathrm{X}}_{\mathrm{T}_4} \\ \tilde{\mathrm{X}}_{\mathrm{T}_5} \\ \tilde{\mathrm{X}}_{\mathrm{T}_6} \end{bmatrix}$$

$$= \begin{bmatrix} \tilde{\mathrm{V}}_1 & \tilde{\mathrm{V}}_2 & \tilde{\mathrm{V}}_3 \end{bmatrix} \begin{bmatrix} \tilde{\mathrm{X}}_{\mathrm{T}_2} \\ \tilde{\mathrm{X}}_{\mathrm{T}_4} \\ \tilde{\mathrm{X}}_{\mathrm{T}_6} \end{bmatrix}$$

$$= \tilde{\mathrm{V}}\mathrm{X}_{\tilde{\mathrm{W}}},$$

noting that $c_3 = 1$ and $c_5 = 13$, and hence, $15c_3 + 12c_5 = 1$, $c_3 + 13c_5 = 0$, and $4c_5 = 1$.

Note that for this example, the proposed protocol achieves the rate $(\lfloor \frac{K}{D} \rfloor + \frac{R}{S})^{-1} = 1/4$, whereas the Specialized MDS Code protocol of Section 3.4 achieves a lower rate $L/(K - D + L) = 2/17$.

Example 2. Consider a scenario where the server has $K = 24$ messages, $X_1, \ldots, X_{24} \in \mathbb{F}_{17}$, and the user wishes to compute $L = 2$ linear combinations of $D = 7$ messages X_2, X_4, X_7, X_{10}, X_{15}, X_{18}, X_{23}, say,

$$Z_1 = 2X_2 + 15X_4 + 6X_7 + 4X_{10} + 11X_{15} + 13X_{18} + 9X_{23},$$
$$Z_2 = 6X_2 + 9X_4 + 3X_7 + 15X_{10} + 13X_{15} + 8X_{18} + X_{23}.$$

For this example, $\mathrm{W} = \{2, 4, 7, 10, 15, 18, 23\}$, and

$$\mathrm{V} = \begin{bmatrix} 2 & 15 & 6 & 4 & 11 & 13 & 9 \\ 6 & 9 & 3 & 15 & 13 & 8 & 1 \end{bmatrix}.$$

Let $\tilde{\mathrm{W}}$ be a sequence of length 7 that the user initially constructs by randomly permuting the elements in W, for example, $\tilde{\mathrm{W}} = \{10, 4, 7, 23, 18, 2, 15\}$, and let $\tilde{\mathrm{V}}$ be a 2×7 matrix that the user initially constructs by applying the same permutation on the columns of the matrix V, i.e.,

$$\tilde{\mathrm{V}} = \begin{bmatrix} 15 & 4 & 6 & 9 & 13 & 2 & 11 \\ 9 & 15 & 3 & 1 & 8 & 6 & 13 \end{bmatrix}.$$

For this example, $R = K \pmod{D} = 3$, $S = \gcd(D + R, R) = 1$, $n = \lfloor \frac{K}{D} \rfloor - 1 = 2$, and $m = \frac{R}{L} + 1 = \frac{5}{2}$. Note that $L = 2 > S = 1$.

The query of the user consists of a 9×24 matrix G and a permutation π on $\{1, \ldots, 24\}$, constructed as follows. The matrix G is constructed using three matrices (blocks) G_1, G_2, and G_3 of size 2×7, 2×7, and 7×10, respectively,

$$\mathrm{G} = \begin{bmatrix} \mathrm{G}_1 & 0_{2\times 7} & 0_{2\times 7} \\ 0_{2\times 7} & \mathrm{G}_2 & 0_{2\times 7} \\ 0_{5\times 7} & 0_{5\times 7} & \mathrm{G}_3 \end{bmatrix}, \tag{4.18}$$

where the construction of the blocks G_1, G_2, and G_3 is described below.

The user randomly selects one of the blocks $\mathrm{G}_1, \mathrm{G}_2, \mathrm{G}_3$, where the probability of selecting the block G_1 is $\frac{7}{20}$, the probability of selecting the block G_2 is $\frac{7}{20}$, and the probability of selecting the block G_3 is $\frac{10}{20}$. Depending on G_1, G_2, or G_3 being selected, the construction of each of these blocks is different. In this example, we consider the case that the user selects G_3. In this case, the user takes G_1 and G_2 to be two randomly generated MDS matrices, each of size 2×7, say,

$$\mathrm{G}_1 = \begin{bmatrix} 11 & 5 & 10 & 1 & 15 & 2 & 7 \\ 16 & 10 & 16 & 6 & 1 & 1 & 13 \end{bmatrix}, \qquad \mathrm{G}_2 = \begin{bmatrix} 5 & 8 & 14 & 7 & 4 & 3 & 16 \\ 3 & 5 & 8 & 1 & 6 & 2 & 15 \end{bmatrix}. \tag{4.19}$$

The user then constructs G_3 using a similar technique as in the Specialized MDS Code protocol of Section 3.4. The details of the construction of G_3 are as follows. Recall that $\tilde{\mathrm{V}}$ generates a $[7, 2]$ MDS code. Thus, the user can obtain the parity-check matrix Λ of the MDS code generated by $\tilde{\mathrm{V}}$ as

$$\Lambda = \begin{bmatrix} 8 & 5 & 9 & 6 & 14 & 11 & 13 \\ 15 & 6 & 13 & 12 & 6 & 16 & 3 \\ 9 & 14 & 15 & 7 & 5 & 14 & 2 \\ 2 & 10 & 16 & 14 & 7 & 8 & 7 \\ 8 & 12 & 8 & 11 & 3 & 7 & 16 \end{bmatrix}.$$

Note that Λ also generates a $[7, 5]$ MDS code. Then, the user randomly chooses a $D = 7$-subset of $\{1, \ldots, 10\}$, say, $\{h_1, \ldots, h_7\} = \{1, 3, 4, 6, 7, 8, 10\}$, and randomly generates a 2×10 MDS matrix H such that the submatrix of H restricted to columns indexed by $\{h_1, \ldots, h_7\} = \{1, 3, 4, 6, 7, 8, 10\}$ (and all rows) is the matrix Λ. For this example, suppose that the user constructs the matrix H as

$$\mathrm{H} = \begin{bmatrix} \mathbf{8} & 1 & \mathbf{5} & \mathbf{9} & 2 & \mathbf{6} & \mathbf{14} & \mathbf{11} & 4 & \mathbf{13} \\ \mathbf{15} & 6 & \mathbf{6} & \mathbf{13} & 3 & \mathbf{12} & \mathbf{6} & \mathbf{16} & 3 & \mathbf{3} \\ \mathbf{9} & 2 & \mathbf{14} & \mathbf{15} & 13 & \mathbf{7} & \mathbf{5} & \mathbf{14} & 15 & \mathbf{2} \\ \mathbf{2} & 12 & \mathbf{10} & \mathbf{16} & 11 & \mathbf{14} & \mathbf{7} & \mathbf{8} & 7 & \mathbf{7} \\ \mathbf{8} & 4 & \mathbf{12} & \mathbf{8} & 8 & \mathbf{11} & \mathbf{3} & \mathbf{7} & 1 & \mathbf{16} \end{bmatrix}.$$

Since H generates an $[10, 5]$ MDS code, it can also be thought of as the parity-check matrix of a $[10, 5]$ MDS code. The user then takes G_3 to be the generator matrix of the $[10, 5]$ MDS code defined by the parity-check matrix H,

$$\mathrm{G}_3 = \begin{bmatrix} \mathbf{3} & 14 & \mathbf{11} & \mathbf{8} & 4 & \mathbf{10} & \mathbf{8} & \mathbf{5} & 5 & \mathbf{6} \\ \mathbf{12} & 16 & \mathbf{3} & \mathbf{4} & 6 & \mathbf{3} & \mathbf{1} & \mathbf{15} & 8 & \mathbf{4} \\ \mathbf{14} & 11 & \mathbf{7} & \mathbf{2} & 9 & \mathbf{6} & \mathbf{15} & \mathbf{11} & 6 & \mathbf{14} \\ \mathbf{5} & 15 & \mathbf{5} & \mathbf{1} & 5 & \mathbf{12} & \mathbf{4} & \mathbf{16} & 13 & \mathbf{15} \\ \mathbf{3} & 5 & \mathbf{6} & \mathbf{9} & 16 & \mathbf{7} & \mathbf{9} & \mathbf{14} & 14 & \mathbf{10} \end{bmatrix}. \tag{4.20}$$

Combining G_1, G_2, and G_3 given by (4.19) and (4.20), the user constructs the matrix G as in (4.18).

Next, the user constructs a permutation π on $\{1, \ldots, 24\}$. Note that the columns 15, 17, 18, 20, 21, 22, 24 of G are constructed based on the columns $1, \ldots, 7$ of Λ, the columns $1, \ldots, 7$ of Λ are constructed based on the columns $1, \ldots, 7$ of $\tilde{\mathrm{V}}$, and the columns $1, \ldots, 7$ of $\tilde{\mathrm{V}}$ correspond to the messages $X_4, X_{10}, X_7, X_{23}, X_{18}, X_2, X_{15}$, respectively. Thus, the user constructs the permutation π such that $\{\pi(4), \pi(10), \pi(7), \pi(23), \pi(18), \pi(2), \pi(15)\} = \{15, 17, 18, 20, 21, 22, 24\}$. For any $i \in \{1, \ldots, 24\} \setminus \{2, 4, 7, 10, 15, 18, 23\}$, the user then randomly chooses $\pi(i)$ (subject to the constraint that π forms a valid permutation on $\{1, \ldots, 24\}$). For this example, suppose that the user takes $\{\pi(1), \pi(3), \pi(5), \pi(6), \pi(8), \pi(9), \pi(11), \ldots, \pi(14), \pi(16), \pi(17), \pi(19), \ldots, \pi(22), \pi(24)\}$ $= \{12, 8, 16, 13, 1, 19, 23, 14, 7, 2, 6, 3, 5, 9, 11, 4, 10\}$.

Then, the user sends the matrix G and the permutation π to the server as the query. Upon receiving the user's query, the server first permutes the components of the vector $\mathrm{X} = [X_1, \ldots, X_{24}]^{\mathsf{T}}$ according to the permutation π to obtain the vector $\tilde{\mathrm{X}} = \pi(\mathrm{X})$, i.e., $\tilde{X}_{\pi(i)} = X_i$ for $i \in \{1, \ldots, 24\}$. For this example, the vector $\tilde{\mathrm{X}}$ is given by

$$\begin{aligned} \tilde{\mathrm{X}} = [&X_8, X_{14}, X_{17}, X_{22}, X_{19}, X_{16}, X_{13}, X_3, X_{20}, X_{24}, X_{21}, \\ &X_1, X_6, X_{12}, X_4, X_5, X_{10}, X_7, X_9, X_{23}, X_{18}, X_2, X_{11}, X_{15}]^{\mathsf{T}}. \end{aligned}$$

Then the server computes $\mathrm{y} = \mathrm{G}\tilde{\mathrm{X}}$, and sends the vector y back to the user as the answer. To recover their demand, the user proceeds as follows. Let $\mathrm{T}_1 = \{1, \ldots, 7\}$, $\mathrm{T}_2 = \{8, \ldots, 14\}$, and $\mathrm{T}_3 = \{15, \ldots, 24\}$. For any $\mathrm{T} \subset \{1, \ldots, 24\}$, we denote by $\tilde{\mathrm{X}}_{\mathrm{T}}$ the vector $\tilde{\mathrm{X}}$ restricted to its components indexed by T. Note that $\mathrm{y} = [\mathrm{y}_1^{\mathsf{T}}, \mathrm{y}_2^{\mathsf{T}}, \mathrm{y}_3^{\mathsf{T}}]^{\mathsf{T}}$, where $\mathrm{y}_1 \triangleq \mathrm{G}_1\tilde{\mathrm{X}}_{\mathrm{T}_1}$, $\mathrm{y}_2 \triangleq \mathrm{G}_2\tilde{\mathrm{X}}_{\mathrm{T}_2}$, and $\mathrm{y}_3 \triangleq \mathrm{G}_3\tilde{\mathrm{X}}_{\mathrm{T}_3}$. Then, the user recovers $[Z_1, Z_2, Z_3]^{\mathsf{T}} = \mathrm{V}\mathrm{X}_{\mathrm{W}} = \tilde{\mathrm{V}}\mathrm{X}_{\tilde{\mathrm{W}}}$ by computing

$$\begin{bmatrix} 6 & 4 & 13 & 1 & 0 \\ 0 & 6 & 4 & 13 & 1 \end{bmatrix} \mathrm{y}_3 = \begin{bmatrix} 6 & 4 & 13 & 1 & 0 \\ 0 & 6 & 4 & 13 & 1 \end{bmatrix} \mathrm{G}_3\tilde{\mathrm{X}}_{\mathrm{T}_3}$$

$$= \begin{bmatrix} 6 & 4 & 13 & 1 & 0 \\ 0 & 6 & 4 & 13 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{3} & 14 & \mathbf{11} & \mathbf{8} & 4 & \mathbf{10} & \mathbf{8} & \mathbf{4} & 5 & \mathbf{6} \\ \mathbf{12} & 16 & \mathbf{3} & \mathbf{4} & 6 & \mathbf{3} & \mathbf{1} & \mathbf{12} & 8 & \mathbf{4} \\ \mathbf{14} & 11 & \mathbf{7} & \mathbf{2} & 9 & \mathbf{6} & \mathbf{15} & \mathbf{2} & 6 & \mathbf{14} \\ \mathbf{5} & 15 & \mathbf{5} & \mathbf{1} & 5 & \mathbf{12} & \mathbf{4} & \mathbf{6} & 13 & \mathbf{15} \\ \mathbf{3} & 5 & \mathbf{6} & \mathbf{9} & 16 & \mathbf{7} & \mathbf{9} & \mathbf{1} & 14 & \mathbf{10} \end{bmatrix} \tilde{\mathrm{X}}_{\mathrm{T}_3}$$

$$= \begin{bmatrix} \mathbf{15} & 0 & \mathbf{4} & \mathbf{6} & 0 & \mathbf{9} & \mathbf{13} & \mathbf{2} & 0 & \mathbf{11} \\ \mathbf{9} & 0 & \mathbf{15} & \mathbf{3} & 0 & \mathbf{1} & \mathbf{8} & \mathbf{6} & 0 & \mathbf{13} \end{bmatrix} \begin{bmatrix} X_4 \\ X_5 \\ X_{10} \\ X_7 \\ X_9 \\ X_{23} \\ X_{18} \\ X_2 \\ X_{11} \\ X_{15} \end{bmatrix} = \begin{bmatrix} 15 & 4 & 6 & 9 & 13 & 2 & 11 \\ 9 & 15 & 3 & 1 & 8 & 6 & 13 \end{bmatrix} \begin{bmatrix} X_4 \\ X_{10} \\ X_7 \\ X_{23} \\ X_{18} \\ X_2 \\ X_{15} \end{bmatrix}$$

$$= \tilde{\mathrm{V}}\mathrm{X}_{\tilde{\mathrm{W}}}.$$

Note that for this example, the proposed protocol achieves the rate $(\lfloor \frac{K}{D} \rfloor + \frac{R}{L})^{-1} = 2/9$, whereas the Specialized MDS Code protocol of Section 3.4 achieves a lower rate $L/(K - D + L) = 2/19$.

4.6 Main Results for Extensions of IPLT Setting

4.6.1 IPLT with Side Information

The result of Theorem 2 can be extended to IPLT with Side Information (SI). When the identity of every message in the support sets of demand and side information must be protected individually, we can achieve the rate $(\lfloor \frac{K}{D+M} \rfloor + \min\{\frac{R}{S}, \frac{R}{L}\})^{-1}$ for IPLT with USI (or IPLT with CSI, provided that the demand-side coefficient matrix is MDS), where $R = K \pmod{D+M}$ and $S = \gcd(D+M+R, R)$. Recall that in the case of USI, the user initially knows a randomly chosen subset of M messages, and in the case of CSI, L randomly generated MDS coded combinations of M randomly chosen messages are initially known by the user. These results generalize those of [39] and [17] for PIR and PLC with individual privacy.

The achievability scheme for IPLT with CSI is the same as the proposed protocol for IPLT without SI, except with the following modifications: the demand's support index set W is replaced by the set of indices of all messages in the union of the support sets of demand and side information, the demand's coefficient matrix V is replaced by the demand-side coefficient matrix, and D is replaced by $D+M$. Note that this scheme is also applicable for IPLT with USI. However, the optimality of this scheme remains open for both cases of USI and CSI.

4.6.2 IPLT with Non-MDS Coefficient Matrices

The proposed IPLT protocol can be extended to the cases in which the demand's coefficient matrix V is randomly chosen from the ensemble of all $L \times D$ matrices that have full row rank. In the following, we briefly outline the required modifications to the protocol for the two cases of $L \leq S$ and $L > S$. In the case of $L \leq S$, any of the blocks $\mathrm{G}_1, \ldots, \mathrm{G}_n$ which was previously a randomly generated MDS matrix, will be a randomly generated matrix of the same size with full row rank; and the block G_{n+1} will be constructed using a matrix $\mathrm{C} = [\mathrm{C}_1, \ldots, \mathrm{C}_{t+m}]$ with the same size and construction as before, except where the submatrix of C restricted to any $(t+1)$-subset of the column-blocks C_i's is required to have full row rank, instead of satisfying the MDS property. In the case of $L < S$, the blocks $\mathrm{G}_1, \ldots, \mathrm{G}_n$ are constructed the same as above. The construction

of the block G_{n+1} is similar to that of the proposed protocol in Section 3.6 for JPLT with non-MDS coefficient matrices, except where the number of messages K is replaced by $D+R$, and the demand's support index set W is replaced by a randomly chosen D-subset of $\{1,\ldots,D+R\}$. It should be noted that this protocol achieves the same rate as the proposed protocol for IPLT with MDS coefficient matrices. Notwithstanding, it remains open whether this rate is optimal for IPLT with non-MDS coefficient matrices.

5. CONCLUSION AND FUTURE WORK

In this thesis, we introduced the problem of single-server *Private Linear Transformation (PLT)*. This problem includes a single remote server that stores a dataset of K messages, and a user that wants to compute L linear combinations of a D-subset of these messages by downloading the minimum amount of information from the server while protecting the privacy of the D messages required for the computation. The PLT problem generalizes the Private Information Retrieval (PIR) and Private Linear Computation (PLC) problems, which have recently received a significant attention from the information and coding theory community.

We focused on the PLT problem under two different information-theoretic privacy requirements, called *joint privacy* and *individual privacy*, that were recently introduced in the PIR and PLC literature. The joint privacy requirement implies that, from the server's perspective, every D-subset of messages is equally likely to be the support set of the required linear combinations; whereas, the individual privacy requirement implies that, from the perspective of the server, every message is equally likely to belong to the D-subset of messages that constitute the support set of the required linear combinations.

In Chapter 3, for the setting in which the coefficient matrix of the required linear combinations generates a Maximum Distance Separable (MDS) code, we characterized the capacity of PLT with joint privacy, where the capacity is defined as the supremum of all achievable download rates. In addition, we presented lower and/or upper bounds on the capacity of PLT with joint privacy for the settings in which the user has a prior side information about the messages in the dataset, and the settings in which the coefficient matrix of the required linear combinations is not MDS. Our results show that when joint privacy is required, PLT (with or without the help of a prior side information) requires less download overhead when compared to (i) applying a PIR scheme for privately retrieving the messages required for the computation, and performing the computations locally, or (ii) using a PLC scheme multiple times to privately compute each of the required linear combinations separately.

In Chapter 4, we established lower and upper bounds on the capacity of PLT with individual privacy under the assumption of MDS coefficient matrices. We showed that our bounds are tight under certain divisibility conditions. In addition, for the same privacy requirement, we presented lower bounds on the capacity of PLT for the settings with a prior side information and non-MDS coefficient matrices. Our results indicate that, when there is no side information, PLT with individual privacy (IPLT) can be performed much more efficiently than PLT with joint privacy (JPLT), in terms of the download cost; and the advantage of IPLT over JPLT is even more pronounced when the user initially knows a subset of messages or a subspace spanned by them as side information.

In this thesis, we made a significant progress towards characterizing the fundamental limits of the single-server PLT problem. However, there remain several open problems. Below, we have listed a few of these problems.

1. The capacity of JPLT and IPLT with *non-MDS coefficient matrices* remains open. Such matrices are particularly of interest in the scenarios where the number of desired attributes and the dimension of the projected space are relatively large, and the size of the field over which the operations are performed is relatively small.

2. For JPLT and IPLT with side information, we considered the case where the privacy of the identities of messages in the support set of demand and those in the support set of side information must be protected. In some applications, it may be required that only the identities of the messages in the demand's support set must be kept private (jointly or individually), and the identities of the messages in the side information's support set do not need to remain private. Characterizing the capacity of such settings is another direction for future work.

3. The joint and individual privacy conditions can be thought of as two extreme cases of a more general information-theoretic privacy requirement where for a given $1 \leq T \leq D$, the user's query should not change the likelihood of any T-subset of messages to be a subset of D messages in the demand's support set. We refer to this type of privacy as *T-wise privacy*. The T-wise privacy provides a graceful trade-off between the degree of privacy

and the download cost. In particular, joint privacy and individual privacy are two special cases of T-wise privacy for $T = D$ and $T = 1$, respectively. Aside from these two cases, T-wise privacy (for $1 < T < D$) has not been studied in the literature of PIR and PLC. This type of privacy is, however, applicable to several practical scenarios, for instance, the following multi-user PIR scenario. Suppose there are multiple users, each of which wants to privately retrieve T distinct messages. To this end, each user sends the index set of their demands to the same trusted agent. Followed by aggregating the users' requests, the agent then collectively retrieves the demands of all the users from the server, using a PIR scheme that achieves T-wise privacy. This ensures that the identities of T messages required by each user remain private. The design and analysis of efficient PIR or PLC (and more generally, PLT) schemes for achieving T-wise privacy remains an open problem.

4. Another important direction for research is to characterize the fundamental limits of the *multi-server case* of the PLT problem. For PIR and PLC, it was recently shown that, in comparison to the single-server case, the download cost can be substantially lower when there are multiple non-colluding servers that store identical copies of the dataset, see, e.g., [13,14,29]. Similar results were also shown for the cases in which the servers can collude with some physical limitations (see, e.g., [22,23,26]), and the cases in which each server stores a coded version of the messages in the dataset (see, e.g., [22, 25, 30, 31]). These results motivate the study of the PLT problem under the multi-server setting.

5. Besides linear transformation, *non-linear transformations* appear frequently in machine learning and cloud/edge computing applications. For instance, the problem of evaluating multivariate polynomials on a subset of dataset finds applications in distributed stochastic gradient descent for a linear regression task, see, e.g., [52]. Protecting the privacy of the user in such applications, i.e., hiding the identities of the data items required for computation, motivates the design and analysis of privacy-preserving schemes for non-linear transformations.

REFERENCES

[1] W. Itani, A. Kayssi, and A. Chehab, "Privacy as a service: Privacy-aware data storage and processing in cloud computing architectures," in *Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing, 2009. DASC'09.*, pp. 711–716, IEEE, 2009.

[2] H. Takabi, J. B. D. Joshi, and G. Ahn, "Security and privacy challenges in cloud computing environments," *IEEE Security Privacy*, vol. 8, no. 6, pp. 24–31, 2010.

[3] I. Ion, N. Sachdeva, P. Kumaraguru, and S. Čapkun, "Home is safer than the cloud!: privacy concerns for consumer cloud storage," in *Proceedings of the Seventh Symposium on Usable Privacy and Security*, p. 13, ACM, 2011.

[4] E. Stefanov and E. Shi, "Oblivistore: High performance oblivious cloud storage," in *2013 IEEE Symposium on Security and Privacy (SP)*, pp. 253–267, IEEE, 2013.

[5] O. Goldreich, "Secure multi-party computation," *Manuscript. Preliminary version*, pp. 86–97, 1998.

[6] C. Zhao, S. Zhao, M. Zhao, Z. Chen, C.-Z. Gao, H. Li, and Y. an Tan, "Secure multi-party computation: Theory, practice and applications," *Information Sciences*, vol. 476, pp. 357 – 372, 2019.

[7] C. Dwork, "Differential privacy: A survey of results," in *Theory and applications of models of computation*, pp. 1–19, Springer, 2008.

[8] Q. Geng and P. Viswanath, "The optimal mechanism in differential privacy," in *2014 IEEE International Symposium on Information Theory (ISIT)*, pp. 2371–2375, 2014.

[9] D. Boneh, A. Sahai, and B. Waters, "Functional encryption: Definitions and challenges," in *Theory of Cryptography* (Y. Ishai, ed.), (Berlin, Heidelberg), pp. 253–273, Springer Berlin Heidelberg, 2011.

[10] Z. Brakerski and G. Segev, "Function-private functional encryption in the private-key setting," in *Theory of Cryptography* (Y. Dodis and J. B. Nielsen, eds.), (Berlin, Heidelberg), pp. 306–324, Springer Berlin Heidelberg, 2015.

[11] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *Journal of the ACM (JACM)*, vol. 45, no. 6, pp. 965–981, 1998.

[12] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in *IEEE Symposium on Foundations of Computer Science*, pp. 41–50, 1995.

[13] H. Sun and S. A. Jafar, "The capacity of private information retrieval," *IEEE Transactions on Information Theory*, vol. 63, pp. 4075–4088, July 2017.

[14] H. Sun and S. A. Jafar, "The capacity of private computation," *IEEE Transactions on Information Theory*, vol. 65, no. 6, pp. 3880–3897, 2019.

[15] M. Mirmohseni and M. A. Maddah-Ali, "Private function retrieval," in *2018 Iran Workshop on Communication and Information Theory (IWCIT)*, pp. 1–6, April 2018.

[16] S. Kadhe, B. Garcia, A. Heidarzadeh, S. El Rouayheb, and A. Sprintson, "Private information retrieval with side information," *IEEE Transactions on Information Theory*, vol. 66, no. 4, pp. 2032–2043, 2020.

[17] A. Heidarzadeh and A. Sprintson, "Private computation with individual and joint privacy," in *2020 IEEE International Symposium on Information Theory (ISIT)*, pp. 1112–1117, 2020.

[18] E. Kushilevitz and R. Ostrovsky, "Replication is not needed: single database, computationally-private information retrieval," in *38th Annual Symposium on Foundations of Computer Science*, pp. 364–373, Oct 1997.

[19] S. Yekhanin, "Private information retrieval," *Communications of the ACM*, vol. 53, no. 4, pp. 68–73, 2010.

[20] T. Mayberry, E.-O. Blass, and A. H. Chan, "Pirmap: Efficient private information retrieval for mapreduce," in *Financial Cryptography and Data Security* (A.-R. Sadeghi, ed.), (Berlin, Heidelberg), pp. 371–385, Springer Berlin Heidelberg, 2013.

[21] R. Sion and B. Carbunar, "On the computational practicality of private information retrieval," in *Proceedings of the Network and Distributed Systems Security Symposium*, pp. 2006–06, Internet Society, 2007.

[22] R. Tajeddine and S. E. Rouayheb, "Robust private information retrieval on coded data," in *2017 IEEE International Symposium on Information Theory (ISIT)*, pp. 1903–1907, 2017.

[23] R. Tajeddine, O. W. Gnilke, D. Karpuk, R. Freij-Hollanti, C. Hollanti, and S. E. Rouayheb, "Private information retrieval schemes for coded data with arbitrary collusion patterns," in *2017 IEEE International Symposium on Information Theory (ISIT)*, pp. 1908–1912, June 2017.

[24] K. Banawan and S. Ulukus, "Multi-message private information retrieval," in *2017 IEEE International Symposium on Information Theory (ISIT)*, pp. 1898–1902, June 2017.

[25] K. Banawan and S. Ulukus, "The capacity of private information retrieval from coded databases," *IEEE Transactions on Information Theory*, vol. 64, pp. 1945–1956, March 2018.

[26] H. Sun and S. A. Jafar, "The capacity of robust private information retrieval with colluding databases," *IEEE Transactions on Information Theory*, vol. 64, pp. 2361–2370, April 2018.

[27] C. Tian, H. Sun, and J. Chen, "Capacity-achieving private information retrieval codes with optimal message size and upload cost," June 2019.

[28] Z. Chen, Z. Wang, and S. Jafar, "The asymptotic capacity of private search," in *2018 IEEE International Symposium on Information Theory (ISIT)*, pp. 2122–2126, 2018.

[29] K. Banawan and S. Ulukus, "Multi-message private information retrieval: Capacity results and near-optimal schemes," *IEEE Transactions on Information Theory*, vol. 64, pp. 6842–6862, Oct 2018.

[30] S. A. Obead and J. Kliewer, "Achievable rate of private function retrieval from MDS coded databases," *2018 IEEE International Symposium on Information Theory (ISIT)*, pp. 2117–2121, 2018.

[31] S. A. Obead, H.-Y. Lin, E. Rosnes, and J. Kliewer, "Capacity of private linear computation for coded databases," *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 813–820, 2018.

[32] B. Tahmasebi and M. A. Maddah-Ali, "Private sequential function computation," in *2019 IEEE International Symposium on Information Theory (ISIT)*, pp. 1667–1671, 2019.

[33] H. Sun and S. A. Jafar, "Blind interference alignment for private information retrieval," in *2016 IEEE International Symposium on Information Theory (ISIT)*, pp. 560–564, 2016.

[34] R. Tandon, "The capacity of cache aided private information retrieval," in *55th Annual Allerton Conf. on Commun., Control, and Computing*, pp. 1078–1082, Oct 2017.

[35] Y. Wei, K. Banawan, and S. Ulukus, "Cache-aided private information retrieval with partially known uncoded prefetching: Fundamental limits," *IEEE Journal on Selected Areas in Communications*, vol. 36, pp. 1126–1139, June 2018.

[36] Y. Wei, K. Banawan, and S. Ulukus, "Fundamental limits of cache-aided private information retrieval with unknown and uncoded prefetching," *IEEE Transactions on Information Theory*, pp. 1–1, 2018.

[37] A. Heidarzadeh and A. Sprintson, "Private computation with side information: The single-server case," in *2019 IEEE International Symposium on Information Theory (ISIT)*, pp. 1657–1661, July 2019.

[38] A. Heidarzadeh, F. Kazemi, and A. Sprintson, "The role of coded side information in single-server private information retrieval," *IEEE Transactions on Information Theory*, vol. 67, no. 1, pp. 25–44, 2021.

[39] A. Heidarzadeh, S. Kadhe, S. E. Rouayheb, and A. Sprintson, "Single-server multi-message individually-private information retrieval with side information," in *2019 IEEE International Symposium on Information Theory (ISIT)*, pp. 1042–1046, July 2019.

[40] F. Kazemi, E. Karimi, A. Heidarzadeh, and A. Sprintson, "Single-server single-message online private information retrieval with side information," in *2019 IEEE International Symposium on Information Theory (ISIT)*, pp. 350–354, July 2019.

[41] A. Heidarzadeh, B. Garcia, S. Kadhe, S. E. Rouayheb, and A. Sprintson, "On the capacity of single-server multi-message private information retrieval with side information," in *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 180–187, 2018.

[42] S. Li and M. Gastpar, "Single-server multi-message private information retrieval with side information," in *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 173–179, 2018.

[43] Z. Chen, Z. Wang, and S. A. Jafar, "The capacity of T-private information retrieval with private side information," *IEEE Transactions on Information Theory*, vol. 66, no. 8, pp. 4761–4773, 2020.

[44] S. P. Shariatpanahi, M. J. Siavoshani, and M. A. Maddah-Ali, "Multi-message private information retrieval with private side information," in *2018 IEEE Information Theory Workshop (ITW)*, pp. 1–5, 2018.

[45] A. Heidarzadeh, F. Kazemi, and A. Sprintson, "Capacity of single-server single-message private information retrieval with private coded side information," in *2019 IEEE International Symposium on Information Theory (ISIT)*, pp. 1662–1666, July 2019.

[46] F. Kazemi, E. Karimi, A. Heidarzadeh, and A. Sprintson, "Private information retrieval with private coded side information: The multi-server case," in *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 1098–1104, 2019.

[47] J. P. Cunningham and Z. Ghahramani, "Linear dimensionality reduction: Survey, insights, and generalizations," *Journal of Machine Learning Research*, vol. 16, no. 89, pp. 2859–2900, 2015.

[48] A. Heidarzadeh, F. Kazemi, and A. Sprintson, "Capacity of single-server single-message private information retrieval with coded side information," in *2018 IEEE Information Theory Workshop (ITW)*, pp. 1–5, Nov 2018.

[49] E. Bingham and H. Mannila, "Random projection in dimensionality reduction: Applications to image and text data," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '01, (New York, NY, USA), p. 245–250, Association for Computing Machinery, 2001.

[50] R. Roth, *Introduction to Coding Theory*. New York, NY, USA: Cambridge University Press, 2006.

[51] H. Marchand, A. Martin, R. Weismantel, and L. Wolsey, "Cutting planes in integer and mixed integer programming," *Discrete Applied Mathematics*, vol. 123, no. 1, pp. 397 – 446, 2002.

[52] 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 *Proceedings of Machine Learning Research* (K. Chaudhuri and M. Sugiyama, eds.), vol. 89, pp. 1215–1225, 2019.